

Année 2007

N° d'ordre : 72

UNIVERSITÉ DE TECHNOLOGIE
DE BELFORT-MONTBÉLIARD

UNIVERSITÉ DE
FRANCHE-COMTÉ

École Doctorale *Sciences Pour l'Ingénieur et Microtechniques*

THÈSE

Présentée pour obtenir le grade de

DOCTEUR de l'Université de Technologie Belfort-Montbéliard
et de l'Université de Franche-comté

DISCIPLINE : **INFORMATIQUE**

ÉTUDES EN RECHERCHE LOCALE ADAPTATIVE POUR L'OPTIMISATION COMBINATOIRE

Par

Isabelle DEVARENNE

Soutenue le 30 novembre 2007

Jury

Rapporteur	HAO Jin-Kao	Professeur, Université d'Angers
Rapporteur	SEVAUX Marc	Professeur, Université de Bretagne Sud
Examineur	BAHI Jacques	Professeur, Université de Franche-Comté
Examineur	DEFAIX Thierry	Ingénieur de recherche, CELAR
Directeurs de Thèse	CAMINADA Alexandre	Professeur, UTBM
	MABED Hakim	Enseignant Chercheur, UTBM

Remerciements

Je tiens tout d'abord à remercier Alexandre Caminada, Professeur à l'Université de Technologie Belfort Montbéliard UTBM, et Hakim Mabed, Enseignant Chercheur à l'UTBM, pour leur encadrement et leur soutien qui m'ont permis de mener à bien cette thèse.

Je remercie Jin-Kao Hao, Professeur à l'Université d'Angers, et Marc Sevaux, Professeur à l'Université de Bretagne Sud, d'avoir accepté de rapporter ma thèse. Leur aide a été précieuse et leurs remarques très pertinentes. Je remercie aussi Jacques Bahi, Professeur à l'Université de Franche Comté, et Thierry Defaix, Ingénieur de recherche au CELAR, pour leur participation au jury de thèse.

Je tiens à remercier particulièrement Jean-Noel Martin, Professeur agrégé et docteur au laboratoire SET, qui m'a beaucoup aidé et Sid Lamrous, Maître de conférences à l'UTBM, pour son aide sur la partie statistique du travail.

Pour finir, je remercie les membres du laboratoire SET (Systèmes et Transports) ainsi que mes amis pour leur aide et leur soutien au cours de ces trois années.

Table des matières

Introduction	15
1 Adaptation en recherche locale	19
1.1 Notions de base	20
1.1.1 Optimisation combinatoire	20
1.1.2 Les méthodes de résolution	21
1.1.3 Notions de voisinage	24
1.1.4 Mémoire et adaptation	28
1.1.4.1 Qu'est ce que l'adaptation ?	29
1.1.4.2 Quel type de mémoire est utilisé ?	31
1.2 L'adaptation en recherche locale	33
1.2.1 Adaptation avec une seule structure de voisinage	33
1.2.1.1 Méthodes générales	33
1.2.1.2 Méthodes avec durée Tabou dépendante du temps de calcul	35
1.2.1.3 Méthodes avec durée Tabou variable dans un intervalle . .	35
1.2.1.4 Méthodes avec durée Tabou réactive	35
1.2.1.5 Méthodes avec durée Tabou adaptative	37
1.2.2 Adaptation avec plusieurs structures de voisinage	39
1.2.2.1 Méthodes de recherche à voisinage variable	39
1.2.2.2 Méthodes de recherche locale réitérée	42
1.2.2.3 Méthodes de recherche à voisinage large	45
1.2.2.4 Méthodes de type hyperheuristique	47
1.3 Discussion	49
2 Recherche Locale Adaptative et k-coloration de graphe	53
2.1 Contexte de travail	54
2.1.1 Quelques notions de théorie des graphes	54
2.1.2 Quelques notions de coloration	55
2.1.3 Description des problèmes de coloration	55
2.1.4 Benchmarks utilisés	56
2.1.4.1 Instances CNET	56
2.1.4.2 Instances DIMACS	57
2.1.5 Recherche locale pour la coloration de graphe	60
2.2 Étude de la méthode RLA	61
2.2.1 Recherches locales de base	61

2.2.1.1	Les méthodes retenues	62
2.2.1.2	Comparaison des méthodes	62
2.2.2	Détection de boucle et liste Tabou	65
2.2.2.1	Détection de boucle	65
2.2.2.2	Liste Tabou	67
2.2.2.3	Résultat et analyse	67
2.2.3	Adaptation de la durée Tabou	72
2.2.3.1	Description de l'adaptation	72
2.2.3.2	Étude de la répartition des durées Tabou	74
2.2.3.3	Étude de corrélation entre nombres de boucles et de visites	76
2.2.4	Adaptation de la détection de boucle	82
2.2.4.1	Description de l'adaptation	82
2.2.4.2	Résultat et analyse	85
2.3	Comparaison des résultats	96
2.4	Conclusion	98
3	Recherche Locale Adaptative et affectation de fréquences	101
3.1	Formalisation du problème	102
3.1.1	Description physique du problème	102
3.1.2	Les contraintes du problème	105
3.1.2.1	Les contraintes liées aux réseaux	105
3.1.2.2	Les contraintes liées aux plans de fréquences	107
3.1.2.3	Les contraintes liées aux interférences	108
3.1.3	Les objectifs à optimiser	111
3.1.3.1	Les interférences	111
3.1.3.2	L'utilisation des fréquences	111
3.1.3.3	La fonction d'évaluation	112
3.1.3.4	Réalisabilité	113
3.1.4	Les instances	113
3.1.4.1	Les instances privées	114
3.1.4.2	Les instances publiques	116
3.1.4.3	Étude de la combinatoire des instances	116
3.2	Modèles de référence et méthodes	117
3.3	Application de la méthode RLA	120
3.3.1	Schéma général de la méthode	120
3.3.2	La solution initiale	122
3.3.2.1	Description globale	122
3.3.2.2	Résultats et analyse	125
3.3.3	Le processus itératif	127
3.3.3.1	Détection de boucle et liste Tabou	127
3.3.3.2	Choix de la sous-bande	130
3.3.3.3	Les opérateurs de mouvement	133
3.3.3.4	Le contrôle de dégradation	142
3.4	Résultats et analyse sur les scénarios	146

3.4.1 Scénarios publics	147
3.4.2 Scénarios privés	148
3.5 Conclusion	150
Conclusions et Perspectives	153
Publications	157
Bibliographie	158

Table des figures

1.1	Courbe d'une fonction évaluation	21
1.2	Exemple de structures de voisinage	25
1.3	Exemple d'opérateurs de sélection	28
1.4	Schéma de fonctionnement d'une méthode adaptative	30
1.5	La méthode COSEARCH (extraite de [8])	49
2.1	4.75.20 : Application de la méthode déterministe $\mathbf{P M}$	64
2.2	Fonctionnement de la détection de boucle	66
2.3	8.15.20 : Nombre d'itérations aléatoires entre deux itérations déterministes en fin d'exécution	67
2.4	Étude sur l'instance DSJC125.1 (méthode DB+TD)	74
2.5	Étude sur l'instance DSJC500.1 (méthode DB+TD)	75
2.6	Étude sur l'instance DSJC125.1 (méthode DB+TA)	75
2.7	Étude sur l'instance DSJC500.1 (méthode DB+TA)	75
2.8	DSJC125.1 : Corrélacion entre nombre de visites et nombre de boucles par nœud par la méthode DB+TD avec $\alpha = 5\%$ ($\text{corr}((a),(b)) = 0.9622$, évaluation=0)	79
2.9	DSJC1000.5 : Corrélacion entre nombre de visites et nombre de boucles par nœud par la méthode DB+TD avec $\alpha = 1\%$ ($\text{corr}((a),(b)) = 0.9903$, évaluation=39)	80
2.10	DSJC1000.5 : Corrélacion entre nombre de visites et nombre de boucles par nœud par la méthode DB+TD avec $\alpha = 5\%$ ($\text{corr}((a),(b)) = 0.5772$, évaluation=44)	80
2.11	DSJC1000.5 : Corrélacion entre nombre de visites et nombre de boucles par nœud par la méthode DB+TA avec $\alpha = 1\%$ ($\text{corr}((a),(b))=0.9923$, évaluation=33)	81
2.12	DSJC1000.5 : Corrélacion entre nombre de visites et nombre de boucles par nœud par la méthode DB+TA avec $\alpha = 5\%$ ($\text{corr}((a),(b))=0.7350$, évaluation=42)	81
2.13	DSJC500.1 : Nombre d'occurrence par nœud pour les trois méthodes utili- sant l'adaptation du paramètre de détection de boucle (instance parfois résolue)	88
2.14	le450_25d : Nombre d'occurrence par nœud pour les trois méthodes utili- sant l'adaptation du paramètre de détection de boucle (instance non résolue)	89

2.15	DSJC1000.5 : Corrélation entre nombre de visites et nombre de boucles par nœud par la méthode DBA+TA ($\text{corr}((a),(b)) = 0.8008$, évaluation=38) .	90
2.16	DSJC1000.5 : Corrélation entre nombre de visites et nombre de boucles par nœud par la méthode DBA+TA ($\text{corr}((a),(b)) = 0.7831$, évaluation=28) .	91
2.17	DSJC1000.5 : Utilisation de la mémoire par chacune des 4 méthodes	92
2.18	DSJC500.1 : Évolution des nœuds en conflit à différents stades de la recherche (instance résolue)	94
2.19	DSJC500.5 : Évolution des nœuds en conflit à différents stades de la recherche (instance non résolue)	95
2.20	Évolution du nombre de nœuds en conflit par rapport au nombre de nœuds en mémoire au cours de la recherche	95
3.1	Exemple de déploiement de cinq réseaux	102
3.2	Changement de fréquences au cours du temps	104
3.3	Domaines de ressource du scénario PUB02	106
3.4	Domaines de ressource du scénario PUB03	106
3.5	Domaines de ressource du scénario PUB07	106
3.6	TEB élémentaire	109
3.7	Diagramme du fonctionnement de la méthode	121
3.8	Schéma de construction de la solution initiale	123
3.9	Représentation graphique des solutions initiales sur le scénario public PUB10126	
3.10	Mécanisme de détection de boucle	129
3.11	Solution initiale pour la méthode gloutonne "ordonnée"	131
3.12	Choix de sous-bande aléatoire	131
3.13	Choix de sous-bande proportionnel	132
3.14	Choix de sous-bande de contribution maximale	132
3.15	Les possibilités d'agrandissement d'une sous-bande	135
3.16	Chevauchement de fréquences interdites dans le scénario SC02	136
3.17	Présence d'un grand peigne dans le scénario SC03	138
3.18	Impact des mouvements sur la qualité sur le scénario SC01	139
3.19	Impact des mouvements sur la qualité sur le scénario SC03	139
3.20	Variation de la fonction de coût sans contrôle de dégradation (scénario SC01)	143
3.21	Influence du paramètre a sur l'évolution de l'amplitude des dégradations acceptées	145

Liste des tableaux

2.1	Caractéristiques des instances CNET	57
2.2	Caractéristiques des instances DIMACS	58
2.3	Comparaison des recherches locales de base	63
2.4	Comparaison des méthodes sur les instances CNET	68
2.5	Comparaison des méthodes sur les instances DIMACS	70
2.6	Comparaison des méthodes sur les instances DSJC	72
2.7	Influence de l'intervalle de la durée Tabou dynamique	73
2.8	Durée Tabou adaptative sur les instances Leighton	76
2.9	Adaptation de la durée Tabou sur les instances DSJC	77
2.10	Influence de α sur les instances DSJC	77
2.11	Influence du paramètre de détection de boucle sur la méthode DB+TD	83
2.12	Caractéristiques des instances DSJC et Leighton	85
2.13	Influence de l'adaptation du paramètre de détection de boucle	86
2.14	Comparaison avec la littérature	97
3.1	Représentation graphique des perturbations et des liens associés	103
3.2	Notations utilisées sur les réseaux	107
3.3	Notations utilisées sur les plans de fréquences	108
3.4	Notations utilisées sur les interférences	110
3.5	Notations utilisées pour les instances	114
3.6	Description des 20 scénarios privés	115
3.7	Description des 10 scénarios publics	116
3.8	Comparaison des solutions initiales	125
3.9	Influence du type de choix de la sous-bande	130
3.10	Mouvement de changement de pas avec découpage	137
3.11	Mouvement de changement de pas avec agrandissement	138
3.12	Choix adaptatif et choix aléatoire des mouvements	142
3.13	Classement des méthodes de contrôle de dégradation	145
3.14	Procédures de contrôle de dégradation sur les instances privées	146
3.15	Influence de la détection de boucle et de la liste Tabou	147
3.16	Combinaison de la détection de boucle et de la liste Tabou (sur 5 exécutions)	148
3.17	Résultat du projet sur les scénarios privés	150

Liste des algorithmes

1	Méthode <i>Basic VNS</i>	40
2	Méthode <i>ILS</i>	43
3	Méthode <i>LNS</i>	46
4	Méthode <i>Recherche Locale Adaptative</i>	52

Introduction

De l'élaboration d'emploi du temps à la recherche du plus court chemin reliant plusieurs villes, nous sommes souvent confrontés à des problèmes difficilement résolubles sans l'aide d'un algorithme d'optimisation. Un problème d'optimisation consiste à rechercher la meilleure solution optimisant un critère ou encore une fonction donnée. L'objectif est de trouver une solution dite optimale, ou parfois seulement de bonne qualité, minimisant ou maximisant une fonction d'évaluation. En fonction de leur difficulté de résolution les problèmes d'optimisation dénombrable ou combinatoire sont classés en différentes catégories. Lorsque le problème à optimiser est NP-difficile, l'utilisation de méthodes exactes garantissant la résolution optimale du problème est beaucoup trop longue en terme de temps de calcul. Dans ce cas, l'utilisation d'heuristiques ou de métaheuristiques permet d'obtenir des solutions de bonnes qualités, proches de l'optimalité, en un temps raisonnable.

Dans la littérature, il existe un grand nombre de méthodes approchées permettant de traiter les problèmes. On observe que certaines méthodes sont plus intéressantes sur certains types de problèmes sans que l'on puisse en expliquer la raison. C'est à la fois une grande frustration et une grande difficulté pour la mise au point de celles-ci. Dès lors, les méthodes sont souvent comparées uniquement sur les résultats finaux auxquels elles ont aboutis. Nous proposons dans cette thèse de mener un travail d'ingénierie algorithmique en étudiant le comportement interne des méthodes au cours de la recherche. L'ingénierie algorithmique est un procédé de conception d'algorithmes intégrant des données et des situations obtenues à partir de simulations et d'analyses. Elle a pour objectif de permettre la conception d'algorithmes en suivant des principes issus de l'ingénierie des systèmes en général. Afin de comprendre le fonctionnement d'une méthode sur un problème ou une instance, il faut archiver et analyser des informations liées à la simulation, en d'autres termes liées au parcours effectué par la méthode au cours de son exécution. Il existe différents outils dans le domaine de l'analyse de données et des statistiques pouvant être appliqués à l'étude du comportement d'une méthode d'optimisation. Nous avons ainsi basé une partie de notre travail sur l'analyse de spectrogrammes et de coefficients de corrélation issus de l'exécution d'algorithmes.

Dans ce cadre méthodologique de l'ingénierie algorithmique, cette thèse a pour objet de mettre au point une méthode pouvant être utilisée pour des problèmes dont l'évaluation des solutions est très coûteuse en terme de temps et dont l'espace de recherche est composée de nombreux minimums locaux. Par exemple, des problèmes où l'évaluation pourrait

faire appel à un simulateur ou à une boîte noire de calcul dans le contexte d'un processus d'optimisation/simulation comme c'est le cas dans certains problèmes de réseaux et notamment dans un problème réel que nous avons traité. Une évaluation coûteuse a un impact fort sur l'optimisation : elle limite le nombre de solutions voisines qui peuvent être évaluées avant de prendre une décision de mouvement de l'algorithme. C'est pour ces raisons de complexité de l'évaluation que nous nous sommes intéressés aux méthodes de résolution approchée et non aux méthodes exactes, et aussi aux méthodes de recherche locale plutôt qu'aux méthodes à population.

Une méthode de recherche locale est basée sur l'évolution itérative d'une solution unique. Le passage d'une solution vers une autre se fait grâce à la définition de structure de voisinage qui est un élément très important dans la définition de ce type de méthode. En particulier se pose la question de l'adaptation d'une méthode de recherche locale pour permettre à celle-ci de parcourir l'espace de recherche malgré la présence de minimums locaux. Le principal inconvénient des méthodes de recherche locale est leur difficulté à s'extraire des optimums locaux, les empêchant ainsi d'aboutir à un optimum global. La plupart des méthodes de recherche locale utilisent une seule structure de voisinage ce qui rend très difficile l'optimisation de problèmes disposant de nombreux minimums locaux. Mais récemment, différents travaux ont été publiés sur des méthodes de recherche locale combinant différentes structures de voisinage ou faisant varier l'ensemble des solutions accessibles via un mécanisme d'extension et/ou de restriction du voisinage. Ces méthodes se sont montrées très prometteuses sur différents problèmes. La combinaison de différentes structures de voisinage ou de mécanismes d'extension et de restriction du voisinage peut permettre à la méthode de s'extraire plus facilement d'un optimum local et ainsi de devenir performante.

La plupart des méthodes alternent les extensions et les restrictions du voisinage de façon totalement déterministe ou bien totalement aléatoire sans étudier le comportement antérieur de la méthode. L'étude du comportement nécessiterait l'utilisation d'une mémoire afin de retenir les événements passés dans le but de les analyser et de conduire l'adaptation. De plus en plus de méthodes utilisent une mémoire, et notamment la notion de liste Tabou qui était à l'origine de la méthode de Recherche Tabou, cependant toutes ne l'utilisent pas à des fins d'adaptation. L'adaptation et la mémoire sont des termes souvent confondus. Alors que la mémoire permet de sauvegarder des événements et désigne la capacité à retrouver des expériences passées, l'adaptation utilise ces informations sauvegardées afin d'influencer le comportement de la méthode. En réalité, la combinaison des deux types de mécanismes est nécessaire pour disposer d'une méthode réellement adaptative.

Nous avons donc travaillé dans cette thèse sur les mécanismes de mémoire et d'adaptation dans le but de mettre au point une méthode de recherche locale adaptative combinant des mécanismes d'extension et de restriction du voisinage. L'extension du voisinage sera définie par une procédure de détection de blocages de la recherche en étudiant les choix effectués par la méthode afin d'intervenir sur son comportement. Le mécanisme de restric-

tion quant à lui sera basé sur l'utilisation d'une liste Tabou à paramétrage adaptatif pour gérer l'accès aux variables. La combinaison entre ces deux mécanismes, l'un étendant le voisinage l'autre le restreignant, nous a paru un point-clé pour mettre au point une méthode adaptative. Cela s'avèrera en fait déterminant pour définir une méthode robuste capable de s'auto-paramétrer sur un ensemble de problèmes.

Ce mémoire de thèse est divisé en trois chapitres. Le premier présente des définitions ainsi qu'un état de l'art sur les méthodes de recherches locales adaptatives. Les deux suivants se rapportent à une mise au point et une application d'une nouvelle méthode de recherche locale adaptative sur deux problèmes distincts : un problème académique, la k -coloration, et un problème réel, l'affectation de fréquences en réseaux de radiocommunications.

Dans le premier chapitre, nous présenterons tout d'abord les différents problèmes d'optimisation suivi d'une description globale des méthodes utilisées pour les résoudre. Ensuite, nous définirons la notion de structure de voisinage, principale composante des méthodes de recherche locale. Enfin, nous présenterons le lien entre la mémoire et l'adaptation. Ces deux notions sont très liées : la mémoire est nécessaire à l'adaptation cependant l'utilisation d'une mémoire ne signifie pas que la méthode soit adaptative. Avant d'adapter une méthode, il faut définir plusieurs points importants : le type de mémoire à utiliser, la nature des informations sauvegardées, la durée pendant laquelle ces informations sont sauvegardées et/ou utilisées et enfin, il reste à définir comment influencer le comportement d'une méthode. Pour finir, ce chapitre présente les concepts généraux utilisés dans la méthode de recherche locale adaptative que nous proposons.

Le second chapitre présente le travail effectué sur le problème de k -coloration. Ce problème a été très utilisé dans la littérature à cause de ses nombreuses applications. Du fait du très grand nombre de méthodes développées sur ce problème, il constitue une très bonne plate-forme d'étude pour notre travail. La première partie de ce chapitre est consacrée à la description du contexte de travail. Dans cette partie, nous commencerons par définir les notions liées à la théorie des graphes ainsi que celles liées à la coloration suivi de la description du problème de k -coloration. Pour finir cette première partie, une étude des méthodes utilisées dans la littérature pour la coloration est effectuée. La seconde partie de ce chapitre présente toutes les études réalisées pour mettre au point la méthode de recherche locale adaptative. Cette méthode a été créée en fonction des observations effectuées sur son comportement sur la base de nombreuses simulations. Des analyses très approfondies des trajectoires de l'algorithme dans l'espace de recherche ont été conduites en utilisant des outils statistiques. Finalement, l'avant dernière partie de ce chapitre présente une comparaison des résultats sur des problèmes DIMACS avec d'autres méthodes publiées dans la littérature suivi d'une conclusion.

Nous terminerons par un troisième chapitre présentant l'application de la méthode de recherche locale adaptative à un problème d'affectation de listes de fréquences en évocation de fréquences pour des réseaux de radiocommunications. Ce problème réel a été

proposé lors d'un contrat de recherche avec le CELAR où trois équipes de recherche étaient concurrentes. Bien que ce problème soit un problème d'affectation de fréquences, le problème que nous avons traité est très spécifique. Les principales caractéristiques sont la structure des listes de fréquences à allouer à chaque réseau du problème ainsi que le mode de calcul de l'interférence entre les listes de fréquences qui provient d'un simulateur agissant comme une boîte noire. Une première partie présentera la formulation physique et mathématique de ce problème. Il y a plusieurs objectifs et plusieurs contraintes à considérer sur le niveau de qualité et la taille de plans de fréquences. La partie suivante est consacrée à l'étude des problèmes existants dans la littérature se rapprochant du problème à traiter ainsi que des méthodes utilisées pour résoudre ces problèmes. Ensuite, nous présenterons l'application de la méthode de recherche locale adaptative à ce problème et plusieurs analyses de son comportement. Étant donné les qualités particulières de ce problème, différentes heuristiques spécifiques ont été développées et ajoutées à la méthode de recherche locale adaptative. Pour finir, une partie de résultats comparatifs avec d'autres méthodes clôturera le chapitre.

Chapitre 1

Adaptation en recherche locale

L'objectif de ce premier chapitre est de situer et de comprendre les mécanismes d'adaptation d'algorithmes de recherche locale en optimisation combinatoire. Ce premier chapitre est divisé en deux parties distinctes. Dans un premier temps des termes liés à l'optimisation combinatoire et aux méthodes de résolution sont définis. Puis nous présentons les notions de voisinage avant d'introduire les mécanismes généraux d'adaptation et de mémoire dans les méthodes. Dans une seconde partie, nous proposons une analyse détaillée des différents procédés d'adaptation utilisés en recherche locale en distinguant les cas où l'adaptation considère une seule ou plusieurs structures de voisinage. En dernière partie, nous introduisons notre travail par une discussion dans le domaine de l'adaptation en recherche locale.

Sommaire

1.1	Notions de base	20
1.1.1	Optimisation combinatoire	20
1.1.2	Les méthodes de résolution	21
1.1.3	Notions de voisinage	24
1.1.4	Mémoire et adaptation	28
1.2	L'adaptation en recherche locale	33
1.2.1	Adaptation avec une seule structure de voisinage	33
1.2.2	Adaptation avec plusieurs structures de voisinage	39
1.3	Discussion	49

1.1 Notions de base

Cette partie présente quelques notions de base utilisées dans cette thèse. Tout d'abord, des notions liées aux problèmes et aux méthodes en optimisation combinatoire sont définies. Ensuite, les notions de voisinage en recherche locale sont abordées. Nous travaillons essentiellement sur les méthodes de recherche locale qui font évoluer une solution en utilisant la notion de voisinage et un ensemble de règles ou d'opérateurs permettant de définir un sous-ensemble de solutions accessibles parmi les solutions voisines d'une solution donnée. Ensuite nous abordons les notions d'adaptation des algorithmes et de mémorisation d'informations découvertes pendant la recherche.

1.1.1 Optimisation combinatoire

On peut trouver de nombreuses définitions de problèmes d'optimisation combinatoire, nous avons retenu celle de Collette et Siarry [31] qui propose la définition suivante :

Définition 1.1. *Problème d'optimisation*

Un problème d'optimisation se définit comme la recherche du minimum ou du maximum d'une fonction donnée. Mathématiquement, dans le cas d'une minimisation, un problème d'optimisation se présentera sous la forme suivante :

$$\begin{aligned} &\text{minimiser } f(s) \text{ (fonction à optimiser)} \\ &\text{avec } g(s) \leq 0 \text{ (m contraintes d'inégalités)} \\ &\text{et } h(s) = 0 \text{ (p contraintes d'égalités)} \end{aligned} \quad (1.1)$$

En d'autres termes, résoudre un problème d'optimisation $P(S, f)$ revient à déterminer une solution $s^* \in S$ minimisant ou maximisant la fonction f avec S l'ensemble des solutions ou l'espace de recherche et $f : S \mapsto Y$ une application ou une fonction d'évaluation qui à chaque configuration s associe une valeur $f(s) \in Y$.

Il est possible de passer d'un problème de maximisation à un problème de minimisation [107] grâce à la propriété suivante :

$$\max_{s \in S} f(s) \simeq \min_{s \in S} (-f(s)) \quad (1.2)$$

Généralement, une solution $s \in S$ est un vecteur d'un espace à N dimensions.

Définition 1.2. *Problème d'optimisation continue*

Dans le cas de variables réelles, on a $a : S \subseteq \mathbb{R}^N$. On parle alors de problème d'optimisation en variables continues.

Un problème d'optimisation continue (PO) peut être formulé de la façon suivante :

$$(PO) \min_{s \in \mathbb{R}^N} f(s) \quad (1.3)$$

Définition 1.3. *Problème d'optimisation combinatoire*

Un problème d'optimisation combinatoire est un problème d'optimisation dans lequel l'espace de recherche S est dénombrable.

Un problème d'optimisation combinatoire (*POC*) peut être formulé ainsi :

$$(POC) \min_{s \in \mathbb{Z}^N} f(s) \quad (1.4)$$

où une solution s est un vecteur composé de N valeurs entières, soit $S \subseteq \mathbb{Z}^N$.

La principale différence entre problème d'optimisation continue et problème d'optimisation combinatoire repose sur l'utilisation de variables discrètes. Dans les deux catégories de problèmes, une solution $s \in S$ est une instantiation des variables $x_i \in X$, où i est l'indice de la variable dans $[1; N]$, et X est le vecteur de dimensions N correspondant à la solution, et $f(s)$ est son évaluation. Résoudre ces problèmes revient à trouver une *solution optimale* appelée aussi *optimum global*.

Définition 1.4. *Optimum global*

Une solution est un *optimum global* à un problème d'optimisation s'il n'existe pas d'autres solutions de meilleure qualité. La solution $s^* \in S$ est un optimum global ssi :

$$\forall s \in S, \begin{cases} f(s^*) \leq f(s) & \text{dans le cas de minimisation} \\ f(s^*) \geq f(s) & \text{dans le cas de maximisation} \end{cases} \quad (1.5)$$

La figure 1.1 schématise la courbe d'une fonction d'évaluation en faisant apparaître l'optimum global dans le cas d'un problème de minimisation.

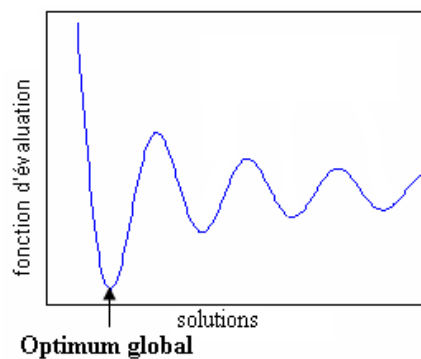


FIG. 1.1 – Courbe d'une fonction évaluation

1.1.2 Les méthodes de résolution

Il existe un grand nombre de problèmes d'optimisation. Les problèmes les plus classiques de programmation linéaire utilisent une fonction objectif ainsi que des contraintes linéaires. Un problème d'optimisation exprimé sous la forme d'un programme linéaire

cherche à maximiser une fonction de N variables x_i en respectant m contraintes linéaires. Il peut s'écrire de la façon suivante [15] :

$$(PL) \begin{cases} \max f(x_1, x_2, \dots, x_N) = c_0 + c_1x_1 + \dots + c_Nx_N \\ \text{s.c.} \\ a_{i1}x_1 + a_{i2}x_2 + \dots + a_{iN}x_N \leq b_i & i = 1, \dots, m \\ x_j \in \mathbb{R} & j = 1, \dots, p \\ x_j \in \mathbb{R}^+ & j = p + 1, \dots, q \\ x_j \in \mathbb{N} & j = q + 1, \dots, N \end{cases} \quad (1.6)$$

avec c_i les coefficients de la fonction à maximiser, a_{ij} et b_i les coefficients des contraintes et \mathbb{R}^+ l'ensemble des réels positifs ou nuls.

Les problèmes de programmation linéaire peuvent utiliser des variables réelles, des variables réelles positives ou nulles et des variables entières. Dans ce cas, ils sont appelés programmes linéaires en variables mixtes. Si toutes les variables du problème sont entières on parle alors de programmes linéaires en variables entières ou encore de programme linéaire en 0-1 si les variables sont de type booléen. Ce type de problème est résoluble de manière exacte via différentes méthodes comme la méthode du simplexe par exemple pour les problèmes en variables continues. La principale caractéristique d'une méthode exacte est la garantie de trouver un optimum global. Beaucoup d'études ont été menées afin de modéliser sous la forme d'un problème de programmation linéaire les problèmes d'optimisation combinatoire en général. Certains problèmes combinatoires ont pu être modélisés sous forme linéaire cependant la plupart d'entre eux ne le sont pas.

Par ailleurs parfois les méthodes *exactes* ne sont pas applicables à cause du temps de calcul. En effet, pour des instances de grande taille le temps nécessaire à la résolution peut être très important et pour certains problèmes de références ou de l'industrie l'optimisation doit être réalisée avec une contrainte de temps de calcul. La méthode Branch and Bound [91, 95] qui consiste à faire une énumération exhaustive de manière implicite en décomposant le problème a été appliquée à différents problèmes d'optimisation comme le problème de coloration de graphe et testée sur les instances DIMACS [95]. Plusieurs instances ont été résolues en quelques heures. Par exemple, une instance aléatoire composée de 125 nœuds a été résolue de façon optimale en deux heures alors que certaines méthodes approchées trouvent une solution de même qualité en moins d'une minute.

Ainsi lorsque l'on dispose d'un temps de calcul limité ou que les problèmes sont trop grands, on peut être amené à se contenter de rechercher une solution de "bonne qualité" sans utiliser de méthodes exactes.

C'est pour ces raisons que des méthodes *approchées* ont été développées. Ces méthodes permettent de trouver des solutions approchées ou que l'on espère proches de la solution optimale en un temps raisonnable. En contrepartie, avec ces méthodes il n'est pas garanti de trouver un optimum global. Parmi ces méthodes approchées, deux classes de méthodes peuvent être identifiées : les heuristiques et les métaheuristiques [76]. Les premières sont

des méthodes dédiées à un problème alors que les secondes sont des méthodes génériques dont les principes généraux sont applicables à différents problèmes d'optimisation combinatoire.

Il n'existe aucune définition reconnue par tous des métaheuristiques, cependant Widmer *et al.* [136] ont regroupé les différentes définitions en proposant un ensemble de propriétés fondamentales. Les caractéristiques des métaheuristiques sont les suivantes :

- Les métaheuristiques sont des stratégies qui permettent de guider la recherche d'une solution optimale.
- Le but visé par les métaheuristiques est d'explorer l'espace de recherche efficacement afin de déterminer des solutions (presque) optimales.
- Les techniques qui constituent des algorithmes de type métaheuristique vont de la simple procédure de recherche locale à des processus d'apprentissage complexes.
- Les métaheuristiques sont en général non-déterministes et ne donnent aucune garantie d'optimalité.
- Les métaheuristiques peuvent contenir des mécanismes qui permettent d'éviter d'être bloqué dans des régions de l'espace de recherche.
- Les concepts de base des métaheuristiques peuvent être décrits de manière abstraite, sans faire appel à un problème spécifique.
- Les métaheuristiques peuvent faire appel à des heuristiques qui tiennent compte de la spécificité du problème traité mais ces heuristiques sont contrôlées par une stratégie de niveau supérieur.
- Les métaheuristiques peuvent faire usage de l'expérience accumulée durant la recherche de l'optimum pour mieux guider la suite du processus de recherche.

Une autre caractéristique des méthodes approchées est souvent utilisée pour effectuer une classification. Elle fait référence au nombre de solutions employées par la méthode. On distingue les méthodes utilisant une population de solutions des méthodes utilisant une unique solution appelées méthodes de recherche locale. Ce type de classification est très courant dans la littérature [47, 109].

Dans le premier cas, les méthodes sont basées sur la notion de population [23]. Elles manipulent un ensemble de solutions en parallèle lors de chaque itération. Les algorithmes évolutionnaires [45, 71] et les algorithmes de colonies de fourmis sont des exemples de méthodes à base de population. Dans le second cas, les méthodes de recherche locale sont basées sur la notion de parcours faisant évoluer une unique solution. Elles sont aussi appelées méthodes de trajectoire. La notion de voisinage est alors primordiale. Il existe un grand nombre de méthodes de recherche locale [4, 60, 103, 121]. Les plus connues sont probablement la Recherche Tabou [12, 46, 75], le recuit simulé [83, 126] et la recherche à voisinage variable [67, 70, 97].

Enfin, il y a aussi des hybridations de méthodes de différentes catégories. Talbi [125] a proposé une taxonomie de l'hybridation en présentant différents niveaux possibles de

combinaison entre les méthodes. Globalement, les méthodes hybrides combinent méthodes exactes et méthodes approchées [6, 48, 49, 101, 104] ou des méthodes approchées entre elles : méthodes de recherche locale et à population par exemple [57, 59, 61].

Comme le titre l'indique, le travail que nous avons effectué durant cette thèse rentre clairement dans le cadre des méthodes de recherche locale pour traiter des problèmes d'optimisation combinatoire.

1.1.3 Notions de voisinage

Dans cette partie, nous nous intéressons à la définition de notions autour du voisinage. Duhamel [47] propose un modèle conceptuel des méthodes par amélioration itérative. Les points communs de ces méthodes peuvent être résumés en trois règles définissant l'exploration du voisinage, la sélection de voisins parmi les solutions du voisinage et enfin la modification effectuée. Le premier groupe de règles permet de construire l'ensemble des solutions voisines de la solution courante, il correspond à ce que nous appellerons structure de voisinage. Le second organise la recherche dans l'ensemble des solutions voisines, nous le noterons opérateur de sélection. Le dernier correspond aux mouvements ou aux règles de passage vers la nouvelle solution à partir du sous-ensemble de voisines sélectionnées.

La définition de la structure de voisinage est une étape clé dans la mise au point d'un algorithme de recherche locale puisqu'elle permet de définir l'ensemble des solutions qu'il est possible d'atteindre à partir d'une solution donnée par une série de transformations. En particulier, la définition du voisinage permet d'identifier toutes les paires de solutions voisines.

Définition 1.5. *Voisinage*

Le voisinage de s est un sous-ensemble de configurations de S directement atteignable à partir d'une transformation donnée de s . Il est noté $V(s)$ et une solution $s' \in V(s)$ est dite voisine de s .

Cette notion de voisinage structure l'espace de recherche dans le sens où elle permet de définir des sous-ensembles de solutions. En particulier, à partir d'une solution donnée, on peut établir plusieurs structures de voisinage selon la transformation que l'on s'autorise, celle-ci étant définie comme une application $V : S \mapsto \mathcal{P}(S)$. Chaque structure de voisinage fournit un voisinage, ou un ensemble de solutions, précis via la transformation définie.

Pour illustrer cette notion, la figure 1.2 présente deux voisinages distincts. Une solution est composée ici de 3 variables binaires. Selon le voisinage V_1 , deux solutions sont voisines si et seulement si une seule composante diffère entre elles. Le voisinage V_2 définit deux solutions voisines si et seulement si une permutation entre deux composantes permet de passer de l'une à l'autre. Depuis une solution s quelconque l'ensemble $V_1(s)$ est différent de $V_2(s)$. Cependant les voisinages se recoupent souvent très largement ; par exemple certaines solutions appartenant à l'ensemble $V_2(s)$ sont accessibles à partir de l'ensemble $V_1(s_1)$. La structure du voisinage permet de se déplacer d'une solution à une autre dans

l'espace de recherche en établissant des chemins entre les solutions.

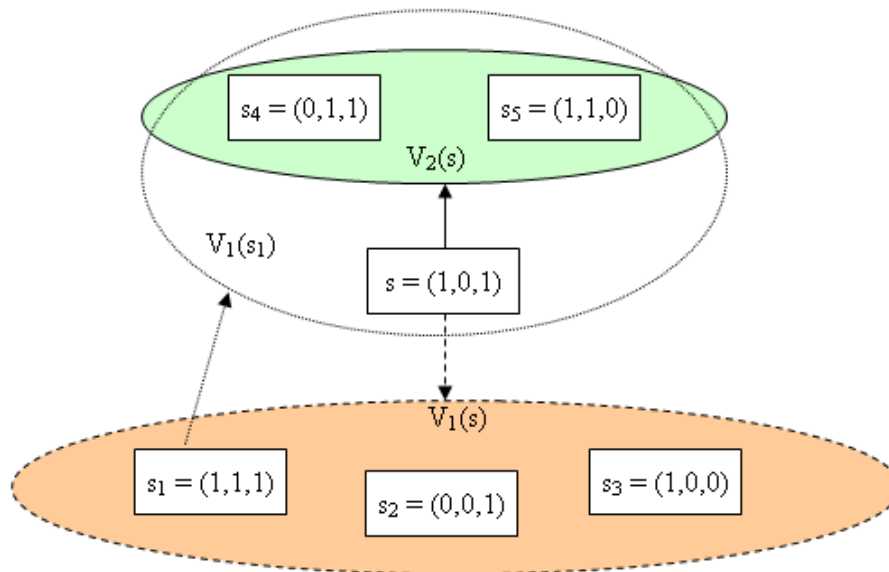


FIG. 1.2 – Exemple de structures de voisinage

Pour caractériser une structure de voisinage dans le contexte de l'optimisation, elle peut être définie en fonction de différentes propriétés. Selon Duhamel [47], il en existe trois principales :

1. La *complexité spatiale* qui détermine la relation entre la taille du voisinage d'une solution et la taille de l'espace de recherche du problème.
2. La *complexité temporelle* qui correspond au temps nécessaire pour l'évaluation des solutions voisines en fonction de la taille du voisinage.
3. La *portée* qui permet de mesurer le degré de transformation entre la solution courante et les solutions voisines. Trois degrés de portée sont proposés dont les qualifications générales sont assez imprécises mais qu'il convient d'apprécier pour chaque problème. Le premier degré de portée est la portée locale qui est caractérisée par l'introduction de peu de modifications dans la solution courante. Le second degré est nommé régional car une plus grande quantité de modifications est introduite dans les solutions voisines par rapport à la solution courante. Enfin, les structures de voisinage de portée globale transforment beaucoup les caractéristiques de la solution initiale.

La portée permet de différencier deux structures de voisinage en fonction de l'éloignement de l'ensemble des solutions accessibles par rapport à la solution initiale. Il existe différentes manières de calculer cette distance ; Weinberg en propose plusieurs dans sa

thèse [132]. L'exemple le plus courant reste cependant la distance de Hamming correspondant au nombre de composantes différentes entre deux solutions voisines¹.

Définition 1.6. *Distance de Hamming*

La distance de Hamming, baptisée du nom de Richard Hamming, désigne le nombre d'éléments différents entre deux ensembles de même taille. Elle mesure le nombre de substitutions requises pour transformer l'un en l'autre.

Mathématiquement, la distance de Hamming d_H entre deux solutions s et s' est calculée de la façon suivante :

$$d_H(s, s') = \sum_{i=1}^N \delta_{s(x_i), s'(x_i)} \quad (1.7)$$

où $\delta_{i,j}$ est le symbole de Kronecker :

$$\delta_{i,j} = \begin{cases} 1 & \text{si } i = j \\ 0 & \text{sinon} \end{cases} \quad (1.8)$$

et $s(x_i)$ la valeur de la composante x_i dans la solution s , et N la taille du vecteur solution.

Définition 1.7. *Portée d'une structure de voisinage*

La portée d'une structure de voisinage $V(s)$ est la distance d maximale des solutions $s' \in V(s)$ par rapport à la solution s .

Propriété 1.1. Deux structures de voisinage V_1 et V_2 sont de portées identiques si et seulement si :

$$d(s, V_1(s)) = d(s, V_2(s)), \forall s \quad (1.9)$$

avec,

$$d(s, V_k(s)) = \max_{s' \in V_k(s)} (d(s, s')) \quad (1.10)$$

et $d(s, s')$ la distance entre les solutions s et s' . Elles sont de portées différentes dans le cas contraire.

On peut maintenant définir plus précisément les notions d'optimum local et global.

Définition 1.8. *Optimum local*

Une solution $s \in S$ est un optimum local si et seulement si il n'existe pas de solution $s' \in V(s)$ dont l'évaluation est de meilleure qualité que s , soit :

$$\forall s' \in V(s), \begin{cases} f(s) \leq f(s') & \text{dans le cas d'un problème de minimisation} \\ f(s) \geq f(s') & \text{dans le cas d'un problème de maximisation} \end{cases} \quad (1.11)$$

avec $V(s)$ l'ensemble des solutions voisines de s .

¹La distance de Hamming est particulièrement adaptée aux solutions composées de variables binaires.

La définition d'un optimum local est liée à la structure de voisinage. En d'autres termes, un optimum local pour une structure de voisinage donnée ne l'est pas forcément pour une autre structure de voisinage.

Définition 1.9. *Optimum global*

Une solution s qui vérifie la propriété précédente pour toutes les structures de voisinage du problème est appelée optimum global.

Beaucoup d'algorithmes de recherche spécifiques à un problème (heuristiques) ou généraux (métaheuristiques) utilisent une seule structure de voisinage. L'intérêt est la simplicité de mise en œuvre et la meilleure compréhension de ce que fait l'algorithme. Cependant, l'utilisation de plusieurs structures de voisinage présente un intérêt certain : une solution s reconnue comme un optimum local pour une structure de voisinage donnée peut ne pas être un optimum local pour une autre structure de voisinage. En effet, un optimum local est défini en fonction de la structure de voisinage selon la définition que nous avons donnée ci-dessus. Il peut donc être intéressant d'utiliser des algorithmes combinant plusieurs structures de voisinage pour la recherche dans les problèmes disposant de nombreux minimums locaux. La méthode de recherche à voisinage variable (VNS) [13, 37], la méthode de recherche locale réitérée (ILS) [92, 94] ou encore la méthode multi-voisinages multi-opérateurs [137] en sont des exemples récents. Les algorithmes génétiques en sont aussi un exemple en combinant la plupart du temps une structure de courte portée via la mutation et une autre de longue portée via le croisement [64]. Nous reviendrons sur le fonctionnement précis de ces méthodes ultérieurement.

A partir d'une structure de voisinage donnée, on peut définir des règles qui permettent de sélectionner une ou plusieurs solutions accessibles dans le voisinage d'une solution quelconque pour permettre aux algorithmes de s'adapter. Ces règles conduisent donc à restreindre le voisinage afin de limiter le choix de la solution voisine sur laquelle pourra se déplacer l'algorithme. Par abus de langage, on peut considérer que cela amène à redéfinir le voisinage. La liste Tabou [46] est probablement le plus connu de ces mécanismes ; il y a plusieurs façons de gérer le statut Tabou de solutions voisines mais il s'agit toujours de restreindre la liste des solutions accessibles à l'intérieur d'une structure de voisinage donnée. Nous définissons ces règles comme des opérateurs de sélection dans le voisinage, un peu à la manière dont le font les algorithmes génétiques pour d'autres motivations.

Définition 1.10. *Opérateur de sélection*

Un opérateur de sélection est défini par un ensemble de règles permettant de choisir un sous-ensemble de solutions parmi l'ensemble des solutions voisines d'une solution quelconque pour une structure de voisinage donnée.

Propriété 1.2. Deux opérateurs de sélection sont considérés comme identiques si les ensembles de solutions accessibles en appliquant ces opérateurs sur un voisinage donné d'une même solution sont identiques.

Notons S l'espace des solutions, V_k une structure de voisinage utilisée, et ϕ_1 et ϕ_2 deux opérateurs définis de la façon suivante : $\phi_{1,k} : s \rightarrow S_{1,k} \subseteq V_k(s)$ et $\phi_{2,k} : s \rightarrow S_{2,k} \subseteq V_k(s)$. On définit formellement deux opérateurs de sélection identiques par :

$$\phi_{1,k} = \phi_{2,k} \iff (\forall s \in S, S_{1,k} = S_{2,k}) \quad (1.12)$$

La figure 1.3 présente deux opérateurs distincts définis pour une même structure de voisinage V_1 et conduisant à des ensembles différents de solutions accessibles. Le premier opérateur ϕ_1 est par exemple l'ensemble des solutions améliorant la solution courante soit :

$$\phi_1(s) = \{s' \mid f(s') < f(s), s' \in V(s)\} \quad (1.13)$$

Le second opérateur ϕ_2 correspond par exemple à une descente de plus grande pente. Il est défini de la façon suivante :

$$\phi_2(s) = \{s' \mid \forall s'' \in V(s), f(s') < f(s''), s' \in V(s)\} \quad (1.14)$$

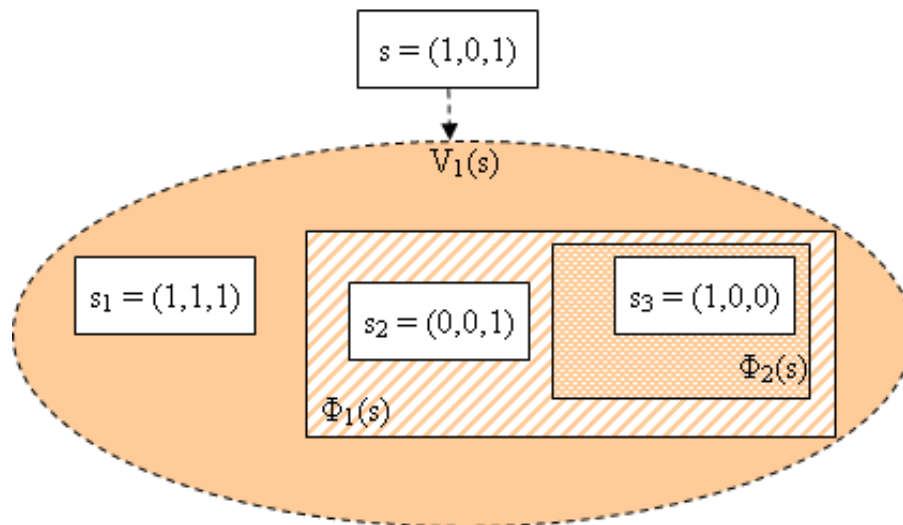


FIG. 1.3 – Exemple d'opérateurs de sélection

Ces deux opérateurs conduisent respectivement aux ensembles $\{s_2, s_3\}$ pour ϕ_1 et $\{s_3\}$ pour ϕ_2 . Ils restreignent simplement le voisinage de s avant le choix de la solution suivante de l'algorithme via un opérateur de mouvement comme par exemple un tirage aléatoire. Par exemple, une liste Tabou [46] est un mécanisme générique de restriction du voisinage avant le mouvement mais de très nombreuses règles spécifiques sont aussi utilisées comme opérateur de sélection.

1.1.4 Mémoire et adaptation

Certains algorithmes utilisent une mémoire afin d'exploiter l'historique de la recherche au cours de l'optimisation alors que d'autres n'ont aucune mémoire de leur passé. Pour

les algorithmes sans mémoire, l'action à réaliser est totalement déterminée par la situation courante. Les algorithmes utilisant une mémoire peuvent choisir l'action à effectuer à partir de la solution courante et en exploitant l'historique de leur propre recherche. On considère plusieurs étendues de l'historique : l'utilisation d'une mémoire à long terme [112] qui peut prendre en compte l'évolution de l'algorithme depuis le début de l'exécution et celle d'une mémoire à court terme [46] qui ne s'intéresse qu'à un nombre limité d'évènements récents. En général, les méthodes utilisant une mémoire pour diriger leur action sont qualifiées, à tort ou à raison, d'adaptative ; c'est une question ouverte car en effet, on peut considérer que l'adaptation peut qualifier aussi une simple réaction à la situation courante.

Cette partie présente dans un premier temps une définition du terme d'adaptation. Ensuite, dans l'hypothèse d'une adaptation liée à la mémorisation d'un historique de la recherche de l'algorithme, les différents types de mémoire et les différents types d'informations mémorisées utilisés par les mécanismes adaptatifs sont expliqués. Enfin nous présenterons un état de l'art des mécanismes adaptatifs utilisés dans la littérature.

1.1.4.1 Qu'est ce que l'adaptation ?

Le verbe *adapter* signifie "modifier une conduite ou une situation", "s'ajuster à une fonction ou à une circonstance"². En optimisation, sa définition est un peu plus spécifique. En général, l'adaptation signifie une modification du comportement de la méthode en fonction de connaissances acquises au cours de la recherche. Ces connaissances peuvent être liées à un apprentissage ou non, l'apprentissage étant l'acquisition par l'expérience de connaissances pouvant être utilisées. C'est dans ce cadre de production et d'utilisation de connaissances acquises que la mémoire intervient ; sans mémoire, il peut y avoir adaptation mais alors elle se fait sans expérience. Bien évidemment, une adaptation avec ou sans expérience ne conduit pas forcément au même résultat. Dans l'aventure humaine, et notamment en résolution de problème, l'expérience est considérée en général comme une valeur ajoutée.

La figure 1.4 schématise le fonctionnement de l'adaptation dans le cadre de l'optimisation. Sur ce schéma, nous avons présenté deux cas : avec et sans mémoire. Le fonctionnement sans mémoire correspond au flot décrit par les flèches pleines et celui avec mémoire aux flèches en pointillées. Sans la mémoire, la méthode fonctionne sans expérience. Si on considère que l'expérience est incluse dans la solution elle-même sans être explicitement identifiée, on se rapproche d'un processus d'évolution dans lequel l'adaptation à l'environnement est directement codée dans les gènes comme pour les algorithmes évolutionnistes [124]. Dans le cas d'une mémoire explicite, des informations liées directement à l'historique de la recherche ou à son analyse sont stockées et sont utilisées pour guider le comportement de la méthode.

Nous avons observé dans la littérature deux grandes catégories de méthodes adap-

²extrait de la définition du centre national de ressources textuelles et lexicales <http://www.cnrtl.fr/lexicographie/>

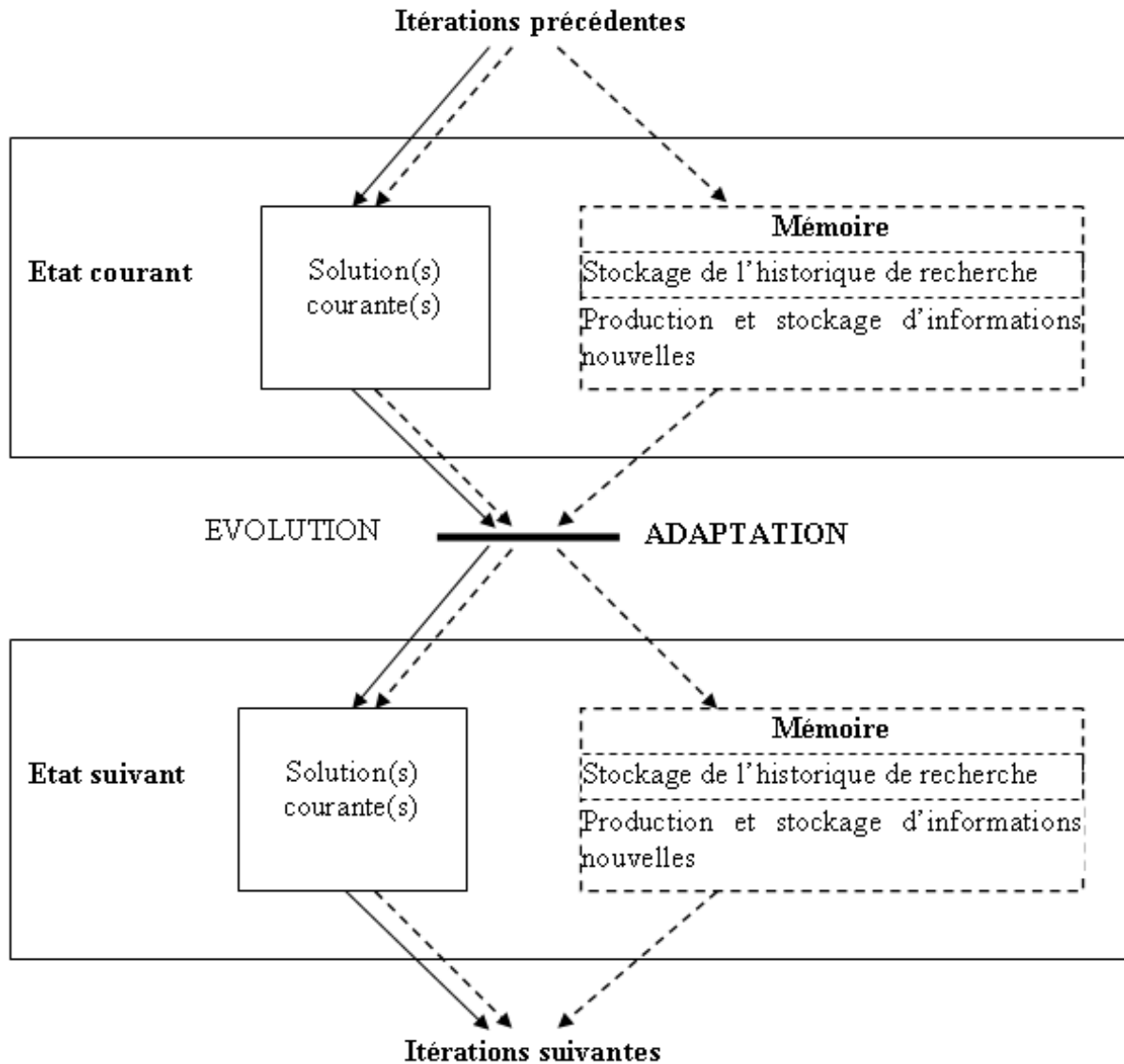


FIG. 1.4 – Schéma de fonctionnement d'une méthode adaptative

tatives dépendant de la période pendant laquelle l'historique de recherche influence les choix effectués par la méthode : cette influence peut être continue au cours de l'exécution ou périodique. Nous les appellerons *adaptation continue* pour la première et *adaptation périodique* pour la seconde.

Définition 1.11. *Adaptation continue*

L'adaptation continue consiste à ajuster en permanence les paramètres de la méthode à partir d'informations extraites en continu.

Définition 1.12. *Adaptation périodique*

L'adaptation périodique consiste à modifier à échéances fixes les paramètres de la méthode

à partir d'informations extraites au cours de périodes spécifiques de la recherche.

Parmi les méthodes adaptatives périodiques, nous distinguons deux options. D'une part, les méthodes qui utilisent une période en début d'exécution pendant laquelle un (ou plusieurs) paramètre(s) sont ajustés et à la fin de cette période d'ajustement les paramètres ne sont plus modifiés ; c'est par exemple le cas du recuit simulé qui définit les paliers de température au début du processus après une étape d'exploration. Et d'autre part les méthodes qui alternent continûment des phases d'ajustement de paramètres et des phases d'utilisation de ceux-ci. Pendant les phases d'utilisation, aucune modification des paramètres n'est effectuée.

1.1.4.2 Quel type de mémoire est utilisé ?

Pour adapter une méthode ou un comportement en général en fonction de l'expérience acquise, il est nécessaire de mémoriser des informations sur ces expériences, en l'occurrence ici sur l'historique de la recherche. Il est donc important de définir la notion de mémoire. D'après le dictionnaire HachetteTM, la mémoire est la "fonction par laquelle s'opèrent dans l'esprit la conservation et le retour d'une connaissance antérieurement acquise". Nous nous sommes intéressés à la définition de la mémoire utilisée en psychologie qui nous donne plus d'informations sur cette fonction.

Définition 1.13. *Mémoire*

La mémoire est un système de stockage et de récupération d'informations qui nous permet de connaître, de nous repérer et d'évoluer dans le monde qui nous entoure³.

En psychologie cognitive, on distingue mémoire à court terme et mémoire à long terme. La mémoire à court terme est une mémoire temporaire limitée en temps à environ 1 minute dans le contexte humain. La mémoire à long terme permet le codage et le stockage durable des informations de manière organisée. Cette mémoire se divise en deux types principaux : la mémoire implicite et la mémoire explicite. La mémoire implicite regroupe les savoir-faire et habitudes acquises alors que la mémoire explicite rassemble les faits et les évènements.

La mémoire implicite est définie ainsi : "La mémoire implicite est inconsciente, on apprend sans retenir l'expérience du passé". La mémoire implicite nous permet par exemple de marcher sans en avoir conscience. En optimisation, l'état courant de la recherche est vue comme une "mémoire implicite" : la solution courante est le résultat des itérations passées et elle contient, ou devrait contenir, les meilleures acquisitions. Cependant, aucun souvenir des choix et des actions ayant permis d'aboutir à la solution courante n'est conservé. Cette qualité est particulièrement pertinente pour les algorithmes évolutionnaires [45] par analogie avec l'expérience acquise par les individus au cours des évolutions, soit des générations de la population qui contient les solutions.

³<http://schwann.free.fr/lamemoire.html>

En psychologie, la mémoire explicite permet l'acquisition de connaissances de façon consciente afin d'apprendre et ainsi de modifier le comportement futur de l'individu : "La mémoire explicite est consciente, elle a la capacité de garder les événements passés". Par exemple, ce type de mémoire permet le stockage de souvenirs. Typiquement, la Recherche Tabou [62] possède et utilise une mémoire explicite d'itérations passées en conservant les informations plus ou moins complètes de la recherche déjà effectuée : solutions complètes, variables choisies ou mouvements réalisés.

Toutes les méthodes ne se classent pas aussi simplement selon leurs fonctions de mémorisation, certains mécanismes mis en oeuvre en optimisation sont plus difficiles à identifier. Par exemple lorsque la méthode se base sur un apprentissage par pénalité/récompense. Il y a des informations supplémentaires à la solution mais seule la valeur courante de ces paramètres est connue à chaque instant ; cette valeur représente une analyse de toutes les itérations passées. L'apprentissage par pénalité/récompense, ou apprentissage par renforcement [81, 129], est un mécanisme d'apprentissage non-supervisé reposant sur le principe suivant : "un système autonome doit apprendre un comportement en tirant parti de sa propre expérience". Cette expérience est renforcée via une récompense lorsque le comportement est jugé "bon" ou au contraire abandonnée via une pénalité dans le cas inverse. On parlera plutôt de mémoire implicite car les faits et les événements passés ne sont pas précisément marqués, on a simplement une vue cumulative des événements qui correspond à un savoir-faire acquis. Hvattum *et al.* [77] utilisent cette forme d'apprentissage pour modifier la fonction de coût et adapter le fonctionnement de leur méthode au cours de la recherche. La fonction de coût inclut un poids dynamique recalculé après chaque itération. Il permet d'intensifier la recherche autour de solutions "réalisables" qui respectent les contraintes du problème. De même Kulturel-Konak *et al.* [114] utilisent une fonction de pénalité qu'ils ajoutent à la fonction de coût au sein d'une Recherche Tabou. Cette pénalité dépend de la distance de la solution à la meilleure solution obtenue précédemment. Une autre méthode utilise aussi la modification de la fonction d'évaluation, c'est la recherche locale guidée GLS (*Guided Local Search*) [82, 131]. Cette méthode est basée sur l'utilisation d'une méthode de recherche locale. Lorsque la recherche locale aboutit à un optimum local, la fonction de coût est modifiée et la méthode de recherche locale est relancée. Cette modification de la fonction de coût se fait grâce à des facteurs de pénalités.

Pour influencer le comportement d'une méthode, les faits et les événements qui se produisent et qui peuvent être stockés dans une mémoire peuvent être de différente nature. Laguna [86] et Widmer *et al.* [136] en ont recensé quatre :

1. La fréquence : les mémoires basées sur la fréquence comptabilisent le nombre de fois qu'une solution, ou qu'une variable, ou qu'une valeur ou encore qu'un mouvement a été utilisé.
2. La récence : les mémoires basées sur la récence s'intéressent non pas à quantifier le nombre de visites mais à dater la dernière visite du critère à mémoriser.
3. La qualité : les mémoires basées sur la qualité retiennent tout ou partie des meilleures solutions rencontrées.

4. L'influence : les mémoires basées sur l'influence mémorisent les choix judicieux effectués au cours de la recherche ayant abouti par exemple à l'amélioration de la meilleure solution.

Toutes les informations mémorisées peuvent être utilisées à la production de comportement adapté à une situation. Il reste que l'information stockée ne fait pas tout : il s'agit de l'utiliser de manière pertinente pour une adaptation réussie.

1.2 L'adaptation en recherche locale

Dans l'hypothèse de l'utilisation d'une mémoire capable de stocker des faits recensés au cours de la recherche, cette partie présente les différents procédés utilisés dans la littérature pour adapter et donc modifier le comportement d'une méthode de recherche locale en fonction de la recherche. Nous avons relevé dans la littérature deux familles d'adaptation :

- **Adaptation avec une seule structure de voisinage :**

Dans cette catégorie de méthodes, on ne change pas la nature de la transformation qui produit le voisinage, on effectue plutôt une sélection sur le voisinage par introduction d'une règle d'accessibilité aux solutions voisines via un opérateur de sélection du voisinage. La section 1.2.1 est consacrée à différentes méthodes de recherche locale avec extension et restriction du voisinage à l'intérieur d'une structure établie.

- **Adaptation avec plusieurs structures de voisinage :**

On se place dans le cadre de l'utilisation de plusieurs structures de voisinage par la recherche locale. Récemment, de nombreux algorithmes utilisant différentes structures de voisinage ont vu le jour. La plupart d'entre eux utilise une méthode de sélection de la structure de voisinage prédéfini et invariable au cours de l'exécution. Cependant, quelques travaux sur l'adaptation de ce choix ont été menés. La section 1.2.2 présente ces différents travaux.

1.2.1 Adaptation avec une seule structure de voisinage

1.2.1.1 Méthodes générales

Nous évoquerons deux méthodes générales de recherche locale, souvent classées dans la catégorie des métaheuristiques, dont la recherche de solution est basée sur la restriction du voisinage à partir de deux mécanismes très distincts.

La méthode du recuit simulé mise au point par trois chercheurs de la société IBM, Kirkpatrick *et al.* en 1983 [83], est une méthode de recherche locale inspirée d'un processus utilisé en métallurgie. Ce processus alterne des cycles de refroidissement lent et de réchauffage ou de recuit qui tendent à minimiser l'énergie du matériau. Cette méthode utilise un critère de sélection d'une solution qui est relaxé par rapport à une descente qui

n'accepte que des améliorations de la solution courante. En effet, le recuit simulé peut accepter des solutions de qualité moindre en fonction de la dégradation de la solution considérée. Afin de fixer le niveau de dégradation acceptable à une itération donnée un paramètre de température a été introduit. Ce paramètre de température décroît au cours du temps. Il s'agit d'une méthode adaptative dans la mesure où la courbe de descente de la température et donc la réduction progressive du voisinage sont estimées au début de l'algorithme après quelques itérations [14]. Le paramètre de température est utilisé pour restreindre progressivement, et indépendamment de l'historique de recherche, le voisinage aux solutions qui améliorent la solution courante ou qui ne la dégradent pas au-delà d'un écart de la fonction de coût. Cet écart est important au début pour favoriser l'exploration et progressivement de plus en plus faible pour intensifier. Cet algorithme est toujours très utilisé notamment dans le domaine des réseaux [24].

La Recherche Tabou est une recherche locale dont le fonctionnement consiste à effectuer une exploration complète du voisinage et de garder la trace d'évènements passés dans une mémoire appelée liste Tabou afin de ne pas les reproduire. En pratique, cette méthode introduite par Glover [62, 63] et par Hansen et Jaumard [66] utilise la mémoire pour exclure certains choix et restreindre le voisinage de la solution courante à un sous-ensemble de $V(s)$ de solutions accessibles. Les composantes de solutions, certains mouvements ou les solutions complètes peuvent être enregistrés. Un paramètre de durée Tabou a été introduit afin d'interdire les évènements pendant un nombre donné d'itérations. Lorsque seules des composantes de solutions sont mémorisées l'accès à de bonnes solutions peut être tabou, aussi un critère d'aspiration a été introduit dans la méthode pour spécifier les conditions de retrait du statut tabou.

La durée du statut Tabou peut être soit statique soit dynamique. En durée tabou statique, la valeur de la durée Tabou est fixe durant toute la recherche. En d'autres termes, le nombre d'itérations pendant lequel l'évènement est tabou est constant comme par exemple dans [73]. Une durée Tabou dynamique varie au cours de l'exécution. Plusieurs études, comme celles réalisées par Hao *et al.* [72] sur les listes Tabou dynamiques ont montré que les résultats obtenus pouvaient être plus intéressants que les résultats obtenus avec une liste statique.

La durée Tabou est un paramètre important, il influence énormément les performances de la méthode. Nous présentons dans les parties suivantes différents modèles de durées Tabou dynamiques utilisées dans la littérature. Toutes les méthodes présentées utilisent une liste Tabou cependant elles ne sont pas toutes des Recherches Tabou. Les durées Tabou ont été regroupées en quatre types de fonctionnement. Dans un premier temps, les approches basées sur le temps de calcul sont analysées. Ensuite, la seconde approche utilise un intervalle au cours du temps. Puis, viennent les durées Tabou définies en fonction de l'état courant de la recherche. Enfin, la dernière approche regroupe les durées Tabou évoluant en fonction d'un historique de la recherche.

1.2.1.2 Méthodes avec durée Tabou dépendante du temps de calcul

Dans cette famille d'approches, la durée Tabou est ajustée progressivement en fonction du temps de calcul ou du nombre d'itérations déjà effectuées. Le principe est de réduire la durée Tabou. Cela conduit donc à une intensification de la recherche en fin de processus un peu comme le fait le recuit simulé.

Plusieurs exemples existent dans la littérature. Montemanni *et al.* [98, 99] utilisent une durée Tabou de cette catégorie dans une Recherche Tabou pour résoudre le problème d'affectation de fréquences. Dans cette méthode, la durée Tabou T est décrémentée progressivement pendant la recherche indépendamment de la qualité de la progression de celle-ci. Toutes les It_s itérations, la durée Tabou T est réduite selon la formulation suivante :

$$T = \beta \times T, \text{ avec } 0 < \beta < 1 \quad (1.15)$$

La valeur de β est fixe pendant la recherche. Si $\beta = 1$, on obtient alors une liste statique. La valeur initiale de T est notée T_{init} . La durée Tabou ne peut pas être inférieure à une valeur seuil T_{min} . Les paramètres T_{min} , β et It_s sont respectivement fixés à 10, 0.96 et 5×10^4 pour toutes les instances testées. La valeur initiale T_{init} est définie empiriquement pour chaque instance. La valeur de β assez proche de 1 donne des paliers de réduction assez faible d'un cycle à l'autre. Dans [99], l'utilisation de la durée Tabou dépendante du temps de calcul a été comparée à trois valeurs différentes de durées Tabou statique. La méthode utilisant la durée Tabou dépendante du temps de calcul obtient de moins bons résultats uniquement sur 2 des 39 instances testées.

1.2.1.3 Méthodes avec durée Tabou variable dans un intervalle

Dans ce deuxième cas, la durée Tabou est choisie aléatoirement dans un intervalle au cours de chaque itération. Ce principe est le plus utilisé dans la littérature et fait office de référence. Les bornes de l'intervalle sont en général choisies en fonction des caractéristiques du problème. Par exemple, Di Gaspero et Schaerf [42] utilisent un intervalle I défini en fonction du nombre N de variables. Plus précisément, l'intervalle I est égal à $[\frac{2}{3} \times N; N]$. Taillard [123] propose une Recherche Tabou appelée (*Robust Taboo Search Method*) pour le problème d'affectation quadratique. Dans cette méthode, la durée Tabou est choisie aléatoirement toutes les $2 \times s_{max}$ itérations dans l'intervalle suivant : $[s_{min}; s_{max}]$, avec $s_{min} = \lfloor 0.9 \times N \rfloor$ et $s_{max} = \lceil 1.1 \times N \rceil^4$. Un autre exemple est donné par Bachelet et Talbi [8]. La durée Tabou utilisée est choisie aléatoirement dans l'intervalle $[\frac{N}{2}; \frac{3 \times N}{2}]$. Dans l'ensemble de ces études, les intervalles sont construits de façon empirique pour chaque problème. Comme indiqué précédemment, ce principe a permis d'améliorer le fonctionnement de la Recherche Tabou à durée fixe [72].

1.2.1.4 Méthodes avec durée Tabou réactive

Dans cette troisième approche, la variation de la durée Tabou est une réaction à l'évolution de la solution courante. Des informations extraites de la solution courante permettent

⁴ $\lfloor \rfloor$ et $\lceil \rceil$ représentent respectivement la partie entière inférieure et supérieure.

de définir la valeur de la durée Tabou lors de chaque itération. Les deux premières méthodes ci-dessous utilisent une durée Tabou dépendant de l'évaluation de la qualité de la solution courante. Les deux suivantes dépendent respectivement du nombre de variables en conflit et du nombre de voisins de la solution courante. Ces travaux ont porté sur des problèmes de coloration et d'emploi du temps.

Galinier et Hao [59] ont proposé d'incrémenter la durée Tabou en fonction de l'évaluation $F(s)$ de la solution courante s . Pour cela, deux paramètres L (choisi aléatoirement dans l'intervalle $[0; 9]$) et λ (fixé expérimentalement à 0.6) ont été utilisés selon la formule suivante :

$$T = L + \lambda \times F(s) \quad (1.16)$$

Dans ce travail, la durée Tabou est proportionnelle au coût de la solution courante. Les valeurs de ces paramètres sont très bonnes pour les instances testées. Cependant, pour d'autres instances, ces paramètres devraient être redéfinis au cas par cas.

Di Gaspero *et al.* [41] ont comparé différentes méthodes sur la base de trois composants : l'exploration du voisinage, les éléments tabous et le calcul de la durée Tabou. Les éléments tabous font référence à la nature des attributs qui apparaissent dans la liste Tabou. Deux durées Tabou sont comparées. Pour la première, la durée Tabou est choisie aléatoirement dans un intervalle $[t_1; t_2]$, avec t_1 et t_2 les meilleures valeurs obtenus suite à différents tests. Pour la seconde, la durée Tabou est celle utilisée par Galinier et Hao [59]. Différentes valeurs des paramètres L et λ ont été utilisées. Deux problèmes ont été utilisés pour la comparaison : la coloration de graphes et la planification des salles d'examen. Pour la coloration, les instances utilisées ont été générées par le générateur de Culberson⁵ et pour le second problème, les instances *Carter's data set* ont été utilisées. Sur ces instances, pour toutes les valeurs testées, les performances de la méthode utilisant une durée Tabou réactive sont assez proches de ceux obtenus avec l'intervalle empirique.

Arntzen et Lokketangen [5] utilisent une Recherche Tabou pour le problème d'emploi du temps dans les universités. Après avoir déplacé un événement E à la place P du jour d , l'évènement E devient tabou. La durée Tabou T de l'évènement E représente le nombre d'itérations pendant lequel tout mouvement impliquant un événement du jour d est interdit. La valeur de la durée Tabou T est calculée par l'équation suivante :

$$T = T_d + r \quad (1.17)$$

avec r un nombre entier choisi aléatoirement entre 0 et 3 et T_d une durée Tabou dynamique choisie dans l'intervalle $[T_b; 2 \times T_b]$. La durée Tabou dynamique est en fonction de l'évaluation de la solution courante. Les plus grandes valeurs de T_d sont choisies lorsque la solution courante est éloignée de la meilleure solution. La durée Tabou T_b basique est fixe pour un grand nombre d'itérations. Elle peut prendre des valeurs comprises entre 5 et 9, et change aléatoirement au cours de la recherche toutes les L itérations avec L un entier aléatoire compris entre 50 et 200. Arntzen *et al.* ont observé que la recherche stagne au

⁵Disponible sur www.cs.alberta.ca/~joe/Coloring/

moins la moitié du temps.

Pour la variation de la durée Tabou en fonction du nombre de variables en conflit $|CN|$, le principe est utilisé par Dorne et Hao [45, 46] selon la formule suivante :

$$T = \text{rand}(10) + 2 \times |CN| \quad (1.18)$$

Dans [45], la durée Tabou est utilisée par une méthode hybride combinant algorithme génétique et Recherche Tabou. Les tests ont été réalisés sur des instances DIMACS telles que les graphes aléatoires DSJC de 500 et 1000 nœuds. La méthode a amélioré les meilleurs résultats précédemment connus en terme de qualité de solution mais aussi en temps. Pour les instances DSJC de 500 et de 1000, les résultats obtenus sont toujours à l'heure actuelle les meilleurs résultats connus. Bien que la durée Tabou soit identique pour toutes les instances, les meilleurs résultats ont été obtenus avec des paramètres différents pour chaque instance.

Dans un autre travail, Hao *et al.* [72] ont utilisé le nombre de solutions voisines de la solution courante. Les solutions s et s' sont dites voisines si et seulement si elles diffèrent uniquement par la valeur d'une variable en conflit. La durée Tabou est calculée de la façon suivante :

$$T = \gamma \times |V(s)| \quad (1.19)$$

où $V(s)$ est l'ensemble des solutions voisines de la solution courante s . De plus, la durée Tabou est limitée par une borne inférieure (respectivement supérieure) égale à N (respectivement $10 \times N$). Cette méthode s'est montrée très compétitive et prometteuse sur les instances CNET.

1.2.1.5 Méthodes avec durée Tabou adaptative

La quatrième et dernière approche regroupe les méthodes utilisant des durées Tabou adaptatives. Ici la valeur de la durée Tabou est ajustée au cours du temps en fonction d'un historique de la recherche et pas seulement de l'état de la solution courante. Six méthodes sont présentées dans cette partie.

La première a été proposée par Capone et Trubian [25]. Seule la dernière itération est utilisée. La durée Tabou T est décrétementée d'une unité après chaque itération améliorante. Une itération améliorante est une itération ayant permis d'améliorer la qualité de la solution courante ; l'analyse de l'historique se limite donc à l'action précédente. Les valeurs négatives de la durée Tabou sont impossibles. Cette définition de la durée Tabou a été utilisée dans une Recherche Tabou pour résoudre le problème d'affectation de fréquences. Les résultats obtenus sont très proches de ceux obtenus avec une durée Tabou statique. A résultats équivalents, l'adaptation de la durée Tabou est plus avantageuse parce qu'elle évite toute une étape de paramétrage de la durée Tabou statique.

La seconde méthode est celle de Battiti et Tecchiolli [9, 12] nommée *Reactive Tabu Search Method*. Cette méthode utilise une mémoire des solutions visitées précédemment.

Après chaque mouvement, l'algorithme vérifie si la solution courante a déjà été rencontrée et il réagit en fonction. La durée Tabou T est incrémentée lorsqu'une configuration est répétée et décrémente lorsque cette solution n'a pas été répétée depuis un temps suffisamment long. Initialisée à 1, la durée Tabou T augmente graduellement en cas de répétition de la façon suivante :

$$T = \min(\max(T \times 1.1, T + 1), L - 2) \quad (1.20)$$

avec L le nombre de variables binaires du problème. Dans le pire des cas, lorsque la durée Tabou est très grande, deux mouvements sont toujours possibles. Dans [12], la méthode a été appliquée au problème d'affectation quadratique QAP (*Quadratic Assignment Problem*). Comparée à la Recherche Tabou utilisant une valeur fixe de durée Tabou ou une valeur choisie aléatoirement dans un intervalle, la méthode présentée s'est montrée très compétitive en particulier pour les problèmes de grande taille.

Les deux exemples suivants sont présentés dans la thèse de Blöchliger [16]. Tout d'abord, la méthode FOO (*Fluctuation Of the Objective function*) est basée sur l'observation de l'évolution de la fonction d'évaluation pendant la recherche. Si aucun changement significatif n'est observé alors la durée Tabou est augmentée. Plus précisément, au cours des ϕ dernières itérations, si la différence δ_f entre la plus grande et la plus petite valeur de fitness est plus petite qu'une valeur seuil b alors la durée Tabou est incrémentée de la valeur η . Dans le cas contraire, elle est décrémente. Les valeurs des paramètres ϕ , η et b sont choisies aléatoirement toutes les ϕ itérations parmi des intervalles propres à chacune des variables. Ces intervalles sont déterminés de façon empirique pour chaque instance de problème. Cette méthode a été comparée à celle proposée par Galinier et Hao [59] sur le problème de coloration de graphes sur des instances DIMACS. Globalement, l'utilisation de la définition de la durée Tabou de Galinier donne de meilleurs résultats que la méthode FOO bien que cette dernière ait obtenu de très bon résultats sur certaines instances.

Blöchliger [16] a proposé un autre mécanisme d'adaptation de la durée Tabou, nommé ACD (*Approximated Cycle-Detection scheme*). L'objectif de cette méthode est de détecter des cycles sans mémoriser l'ensemble des solutions visitées. Pour cela, une solution dite de référence ainsi qu'une méthode de calcul de distance entre deux solutions sont utilisées. A chaque itération, la distance entre la solution courante et la solution de référence est calculée. Si cette distance est nulle alors un cycle est détecté et la durée Tabou est incrémentée d'une valeur η . Dans le cas contraire, la durée Tabou est lentement décrémente. Le paramètre η varie pendant la recherche. Initialisé à 5, il est incrémenté de 5 lorsqu'un cycle est détecté et décrémente de 1 toutes les 15 000 itérations. L'efficacité de cette méthode dépend essentiellement de la mise à jour de la solution de référence et de la distance utilisée. La solution proposée est d'effacer la solution de référence lorsque la solution courante est de meilleure qualité en terme d'évaluation. La solution courante devient ainsi la nouvelle solution de référence. De plus, si la solution courante est très éloignée de la solution de référence, alors on considère que la recherche a définitivement quitté la région de la solution de référence et la solution courante devient alors la nouvelle solution de référence. Les questions de symétrie dans le problème ne sont pas prises en

compte pour les distances, ainsi deux solutions en apparence éloignées peuvent en réalité être identique à une permutation près. Sur le problème de coloration, ce mode de calcul de la durée Tabou a été comparé à la définition proposée par Galinier et Hao [59]. Comme pour la méthode FOO, la méthode ACD a obtenu pour certaines instances de meilleurs résultats mais globalement la méthode de Galinier est plus robuste.

Un autre travail concerne la méthode CN-Tabou (*Consistent-Neighborhood Tabu*) [51, 52] qui utilise aussi une durée Tabou adaptative. Cette méthode utilise des solutions partielles à l'inverse des autres méthodes citées précédemment qui utilisaient toutes des solutions complètes. Lors de chaque itération de la méthode, une nouvelle allocation (x_i, v_i) est réalisée, avec x_i la variable modifiée et v_i sa nouvelle valeur, pour compléter la configuration courante. Pour chaque variable x_j adjacente à x_i et pour chaque valeur potentiellement conflictuelle v_k avec la nouvelle affectation, le couple (x_j, v_k) est mis tabou. La durée Tabou de chaque couple (x_j, v_k) est déterminée par la fréquence d'affectation $nb_{mvt}(x_j, v_k)$ de la variable x_j à la valeur v_k . Plus cette valeur aura été affectée à cette variable, plus la durée Tabou sera importante.

$$T(x_j, v_k) = nb_{mvt}(x_j, v_k) \quad (1.21)$$

Sevaux et Thomin [117, 118] ont proposé une méthode de Recherche Tabou pour un problème d'ordonnancement sur machines parallèles. La durée Tabou repose sur la détection de cycles au cours de la recherche. Un cycle est défini comme la succession des mêmes configurations. La longueur d'un cycle est forcément supérieure à la durée Tabou. Si un cycle est détecté, la durée Tabou est augmentée ; elle est fixée à la taille du cycle détecté.

A l'exception des deux dernières méthodes, dans les autres méthodes citées avec des durées Tabou adaptatives, aucune information historique n'est employée pour diminuer ou augmenter la valeur de la durée Tabou de façon adaptative. Dans ces méthodes, l'historique de la recherche est employé pour déterminer quand la durée Tabou doit être modifiée mais pas la valeur de l'incrément ou du décrétement qui suit une procédure indépendante avec des valeurs fixes. Au contraire, dans CN-Tabu, l'historique est utilisé pour calculer la valeur de l'incrément mais pas pour déterminer le moment où chaque variable devient tabou.

1.2.2 Adaptation avec plusieurs structures de voisinage

1.2.2.1 Méthodes de recherche à voisinage variable

Proposée par Hansen et Mladenovic en 1997 [97], la méthode de recherche à voisinage variable VNS (*Variable Neighborhood Search*) est une métaheuristique récente qui exploite le changement de structure de voisinage, que ce soit en descente vers un optimum local ou pour s'échapper de ces optimaux locaux.

La méthode de base *Basic VNS* [67, 69] fonctionne en plusieurs étapes décrite par l'algorithme 1. Tout d'abord elle nécessite la définition de différentes structures de voisinage, notées $V_1, \dots, V_{k_{max}}$, généralement de portées différentes. Une relation d'ordre entre les

structures de voisinage est à définir en fonction du problème. A partir d'une solution de départ s , un voisin de s est sélectionné en utilisant la structure de voisinage V_1 . Si la solution obtenue suite à l'application d'une méthode de recherche locale (qui est à définir) est meilleure que la solution de départ alors la recherche reprend à partir de cette solution. Sinon, une solution est choisie en utilisant un voisinage de portée supérieure, soit ici le voisinage V_2 . La recherche locale est appliquée sur cette solution, et ainsi de suite. Cette méthode permet de sortir d'un optimum local si celui-ci est un optimum local pour une portée de voisinage plus petite que la valeur k_{max} .

Algorithme 1 Méthode *Basic VNS*

ENTRÉES: solution initiale s_0 , un ensemble de structures de voisinage V_k , avec $k = 1, \dots, k_{max}$

SORTIES: meilleure solution s^*

```

1:  $k \leftarrow 1$ 
2:  $s^* \leftarrow s_0$ 
3:  $s \leftarrow \text{RechercheLocale}(s_0)$ 
4: tantque  $k \leq k_{max}$  faire
5:    $s' \leftarrow \text{ChoixAléatoire}(V_k(s))$ 
6:    $s'' \leftarrow \text{RechercheLocale}(s')$ 
7:   si  $f(s'') < f(s)$  alors
8:     la nouvelle solution de départ est modifiée :  $s \leftarrow s''$ 
9:      $s^* \leftarrow s$ 
10:   retourner au premier voisinage  $V_1$  ( $k \leftarrow 1$ )
11: sinon
12:    $k \leftarrow k + 1$ 
13: fin
14: fin tantque

```

Avanthay *et al.* [7] ont utilisé cette méthode pour résoudre le problème de k -coloration de graphe. Pour toutes les instances DIMACS testées, la méthode VNS obtient de meilleurs résultats que la méthode Tabucol utilisée seule. La méthode Tabucol est une Recherche Tabou appliquée aux problèmes de coloration proposée par Hertz *et al.* [74, 75]. Les paramètres de [7] sont les suivants :

- **Solution initiale** : générée aléatoirement.
- **Condition d'arrêt** : un nombre Max_{VNS} d'itérations sans amélioration de la solution s^* est utilisé. Il est défini en fonction de résultats obtenus suite aux différents tests effectués.
- **Méthode de recherche locale** : en guise de recherche locale la méthode Tabucol est utilisée. A partir d'une solution s , la méthode génère une solution s' en modifiant la couleur d'un nœud en conflit dans la solution s . De plus, l'association nœud et ancienne couleur est mise dans une liste Tabou afin d'éviter d'effectuer le mouvement

inverse. Au bout de Max_{Tabu} itérations sans amélioration de la meilleure solution, la méthode Tabucol est arrêtée. Cette valeur de Max_{Tabu} est très importante ; une valeur trop petite ne permettra pas à la méthode Tabucol d'améliorer la solution s alors qu'une valeur trop grande ralentira énormément la méthode VNS. Après expérimentation, la valeur de $10 \times N$ a été choisie, la valeur de 10 correspondant à la taille de la liste Tabou et N au nombre de nœuds du graphe.

A partir du squelette de la méthode *Basic VNS* présenté en algorithme 1, différentes variantes ont été conçues. Dans la méthode GVNS (*General VNS*) [97] les seuls points qui diffèrent sont l'utilisation d'une première méthode pour améliorer la solution initiale et la méthode de recherche locale utilisée. Afin d'améliorer la solution initiale, la méthode de recherche de voisinage variable réduit appelée RVNS (*Reduced Variable Neighborhood Search*) est utilisée. Elle consiste à choisir aléatoirement un voisin s' de s dans le voisinage V_k . Si celui-ci a une meilleure évaluation que la solution s , alors la recherche reprend à partir de cette solution s' . Sinon, un autre voisin s'' est choisi aléatoirement dans le voisinage V_{k+1} . Le voisinage exploré est très réduit, une seule solution est choisie aléatoirement dans chaque voisinage.

Ensuite, en tant que méthode de recherche locale, les auteurs emploient la méthode de descente à voisinage variable VND [37, 68] (*Variable Neighborhood Descent*). Cette méthode déterministe repose sur la méthode de plus grande descente combinée à l'utilisation de voisinages différents. La méthode de plus grande descente explore tout le voisinage d'une solution s à chaque étape. Il s'agit de trouver le voisin s' ayant la plus petite fonction d'évaluation parmi les voisins de s dans le cas d'un problème de minimisation. Si, dans l'ensemble des voisins de V_k , aucune solution n'est meilleure que la solution de départ, le voisinage V_{k+1} est exploré. Cette méthode nécessite une exploration complète du voisinage V_k d'une solution donnée. Le temps nécessaire peut donc être très important, surtout lorsque le voisinage est grand. Den Besten et Stützle [37] ont utilisé cette méthode sur différents problèmes d'ordonnancement et ont obtenu des résultats particulièrement prometteurs.

L'utilisation de plusieurs structures de voisinage nécessite la définition de règles de sélection. Lorsque le choix du voisinage à utiliser se fait sans prise en compte de l'historique de la recherche alors le mécanisme n'est pas adaptatif. La méthode de recherche à voisinage variable basique *Basic VNS* utilise différentes structures de voisinage dans un ordre défini préalablement. Cette méthode utilise un mécanisme de sélection du voisinage non adaptatif.

Puchinger et Raidl ont proposé une méthode de recherche à voisinage variable guidée par une fonction de relaxation RGVNS (*Relaxation Guided VNS*) [108] qui adapte le choix du voisinage à utiliser. Le meilleur voisinage ou le plus prometteur est sélectionné au moment voulu. Cette méthode a été comparée à la méthode *General VNS* sur le problème du sac à dos. Sur 7 des 9 instances testées, la méthode RGVNS est meilleure que la méthode GVNS.

L'adaptation à voisinage variable peut aussi être considérée avec une modification de la fonction d'évaluation. Braysy [18] utilise la méthode de recherche à voisinage variable réactive. Cette méthode repose sur l'utilisation d'un cycle de quatre méthodes de recherche locale utilisées de façon coopérative. La solution initiale d'une des recherches locales est la meilleure solution obtenue par la méthode effectuée précédemment. Après chaque cycle améliorant, la structure de voisinage est modifiée. Une structure de voisinage de portée plus grande est choisie et un nouveau cycle est effectué. Un cycle améliorant est un cycle pendant lequel la solution initiale de la première méthode a été améliorée. Chaque méthode de recherche locale est effectuée pour un nombre identique d'itérations. Dans le cas d'un cycle non améliorant, la fonction d'évaluation est modifiée via des poids présents dans celle-ci. De plus, après chaque méthode de recherche locale, la valeur de certains paramètres limitant l'espace de recherche est incrémentée. On a donc à la fois une adaptation par changement de structure de voisinage et d'évaluation. L'intérêt de ce mécanisme est de modifier la dimension de l'évaluation des solutions dans l'espace de recherche et donc de s'échapper d'optimums locaux en modifiant aussi le pesage des solutions. Appliquer au problème de tournées de véhicules avec fenêtre de temps, les résultats obtenus par cette méthode ont été comparés à différentes méthodes publiées dans la littérature. La méthode de recherche à voisinage variable réactive a permis d'améliorer les résultats obtenus en terme de nombre de véhicules sur les instances testées. Différentes méthodes [3, 53, 54, 77] utilisent ce mécanisme sans forcément le combiner avec une modification de la structure du voisinage.

1.2.2.2 Méthodes de recherche locale réitérée

La méthode de recherche locale réitérée ILS (*Iterated Local Search*) [30, 94, 120] repose sur la combinaison d'une méthode de recherche locale et de perturbations. La méthode de recherche locale utilisant une structure de voisinage assez simple ne permet pas d'échapper aux optimums locaux. Les perturbations basées sur l'utilisation d'un voisinage plus large permettent de modifier une grande partie de la solution et ainsi de tenter d'échapper aux optimums locaux. A partir d'une solution initiale s_0 donnée, une méthode de recherche locale est appliquée. A partir de la solution obtenue s , une perturbation est effectuée jusqu'à la solution s' . La méthode de recherche locale est alors appliquée sur cette solution pour obtenir la solution s'' . Si cette dernière est meilleure que la solution s , alors le nouveau point de départ est s'' . Sinon, un voisin différent de s est utilisé. Ce processus est répété tant qu'il existe un voisin améliorant le candidat s .

Les points importants de cette méthode sont la génération de la solution initiale, la méthode de recherche locale, la perturbation et enfin le critère d'acceptation. La performance de la méthode ILS repose sur tous ces critères. Il existe une version de base de cette méthode qui est définie ainsi :

- La solution initiale peut être générée aléatoirement ou par une méthode constructive gloutonne ; par exemple, Chiarandini et Stützle [30] utilise la méthode DSATUR [19] pour générer la solution initiale.

Algorithme 2 Méthode *ILS*

ENTRÉES: solution initiale s_0 **SORTIES:** meilleure solution s^*

```

1:  $s^* \leftarrow s_0$ 
2:  $s \leftarrow \text{RechercheLocale}(s_0)$ 
3: répéter
4:    $s' \leftarrow \text{Perturbation}(s, \text{historique})$ 
5:    $s'' \leftarrow \text{RechercheLocale}(s')$ 
6:    $s \leftarrow \text{CritèreAcceptation}(s, s'', \text{historique})$ 
7:   si  $f(s) < f(s^*)$  alors
8:      $s^* \leftarrow s$ 
9:   fin
10: jusqu'à critère d'arrêt non vérifié

```

- La méthode de recherche locale utilise un voisinage très simple pour la plupart des problèmes.
- En ce qui concerne le mouvement de perturbation, un mouvement aléatoire dans le voisinage d'un degré plus grand que celui utilisé par la méthode de recherche locale peut être effectué. La structure de voisinage utilisé par la perturbation est de portée plus importante que celle utilisée par la méthode de recherche locale.
- Le critère d'acceptation utilisé est d'accepter toutes les améliorations, pas les dégradations.

Beaucoup de variantes de cette méthode ont été développées sur différents problèmes [106]. L'élément novateur principal étant la combinaison d'une perturbation avec une recherche locale, nous ne nous intéresserons ici qu'aux différentes définitions et adaptations de cet opérateur de perturbation.

Selon Lourenço *et al.* [93,94], "Une bonne perturbation transforme une bonne solution en un excellent point de départ pour une méthode de recherche locale". La perturbation et surtout la portée de son voisinage ou en d'autres termes, le nombre de composants de la solution modifiés par la perturbation, ont une influence capitale sur le fonctionnement de la méthode. Une trop grande perturbation revient à effectuer une relance aléatoire de la méthode et ne permet pas de converger. Si au contraire elle est trop faible, elle ne permettra pas à la méthode de s'échapper des optimums locaux et de diversifier la recherche. La portée de cette perturbation est définie de façon statique ou dynamique selon plusieurs variantes de cette méthode.

- *Valeur statique* : Différentes perturbations de taille fixe ont été testées par Chiarandini et Stützle [30] et par Paquete et Stützle [106] pour résoudre le problème de coloration de graphes. Dans [106], quatre perturbations sont proposées. Ces per-

turbations consistent (i) à affecter aléatoirement une nouvelle couleur à un certain nombre de nœuds sélectionnés aléatoirement, (ii) à affecter aléatoirement une nouvelle couleur à un certain nombre de nœuds en conflit, (iii) à ajouter un certain nombre d'arêtes au graphe initial pendant quelques itérations de la Recherche Tabou utilisée et les supprimer après la perturbation et enfin (iv) à effectuer p itérations fixes à l'avance pendant lesquelles le même mécanisme est utilisé : une itération aléatoire ou une itération de la Recherche Tabou utilisant une grande durée Tabou. Cette dernière méthode de perturbation a donné les meilleurs résultats.

Réutilisée par Chiarandini et Stützle dans [30], cette dernière perturbation a été comparée à deux autres méthodes de perturbations. Un certain nombre de couleurs sont effacées puis les nœuds sont recoloriés soit (v) de façon aléatoire, différente de l'ancienne couleur, soit (vi) en utilisant la méthode DSATUR. Les résultats obtenus sur les instances structurées Leighton et Queens (DIMACS) montrent que les perturbations (v) et (vi) donnent de meilleurs résultats que la perturbation (iv). Les résultats inverses sont obtenus sur les instances aléatoires DSJC (DIMACS). Globalement, la valeur de la taille de la perturbation influençant beaucoup la qualité des résultats obtenus, de nombreux tests pour obtenir la valeur la plus adaptée au problème posé sont nécessaires.

- *Valeur dynamique* : Deux possibilités ont été identifiées par Lourenço *et al.* dans [94] sur l'adaptation dynamique de la portée de la perturbation pour cette méthode. La première consiste à exploiter l'historique de la recherche en utilisant une méthode de Recherche Tabou comme perturbation [11]. La perturbation (iv) de Paquete et Stützle [106] est aussi une Recherche Tabou. La méthode de recherche réitérée utilisant cette perturbation a été comparée à une Recherche Tabou sur le problème de coloration de graphe, plus particulièrement sur certaines instances DIMACS. Les résultats obtenus sont très compétitifs en terme de nombre moyen d'itérations mais aussi de pourcentage de réussite. La seconde possibilité est de modifier de façon déterministe la taille de la perturbation comme le fait la méthode *Basic VNS* par exemple.

Les perturbations peuvent aussi être définies de façon différente, par exemple en générant la solution s' à partir de la méthode de perturbation appliquée à la meilleure solution s^* , et non à la solution courante s [20]. Cette méthode réalise toujours une perturbation de portée comprise entre 0 et N , N étant le nombre d'éléments de la solution. En repartant de la meilleure solution, la portée de la perturbation n'est pas définie à l'avance et est différente d'une perturbation à une autre.

Un autre exemple est donné par Thierens [128] qui utilise une méthode hybride, notée PILS (*Population-Based Iterated Local Search*) combinant une méthode à population et la méthode de recherche locale réitérée. Deux solutions sont choisies aléatoirement dans la population : la première sera la solution à perturber alors que la seconde servira de référence. Si pour une variable les valeurs sont différentes, une probabilité indique si la

valeur de cette variable dans la solution à perturber est modifiée ou non. Si elle est modifiée, la valeur associée à l'autre solution sera utilisée. Si les valeurs sont identiques, aucune modification n'est effectuée. La portée du voisinage utilisé par cette perturbation dépend donc des différences entre ces deux solutions. Ce mécanisme est semblable à un croisement génétique multi-point où chaque variable peut prendre la valeur d'une autre solution.

Il est important de noter dans cette catégorie d'algorithme que plus une perturbation modifie la solution plus le temps de calcul pour converger vers une nouvelle solution risque d'augmenter. De plus, il est préférable que la méthode de recherche locale ne puisse pas effectuer les modifications inverses de celles effectuées par une perturbation car pour cette méthode le piège classique est de revenir au même minimum local si la portée de la perturbation est insuffisante. Tout ceci est donc très difficile à paramétrer en fonction du problème.

1.2.2.3 Méthodes de recherche à voisinage large

La méthode de recherche à voisinage large LNS (*Large Neighborhood Search*) introduite par Shaw [119] pour résoudre le problème de tournées de véhicules s'est montrée très performante. Une demande consiste à prendre une quantité de marchandise d'un endroit donné à un autre. De plus, une fenêtre de temps est définie pour le retrait et la livraison de ces marchandises avec un nombre limité de véhicules. La méthode permet à chaque itération de modifier plusieurs attributs de la solution. On agit donc directement sur la portée de l'algorithme. Inspirée de la méthode *Ruin and Recreate* [115], une modification se fait en deux étapes distinctes. Tout d'abord, un nombre q d'attributs de la solution est effacé. Puis, de nouvelles valeurs de ces q attributs sont réinsérées dans la solution. Ce principe a aussi inspiré une partie du fonctionnement de la méthode CN-Tabu. La performance et la robustesse de cette méthode dépendent essentiellement du choix des procédures de suppression et de réinsertion. L'algorithme général de cette méthode est présenté ci-après.

Le paramètre q détermine la portée du voisinage utilisé : $q \in [0; N]$, avec N le nombre d'attributs de la solution. Si la valeur de q est nulle alors aucun attribut n'est effacé ni réinséré et aucune solution voisine n'est explorée. A l'inverse, si q est égal à N , alors à chaque itération la solution est entièrement effacée et reconstruite. C'est comme si la méthode effectuait un *random restart*. En général, plus q est grand, plus il est facile de se déplacer dans l'espace de recherche. Mais, plus q est grand, plus la méthode de réinsertion est ralentie.

Laborie et Godard [85] ont développé une version adaptative de cette méthode appelée SA-LNS (*Self-adaptive Large Neighborhood Search*). Pour cela trois structures de voisinage ont été définies notée LN_i avec i allant de 1 à 3. A chaque cycle de la méthode LNS, un voisinage large est sélectionné. Sa probabilité d'être sélectionnée dépend d'un poids qui est mis à jour par rapport à l'efficacité du voisinage. Cette efficacité est définie de la façon

Algorithme 3 Méthode *LNS***ENTRÉES:** solution initiale s_0 , $q \in \mathbb{N}$ **SORTIES:** meilleure solution s^*

```

1:  $s \leftarrow s_0$ 
2:  $s^* \leftarrow s_0$ 
3: répéter
4:    $s' \leftarrow s$ 
5:   effacer  $q$  attributs dans la solution  $s'$ 
6:   réinsérer les  $q$  attributs dans  $s'$ 
7:   si  $f(s') < f(s^*)$  alors
8:      $s^* \leftarrow s'$ 
9:   finsi
10:  si  $accept(s', s)$  alors
11:     $s \leftarrow s'$ 
12:  finsi
13: jusqu'à critère d'arrêt non vérifié

```

suivante :

$$r = \frac{\Delta c}{\Delta t} \quad (1.22)$$

avec Δc l'amélioration en terme de coût sur la dernière utilisation (0 en cas de dégradation) et Δt le temps CPU utilisé pour effectuer le cycle. A partir de ce ratio, le poids de chaque structure de voisinage est mis à jour de la façon suivante :

$$poids_{t+1} = (1 - \alpha) \times poids_t + \alpha \times r \quad (1.23)$$

avec $\alpha \in [0; 1]$. Globalement, la méthode SA-LNS a permis d'obtenir des résultats très prometteurs dépassant ceux des algorithmes de référence sur un problème d'ordonnancement.

Ropke et Pisinger [112] ont défini une méthode adaptative de recherche à voisinage large ALNS (*Adaptive Large Neighborhood Search*) basée sur la méthode LNS, décrite dans la partie précédente, pour résoudre le problème de tournées de véhicules. Ce problème consiste à servir chaque demande. Afin de résoudre ce problème, la méthode ALNS emploie plusieurs procédures d'effacement et de réinsertion des attributs. A chaque itération, la méthode choisit la procédure de suppression et de réinsertion à utiliser en fonction des performances précédentes de chacune des procédures de suppression et de réinsertion. Les performances sont calculées à partir d'une variable poids w_j pour chaque méthode j . Le choix se fait selon le principe de la roulette biaisée; si on a k méthodes de poids w_i , $i \in \{1, \dots, k\}$, la probabilité p_j de sélectionner la méthode j est :

$$p_j = \frac{w_j}{\sum_{i=1}^k w_i} \quad (1.24)$$

Initialisé à la même valeur pour toutes les méthodes, le poids d'une méthode est incrémenté ou décrémenté d'une valeur différente selon les situations. On distingue trois

cas particuliers pour chaque méthode de suppression et de réinsertion. La méthode a conduit à améliorer la meilleure solution, à améliorer la solution courante ou à dégrader la solution courante. Le poids des deux méthodes utilisées est donc mis à jour régulièrement ce qui favorise les bonnes associations de méthode de suppression et de réinsertion. Toutes les 100 itérations, les poids de toutes les méthodes sont réajustés. Le choix de la procédure de suppression et de celle de réinsertion se fait donc en fonction des performances passées de toutes les autres méthodes. A chaque itération, le nombre q d'éléments à supprimer est choisi aléatoirement dans l'intervalle $[4; \alpha \times N]$ avec N la taille du problème traité. La meilleure valeur de α est de 0.4. La méthode ALNS est beaucoup plus robuste que la méthode LNS sur le problème de tournées de véhicules.

1.2.2.4 Méthodes de type hyperheuristique

Les hyperheuristiques [21] sont des (meta)heuristiques de haut-niveau dont le rôle est de superviser la sélection d'une (meta)heuristique parmi un ensemble de (meta)heuristiques de bas-niveau. Dans [22], Burke et Kendall définissent les hyperheuristiques comme des "heuristics to choose heuristics", c'est-à-dire comme des heuristiques permettant de choisir d'autres heuristiques. La méthode de sélection des heuristiques est une caractéristique très importante pour une hyperheuristique.

Cowling *et al.* [35] ont utilisé une hyperheuristique sur différents problèmes d'ordonnement dont le problème de planification d'examens d'une université. Ce problème consiste à allouer à chaque étudiant une session d'examen pendant laquelle il devra présenter son travail devant un jury composé de trois personnes. Une session regroupe six présentations d'étudiants. Huit heuristiques différentes ont été utilisées. Chacune utilise une unique structure de voisinage basée sur le déplacement d'un membre du jury d'une session vers une autre, sur le déplacement de la présentation d'un étudiant dans une session différente ou sur la permutation de deux membres de jury de sessions différentes. Globalement, l'hyperheuristique utilise donc un ensemble de structures de voisinage. Le choix du (ou des) membre(s) du jury ou de la présentation à déplacer est spécifique à chaque heuristique.

L'hyperheuristique proposée par Cowling *et al.* [33–35] utilise une fonction de choix pour sélectionner l'heuristique à appeler. Cette fonction de choix est basée sur la performance récente de chaque heuristique (représentée par la fonction f_1), la performance récente de paires d'heuristiques (fonction f_2) et la récence du dernier appel de l'heuristique (fonction f_3). Ainsi :

$$f(H_k) = \alpha \times f_1(H_k) + \beta \times f_2(H_j, H_k) + \gamma \times f_3(H_k) \quad (1.25)$$

avec

- H_k la k^e heuristique,
- α , β et γ des poids reflétant l'importance de chaque terme. Ces poids sont modifiés de façon adaptative dans [33],
- $f_1(H_k)$ est la performance récente de l'heuristique H_k ,

- $f_2(H_j, H_k)$ est la performance récente de la paire d'heuristiques H_j, H_k ,
- $f_3(H_k)$ est la récence du dernier appel de l'heuristique H_k .

Les facteurs f_1 et f_2 permettent d'intensifier la recherche alors que la facteur f_3 ajoute un degré de diversification en favorisant les heuristiques qui n'ont pas été choisi depuis un long moment. L'heuristique choisie est celle ayant la valeur maximale $f(H_k)$.

Dans [33], les paramètres de poids α , β et γ varient au cours de l'exécution en fonction de la volonté d'intensifier ou de diversifier la recherche de l'hyperheuristique. Suite à l'amélioration de la solution courante, les paramètres d'intensification (α et β) sont augmentés. L'augmentation est proportionnelle à l'importance de l'amélioration de la solution. Dans le cas inverse, le paramètre d'intensification ayant la plus grande valeur est diminuée. Ce paramètre est celui qui a contribué le plus au choix de l'heuristique qui a conduit à la dégradation. La pénalité est également proportionnelle à l'importance de détérioration de la solution. Après un certain nombre d'itérations sans amélioration (correspondant au nombre d'heuristiques), le paramètre de diversification (γ) augmente. Ensuite, il est fixé à une valeur suffisamment petite pour permettre le choix d'une heuristique pouvant améliorer efficacement la solution. L'hyperheuristique a été comparée aux différentes heuristiques utilisées seules. Sur les instances testées, l'hyperheuristique a été plus performante que les heuristiques utilisées séparément.

Une autre hyperheuristique est la méthode COSEARCH [8, 133, 134] qui est une méthode co-évolutionniste. Elle est basée sur l'utilisation de trois agents complémentaires, chacun d'eux ayant un rôle bien défini et combinant des structures de voisinage de différentes portées notamment via un algorithme génétique. Ces trois agents évoluent en parallèle et coopèrent via une mémoire centrale et partagée (*Adaptive Memory*, AM). Pendant la recherche, la mémoire centrale regroupe l'ensemble des connaissances acquises par les trois agents. Chaque agent est en fait une métaheuristique ayant un rôle particulier.

- **L'agent SA ou Searching Agent** : cet agent utilise une fonction d'amélioration basée sur le voisinage. Au commencement du programme, l'agent SA doit améliorer la solution initiale. Pour cela, SA utilise la Recherche Tabou.
- **L'agent DA ou Diversifying Agent** : à partir des régions déjà visitées, l'agent DA fournit de nouvelles solutions les plus différentes possibles des solutions visitées appartenant aux régions inexplorées de l'espace de recherche. L'opérateur DA est un algorithme génétique utilisant deux méthodes de cross-over spécifiques FX et SX.
- **L'agent IA ou Intensifying Agent** : cet agent génère de nouvelles solutions à partir des régions prometteuses déjà visitées. Pour cela, une méthode *Short Random Walk* est utilisée. Cette méthode génère, à partir d'une bonne solution, une nouvelle solution qui sera utilisée comme point de départ par l'agent SA.

Cette méthode combine à la fois une stratégie de diversification et une stratégie d'in-

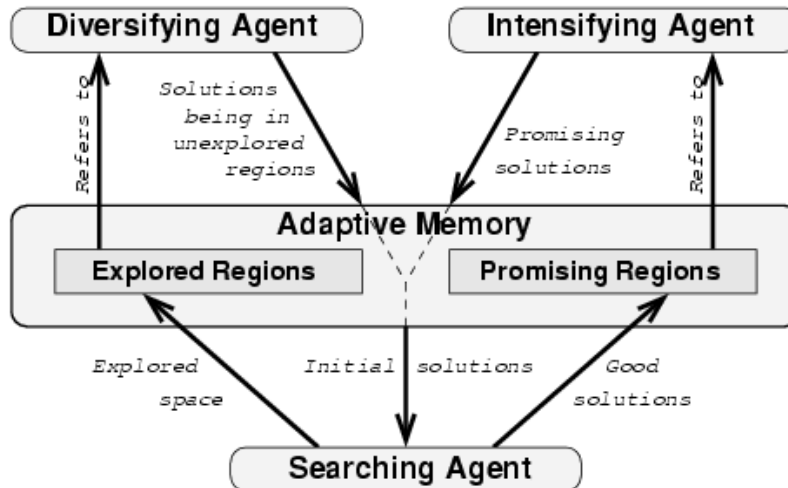


FIG. 1.5 – La méthode COSEARCH (extraite de [8])

tensification. Ainsi, de nouvelles régions sont visitées et les régions déjà visitées font l'objet d'une étude plus précise. Cette méthode paraît très intéressante du point de vue de la combinaison de plusieurs méthodes différentes en parallèle. Toutefois, les résultats en terme de temps de calcul ne sont pas les meilleurs obtenus jusqu'à présent. Mais en terme de qualité cette méthode donne de très bons résultats. Elle a notamment été utilisée sur des problèmes d'allocation de fréquences en réseaux cellulaires [133, 134]. Les solutions explorées (*Explored Regions*) ainsi que les solutions prometteuses (*Promising Regions*) sont enregistrées via la mémoire partagée AM. Une matrice M de taille $N * NF$, avec N le nombre de cellules et NF le nombre de fréquences, représente les solutions déjà visitées. L'élément $M[f][c]$ correspond au nombre d'allocations de la fréquence f pour la cellule c depuis le début de la recherche. Ainsi toutes les affectations réalisées par toutes les métaheuristiques sont enregistrées et utilisées pour intensifier ou diversifier.

1.3 Discussion

Dans ce chapitre, nous avons dans un premier temps défini des notions de base sur les problèmes d'optimisation combinatoire et les différentes méthodes utilisées pour les résoudre. Dans cette thèse, nous nous intéresserons plus particulièrement aux méthodes de recherche locale. Ces méthodes sont basées sur les notions de voisinage qui influent énormément sur leur performance. Une structure de voisinage est définie comme l'ensemble des solutions atteignables à partir de règles de transformation des solutions. On peut ensuite restreindre ce voisinage en définissant des sous-ensembles de solutions accessibles dans le voisinage comme le fait la liste Tabou. Dans la littérature ces deux notions sont souvent confondues mais dans la partie état de l'art des méthodes adaptatives, nous avons utilisé cette distinction car cela conduit à des procédés très différents dans les algorithmes de

recherche.

Ensuite, nous avons défini les notions de mémoire et d'adaptation. Toutes les méthodes utilisant une mémoire ne sont pas forcément adaptatives. Dans cette partie, nous avons distingué deux types de méthodes : les méthodes dites *évolutives* et les méthodes dites *adaptatives*. Une méthode adaptative utilise un mécanisme d'adaptation basé sur des données extraites d'un historique de recherche pour influencer ses futurs choix. Au contraire, une méthode évolutive utilise seulement les informations intrinsèquement contenues dans la solution courante. L'objectif principal de l'adaptation des méthodes est double. Tout d'abord, certaines méthodes obtiennent de très bons résultats avec des valeurs de paramètres différentes pour chaque instance où elles sont appliquées. L'adaptation peut permettre d'éviter cette phase de paramétrage par instance qui est souvent longue et fastidieuse. Ensuite, l'adaptation peut permettre de travailler sur des espaces de recherche présentant un grand nombre d'optimums locaux tout en utilisant une recherche locale. A travers la littérature nous avons vu que l'adaptation de la recherche locale a suscité beaucoup d'intérêt depuis quelques années. Nous avons regroupé les différents mécanismes d'adaptation en deux familles distinctes en fonction de l'utilisation d'une unique ou de plusieurs structures de voisinage. En général, l'adaptation du choix de la structure ou de la restriction du voisinage a permis d'améliorer la performance des méthodes. Cependant, les méthodes trouvées dans la littérature combinent rarement ces deux possibilités d'adaptation car l'ensemble devient difficile à gérer.

L'adaptation dans les méthodes utilisant une seule structure de voisinage se fait par des mécanismes d'extension et de restriction du voisinage. On ne change pas la nature de la transformation qui produit le voisinage, on effectue plutôt une sélection sur le voisinage. Les méthodes utilisant une liste Tabou peuvent être classées dans cette catégorie. En effet, l'utilisation d'une liste Tabou ne modifie pas la structure de voisinage mais elle permet de le restreindre. Dans la littérature, beaucoup de méthodes utilisent une liste Tabou avec des durées Tabou définies différemment. Nous avons recensé quatre catégories de paramètres pour la durée Tabou : durée dépendante du temps de calcul, durée choisie dans un intervalle, durée réactive et durée adaptative. Les durées Tabou réactives utilisent les caractéristiques de la solution courante alors que les durées adaptatives sont définies en fonction d'un historique de la recherche. L'utilisation de cet historique peut être double ; il peut permettre de définir le moment où la variable devient Tabou et la valeur de la durée Tabou.

Les méthodes utilisant plusieurs structures de voisinage sont très récentes. Ces méthodes utilisent en général des structures de voisinage de portées différentes. L'avantage de l'utilisation de plusieurs structures repose sur le fait qu'une solution est considérée comme un optimum local pour une structure de voisinage donnée. L'utilisation de plusieurs structures permet donc de ne pas être définitivement bloqué dans un optimum local. La plupart de ces méthodes utilisent les différentes structures de voisinage dans un ordre prédéfini. Certains travaux portent cependant sur l'adaptation de ce choix entre les structures de voisinage. Le second élément d'adaptation utilisé par les méthodes combinant

plusieurs structures de voisinage est basé sur des mécanismes d'extension et de restriction du voisinage définis par la structure de voisinage sélectionnée. La plupart des méthodes utilisent la notion de poids afin d'effectuer le choix adaptatif. Par exemple, chaque heuristique est sélectionnée en fonction d'un poids correspondant à sa performance passée dans le cas des hyperheuristiques. L'apprentissage par renforcement est très souvent utilisé afin d'adapter le choix de la structure de voisinage ou le choix de l'opérateur de sélection.

En conclusion l'adaptation peut répondre à certains problèmes comme la suppression de la phase manuelle de calibrage nécessaire lors de l'utilisation de paramètres fixes et la possibilité de modifier le voisinage en changeant ou non de portée pour échapper aux minimums locaux. Néanmoins, comme nous l'avons vu dans l'état de l'art les méthodes utilisant des mécanismes d'adaptation dans la littérature emploient en général un mécanisme d'adaptation qui est principalement basé sur l'état courant de la recherche. L'historique de la trajectoire de la recherche reste peu utilisé pour dynamiser l'adaptation et la rendre continue au fur et à mesure de la progression. Le travail le plus probant réalisé dans ce domaine est probablement celui de l'hyperheuristique COSEARCH avec la mémoire centrale partagée par plusieurs métaheuristiques. Mais cette méthode nécessite un contexte de ressources de calcul très important qui ne convient pas dans le cadre de contraintes fortes sur les ressources (machines, temps). La difficulté de la mise au point de méthodes adaptatives continues est liée à la nécessaire compréhension de ce que fait l'algorithme. On a besoin de comprendre l'influence de chaque élément adaptatif sur la méthode or justement très peu d'études de comportement interne sont réalisées. On se contente souvent de donner les entrées et les résultats obtenus sans faire d'analyses numériques sur le chemin parcouru par la méthode pour passer de l'un à l'autre ni sur la façon dont agit l'adaptation sur le comportement interne de la méthode.

Nous proposons dans cette thèse de mettre au point une méthode de recherche locale adaptative utilisant différents mécanismes d'adaptation définis en réponse aux observations effectuées au cours d'études comportementales de la méthode. On souhaite que l'adaptation se fasse pour éviter le paramétrage spécifique à chaque instance et pour sortir des minimums locaux en se reposant sur l'adaptation continue des mécanismes d'extension et de restriction de voisinage. Le mécanisme d'extension utilisé sera une *détection de boucles* pendant la recherche. La notion de boucle est relative aux répétitions successives d'un même choix de variable et est symptomatique d'un optimum local pour les méthodes déterministes. En cas de boucle détectée, une extension du voisinage sera utilisée par le biais d'un opérateur de diversification. Le mécanisme de restriction sera représenté par une liste Tabou. Les durées Tabou que nous avons classées comme adaptatives définissent soit la valeur de durée Tabou soit le moment où la variable devient tabou de façon adaptative. Nous proposons ici de définir la valeur de la durée Tabou (incrément ou décrétement) et le moment où la variable devient tabou de façon totalement adaptative. Les informations extraites de l'historique de la recherche seront employées pour déterminer le moment où une variable devient tabou ainsi que sa période de prohibition. Nous associerons complètement les deux mécanismes car a priori nous considérons que seules les variables à l'origine des blocages de l'algorithme devraient justifier le statut Tabou. En effet les boucles sont

révélatrices de sur-sélections de certaines variables qui traduisent l'incapacité de l'algorithme à se déplacer, au moins momentanément, dans l'espace de recherche. Ce point sera particulièrement étudié.

La méthode proposée utilisera donc deux opérateurs de sélection du voisinage en fonction de la détection ou non d'une boucle alternant ainsi itération *intensifiante* avec itération *diversifiante*. On pourra ainsi contrôler le rapprochement ou l'éloignement de l'algorithme de certaines zones de recherche. L'itération intensifiante reposera sur la sélection d'une variable à partir d'un ensemble restreint de variables critiques; la notion de variable critique est à définir pour chaque problème. Il peut s'agir par exemple de variables disposant de nombreux conflits si le problème contient des contraintes. L'itération diversifiante quant à elle permettra la sélection d'une autre catégorie de variables regroupant un ensemble plus large de variables du problème. Cette seconde opération sera appliquée uniquement lorsqu'un blocage sera détectée. La différence entre ces deux opérateurs de sélection reposera sur le choix de la variable à modifier et non sur le choix de la nouvelle valeur à lui allouer. Dans les deux cas, s'il est possible par exemple de connaître la meilleure valeur différente de l'ancienne, alors cette valeur pourrait être allouée mais cela dépend complètement du problème et de la difficulté d'évaluer le voisinage. Nous traiterons les deux cas dans les chapitres suivants. Les mécanismes généraux de la méthode RLA sont présentés dans l'algorithme suivant :

Algorithme 4 Méthode *Recherche Locale Adaptative*

ENTRÉES: méthodes de détection de blocage et de calcul de la durée tabou

SORTIES: meilleure solution s^*

```

1:  $s \leftarrow \text{GenererSolutionInitiale}()$ 
2:  $s^* \leftarrow s$  // la solution initiale est la meilleure solution
3:  $(s, x) \leftarrow \text{Intensifiant}()$  // retourne  $x$  la variable choisie et  $s$  la nouvelle solution
4: tantque condition d'arrêt non vérifiée faire
5:   si  $\text{blocage}(x)$  alors
6:      $\text{MettreTabou}(x, dt)$  // ajouter  $x$  à la liste Tabou pour une durée  $dt$ 
7:      $(s, x) \leftarrow \text{Diversifiant}()$ 
8:   sinon
9:      $(s, x) \leftarrow \text{Intensifiant}()$ 
10:  fin
11:  si  $s$  meilleure que  $s^*$  alors
12:     $s^* \leftarrow s$ 
13:  fin
14: fin tantque

```

Il reste à définir tous les paramètres de l'adaptation au sein de ces mécanismes généraux. Ce travail a été réalisé sur deux problèmes différents : le problème de k -coloration et le problème d'affectation de listes de fréquences présentés dans les deux chapitres suivants. Le premier problème a été très utilisé comme problème académique dans la littérature alors que le second correspond à une application réelle.

Chapitre 2

Recherche Locale Adaptative et k -coloration de graphe

Cette partie présente les études en recherche locale adaptative sur le problème de k -coloration de graphe. Tout d'abord, les principes du problème de k -coloration seront abordés. Puis, un état de l'art spécifique à la k -coloration de graphe sera présenté. Enfin, nous détaillerons nos études en recherche locale adaptative sur différents benchmarks avec une comparaison aux résultats obtenus par d'autres méthodes sur les mêmes problèmes.

Ce travail a fait l'objet de plusieurs publications notamment dans le journal *IEEE International Journal on Artificial Intelligence Tools* et dans les conférences *20th European Simulation and Modeling Conference (ESM 2006)* et *6th Metaheuristics International Conference (MIC 2005)*.

Sommaire

2.1	Contexte de travail	54
2.1.1	Quelques notions de théorie des graphes	54
2.1.2	Quelques notions de coloration	55
2.1.3	Description des problèmes de coloration	55
2.1.4	Benchmarks utilisés	56
2.1.5	Recherche locale pour la coloration de graphe	60
2.2	Étude de la méthode RLA	61
2.2.1	Recherches locales de base	61
2.2.2	Détection de boucle et liste Tabou	65
2.2.3	Adaptation de la durée Tabou	72
2.2.4	Adaptation de la détection de boucle	82
2.3	Comparaison des résultats	96
2.4	Conclusion	98

2.1 Contexte de travail

La première application du problème de coloration de graphe remonte au milieu du XIX^e siècle. A cette époque, il s'agissait de colorier les pays sur une carte : deux pays voisins ne devaient pas être coloriés de la même couleur. Ce problème est un cas particulier des problèmes de coloration de graphe. En effet, les pays sont représentés par les nœuds, les arêtes reliant les pays voisins. Ce graphe a cependant une particularité, il est planaire. La représentation de ce graphe ne présente pas de superpositions d'arêtes.

Depuis, le problème de coloration de graphe a été appliqué à beaucoup d'autres applications réelles comme les problèmes d'allocation de ressources par exemple. Afin de s'adapter aux spécificités de ces applications, différentes variantes du problème de coloration de graphe ont vu le jour. Nous ne présenterons ici que les deux variantes que nous avons étudié : le problème de k -coloration ainsi que le problème de coloration de graphe. Avant d'aborder ces deux types de problèmes, quelques définitions mathématiques sont nécessaires. Dans un premier temps, nous définirons des notions liées à la théorie des graphes, suivi des définitions propres aux problèmes de coloration et de la formalisation mathématique des deux problèmes précédemment cités. Pour finir, les différents jeux de tests utilisés dans ce chapitre seront présentés.

2.1.1 Quelques notions de théorie des graphes

Définition 2.1. *Graphe non-orienté*

Un graphe non-orienté est un couple, $G = (V, E)$, composé d'un ensemble de nœuds ou sommets V et d'un ensemble d'arêtes $E \subseteq V \times V$.

Dans les problèmes que nous avons traités, les graphes G à colorier seront tous des graphes non-orientés. Nous noterons $N = |V|$ le nombre de nœuds.

Définition 2.2. *Le voisinage d'un nœud*

Le voisinage d'un nœud $x \in V$ du graphe G , noté $\mathcal{V}(x)$, est l'ensemble des nœuds reliés par une arête au nœud x . Les nœuds appartenant au voisinage de x sont dits *voisins* de x .

Définition 2.3. *Le degré d'un nœud*

Le degré d_x d'un nœud x est le nombre de voisins, soit le nombre de nœuds appartenant au voisinage du nœud x : $|\mathcal{V}(x)|$. Un nœud de degré 0 est dit *isolé*. Le nœud de degré *minimal* (respectivement *maximal*) est noté d_{min} (respectivement d_{max}).

Définition 2.4. *Un graphe complet*

Un graphe complet est un graphe dont les nœuds sont tous reliés deux à deux par une arête. Un graphe complet à N sommets contient $\frac{N \times (N-1)}{2}$ arêtes.

Définition 2.5. *Une clique*

Une clique est un sous-graphe complet, c'est-à-dire un sous-graphe dont les sommets sont tous connectés deux à deux. Une p -clique est une clique de p sommets.

Définition 2.6. *La densité du graphe*

La densité du graphe $G = (V, E)$ représente la probabilité de trouver une arête entre chaque paire de nœuds (x_i, x_j) , avec x_i et x_j deux nœuds distincts. La densité Δ est calculée de la façon suivante :

$$\Delta = \frac{|E|}{\frac{N \times (N-1)}{2}} \quad (2.1)$$

2.1.2 Quelques notions de coloration

Définition 2.7. *Une coloration*

Une coloration d'un graphe $G = (V, E)$ est l'affectation d'une couleur $c(x)$ à chaque sommet $x \in V$. En optimisation, une coloration sera appelée une solution au problème de coloration.

Définition 2.8. *Un conflit*

On appelle conflit dans une coloration le fait que deux sommets reliés par une arête soient de même couleur.

Définition 2.9. *Une coloration valide*

Une coloration valide du graphe $G = (V, E)$ est une coloration ne générant aucun conflit. L'ensemble des nœuds reliés par une arête sont de couleurs différentes.

$$\forall (x, y) \in E, c(x) \neq c(y) \quad (2.2)$$

Un graphe pour lequel il existe une coloration valide utilisant k couleurs est dit *k-coloriable*.

Définition 2.10. *Le nombre chromatique*

Le nombre chromatique χ du graphe $G = (V, E)$ est le plus petit nombre de couleurs pour lequel il existe une coloration valide.

2.1.3 Description des problèmes de coloration

Il existe un très grand nombre de problèmes de coloration. Cependant, cette partie en présente uniquement deux : le *problème de coloration de graphe* et le *problème de k-coloration*. Le lecteur trouvera une description complète des problèmes de coloration dans [78]. Le problème que nous traiterons dans ce chapitre est celui de *k-coloration*. Dans la littérature, la résolution des problèmes de coloration de graphe consistent souvent à relancer une méthode de *k-coloration* avec des valeurs de k décroissantes.

Définition 2.11. *Problème de coloration de graphe*

Étant donné un graphe $G = (V, E)$ non-orienté, le problème de coloration de graphe consiste à rechercher une coloration valide utilisant le plus petit nombre de couleurs k .

Le nombre de couleurs d'une solution s peut par exemple être évalué par la fonction $\mathcal{F}(s)$ suivante :

$$\begin{aligned} \mathcal{F} : \{colorations\ valides\} &\longrightarrow \mathbb{N} \\ s &\longmapsto \mathcal{F}(s) = |\{l \in \mathbb{N} / \exists x \in V, c(x) = l\}| \end{aligned} \quad (2.3)$$

Définition 2.12. *Problème de k -coloration*

Étant donné un graphe $G = (V, E)$ non-orienté, le problème de k -coloration de graphe consiste à rechercher une coloration valide utilisant au plus k couleurs.

Résoudre un problème de k -coloration consiste à trouver une coloration utilisant k couleurs et minimisant par exemple la fonction d'évaluation $F(s)$ suivante que nous avons retenu pour la suite de notre étude :

$$\begin{aligned} F : \{\text{colorations}\} &\longrightarrow \mathbb{N} \\ s &\longmapsto F(s) = |\{(x, y) \in E / c(x) = c(y)\}| \end{aligned} \quad (2.4)$$

Une solution s est valide si $F(s) = 0$.

NB : Les problèmes de T-coloration qui introduisent de nouvelles contraintes et de T-coloration avec ensembles qui se rapportent à l'allocation d'un ensemble de valeurs à des variables sont définis au chapitre 3.

2.1.4 Benchmarks utilisés

Au cours de cette étude, nous avons utilisé différents types de graphes pour traiter des problèmes de k -coloration. Cette partie présente les principales caractéristiques des instances CNET (*Centre National d'Études des Télécommunications*) et des instances DIMACS (*Discrete Mathematics and Theoretical Computer Science*¹) utilisées.

2.1.4.1 Instances CNET

Dans un premier temps, nous avons choisi d'utiliser les benchmarks CNET. Ces jeux de tests ont été créés par le CNET en 1994² dans le but de tester des algorithmes d'affectation de fréquences pour les réseaux de radiocommunications avec les mobiles. Si l'on considère les fréquences à allouer comme des couleurs et les émetteurs comme les nœuds, ce problème est un cas de coloration de graphes. Certaines instances CNET nécessitent l'affectation de plusieurs fréquences sur une même station. Dans ce travail, nous nous sommes intéressés uniquement aux instances prévoyant l'allocation d'une unique fréquence par station. Les instances CNET sont identifiées de la façon suivante : $\chi.N.\Delta$ avec χ le nombre chromatique, $|V| = N$ le nombre de nœuds et Δ la densité du graphe (exprimée en pourcentage). Le tableau 2.1 présente l'ensemble des instances ainsi que leurs caractéristiques.

Quelques méthodes ont été réalisées sur ces jeux de tests comme par exemple une méthode évolutionnaire [44], un algorithme génétique [71] ou encore une Recherche Tabou [72]. Cependant, peu de publications utilisent ces jeux de données. Nous avons utilisé ces instances comme point de départ de nos études car le nombre chromatique est connu et elles sont toutes réalisables. Parmi les trois méthodes citées, deux résolvent tous les problèmes [44, 72], cependant la méthode [72] semble être la plus efficace en nombre d'itérations pour trouver la solution optimale.

¹Pour plus d'informations, voir le site <http://dimacs.rutgers.edu/Challenges/>

²Disponible sur <http://set.utbm.fr/index.php?pge=228>

TAB. 2.1 – Caractéristiques des instances CNET

problèmes	χ	$ V $	Δ
4.75.10	4	75	0.1
4.75.20	4	75	0.2
4.75.30	4	75	0.3
8.75.10	8	75	0.1
8.75.20	8	75	0.2
8.75.30	8	75	0.3
8.150.10	8	150	0.1
8.150.20	8	150	0.2
8.150.30	8	150	0.3

problèmes	χ	$ V $	Δ
15.150.10	15	150	0.1
15.150.20	15	150	0.2
15.150.30	15	150	0.3
15.300.10	15	300	0.1
15.300.20	15	300	0.2
15.300.30	15	300	0.3

2.1.4.2 Instances DIMACS

Notre travail s'est surtout concentré sur les instances DIMACS³ (instances du 2nd challenge DIMACS, réalisé en 1992-1993). Ces instances ont la particularité d'être très utilisées dans la littérature et surtout de regrouper un ensemble très important de graphes différents par leur structure, leur taille et leur difficulté. Parmi ces instances, nous avons choisi de travailler sur les graphes aléatoires, les graphes Leighton et les graphes plats.

- **Graphes aléatoires** : Créés par David Johnson en 1991 [80], les graphes aléatoires sont nommés par *DSJCN.p* où N est le nombre de nœuds et p représente la probabilité que deux nœuds soient liés par une arête. Par exemple, l'instance DSJC1000.9 correspond à un graphe composé de 1000 nœuds et une probabilité d'existence d'une arête entre deux nœuds égale à 90%. Le nombre chromatique de ces graphes n'est pas connu.
- **Graphes Leighton** : Générées par Leighton en 1979 [87], les instances Leighton sont des graphes aléatoires générés en fixant le nombre chromatique χ et en introduisant des cliques de taille 2 à χ dans le graphe. De plus, tous les graphes Leighton utilisent le même nombre de nœuds. Les graphes sont nommés de la façon suivante : *le450_χl*, avec 450 le nombre de nœuds, χ le nombre chromatique et une lettre $l \in \{a, b, c, d\}$ utilisée pour distinguer les différents graphes de même nombre chromatique.
- **Graphes plats** : Les graphes plats sont des graphes quasi-aléatoires générés par Culberson⁴. Ils sont notés de la façon suivante : *flat N_χ_0* avec N le nombre de

³Disponible sur <http://mat.gsia.cmu.edu/COLOR/instances.html>

⁴Le générateur des instances est disponible sur <http://web.cs.ualberta.ca/~joe/Coloring/>

nœuds et χ le nombre chromatique.

Le tableau 2.2 présente les principales caractéristiques des instances DIMACS utilisées. Pour chacune d'entre elles, nous donnons le nombre chromatique χ , le meilleur résultat connu k^* , le nombre de nœuds $|V|$ ainsi que la densité du graphe Δ .

TAB. 2.2 – Caractéristiques des instances DIMACS

problèmes	(χ, k^*)	$ V $	Δ
le450_5a	(5,5)	450	0.0566
le450_5b	(5,5)	450	0.0568
le450_5d	(5,5)	450	0.0966
le450_15a	(15,15)	450	0.0809
le450_15b	(15,15)	450	0.0809
le450_15c	(15,15)	450	0.1651
le450_15d	(15,15)	450	0.1658
le450_25c	(25,25)	450	0.1717
le450_25d	(25,25)	450	0.1725
flat300_20_0	(20,20)	300	0.4766
flat300_26_0	(26,26)	300	0.4823
flat300_28_0	(28,28)	300	0.4837
flat1000_50_0	(50,50)	1 000	0.4905
flat1000_60_0	(60,60)	1 000	0.4922
flat1000_76_0	(76,83)	1 000	0.4939

problèmes	(χ, k^*)	$ V $	Δ
DSJC125.1	(-,5)	125	0.095
DSJC125.5	(-,17)	125	0.5021
DSJC125.9	(-,30)	125	0.8982
DSJC250.1	(-,8)	250	0.1034
DSJC250.5	(-,22)	250	0.5034
DSJC250.9	(-,72)	250	0.8963
DSJC500.1	(-,12)	500	0.0999
DSJC500.5	(-,48)	500	0.502
DSJC500.9	(-,126)	500	0.9013
DSJC1000.1	(-,20)	1 000	0.0994
DSJC1000.5	(-,83)	1 000	0.5002
DSJC1000.9	(-,224)	1 000	0.8998

Les instances DIMACS sont très utilisées dans la littérature. Les instances que nous avons traitées sont classées parmi les instances *difficiles* [27].

Parmi les différentes méthodes utilisées pour résoudre ces instances, nous pouvons citer une méthode hybride combinant une méthode à population et une Recherche Tabou. Cette méthode, appelée AMACOL (*Adaptive Memory Algorithm for k-COLORing*), est proposée par Galinier *et al.* [61]. Elle a permis pour la plupart des instances d'obtenir le plus petit nombre de couleurs connu k^* , elle est classée parmi les meilleures méthodes de coloration. Cette méthode est basée sur l'utilisation d'une mémoire centrale de solutions composées de classes de couleur stables maximales. Dans un premier temps, un ensemble de solutions aléatoires est généré puis la méthode *Tabucol* [74] est appliquée pendant un nombre fixé d'itérations. Sur ces solutions, on applique un opérateur permettant de générer des classes de couleur stables maximales. Une classe de couleur rassemble tous les nœuds de même couleur au sein d'une solution. On appelle classe de couleur stable un ensemble de nœuds non-reliés deux à deux de même couleur. L'opérateur a donc pour

objectif de créer des solutions utilisant des classes de couleur stables de taille maximale, c'est à dire regroupant le plus grand nombre de nœuds.

Des méthodes de voisinage ont aussi été utilisées. Parmi celles-ci, nous citerons une Recherche Tabou, une méthode de recherche locale réitérée (ou *Iterated Local Search*, ILS) et finalement une méthode de recherche locale guidée (*Guided Local Search*, GLS).

La méthode de Recherche Tabou générique (*Generic Tabu Search*, GTS) a été publiée par Dorne et Hao [46]. Cette méthode utilise une durée Tabou réactive (cf. section 1.2.1.4) ainsi qu'une solution initiale générée par l'algorithme DSATUR [19]. La durée Tabou dépend du nombre de nœuds en conflit de la solution courante.

La recherche locale réitérée (*Iterated Local Search* ou ILS), proposée dans [29], est une méthode combinant une méthode de recherche locale (ici, une Recherche Tabou) et une perturbation. Une Recherche Tabou est exécutée jusqu'à ce que la meilleure solution obtenue soit inchangée durant un nombre fixé d'itérations. Une perturbation est alors appliquée à cette solution et la Recherche Tabou est à nouveau exécutée sur la solution perturbée.

Enfin, la recherche locale guidée (*Guided Local Search*, GLS) est basée sur la modification de la fonction de coût pour échapper aux optimums locaux. Une application de cette méthode est proposée par Chiarandini [27]. Cette méthode modifie la fonction de coût au cours de l'exécution en utilisant des poids w_i qui sont associés à chaque composant i de la solution s :

$$f'(s) = f(s) + \lambda \times \sum_{i=1}^N w_i \times I_i \quad (2.5)$$

avec $f(s)$ l'évaluation courante de la solution, $f'(s)$ la nouvelle évaluation et I_i un entier de valeur 1 si le composant i est dans la solution, 0 sinon.

Les poids w_i sont initialisés à 0 et mis à jour après chaque itération améliorante. La mise à jour s'effectue uniquement sur les poids des composants présents dans la solution localement optimale. Pour cela, la contribution $f_i(s)$ d'un composant i de la solution s à la qualité globale de la solution permet d'estimer l'utilité $util_i$ d'incrémenter le poids w_i associé au composant i . Cette utilité est calculée de la façon suivante :

$$util_i = \frac{f_i(s)}{1 + w_i} \quad (2.6)$$

Seuls les composants d'utilité maximale sont mis à jour par $w_i = w_i + 1$.

Nous utiliserons l'ensemble des méthodes analysées dans ce paragraphe pour évaluer nos résultats sur les instances DIMACS.

2.1.5 Recherche locale pour la coloration de graphe

De nombreuses méthodes ont été utilisées pour résoudre les problèmes de coloration de graphes : des méthodes exactes [10,95], des méthodes multi-solutions [2,58], des méthodes mono-solution [28,60,90] et des méthodes hybrides [45,59]. Nous nous intéresserons uniquement aux méthodes de recherche locale dans cette partie.

Une méthode de recherche locale peut être définie en fonction des trois éléments suivants : la composition de l'espace de recherche, la fonction d'évaluation et la structure de voisinage utilisée. Ces trois éléments définissent la stratégie de la recherche locale.

Dans la littérature, beaucoup de travaux ont été récemment publiés sur l'utilisation de structures de voisinage et sur la combinaison de celles-ci dans des méthodes de recherche locale (cf. section 1.2.2). Les méthodes utilisant plusieurs structures de voisinage sont en particulier la méthode de recherche locale réitérée [30,106], la méthode à voisinage variable [7,67,69,97,108], la méthode combinant plusieurs structures de voisinage et plusieurs opérateurs [137] ainsi que la méthode à voisinage large [26,65,112]. Toutes ces méthodes sont présentées dans le chapitre précédent où nous nous sommes intéressés au choix des structures de voisinage. Dans cette partie, nous nous intéresserons plutôt à la définition de ces structures.

Avanthay *et al.* [7] ont étudié l'influence de différents types de voisinage pour la coloration de graphe dans une méthode à voisinage variable. Dans un premier temps, chaque structure de voisinage a été utilisée séparément. Puis, des combinaisons de ces structures de voisinage ont été étudiées. Deux grands groupes de structures de voisinage peuvent être distingués, ils sont expliqués ci-dessous.

- **Le voisinage basé sur les nœuds** : Ce type de voisinage modifie la couleur de nœuds en conflit. Cette catégorie regroupe différentes structures de voisinage, nous ne présenterons que les plus utilisées :
 - **Nœud en conflit** : Le voisinage d'une solution s est l'ensemble des solutions qui modifient la couleur d'un nœud en conflit de la solution s . Cette structure de voisinage est la plus utilisée dans la littérature [10,44,46,72,74,106].
 - **Permutation** : L'ensemble des solutions voisines de s est obtenu en échangeant les couleurs allouées à deux nœuds distincts. Il existe une variante de cette structure de voisinage qui consiste à permuter des nœuds adjacents. Cette structure est assez peu utilisée dans la littérature [26,38,121,122].
 - **Permutation cyclique** : Cette structure de voisinage modifie un ensemble de nœuds de façon cyclique. Un premier nœud est choisi et la couleur allouée à ce nœud est modifiée. Un nœud, colorié par cette nouvelle couleur, est choisi pour être colorié par une autre couleur, et ainsi de suite. Cette structure est présentée par Chiarandini *et al.* [27,28].

- **Le voisinage basé sur les classes de couleur :** Ces structures de voisinage modifient une solution en sélectionnant tout d'abord une classe de couleur en conflit et en modifiant un ou plusieurs nœuds de cette classe de couleur. Avanthay *et al.* [7] proposent quatre structures de voisinage basées sur les classes de couleur. Par exemple, la première structure de voisinage fonctionne de la façon suivante : soient V^* la classe de couleur contenant le plus de nœuds en conflit et p le nombre de nœuds contenus dans cette classe de couleur, tous les nœuds de la classe de couleur V^* sont déplacés dans la meilleure classe de couleur possible, soit dans celle générant le moins de conflits. Puis un nombre p de nœuds en conflit, si possible les nouveaux nœuds en conflit, sont ensuite déplacés dans la classe de couleur V^* .

Avanthay *et al.* [7] et Chiarandini [27] présentent en détail d'autres structures de voisinage moins courantes utilisées pour la coloration de graphes. Chiarandini effectue une comparaison de différentes structures de voisinage et de leurs combinaisons sur des graphes DIMACS. Sur les graphes étudiés, les résultats obtenus avec des structures de voisinage larges semblent prometteurs mais n'atteignent pas les résultats des méthodes présentées en 2.1.4.

La méthode de recherche locale adaptative étudiée au cours de cette thèse et présentée dans la partie suivante travaille avec des structures de voisinages basées sur les nœuds.

2.2 Étude de la méthode RLA

Dans cette section nous nous proposons de décrire les principales composantes fonctionnelles de l'algorithme que nous avons étudié. La méthode RLA (*Recherche Locale Adaptative*) est une procédure itérative. A chaque itération la couleur d'un nœud est modifiée. Pour cela l'algorithme procède à deux choix : le choix du nœud à modifier suivi du choix de la nouvelle couleur. Le choix du nœud dépend d'une procédure de détection de boucle et d'une liste Tabou. Ces deux mécanismes étendent et restreignent le voisinage de la solution. Il faut noter qu'aucun contrôle de dégradation n'est utilisé dans l'ensemble de l'étude présentée.

Cette partie se compose de quatre sous-parties. Dans un premier temps, différentes recherches locales de base sont présentées suivies d'une partie sur les mécanismes généraux de détection de boucle et de liste Tabou. Puis, les adaptations automatiques de ces deux mécanismes sont étudiées dans deux parties distinctes.

2.2.1 Recherches locales de base

La recherche locale est une procédure itérative. Pour la coloration de graphe, à chaque itération la couleur d'un nœud est modifiée. Chaque itération de la recherche locale que nous utilisons est composée de deux choix successifs : le choix du nœud suivi du choix

de la couleur. La première étape nous permet de sélectionner une recherche locale simple comme point de départ des études sur l'adaptation.

2.2.1.1 Les méthodes retenues

Nous avons opté pour l'utilisation de méthodes classiques, simples à mettre en œuvre et à évaluer, utilisant des voisinages basés sur les nœuds en conflit tels que ceux présentés dans l'état de l'art.

Le nœud peut être choisi de trois manières différentes : (A) aléatoirement parmi l'ensemble des nœuds ; (C) aléatoirement parmi les nœuds en conflit, indépendamment du nombre de conflits ; ou (P) aléatoirement parmi les nœuds les plus en conflit. Chaque mécanisme de sélection définit un opérateur spécifique impliquant différents degrés d'exploration : du plus proche d'un mécanisme déterministe car se limitant souvent à un choix dans un ensemble composé d'un élément (méthode (P)) au plus éloigné car le choix est aléatoire (méthode (A)).

Lorsque le nœud à modifier est sélectionné, une nouvelle couleur différente de l'ancienne est choisie pour lui être allouée. Deux mécanismes de choix de la couleur ont été étudiés : (A) aléatoire parmi toutes les couleurs ou (M) aléatoire parmi les couleurs les moins utilisées par les nœuds voisins du nœud sélectionné. Là aussi, la méthode (M) est relativement déterministe à l'opposé de la méthode (A).

Toutes les combinaisons possibles ont été étudiées. Toutes ces méthodes utilisent le principe suivant : un nœud modifié lors d'une itération ne pas être choisi à l'itération suivante. De plus, la nouvelle couleur choisie pour le nœud à modifier est forcément différente de l'ancienne qui lui était allouée quelque soit le type de choix de couleur utilisé. Le nom de chacune des méthodes se composera de la façon suivante : la première partie du nom correspond au mode de sélection du nœud, la seconde partie correspond au mode de modification de la couleur.

- Méthode **A|A** : Le nœud et la couleur choisis aléatoirement.
- Méthode **C|A** : Un nœud est choisi parmi les nœuds en conflit et la couleur est aléatoire.
- Méthode **C|M** : Un nœud parmi les nœuds en conflit et la couleur la moins utilisée par les nœuds voisins.
- Méthode **P|A** : Un nœud parmi les nœuds les plus en conflit et une couleur choisie aléatoirement.
- Méthode **P|M** : Un nœud parmi les nœuds les plus en conflit et la couleur la moins utilisée par les nœuds voisins.

2.2.1.2 Comparaison des méthodes

Pour effectuer les tests de comparaison sur les différents algorithmes définis précédemment, dix exécutions de chacun de ces algorithmes ont été effectuées sur les benchmarks

CNET. Le tableau 2.3 présente les résultats des différentes variantes de la recherche locale sur les problèmes CNET avec un nombre maximum d'itérations fixé à 1 000 000. Trois critères de comparaison sont utilisés :

- Le taux de succès s est une note sur 10 représentant le nombre d'exécutions ayant trouvé l'optimum global sur l'ensemble des exécutions ; plus particulièrement, pour ce tableau il correspond au nombre d'exécutions réussies sur les 10 exécutions réalisées.
- Le nombre moyen d'itérations it nécessaire à l'obtention de cette solution optimale.
- Le nombre moyen de conflits restants c lorsque l'optimum n'est pas obtenu.

Les résultats présentés dans ce tableau sont classés : plus les cellules sont foncées, plus le résultat est bon. Le classement est effectué en priorité par rapport au taux de succès. Lorsqu'il est identique, le nombre moyen de conflits restants, pour les exécutions n'ayant pas trouvé l'optimum, est utilisé suivi du nombre moyen d'itérations.

TAB. 2.3 – Comparaison des recherches locales de base

problèmes	A A		C A			C M			P A			P M		
	s	c	s	it	c	s	it	c	s	it	c	s	it	c
4.75.10	0	34	0	-	18	10	661	-	10	4235	-	10	413	-
4.75.20	0	90	0	-	88	0	-	40	6	2429	26	4	295	34
4.75.30	0	148	0	-	133	10	583	-	10	674	-	0	-	114
8.75.10	0	12	10	288	-	10	29	-	10	134	-	10	20	-
8.75.20	0	38	0	-	18	10	90	-	10	1935	-	10	70	-
8.75.30	0	61	0	-	55	10	1110	-	4	21716	2	8	618	11
8.150.10	0	89	0	-	49	10	148	-	10	950	-	10	114	-
8.150.20	0	205	0	-	201	0	-	47	0	-	24	0	-	46
8.150.30	0	331	0	-	327	0	-	159	1	540926	86	0	-	120
15.150.10	0	42	10	3667	-	10	110	-	10	306	-	10	47	-
15.150.20	0	102	0	-	64	10	118	-	10	7707	-	10	88	-
15.150.30	0	164	0	-	151	9	743	1	0	-	2	8	433	10
15.300.10	0	229	0	-	148	10	220	-	10	2533	-	10	162	-
15.300.20	0	493	0	-	486	0	-	38	0	-	17	0	-	47
15.300.30	0	774	0	-	764	0	-	316	0	-	161	0	-	197

A partir de ces résultats, on remarque plusieurs points intéressants. Tout d'abord, la méthode **A|A** ne trouve jamais de solution optimale quel que soit le problème CNET traité. De plus, elle est toujours classée dernière. En conséquence, elle ne sera plus étudiée

par la suite. La méthode **C|A** ne sera pas non plus conservée puisqu'elle est toujours en 4^e position.

Les trois autres méthodes, **C|M**, **P|A** et **P|M**, sont toujours classées dans les trois premières. Chaque méthode peut être notée selon la notation suivante : $P_1|P_2|P_3$ avec P_1 le nombre de fois que la méthode arrive en première position, P_2 en seconde position et P_3 en troisième position. La méthode **C|M** est notée 4|7|5, **P|A** 5|2|9 et **P|M** 7|7|2.

Globalement, la meilleure méthode est **P|M** puisqu'elle obtient les meilleurs résultats pour sept instances sur les seize présentées et elle est classée troisième uniquement pour deux instances. La seconde méthode est **C|M** parce qu'elle est classée première pour quatre instances et seconde pour sept autres. Finalement, la troisième méthode est **P|A**. Bien que cette méthode soit la meilleure pour cinq instances sur les seize, elle est classée troisième pour neuf instances soit plus de la moitié des instances de ce tableau.

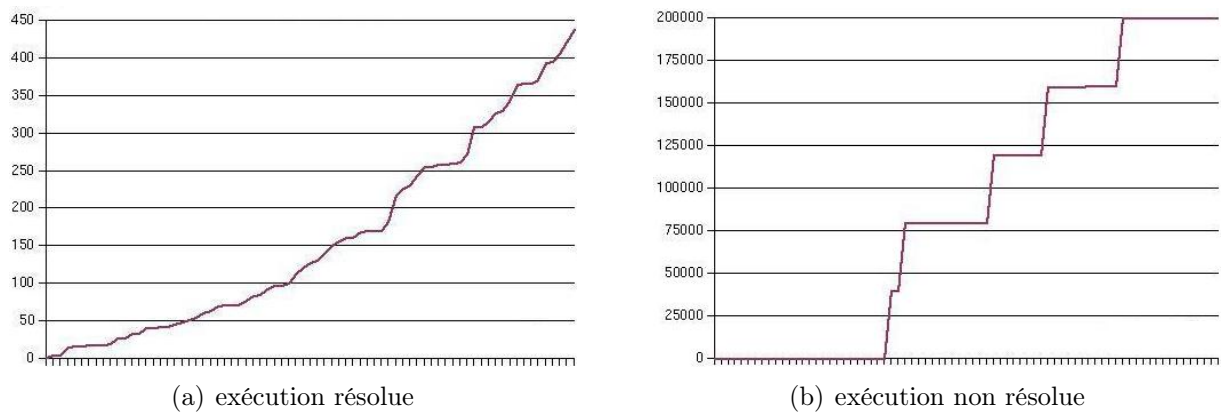


FIG. 2.1 – 4.75.20 : Application de la méthode déterministe **P|M**

Bien que la combinaison **P|M** ait été la plus performante sur les jeux de tests CNET, elle ne trouve pas systématiquement l'optimum. Le point de départ de notre étude est l'examen du comportement de la méthode lorsqu'elle réussit et lorsqu'elle échoue à trouver une solution réalisable. Nous cherchons à comprendre les raisons pour lesquelles la méthode n'a pas trouvé l'optimum. Les figures 2.1 représentent le nombre de visites cumulées de chaque nœud au cours de deux exécutions sur le problème 4.75.20 : la première a permis l'obtention d'une solution optimale (courbe 2.1(a)) tandis que la seconde a abouti à un optimum local (courbe 2.1(b)). En abscisse, les nœuds sont classés par ordre croissant de degré : du plus faible au plus haut degré. L'ordonnée correspond au nombre de visites cumulées par nœud.

Ces deux courbes montrent une répartition du nombre de visites cumulées par nœud très différente. En effet, la courbe (2.1(a)) correspondant à une exécution résolue est plus régulière que la courbe (2.1(b)). On observe une progression en paliers ainsi que des montées parfois très brusques sur la courbe correspondant à une exécution non résolue. Ainsi, au cours de cette exécution certains nœuds ont été très souvent choisis (correspondant

aux montées brusques) et d'autres, à l'inverse, ont été très peu visités (correspondant aux plateaux). La méthode utilisée étant très déterministe dans le choix du nœud mais aussi dans celui de la couleur, lors de l'atteinte d'un optimum local certains nœuds, en l'occurrence les nœuds les plus en conflit, sont beaucoup trop visités par rapport aux autres empêchant l'algorithme de se sortir du puits.

En particulier, les deux méthodes les plus déterministes, qui sont les plus efficaces sur ces instances, ne trouvent jamais de solutions réalisables aux quatre problèmes les plus difficiles notés en gras dans le tableau. La méthode $\mathbf{P|A}$, la plus aléatoire des trois, est classée quatre fois première sur ces problèmes. Idéalement, il faudrait donc combiner les méthodes en s'adaptant au problème pour trouver globalement le meilleur résultat sur toutes les instances.

En partant de la meilleure méthode $\mathbf{P|M}$, nous avons choisi de la combiner avec une seconde méthode. Le tableau 2.3 a montré que, parmi les cinq méthodes présentées, trois permettent d'obtenir le meilleur résultat sur au moins un des problèmes. En conséquence, nous choisissons de tenter de combiner les deux méthodes ayant obtenu les meilleurs résultats sur les instances CNET : la méthode $\mathbf{P|M}$ et la méthode $\mathbf{C|M}$. Pour cela, nous devons répondre à la question suivante : Comment combiner deux méthodes ? La partie suivante apporte une réponse à cette question.

2.2.2 Détection de boucle et liste Tabou

Cette partie est subdivisée en trois sous parties. Dans un premier temps, le mécanisme de détection de boucle est expliqué. Ensuite, l'intérêt d'ajouter une liste Tabou à la méthode est étudié. Pour finir, les résultats et analyses liés à ces deux mécanismes sont présentés.

2.2.2.1 Détection de boucle

Nous voulons tout d'abord identifier les périodes de blocage de la recherche locale de base $\mathbf{P|M}$. Pour cela nous avons mis en place un mécanisme de détection de boucle (notée *DB*). Une boucle est détectée lorsque le nombre de visites d'un nœud dépasse un certain seuil *occ* donné par l'équation (2.7) au cours des M dernières itérations, avec arbitrairement $M = |V|/2$. Il ne s'agit donc pas de détection de la reproduction de série de nœuds mais de la construction et de l'exploitation d'une mémoire des nœuds visités, indépendamment de tout ordre. Le nombre d'occurrence *occ* correspond à la valeur entière supérieure du produit de la taille par le pourcentage α .

$$occ = \lceil \alpha \times M \rceil \quad (2.7)$$

La détection de boucle est un mécanisme pouvant permettre d'identifier les problèmes de saturation liés à l'utilisation d'une méthode trop déterministe. En cas de boucle, nous proposons de changer de méthode et de créer une extension du voisinage de la solution courante. Ainsi, lors de l'itération suivant une détection de boucle, nous proposons que le

mécanisme de choix du nœud soit modifié pour apporter de la diversité : le nœud est choisi aléatoirement parmi l'ensemble des nœuds en conflit à l'exception du nœud à l'origine de la boucle ce qui correspond au transfert de l'algorithme sur la méthode $\mathbf{C|M}$. La figure 2.2 présente le schéma global de la méthode de combinaison.

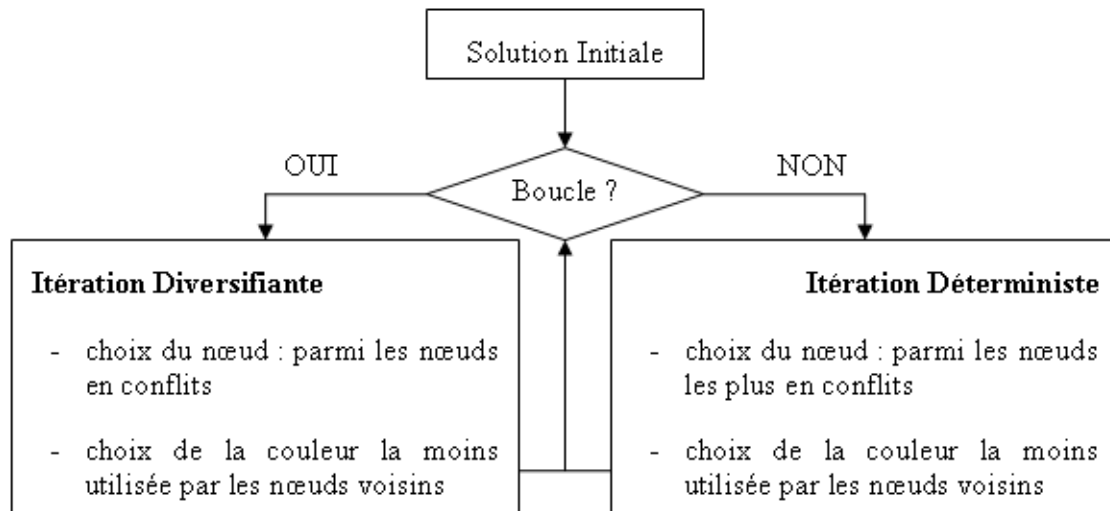


FIG. 2.2 – Fonctionnement de la détection de boucle

Afin d'illustrer et de comprendre l'importance de l'interaction entre les itérations déterministes et diversifiantes, la figure 2.3 présente le nombre d'itérations diversifiantes consécutives entre deux itérations déterministes. Pour ce test, une des plus difficiles instances CNET a été utilisée, l'instance 8.150.20 pour laquelle aucune des méthodes prises séparément n'a trouvé de solution réalisable. Entre deux itérations déterministes, le nombre d'itérations diversifiantes varie globalement entre 0 et 6. Vers la fin de la recherche, le nombre de nœud en conflit diminue ce qui pour la méthode $\mathbf{P|M}$ entraîne une probabilité de choisir un nœud déclenchant une boucle plus importante.

La détection de boucle permet de diversifier le choix du nœud lorsque le choix devient trop restrictif et ainsi d'étendre la règle de voisinage. Cependant, les nœuds provoquant une boucle peuvent être de nouveau choisis lors des itérations suivantes qu'elles soient déterministes ou diversifiantes et persister avec un nombre d'occurrence fort. Ainsi, à court terme les nœuds à l'origine des boucles peuvent se répéter. Pour gérer ce phénomène, l'utilisation d'une interdiction durable d'utilisation du nœud à l'origine de la boucle est introduite. Une liste Tabou telle que celle définie dans la méthode de Recherche Tabou [62, 63] apparaît comme un bon mécanisme pour éviter ainsi les répétitions rapides de choix du nœud.

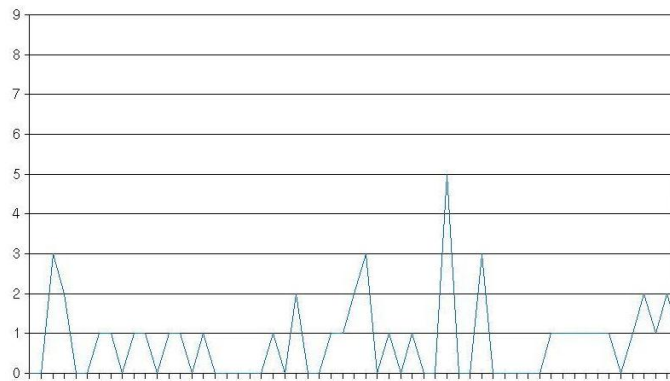


FIG. 2.3 – 8.15.20 : Nombre d'itérations aléatoires entre deux itérations déterministes en fin d'exécution

2.2.2.2 Liste Tabou

La combinaison d'une liste Tabou à la détection de boucle permet de mémoriser durablement les nœuds ayant été détectés dans des boucles et ainsi de les interdire pendant une période donnée. Dans ce cadre coopératif, la détection de boucle garde son rôle de déclencheur du type de voisinage et la liste Tabou restreint le voisinage.

Deux types de liste Tabou ont été étudiés :

- **Une liste Tabou de nœuds** : le choix de ces nœuds est interdit durablement.
- **Une liste Tabou d'associations (nœud, couleur)** : seule l'affectation d'une couleur est taboue pour un nœud.

Deux définitions de la durée Tabou ont été étudiées :

- **Durée Tabou statique** : la durée d'interdiction est la même pour tous les nœuds pendant toute la recherche.
- **Durée Tabou dynamique** : la durée Tabou est choisie aléatoirement dans l'intervalle suivant : $[0.5 \times f(N) ; 1.5 \times f(N)]$ avec $f(N)$ une fonction dépendant du nombre de nœuds du graphe.

Dans la méthode, les solutions voisines de la solution courante ne sont pas explorées et le voisin sélectionné ne l'est pas par rapport à sa qualité. La prochaine section présente une succession d'expérimentations permettant de mesurer les interactions entre la détection de boucle et la liste Tabou.

2.2.2.3 Résultat et analyse

L'objectif de cette section est de discuter plusieurs résultats de la combinaison de ces deux mécanismes.

2.2.2.3.1 Influence de la détection de boucle

Le tableau 2.4 compare les résultats obtenus par les méthodes $\mathbf{P|M}$ et $\mathbf{C|M}$ avec la combinaison par détection de boucle sur les instances CNET. Les deux méthodes présentées correspondent à la méthode basée sur le choix d'un nœud en conflit (méthode $\mathbf{C|M}$) et à celle basée sur le choix du nœud le plus en conflit (méthode $\mathbf{P|M}$). La détection de boucle présentée en colonne DB correspond à une combinaison de ces deux méthodes où les itérations diversifiantes sont réalisées par $\mathbf{C|M}$.

Pour chaque méthode, deux critères de comparaison sont présentés : le taux de succès parmi les 10 exécutions de 300 000 itérations ainsi que le nombre moyen d'itérations nécessaires à la résolution du problème. Au cours de ces tests, la valeur du paramètre α est fixé à 5%.

TAB. 2.4 – Comparaison des méthodes sur les instances CNET

problèmes	méthode sans DB						méthode DB		
	$\mathbf{C M}$			$\mathbf{P M}$			s	it	c
	s	it	c	s	it	c	s	it	c
4.75.10	10	661	-	10	413	-	10	499	-
4.75.20	0	-	40	4	295	34	10	394	-
4.75.30	10	583	-	0	-	114	10	297	-
8.75.10	10	29	-	10	20	-	10	23	-
8.75.20	10	90	-	10	70	-	10	67	-
8.75.30	10	1110	-	8	618	11	10	904	-
8.150.10	10	148	-	10	114	-	10	108	-
8.150.20	0	-	47	0	-	46	10	98625	-
8.150.30	0	-	159	0	-	120	10	1319	-
15.150.10	10	110	-	10	47	-	10	44	-
15.150.20	10	118	-	10	88	-	10	101	-
15.150.30	9	743	1	8	433	10	9	463	1
15.300.10	10	220	-	10	162	-	10	173	-
15.300.20	0	-	38	0	-	47	10	53780	-
15.300.30	0	-	316	0	-	197	10	57283	-

Si on reprend la notation précédente $P_1|P_2|P_3$ avec P_i le nombre de fois que la méthode arrive en i^e position, on obtient 0|4|11 pour la méthode $\mathbf{C|M}$, 4|7|4 pour $\mathbf{P|M}$ et 11|4|0 pour la détection de boucle. Ainsi, la méthode de détection de boucle obtient les meilleurs

résultats pour 11 instances. Pour les instances où la détection de boucle est seconde, on observe que la différence entre le nombre moyen d'itérations utilisées par la méthode classique première et la méthode de détection de boucle est très faible. La détection de boucle a permis de résoudre la totalité des instances CNET dont certaines n'étaient pas résolues par les méthodes **P|M** et **C|M**. Ainsi, la combinaison adaptative selon une règle parfaitement déterministe de deux méthodes de recherche locale a permis d'obtenir des résultats meilleurs que ces deux méthodes prises séparément.

La combinaison des méthodes par détection de boucle ayant montré son intérêt sur ces instances, nous allons maintenant travailler sur les instances DIMACS plus difficiles en évaluant aussi l'apport d'une liste Tabou sur les variables pour gérer leur réutilisation.

2.2.2.3.2 Détection de boucle et liste Tabou

Dans cette section, nous présentons des résultats comparatifs de nos différentes approches sur des instances DIMACS. Le tableau 2.5 expose les résultats obtenus pour les graphes Leighton et Flat. Dix exécutions de 1 000 000 d'itérations ont été effectuées. Pour toutes ces instances, le nombre chromatique est connu (dans la deuxième colonne). Ce tableau compare cinq méthodes. La première est basée sur le mécanisme de détection de boucle sans liste Tabou (première colonne de résultats). Les quatre autres utilisent toutes la détection de boucle et une liste Tabou. Les méthodes deux et trois (respectivement 2^e et 3^e colonnes de résultats) utilisent une durée Tabou statique de valeur $\sqrt{N}/2$, alors que les deux dernières méthodes (4^e et 5^e colonnes de résultats) sont basées sur l'utilisation d'une durée dynamique choisie dans l'intervalle $[0.5 \times \sqrt{N}/2 ; 1.5 \times \sqrt{N}/2]$. Les deuxième et quatrième méthodes utilisent une liste Tabou des nœuds ayant déclenché une boucle. Les deux autres utilisent une liste Tabou de tous les nœuds visités ; les gestions des opérations diversifiantes et des variables taboues sont donc isolées. Pour chaque méthode, deux critères sont présentés : le nombre d'exécutions réussi parmi les 10 exécutions est noté s et le nombre moyen d'itérations est noté it .

Tout d'abord l'association de la liste Tabou à la détection de boucle est convaincante car que ce soit en nombre d'instances résolues, en taux de succès ou en nombre d'itérations, la détection de boucle seule est toujours améliorée avec une option taboue. Analysons comment les options taboues se comportent les unes par rapport aux autres.

A partir de ce tableau, mis à part le problème *flat_26_0*, on remarque que les méthodes trois et cinq qui mettent tous les nœuds tabous permettent d'obtenir une solution optimale lors de chaque exécution ou ne trouvent jamais de solution optimale pour un problème donné. Au contraire, lorsque la liste Tabou est composée des nœuds bouclants, les résultats de la méthode ne semblent pas aussi binaire ; par exemple, avec une durée Tabou statique, sur les 12 problèmes résolus, ce qui est la meilleure performance, seuls 3 le sont lors de chaque exécution. En particulier, pour les problèmes *le450_15a*, *le450_15b*, *le450_15c* et *le450_15d*, les résultats diffèrent en fonction du type de liste Tabou : tous les nœuds ou les nœuds bouclants. Pour les instances *le450_15a* et *le450_15b*, les méthodes 3 et 5 ne trouvent jamais la solution optimale contrairement aux deux autres méthodes.

Pour les instances le450_15c et le450_15d, la méthode où seuls les nœuds bouclants deviennent tabous la trouve parfois.

En première lecture, on aurait donc tendance à préférer utiliser une durée Tabou uniquement pour les nœuds bouclants car le principe présente une meilleure robustesse face à la variété des instances (résultats de la 2^e méthode). Cependant, cette différence peut être due à la sensibilité de la méthode à la valeur du paramètre de durée Tabou et doit donc être confirmée sur d'autres cas.

En ce qui concerne les options de durée Tabou statique ou dynamique, les résultats sont très équilibrés. Sur le pourcentage de succès en 1^{er} critère ou le nombre d'itérations en 2^e critère, les deux principes se classent chacun 6 fois premiers sur les 12 instances résolues au moins une fois (cellules grisées). Comme nous l'avons vu dans le premier chapitre, plusieurs études ont montré l'intérêt des listes Tabou dynamiques par rapport

TAB. 2.5 – Comparaison des méthodes sur les instances DIMACS

		Méthode de détection de boucle									
		sans liste Tabou		durée Tabou statique $\sqrt{N}/2$				durée Tabou dynamique $[0.5 \times \sqrt{N}/2 ; 1.5 \times \sqrt{N}/2]$			
				nœuds bouclants		tous les nœuds		nœuds bouclants		tous les nœuds	
DIMACS	K	s	it	s	it	s	it	s	it	s	it
le450_5a	5	7	377 528	5	234 351	10	9 679	3	302 000	10	7 903
le450_5b	5	4	169 875	1	125 511	10	26 870	1	629 000	10	28 870
le450_5c	5	9	207 041	8	460 771	10	9 070	10	252 120	10	6 107
le450_5d	5	8	102 871	9	230 401	10	3 845	8	396 000	10	5 479
le450_15a	15	4	387 305	9	452 579	0	-	5	304 000	0	-
le450_15b	15	8	44 108	8	279 306	0	-	9	390 000	0	-
le450_15c	15	0	-	1	235 273	10	183 202	0	-	10	194 312
le450_15d	15	0	-	1	128 381	10	294 031	0	-	10	184 193
le450_25a	25	9	2 052	10	2 336	10	368 317	9	2 526	10	179 095
le450_25b	25	10	1 109	10	1 061	10	6 109	10	1 198	10	8 595
le450_25c	25	0	-	0	-	0	-	0	-	0	-
le450_25d	25	0	-	0	-	0	-	0	-	0	-
flat_20_0	20	10	16 482	10	19 426	10	19 427	10	16 665	10	21 199
flat_26_0	26	0	-	1	941 371	3	274 612	2	374 937	7	523 207
flat_28_0	28	0	-	0	-	0	-	0	-	0	-

aux listes statiques dont le résultat est totalement dépendant d'une valeur taboue unique trop spécifique à un problème. Pour l'ensemble de ces raisons nous choisissons par la suite d'étudier la liste dynamique plutôt que la liste statique. Ce choix sera conforté par la suite de l'étude où nous montrerons l'importance de l'adaptabilité de la durée.

2.2.2.3.3 Nœuds bouclants et liste Tabou

En prenant l'option d'une durée Tabou dynamique, nous souhaitons mieux caractériser l'impact des nœuds bouclants mis en statut tabou par rapport aux autres options : aucun nœud tabou et tous les nœuds tabous.

Le tableau 2.6 présente les résultats obtenus pour les graphes aléatoire DSJC des instances DIMACS. Le nombre chromatique est connu uniquement pour la première instance. Pour les autres, le meilleur résultat connu dans la littérature sera utilisé (deuxième colonne). Ce tableau présente quatre méthodes. La première est la méthode **P|M** qui est la recherche locale de base la plus performante sur les problèmes CNET. Toutes les autres méthodes sont basées sur la détection de boucle : sans liste Tabou, avec une liste Tabou des nœuds bouclants et pour finir, une liste Tabou de tous les nœuds visités. La durée Tabou est dynamique autour de la valeur $f(N) = \frac{\sqrt{N}}{2}$.

Comme pour les problèmes CNET, le mécanisme de détection de boucle sans liste Tabou améliore les résultats obtenus par la méthode **P|M** seule sur 75% des problèmes à la fois en taux de succès et en moyenne de conflits restants si la satisfaction n'est pas atteinte.

Suite à l'ajout des listes Tabou de nœuds bouclants et de tous les nœuds les résultats obtenus sont sensiblement meilleurs. Les deux méthodes avec liste Tabou améliorent les scores de 9 problèmes sur 12 avec un net avantage (cellules grisées) lorsque seuls les nœuds bouclants sont mis tabous. Par exemple, la méthode utilisant la liste Tabou de nœuds bouclants permet de résoudre systématiquement l'instance DSJC250.5 alors que cette instance n'est jamais résolue par les autres méthodes. Mais surtout, et c'est le résultat le plus important, cette méthode a permis d'obtenir le nombre moyen le plus faible de conflits restants pour 5 des 6 instances non résolues.

A partir des résultats présentés dans ce tableau, on observe que la méthode combinant une détection de boucle et une liste Tabou des nœuds bouclants est la plus performante autant en taux de succès qu'en nombre d'évaluations. Cette observation implique donc que seuls certains nœuds méritent leur statut tabou pour conduire à une amélioration des résultats de la recherche. En l'occurrence ce sont les nœuds les plus visités pendant une période donnée. La méthode qui sera conservée par la suite combine la détection de boucle à une liste Tabou de nœuds bouclants pour une durée Tabou dynamique ; elle est notée *DB+TD*.

TAB. 2.6 – Comparaison des méthodes sur les instances DSJC

		méthode P M			méthode de détection de boucle									
					sans liste Tabou			liste Tabou dynamique						
								[$0.5 \times \sqrt{N}/2 ; 1.5 \times \sqrt{N}/2$]						
								nœuds bouclants			tous nœuds			
DIMACS	K	s	it	c	s	it	c	s	it	c	s	it	c	
DSJC125.1	5	10	27 959	-	10	24 294	-	10	26 421	-	10	44 395	-	
DSJC125.5	17	10	32 10 ⁶	-	10	563 127	-	10	2 10 ⁶	-	8	31 10 ⁶	1	
DSJC125.9	44	10	23 742	-	10	17 202	-	10	22 109	-	8	21 801	1	
DSJC250.1	8	10	3 10 ⁶	-	10	201 656	-	10	66 514	-	10	2 10 ⁶	-	
DSJC250.5	28	0	-	4	0	-	2	10	30 10 ⁶	-	0	-	4	
DSJC250.9	72	0	-	1	3	172 870	1	5	5 10 ⁶	-	0	-	1	
DSJC500.1	12	0	-	7	0	-	9	0	-	2	0	-	5	
DSJC500.5	48	0	-	17	0	-	16	0	-	5	0	-	16	
DSJC500.9	126	0	-	17	0	-	7	0	-	5	0	-	17	
DSJC1000.1	20	0	-	16	0	-	26	0	-	15	0	-	16	
DSJC1000.5	84	0	-	39	0	-	63	0	-	46	0	-	39	
DSJC1000.9	224	0	-	41	0	-	9	0	-	7	0	-	34	

2.2.3 Adaptation de la durée Tabou

2.2.3.1 Description de l'adaptation

Dans cette section, nous nous intéressons à l'adaptation automatique de la durée Tabou. Le paramètre a a une très grande influence sur la Recherche Tabou comme nous l'avons vu dans le premier chapitre, il en a tout autant sur la méthode RLA que nous présentons. Pour l'illustrer, le tableau 2.7 présente les résultats obtenus par la méthode utilisant une liste Tabou de tous les nœuds visités avec 5 valeurs différentes de $f(N)$ pour une durée dynamique. Trois critères sont utilisés : le taux de succès, le nombre d'itérations nécessaires à la résolution et le nombre de conflits de la meilleure solution obtenue dans le cas où l'optimum n'a pas été trouvé.

Notre définition initiale de la durée Tabou est basée essentiellement sur le nombre de nœuds N , ce qui correspond au choix le plus fréquent dans la littérature. Les deux instances présentées dans le tableau 2.7 possèdent un nombre de nœuds identique (450) pour faire ressortir que cette propriété n'est pas suffisante pour identifier une durée adaptée. Le premier problème est résolu dans 100% des exécutions avec une durée Tabou choisie dans le premier intervalle autour de la valeur de $\sqrt{N}/8$. On observe que le second problème

TAB. 2.7 – Influence de l'intervalle de la durée Tabou dynamique

$f(N)$	$0.5 \times \frac{\sqrt{N}}{8}; 1.5 \times \frac{\sqrt{N}}{8}$			$0.5 \times \frac{\sqrt{N}}{7}; 1.5 \times \frac{\sqrt{N}}{7}$			$0.5 \times \frac{\sqrt{N}}{5}; 1.5 \times \frac{\sqrt{N}}{5}$			$0.5 \times \frac{\sqrt{N}}{4}; 1.5 \times \frac{\sqrt{N}}{4}$		
DIMACS	s	it	c	s	it	c	s	it	c	s	it	c
le450_15a	10	429 118	0	5	147 775	2	0	-	2	0	-	4
le450_15b	0	-	1	5	296 529	2	0	-	1	0	-	4

n'est jamais résolu avec cet intervalle. L'utilisation de l'intervalle suivant permet d'obtenir des résultats avec 50% de réussite sur chaque instance. Pour des intervalles plus grands (les deux dernières colonnes), il n'y a pas de cas résolus ! On voit bien sur ce cas la limite d'une démarche empirique centrée sur la recherche d'une valeur de référence unique pour tous les problèmes : deux instances ayant un même nombre de nœuds ne réagissent pas de la même façon aux différentes valeurs de durée Tabou.

Le tableau 2.7 montre bien l'importance de la valeur de la durée Tabou. Il est très difficile de trouver une durée Tabou idéale pour toutes les instances traitées. C'est pour cette raison que nous avons étudié l'adaptation automatique de ce paramètre. La méthode que nous avons proposée ajuste la durée Tabou en fonction du comportement de la recherche. Un historique est alors employé pour déterminer la période de prohibition de chacune des variables de décision. Une bonne valeur de paramètre de durée Tabou devrait empêcher des cycles de recherche et devrait donc être suffisamment importante pour exclure à bon escient les variables bouclantes et orienter la recherche vers de nouvelles configurations.

Notre proposition est d'adapter le paramètre à chaque problème et à chacune des variables du problème en question en utilisant un historique pour chaque variable sur son implication dans les boucles détectées. Nous avons déjà décelé que le statut tabou est plus pertinent s'il est utilisé sur les variables détectées dans des boucles. En complément, nous souhaitons configurer dynamiquement la durée Tabou en fonction du nombre de boucles impliquant chaque variable du problème. Nous utilisons donc le nombre de boucles détectées pour un nœud par rapport au nombre de boucles détectées pour l'ensemble des nœuds. La durée Tabou devient proportionnelle au nombre de boucles détectées par variable par rapport au nombre moyen de boucles détectées.

A chaque boucle détectée, le nœud à l'origine de la boucle sera donc mis tabou pour une durée spécifique et adaptative selon son historique et l'historique de l'ensemble des nœuds, suivant la formule :

$$AT(x_i) = \text{rand}(DT) + \frac{nbBoucles(x_i)}{\sum_{x_j \in V} nbBoucles(x_j)} \times N \quad (2.8)$$

avec $nbBoucles(x_i)$ le nombre de boucles détectées pour le nœud x_i , et l'intervalle DT

défini comme pour une durée Tabou dynamique :

$$DT = \left[0.5 \times \frac{\sqrt{N}}{2}; 1.5 \times \frac{\sqrt{N}}{2} \right] \quad (2.9)$$

Cette méthode sera notée par la suite $DB+TA$ pour détection de boucle et durée Tabou adaptative sur chaque nœud bouclant.

2.2.3.2 Étude de la répartition des durées Tabou

Ce paragraphe permet d'étudier le comportement de deux méthodes : la méthode utilisant une durée Tabou dynamique, notée $DB+TD$, et celle utilisant la durée Tabou adaptative, notée $DB+TA$. Les deux éléments observés ici sont les valeurs de durée Tabou utilisées par chacun des nœuds ainsi que le nombre de boucles générées par nœud. Les figures 2.4 et 2.6 correspondent au problème DSJC125.1 pour une exécution résolue et les figures 2.5 et 2.7 au problème DSJC500.1 pour une exécution non résolue. Les deux premières figures exposent la méthode $DB+TD$ et les deux suivantes la méthode $DB+TA$. Les spectrogrammes (a) représentent la fréquence d'apparition de chacune des valeurs. Plus les points sont foncés, plus cette valeur a été utilisée. Pour chacune de ces figures, le spectrogramme (a) correspond aux différentes valeurs de durée Tabou utilisées par la méthode $DB+TD$ (spectrogrammes 2.4 et 2.5) ou la méthode $DB+TA$ (spectrogrammes 2.6 et 2.7) et la courbe (b) représente le nombre de boucles provoquées par chaque nœud. Sur toutes ces figures, l'axe des abscisses représente les nœuds classés par ordre croissant de degré.

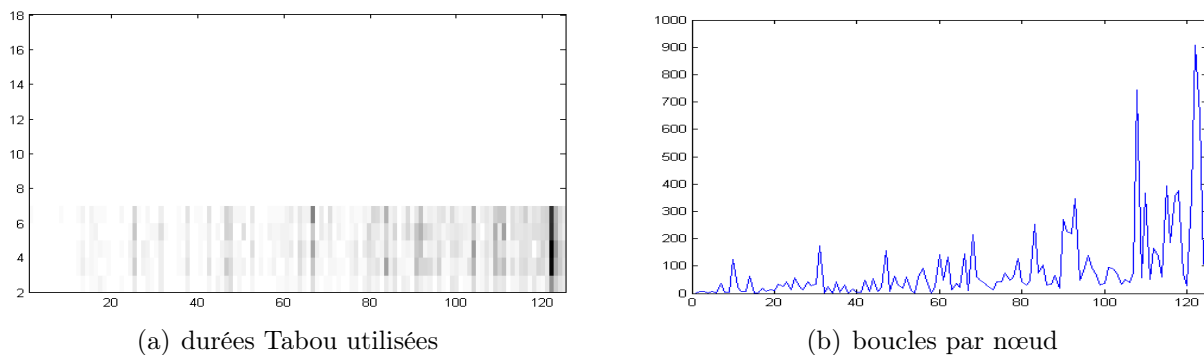


FIG. 2.4 – Étude sur l'instance DSJC125.1 (méthode $DB+TD$)

Dans la méthode $DB+TD$, les valeurs de durée Tabou sont choisies de façon uniforme au sein d'un même intervalle autour de la valeur de $f(N) = \sqrt{N}/2$. Les spectrogrammes 2.4(a) et 2.5(a) montrent ainsi cette répartition uniforme. Pour les nœuds ayant provoqué le plus de boucles (courbes 2.4(b) et 2.5(b)), on observe que toutes les valeurs de durée Tabou sont plus foncées (courbes 2.4(a) et 2.5(a)). En d'autres termes, chaque nœud a utilisé toutes les valeurs de l'intervalle de façon équiprobable.

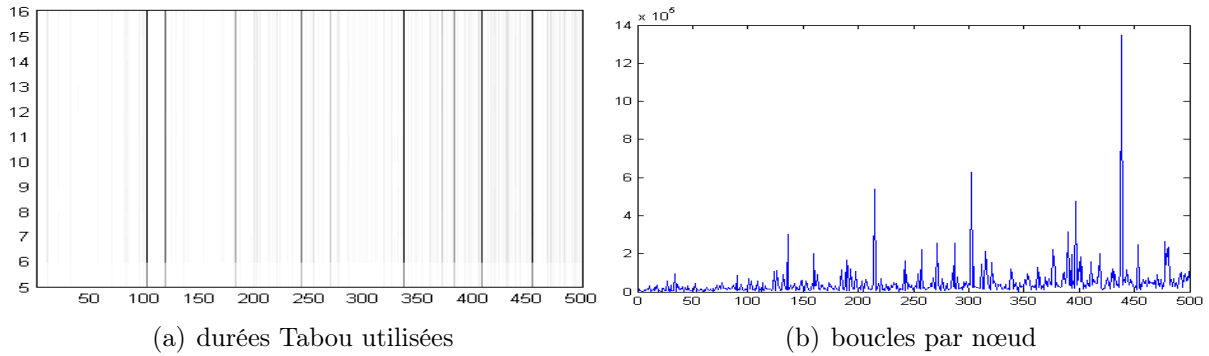


FIG. 2.5 – Étude sur l'instance DSJC500.1 (méthode DB+TD)

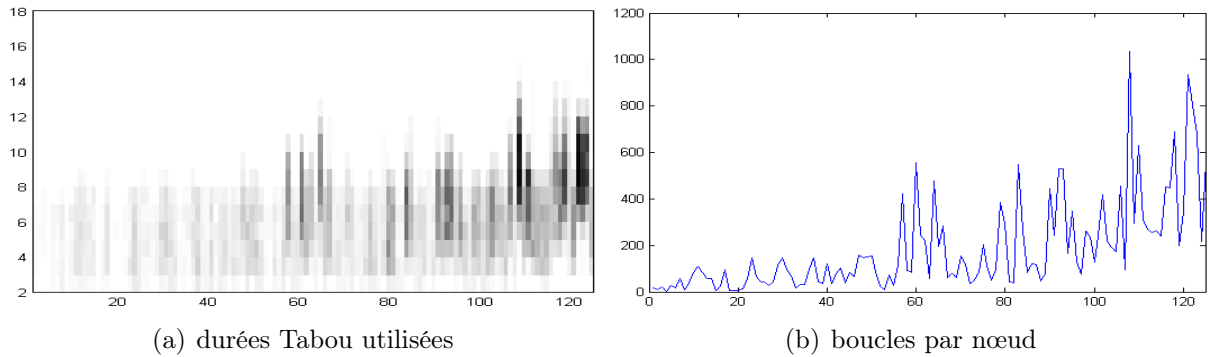


FIG. 2.6 – Étude sur l'instance DSJC125.1 (méthode DB+TA)

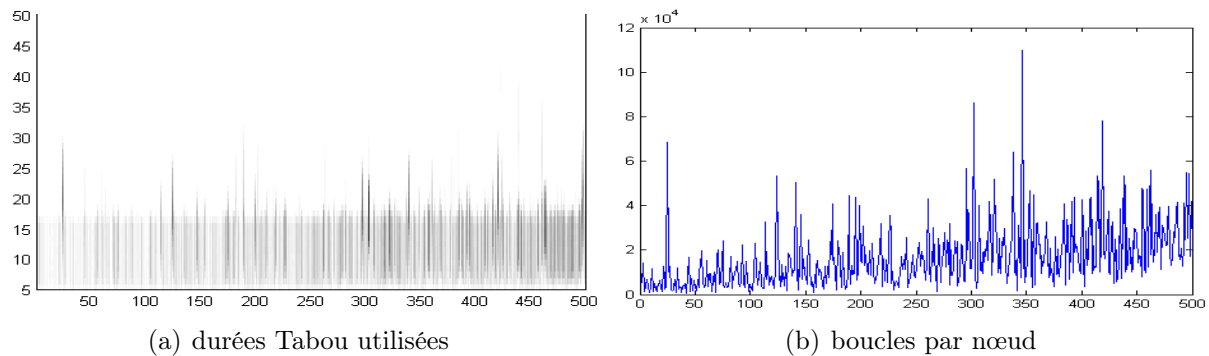


FIG. 2.7 – Étude sur l'instance DSJC500.1 (méthode DB+TA)

Contrairement aux deux premières figures, les spectrogrammes 2.6(a) et 2.7(a) sont très différents : les durées Tabou ne sont pas utilisées de façon uniforme par la méthode DB+TA. Nous espérons donc de ce système une meilleure diversification de la recherche au bénéfice des variables qui ne sont pas impliquées dans les boucles. Par ailleurs, selon la méthode que nous avons proposée, les nœuds provoquant le plus de boucles (courbes 2.6(b) et 2.7(b)) utilisent des valeurs de durée Tabou plus importantes. La valeur de pénalité a augmenté de façon significative les valeurs de durée Tabou utilisées ce qui correspond au

comportement attendu et observé sur plusieurs instances.

Après avoir observé la différence comportementale entre ces deux méthodes du point de vue des valeurs de durée Tabou utilisées pour chaque nœud, le tableau 2.8 présente les résultats comparatifs obtenus avec les durées Tabou dynamiques et adaptatives sur les instances Leighton. Pour ces tests, le nombre d'occurrence occ est égale à $\lceil 5\% \times M \rceil$ pour toutes les méthodes présentées. Deux critères de comparaison sont utilisés dans ce tableau : s le taux de succès sur les 10 exécutions ainsi que it le nombre moyen d'itérations nécessaire à la résolution de chaque instance.

TAB. 2.8 – Durée Tabou adaptative sur les instances Leighton

		$occ = \lceil 5\% \times M \rceil$			
		durée Tabou dynamique DB+TD		durée Tabou adaptative DB+TA	
problèmes	k	s	it	s	it
le450_5a	5	3	302 000	10	326 148
le450_5b	5	1	629 000	2	1 205 950
le450_5c	5	10	252 120	10	251 881
le450_5d	5	8	396 000	2	1 079 031
le450_15a	15	5	304 000	10	1 889 569
le450_15b	15	9	390 000	10	904 067
le450_15c	15	0	–	10	70.6×10^6
le450_15d	15	0	–	4	192.6×10^6

Pour tous les problèmes étudiés, nous observons que la durée Tabou adaptative a permis d'améliorer les résultats en terme de taux de succès ou de nombre d'itérations en 2^e critère pour 7 des 8 instances. Seul le résultat de la méthode dynamique pour le problème le450_5d est meilleur que celui obtenu par la méthode adaptative. Par ailleurs, la méthode s'avère robuste puisqu'elle trouve une solution à tous les problèmes ce qui montre l'efficacité de la combinaison des mécanismes.

Ces résultats se confirment dans le cas des problèmes DSJC (tableau 2.9). Le taux de succès est identique sauf pour le DSJC500.1 mais le nombre moyen de contraintes violées est toujours plus faible pour la méthode AT (les tests ont été réalisés sur 5 exécutions). Cette méthode améliore le nombre moyen de conflits restants pour 5 des 6 instances.

2.2.3.3 Étude de corrélation entre nombres de boucles et de visites

Le but de cette partie est d'étudier le nombre de boucles détectées au cours de la recherche et plus particulièrement, le nombre de boucles détectées par nœud par rapport

TAB. 2.9 – Adaptation de la durée Tabou sur les instances DSJC

problèmes	DB+TD		DB+TA	
	s	c	s	c
DSJC500.1	0	1	6	0.4
DSJC500.5	0	4.8	0	4.2
DSJC500.9	0	3.8	0	3.4
DSJC1000.1	0	15	0	13.4
DSJC1000.5	0	41.2	0	42.8
DSJC1000.9	0	4.4	0	4

au nombre de visites en fonction du type de durée Tabou, dynamique ou adaptative. Les figures 2.4(b), 2.5(b), 2.6(b) et 2.7(b) montrent que le nombre de boucles détectées par nœud n'est pas identique pour tous les nœuds.

Le premier facteur qui influe sur le nombre de boucles est le rapport entre le nombre d'occurrences recherchées, fixé par le paramètre α , et la taille de la fenêtre d'observation, fixée à $N/2$ où N est le nombre de nœuds du graphe selon l'équation (2.7). Pour l'instant, nous avons fixé α à 5% ce qui demande plus d'occurrences du même nœud pour déclencher une boucle pour les plus grands problèmes. Nous avons donc fait varier les valeurs de ce paramètre. Le tableau 2.10 présente les résultats obtenus sur les graphes aléatoires DSJC avec deux valeurs différentes de α . Nous avons conservé les deux méthodes de détection de boucle avec durée Tabou dynamique (DB+TD) et durée Tabou adaptative (DB+TA) pour ce test. Pour chaque méthode, trois critères sont utilisés : le taux de succès (sur 5 exécutions de 3×10^8 itérations), le nombre moyen de conflits de la meilleure solution trouvée et le pourcentage de boucles détectées pendant la recherche (noté *bcle*). Les meilleurs scores sont soulignés.

TAB. 2.10 – Influence de α sur les instances DSJC

problèmes	$\alpha = 5\%$						$\alpha = 1\%$					
	DB+TD			DB+TA			DB+TD			DB+TA		
	s	c	bcle	s	c	bcle	s	c	bcle	s	c	bcle
DSJC500.1	0	1	14.5	6	0.4	12.6	0	2.4	59.3	0	3.2	58.9
DSJC500.5	0	4.8	13.9	0	4.2	13.2	0	28.4	48.8	0	28.2	48.8
DSJC500.9	0	3.8	10.2	0	3.4	9.0	0	15	57.1	0	15.4	56.7
DSJC1000.1	0	15	4.3	0	13.4	2.7	0	3	50.1	0	3.4	49.5
DSJC1000.5	0	41.2	8.5	0	42.8	7.6	0	37.8	40.5	0	40.2	41.8
DSJC1000.9	0	4.4	5.4	0	4	4.7	0	23	44.5	0	22.8	43.8

Pour les instances de grandes tailles DSJC1000.1 et DSJC1000.5, les meilleures valeurs

sont obtenues avec $\alpha = 1\%$. Ces résultats montrent clairement l'influence du paramètre α qui détermine le nombre d'occurrences à rechercher dans la fenêtre d'observation de la mémoire. Les mêmes tendances sont observées avec les deux méthodes.

La durée Tabou adaptative est calculée en fonction du nombre de boucles détectées par nœud par rapport au nombre total de boucles détectées au cours de l'exécution. La détection de boucle est basée sur un pourcentage de visites d'un nœud donné au cours des dernières $N/2$ itérations. Pour les instances de 125 nœuds, une boucle est donc détectée au bout de 3 répétitions d'un nœud au cours des 62 dernières itérations avec $\alpha = 5\%$ ($occ = \lceil \alpha \times 62 \rceil$). Pour les instances à 1000 nœuds, 25 répétitions d'un même nœud sont nécessaires pour détecter une boucle lors des 500 dernières itérations. Cependant, le pourcentage d'itérations diversifiantes mesuré sur plusieurs exécutions (itérations suivant une détection de boucle) par rapport au nombre total d'itérations est très différent pour chaque instance. Pour l'instance DSJC125.1 toujours résolue (125 nœuds), avec α à 5% ce pourcentage avoisine les 40% comparés aux 5% pour l'instance DSJC1000.9 jamais résolue (1000 nœuds). Ce pourcentage est très sensible à la valeur de α ; le nombre de boucles détectées est plus important lorsque la valeur de α est petite. La colonne *bcle* (pour boucle) du tableau 2.10 nous donne des indications sur cette corrélation : pour $\alpha = 1\%$, le nombre moyen de boucles est de 50% alors qu'il est de 10% environ pour $\alpha = 5\%$.

Afin d'étudier globalement le comportement de la méthode, nous avons calculé la corrélation entre le nombre de visites et le nombre de boucles détectées par nœud. La figure 2.8 est composée de deux courbes : la première présente le nombre de visites alors que la seconde montre le nombre de boucles par nœud pour l'instance résolu DSJC125.1. La méthode utilisée est basée sur la durée Tabou dynamique avec $\alpha = 5\%$. Pour ces courbes, l'axe des abscisses représente les nœuds classés par ordre croissant de degré et l'axe des ordonnées représente le nombre de visites (courbe a) ou le nombre de boucles (courbe b). On observe visuellement une grande similarité entre ces courbes. Notamment, les nœuds les plus visités correspondent aux nœuds ayant déclenché le plus de boucles au cours de l'exécution.

Numériquement, l'objectif du calcul de corrélation est de quantifier la relation entre deux ensembles de mesures. En d'autres termes, la corrélation représente leur degré de similitude. Soit deux séries de N mesures X et Y (avec $X(i)$ et $Y(i)$, $i = 1, 2, \dots, N$), la corrélation de Pearson [135] (*the Pearson product-moment correlation coefficient*, ou "sample correlation coefficient") peut être utilisée pour estimer la ressemblance entre X et Y . Elle est calculée de la façon suivante :

$$corr(X, Y) = \frac{\sum_{i=1}^N (X(i) - \bar{X})(Y(i) - \bar{Y})}{(N - 1)(\sigma_X \times \sigma_Y)} \quad (2.10)$$

avec \bar{X} et \bar{Y} la moyenne arithmétique de X et Y , et σ_X et σ_Y l'écart type de X et Y . Le coefficient de corrélation varie entre -1 et 1 . La valeur 0 correspond à la non-corrélation

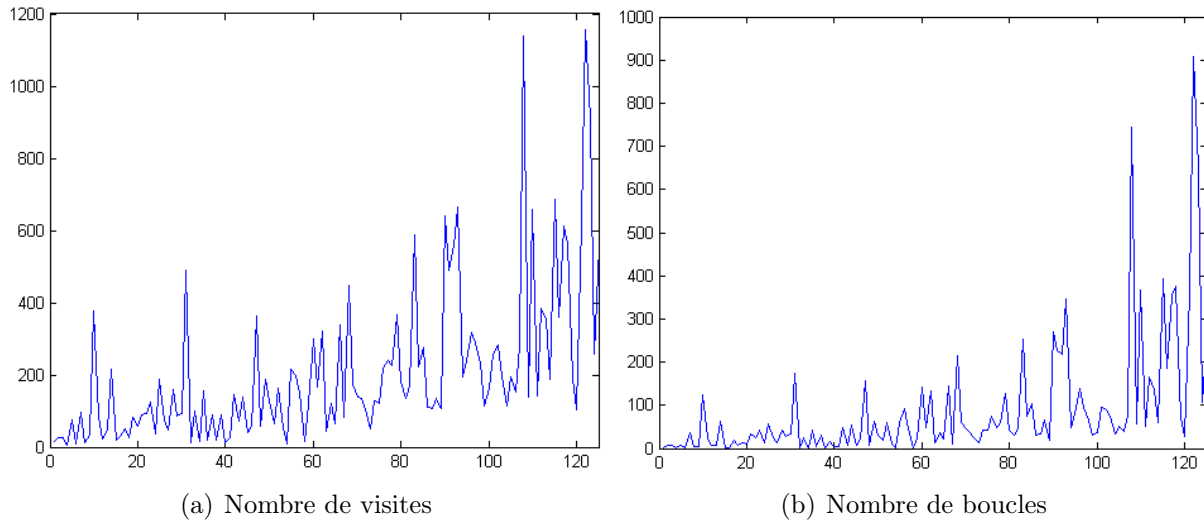


FIG. 2.8 – DSJC125.1 : Corrélation entre nombre de visites et nombre de boucles par nœud par la méthode DB+TD avec $\alpha = 5\%$ ($\text{corr}((a),(b)) = 0.9622$, $\text{évaluation}=0$)

de X et Y (ces deux ensembles de valeurs sont complètement différents) alors que la valeur 1 correspond à une corrélation positive très importante (X et Y sont identiques). Les valeurs négatives correspondent à une corrélation négative : lorsque le coefficient de corrélation est de -1 , les ensembles X et Y sont exactement opposés.

Les courbes présentées sur la figure 2.8 montre un coefficient de corrélation très important (0.9622) entre le nombre de visites et le nombre de boucles par nœud sur un problème résolu systématiquement. Les figures 2.9 et 2.10 présentent les courbes obtenues lors d’une exécution n’ayant pas abouti à la solution optimale de la méthode DB+TD pour l’instance DSJC1000.5. Seule la valeur du paramètre α est différente : 1% pour la figure 2.9, 5% pour la figure 2.10.

Sur les courbes de la figure 2.10, beaucoup de différences sont observables. Notamment, les nœuds les plus visités ne correspondent pas aux nœuds ayant déclenché le plus de boucles. Le coefficient de corrélation est de 0.5772 . Au contraire, les courbes de la figure 2.9 présentent d’importantes similarités ainsi qu’un coefficient de corrélation de 0.9903 . En outre, les meilleures solutions obtenues par ces deux exécutions sont de différentes qualités : pour $\alpha = 5\%$ (figure 2.10), la solution obtenue est de moins bonne qualité (44 conflits) que celle obtenue avec $\alpha = 1\%$ (39 conflits). Bien que la différence d’évaluation soit assez faible, la valeur de 1% est plus appropriée pour cette instance. Le nombre de boucles détectées par nœud est proportionnel au nombre de visites.

Les figures 2.11 et 2.12 représentent les courbes relatives à des exécutions de la méthode DB+TA sur l’instance DSJC1000.5 avec différentes valeurs de α : 1% pour la figure 2.11 et 5% pour la figure 2.12. Les figures 2.11 montrent que la durée Tabou adaptative permet de réduire le nombre de visites des nœuds les plus visités et le nombre de boucles

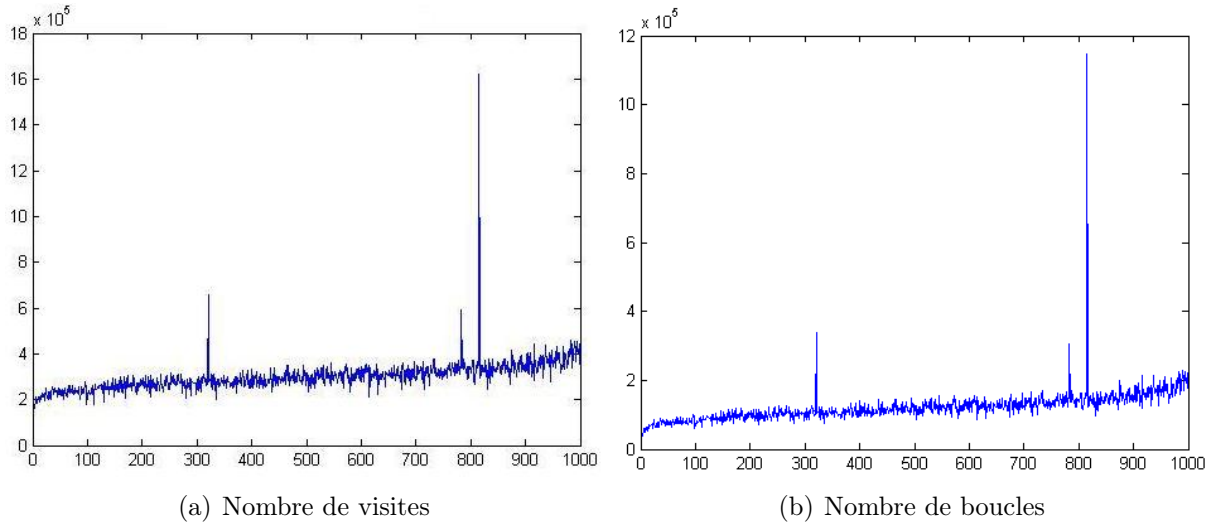


FIG. 2.9 – DSJC1000.5 : Corrélation entre nombre de visites et nombre de boucles par nœud par la méthode DB+TD avec $\alpha = 1\%$ ($\text{corr}((a),(b)) = 0.9903$, évaluation=39)

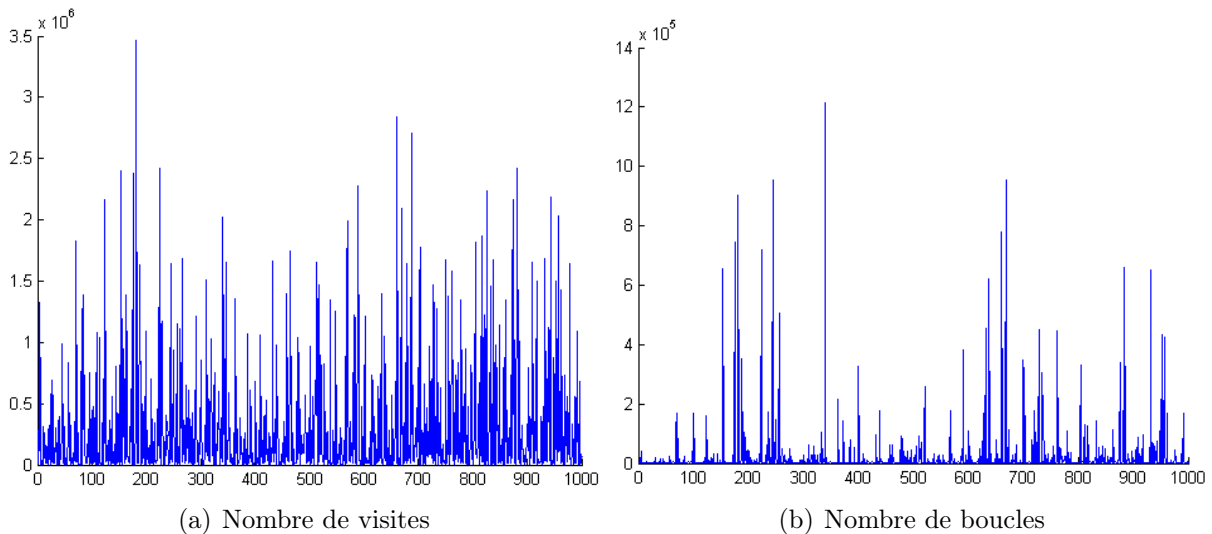


FIG. 2.10 – DSJC1000.5 : Corrélation entre nombre de visites et nombre de boucles par nœud par la méthode DB+TD avec $\alpha = 5\%$ ($\text{corr}((a),(b)) = 0.5772$, évaluation=44)

détectées pour ces nœuds. Par exemple, sur la figure 2.9, le nœud le plus visité est choisi environ 16×10^5 fois. Il est mis tabou à 12×10^5 reprises. Sur la figure 2.11, le nœud le plus visité l'est à environ 15×10^5 reprises et déclenche environ 10×10^5 boucles. De ce fait, les autres nœuds sont visités plus souvent, mais le nombre de boucles détectées pour ces nœuds ne paraît pas plus important.

La durée Tabou adaptative a permis de réduire la différence entre les nœuds les plus visités et ceux les moins visités. Le bénéfice de cette adaptation peut être vu en terme de coefficient de corrélation. Comparé au coefficient de corrélation de la méthode dynamique

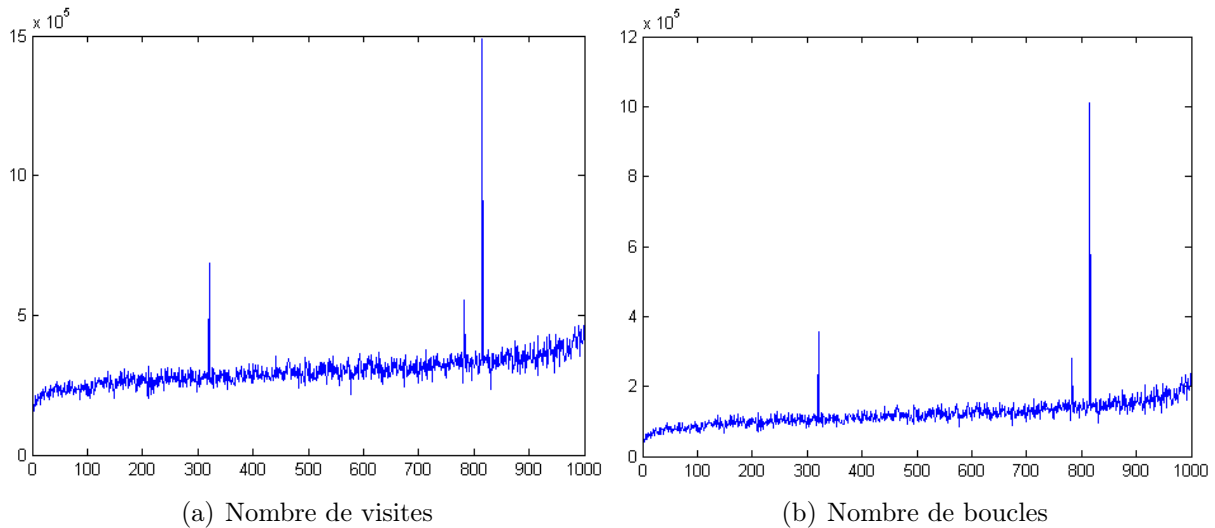


FIG. 2.11 – DSJC1000.5 : Corrélation entre nombre de visites et nombre de boucles par nœud par la méthode DB+TA avec $\alpha = 1\%$ ($\text{corr}((a),(b))=0.9923$, évaluation=33)

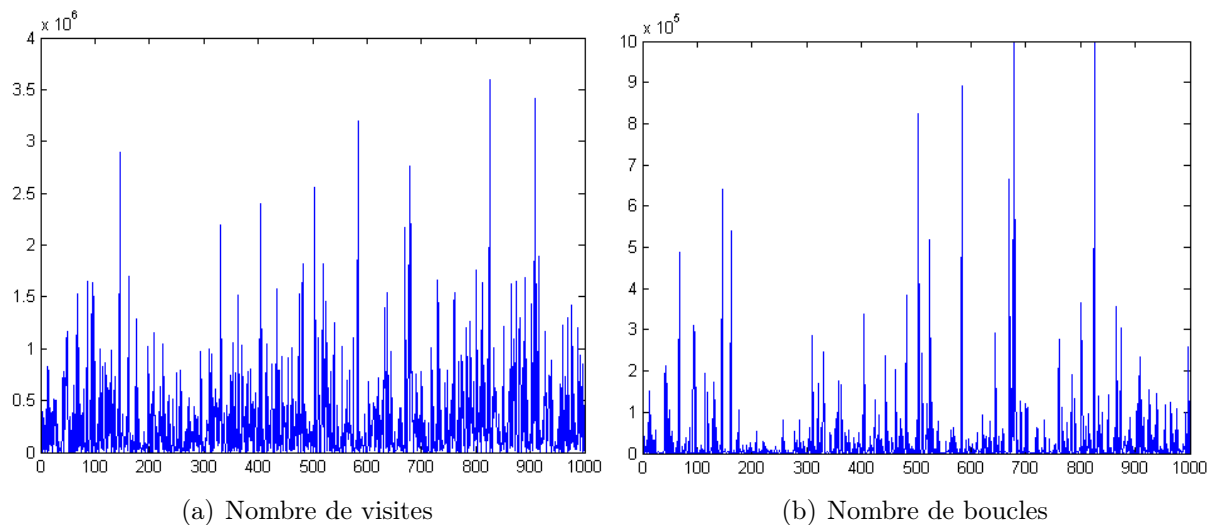


FIG. 2.12 – DSJC1000.5 : Corrélation entre nombre de visites et nombre de boucles par nœud par la méthode DB+TA avec $\alpha = 5\%$ ($\text{corr}((a),(b))=0.7350$, évaluation=42)

(voir figure 2.9), ce coefficient est légèrement supérieur (0.9923 au lieu de 0.9903). De la même façon, la figure 2.12 présente les courbes obtenues avec la durée Tabou adaptative et $\alpha = 5\%$ sur l'instance DSJC1000.5. Comme pour les courbes de la figure 2.10, les courbes de la figure 2.12 sont différentes bien que le coefficient de corrélation ait augmenté (sur la figure 2.9, il est de 0.5772 contre 0.7350 ici) avec 2 points gagnés sur la fonction de coût.

En conclusion, lorsque le nombre d'occurrences recherchées est excessivement grand (cas de $\alpha = 5\%$ sur les problèmes de 1000 nœuds), la dispersion des visites d'un nœud au cours des itérations ne permet pas de détecter des boucles même si ce nœud est fréquem-

ment visité. Et dans ce cas, l'adaptation dynamique de la durée Tabou qui se déclenche en fonction des boucles ne parvient pas à corriger efficacement le nombre de visites de chaque nœud. La corrélation entre le nombre de visites et le nombre de boucles par nœud semble un critère important puisque en général, l'amélioration de la corrélation s'accompagne d'une amélioration des résultats de la méthode. Le paramétrage de la détection de boucle peut permettre d'améliorer cette corrélation étant le responsable de la relation entre le nombre de visites et le nombre de boucles. Dans la section suivante, l'adaptation automatique de ces paramètres est présentée.

2.2.4 Adaptation de la détection de boucle

2.2.4.1 Description de l'adaptation

Pour contrôler la sélection répétitive de nœuds bouclants, les mécanismes de la sélection de boucle doivent pouvoir s'adapter aux particularités du problème. Le tableau 2.10 a montré l'importance du paramètre α avec lequel le pourcentage de boucles détectées peut être très différent d'un problème à un autre. Par exemple, avec α à 5%, sur le problème DSJC500.1 la méthode détecte en moyenne 14.5% de boucles alors que sur le problème DSJC1000.1 la méthode en détecte uniquement 4.3% ce qui rend tout à fait inopérant la combinaison des deux méthodes basées sur la détection de boucle. Tout comme pour la durée Tabou, un paramètre fixe ne suffit pas à réguler le taux de boucles, et donc par conséquent, le taux de diversification à utiliser par la méthode en fonction des propriétés des problèmes. Au lieu d'utiliser un pourcentage α de la taille de la mémoire, nous allons désormais travailler directement sur le nombre d'occurrences recherchées identifié par le paramètre *occ*.

Le tableau 2.11 présente le nombre d'occurrences *occ* utilisé pour différentes valeurs de α . Les résultats sont obtenus avec la méthode utilisant une durée Tabou dynamique DB+TD autour de $f(N) = \sqrt{N}/2$ pour les instances DSJC500.* et DSJC1000.* ainsi que pour deux des instances Leighton pour lesquelles nous n'avons pas de solution optimale à ce stade. DSJC500.* désigne l'ensemble des instances DSJC de 500 nœuds. Pour chaque valeur de α , trois critères sont présentés : le taux de succès sur les 5 exécutions de 300 millions d'itérations, le nombre moyen de conflits c de la meilleure solution obtenue au cours des 5 exécutions et la valeur d'occurrence *occ* correspondante.

Le tableau 2.11 utilise une couleur de cellule indiquant la qualité des résultats : les cellules les plus foncées correspondent aux meilleurs résultats. Ces résultats montrent que le paramètre α est décisif pour la performance de l'algorithme. En réalité, plus que le paramètre α , c'est le nombre d'occurrences de chaque nœud comme paramètre de détection de boucle qui est important.

Nous avons donc étudié la pertinence d'adapter le nombre autorisé de répétitions du choix de chaque nœud avant la détection d'une boucle en utilisant l'historique des visites de chacun d'eux. Le nombre d'occurrence $occ(x_i)$ du nœud x_i est calculé à partir du ratio entre le nombre de visites du nœud et le nombre courant d'itérations. En conséquence,

TAB. 2.11 – Influence du paramètre de détection de boucle sur la méthode DB+TD

problèmes	k	$\alpha = 1\%$			$\alpha = 2\%$			$\alpha = 3\%$			$\alpha = 4\%$			$\alpha = 5\%$		
		s	c	occ	s	c	occ	s	c	occ	s	c	occ	s	c	occ
DSJC500.1	12	0	2.4	3	10	0	5	6	0.4	8	6	0.4	10	0	1	13
DSJC500.5	48	0	28	3	0	5.6	5	0	4	8	0	4.4	10	0	4.8	13
DSJC500.9	126	0	15	3	0	6.8	5	0	4.8	8	0	4	10	0	3.8	13
DSJC1000.1	20	0	3	5	0	5.4	10	0	9.2	15	0	12	20	0	15	25
DSJC1000.5	84	0	37	5	0	27	10	0	35	15	0	40	20	0	41	25
DSJC1000.9	224	0	23	5	0	10	10	0	7	15	0	6	20	0	4.4	25
le450_25c	25	0	42	3	0	19	5	0	8.4	7	0	3.4	9	0	4	12
le450_25d	25	0	42	3	0	18	5	0	9.6	7	0	4	9	0	5	12

plus un nœud est visité par rapport au nombre moyen de visites de l'ensemble des nœuds, plus petit sera le nombre d'occurrence suffisant pour le déclenchement d'une boucle. Nous proposons que le nombre maximum de visites autorisées, noté $occ(x_i)$, soit défini par :

$$occ(x_i) = \lceil \theta(x_i) * (OccMax - OccMin) + OccMin \rceil \quad (2.11)$$

avec $\theta(x_i)$ calculé de la façon suivante :

$$\theta(x_i) = \min \left(\frac{\sum_{j \in [1;N]} nbVisites(x_j)}{nbVisites(x_i) \times N}; 1 \right) \quad (2.12)$$

La valeur de $\theta(x_i)$ varie entre 0 et 1. Ainsi, les paramètres $OccMax$ et $OccMin$ représentent les valeurs maximales et minimales du nombre d'occurrences $occ(x_i)$ de chaque nœud x_i .

2.2.4.1.1 Influence de la densité du graphe

Pour déterminer les valeurs $OccMin$ et $OccMax$, la relation entre le nombre le plus approprié de valeur seuil du nombre d'occurrences occ et les paramètres des instances traitées a été observée sur les résultats du tableau 2.11. Pour les instances de densité $\Delta = 10\%$, DSJC500.1 et DSJC1000.1, les meilleurs résultats ont été obtenus avec une valeur seuil occ égale à 5 correspondant à $\alpha = 2\%$ pour DSJC500.1 et $\alpha = 1\%$ pour DSJC1000.1. Les instances *DSJC* de densité $\Delta = 50\%$, DSJC500.5 et DSJC1000.5, obtiennent leur meilleurs résultats avec une valeur proche de 10. Finalement, les meilleurs résultats des instances *DSJC* de densité $\Delta = 90\%$ sont obtenus avec des valeurs plus grandes en nombre d'occurrences. A priori pour obtenir des solutions de meilleures qualités les graphes très denses requièrent plus de visites de chaque nœud avant de diversifier la recherche. La forte densité implique que les nœuds sont plus contraints et qu'il est plus

difficile de trouver les combinaisons de valeurs qui respectent les contraintes en jeu. En d'autres termes, la recherche doit être intensifiée plus durablement autour de ces nœuds afin de converger vers de meilleures solutions.

Cette observation peut être utilisée pour déterminer les valeurs de $OccMin$ et $OccMax$ en fonction de la densité du graphe, selon la formule suivante :

$$OccMin = D(\Delta) \text{ et } OccMax = 2 \times D(\Delta) \quad (2.13)$$

avec Δ la densité du graphe, et D une fonction calculée par la formule suivante :

$$D(x) = 25 \times x + 3, x \in [0; 1] \quad (2.14)$$

Les paramètres de cette fonction $D(x)$ ont été déterminés par rapport aux nombres d'occurrence ayant permis d'obtenir les meilleurs résultats reportés dans le tableau 2.11. La valeur 25 correspond à la plus grande valeur d'occurrence mais aussi à la valeur ayant obtenu les meilleurs résultats pour l'instance DSJC1000.9. Cette constante est donc dépendante des problèmes que nous avons choisis mais pourrait être déterminée automatiquement ultérieurement sur la base de la densité du graphe traité. La valeur 3 est le nombre minimum de visites d'un nœud avant de provoquer une boucle et est relativement indépendante des problèmes à traiter.

Pour les instances de 1 000 nœuds, on obtient donc les intervalles suivants : [5; 10] pour le problème DSJC1000.1, [15; 30] pour le problème DSJC1000.5 et enfin [25; 50] pour le problème DSJC1000.9. Pour les deux instances Leighton citées dans le tableau 2.11, on obtient le même intervalle [7; 14].

2.2.4.1.2 Influence du degré du nœud

Une autre hypothèse est de travailler sur le degré individuel de chaque nœud plutôt que sur la densité du graphe. Le tableau 2.12 présente quelques caractéristiques des instances *DSJC* et *Leighton*. Pour chaque instance, plusieurs informations sont données : le nombre de nœuds $N = |V|$, la densité du graphe Δ , le degré minimum d_{min} ainsi que le degré maximum d_{max} des nœuds dans le graphe. En fonction de ces informations, en utilisant l'équation (2.16) on calcule le ratio entre le degré d'un nœud et le nombre de nœuds du graphe. Ce ratio est noté δ_{min} pour le nœud de degré le plus faible et δ_{max} pour celui de plus grand degré. Les dernières colonnes présentent les valeurs de $D(\Delta)$, $D(\delta_{min})$, $D(\delta_{max})$ (voir équation (2.14)) et du nombre cible d'occurrences $bestNbOcc$ obtenu à partir des meilleurs résultats du tableau 2.11.

Pour les instances DSJC1000.*, la différence entre le plus grand et le plus petit nombre de nœuds voisins est très petite parce que les instances DSJC sont générées de façon aléatoire. En conséquence, l'écart-type entre les nombres de voisins $d(x_i)$ est très petit (égal à 9.6). Par opposition, le plus grand et le plus petit $d(x_i)$ pour les instances *Leighton* sont très différents. Par exemple, pour l'instance *le450_25c*, le nombre de voisins $d(x_i)$ varie entre 7 et 179, et l'écart-type est de 29.2. L'utilisation de la densité du graphe comme

TAB. 2.12 – Caractéristiques des instances DSJC et Leighton

problèmes	$ V $	Δ	d_{min}	d_{max}	$D(\Delta)$	$D(\delta_{min})$	$D(\delta_{max})$	bestNbOcc
DSJC1000.1	1000	0.0994	68	127	5	4	6	5
DSJC1000.5	1000	0.5002	447	551	15	14	16	10
DSJC1000.9	1000	0.8998	870	924	25	24	26	25
le450_25c	450	0.1717	7	179	7	3	12	9
le450_25d	450	0.1725	11	157	7	3	11	9

référence unique n'est pas appropriée pour ce type d'instance. L'utilisation du degré du nœud est plus adaptée pour ajuster le nombre d'occurrence à chaque nœud.

Pour la plupart des instances, le nombre cible du nombre d'occurrences *bestNbOcc* (dernière colonne du tableau 2.12) est inclus dans l'intervalle défini en fonction de l'équation (2.14). En prenant comme référence le degré de chaque nœud les bornes de l'intervalle $[OccMin; OccMax]$ sont calculées pour chaque nœud x_i en utilisant l'équation suivante :

$$OccMin(x_i) = D(\delta_i) \text{ et } OccMax(x_i) = 2 \times D(\delta_i) \quad (2.15)$$

avec δ_i , une fonction dépendante du degré $d(x_i)$ du nœud x_i calculée comme le ratio entre le degré $d(x_i)$ du nœud x_i et le nombre de nœuds du graphe, excepté le nœud x_i lui-même :

$$\delta_i = \frac{d(x_i)}{N - 1} \quad (2.16)$$

Par exemple, pour l'instance le450_25c, les nœuds de faible degré auront un intervalle du nombre d'occurrence de $[3; 6]$ alors que les nœuds de fort degré auront un intervalle de $[11; 24]$. Avec la densité du graphe, l'intervalle est de $[7; 14]$.

2.2.4.2 Résultat et analyse

2.2.4.2.1 Comparaison des résultats

Le tableau 2.13 présente les résultats obtenus sur les instances Leighton et *DSJC* avec quatre méthodes différentes. La première (colonne 2) correspond à la méthode DB+TA utilisant $\alpha = 5\%$. Les trois autres méthodes (notées DBA+TA) sont basées sur l'utilisation combinée de l'adaptation de la durée Tabou et de celle du paramètre de détection de boucle. La seule différence entre toutes ces méthodes est la définition de l'intervalle $[OccMin; OccMax]$: il est fixé dans un intervalle qui englobe les deux cas de réglage adaptatif (colonne 3), il est défini en fonction de la densité du graphe (colonne 4) ou en fonction du degré des nœuds (dernière colonne). Pour chaque méthode, deux critères sont utilisés : le taux de succès parmi les 5 exécutions réalisées ainsi que le nombre moyen de conflits de la meilleure solution au cours des 5 exécutions. Un code couleur est utilisé : les

cellules les plus foncées correspondent aux meilleurs résultats.

Premièrement, comparé aux résultats obtenus avec la méthode DB+TA (colonne 2), l'adaptation du paramètre de détection de boucle améliore ou égale la qualité des résultats pour 7 problèmes sur 8. En particulier lorsque le degré du nœud est utilisé comme paramètre, on remarque que les cellules de la dernière colonne sont plus foncées que les autres. Pour quatre des huit instances, la dernière colonne est meilleure que les autres méthodes et n'obtient jamais les pires résultats.

Pour les instances Leighton, les deux premières versions de la méthode DBA+TA (colonnes 3 et 4) dégradent une fois sur deux la qualité des solutions par rapport aux résultats de la méthode DB+TA (colonne 2). La dernière version (dernière colonne) permet de les améliorer. Pour les instances DSJC500.* et DSJC1000.* qui sont les problèmes les plus difficiles, la méthode DBA+TA utilisant le degré du nœud permet l'obtention de résultats proches en terme de qualité des meilleures valeurs obtenues avec les différents α testés (tableau 2.11). Globalement, l'utilisation du degré du nœud comme paramètre de l'adaptation de la détection de boucle à chaque problème permet d'améliorer les résultats précédents par rapport à n'importe quelle valeur fixe de α .

2.2.4.2.2 Répartition du nombre d'occurrence par nœud

Afin d'analyser les résultats présentés dans le tableau 2.13, les figures 2.13 et 2.14 présentent des spectrogrammes donnant la distribution des valeurs d'occurrences par nœud pour les trois méthodes DBA+TA pour les instances DSJC500.1 représentant un graphe aléatoire et le450_25d un graphe structuré. Ces figures ont en abscisse les nœuds classés par ordre croissant de degré et en ordonnée les différentes valeurs du nombre d'occurrence.

TAB. 2.13 – Influence de l'adaptation du paramètre de détection de boucle

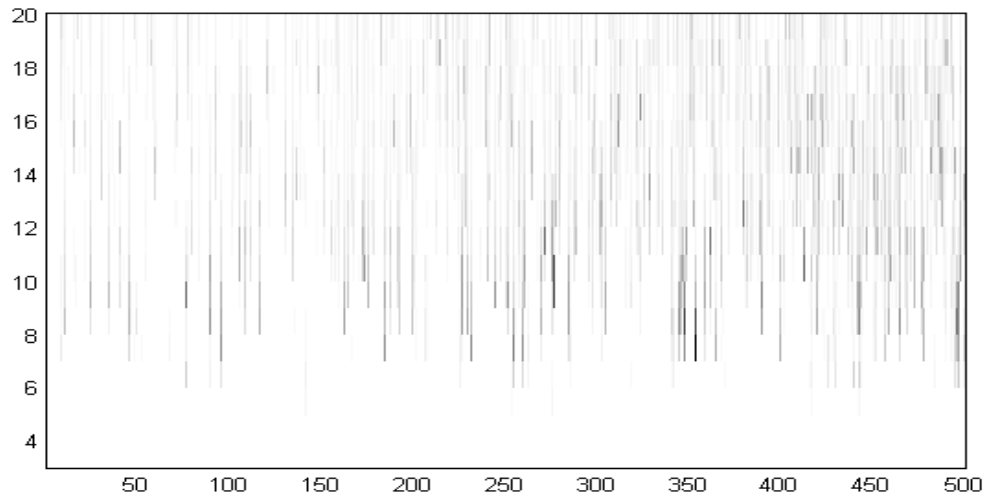
problèmes	DB+TA		DBA+TA					
	$occ = \lceil 5\% \times N \rceil$		$occ \in [3;20]$		$occ \in [D(\Delta); 2 \times D(\Delta)]$		$occ_i \in [D(\delta_i); 2 \times D(\delta_i)]$	
	s	c	s	c	s	c	s	c
DSJC500.1	6	0.4	0	2	6	0.4	8	0.2
DSJC500.5	0	4.2	0	4.4	0	9	0	6
DSJC500.9	0	3.4	0	3.6	0	1.4	0	1.8
DSJC1000.1	0	13.4	0	5.6	0	3.4	0	5.6
DSJC1000.5	0	42.8	0	29.6	0	39	0	37
DSJC1000.9	0	4	0	9	0	5.3	0	4
le450_25c	0	3	0	4.8	0	4	0	2.6
le450_25d	0	5.8	0	5.8	0	4.4	0	3.4

L'intensité de la couleur représente la proportion d'itérations pour lesquelles le nombre d'occurrence a été utilisé pour chaque nœud. Plus ce point est foncé, plus ce nombre d'occurrence a été utilisé pour le nœud correspondant. Sur ces deux figures, trois courbes sont présentées : la première correspond à la méthode utilisant un intervalle fixe, la seconde à l'utilisation de la densité du graphe et la troisième à l'utilisation du degré du nœud.

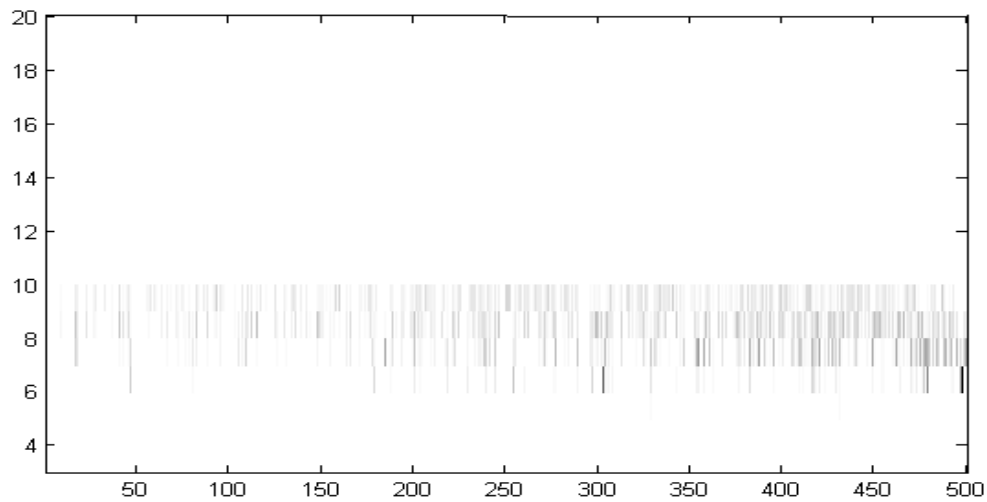
Les spectrogrammes de la figure 2.13 correspondent au problème DSJC500.1 résolu deux fois sur trois. Sur la figure 2.13(a), le nombre d'occurrences de chaque nœud varie de façon uniforme entre 3 et 20 (les bornes de l'intervalle). Les variations de la figure 2.13(b) sont plus limitées : le nombre d'occurrence oscille entre 6 et 10 pour tous les nœuds. On voit que les nœuds de degré plus fort (à droite) sont plus souvent en boucle (marquage plus sombre). Sur la dernière figure (c), l'utilisation du degré du nœud permet d'agrandir l'intervalle uniquement pour les nœuds de degré important, mais la densité du graphe étant de 10% l'accroissement de l'intervalle est assez faible. À l'inverse, cet intervalle est réduit pour les nœuds de plus faible degré. Pour la première moitié des nœuds (les nœuds de plus faible degré), le nombre d'occurrence est la plupart du temps fixé à la borne supérieure $OccMax$ de l'intervalle de valeur du nombre d'occurrence. Pour la seconde moitié des nœuds, la valeur la plus souvent utilisée est la limite basse $OccMin$. Les nœuds de degré fort sont plus souvent visités que les autres, ce qui entraîne le paramétrage à réduire le nombre d'occurrences recherchées pour les boucles, et vice versa pour les nœuds de faible degré. On remarque que l'intensité des couleurs est plus uniforme sur la figure (c) ce qui traduit un certain équilibre dans la distribution du choix des nœuds, ce qui n'est pas le cas sur la figure (b) où la méthode a aussi résolu le problème : il n'y a pas une mais plusieurs trajectoires pour trouver une solution optimale. À noter que dans les spectrogrammes (b) et (c), les bornes de l'intervalle $[OccMin; OccMax]$ sont semblables pour presque tous les nœuds ; cela illustre très bien la structure du graphe aléatoire.

La figure 2.14 est très différente de la figure 2.13. À cause de la structure du graphe, la plupart des nœuds de degré le plus faible (première moitié) sont très peu visités et ne provoquent jamais de boucle quelle que soit la méthode. En conséquence, ils apparaissent rarement sur les graphiques utilisés. Sur la figure 2.14(a), les nœuds de degré important utilisent en général de petites valeurs, des valeurs proches de la limite basse de l'intervalle, ce qui est le comportement attendu de la formule 2.11. La même observation peut être faite sur les figures (b) et (c). Sur les trois cas, la distribution des nombres d'occurrences utilisés par nœud n'est pas du tout uniforme en opposition aux figures 2.13. Seules quelques fréquences sont utilisées pour chaque nœud, ce qui donne un nuage de points dont le centre de gravité correspond aux limites basses des intervalles des nœuds de degré élevé. On voit bien sur ces graphes que l'algorithme insiste sur des nœuds précis.

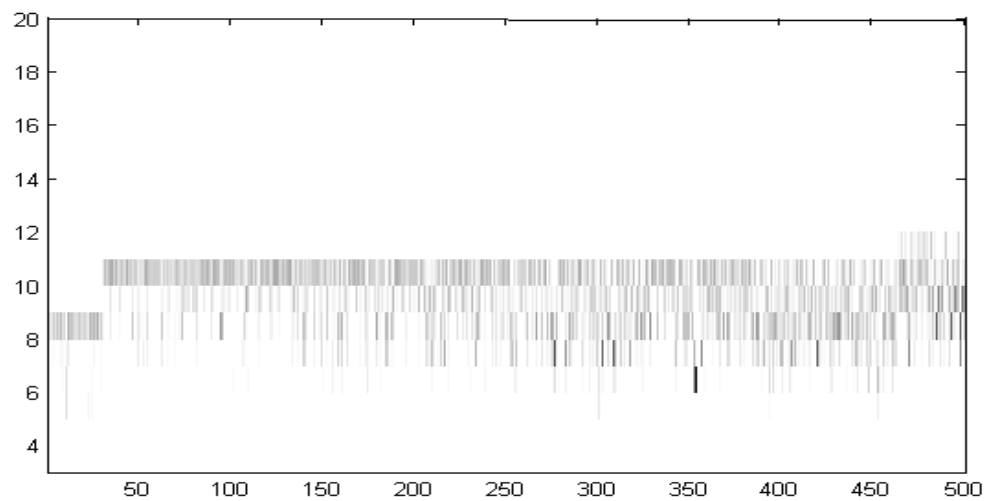
En figure 2.14(c), l'intervalle du nombre d'occurrence est très différent pour chaque nœud. Il reflète le degré du nœud ; les valeurs des bornes inférieure et supérieure sont plus grandes pour les nœuds de plus fort degré, d'où l'utilisation d'un nombre d'occurrence souvent plus grand pour les nœuds de plus fort degré par la troisième méthode. Ainsi, ces nœuds sont plus souvent visités avant de provoquer une boucle ce qui permet à la



(a) Intervalle fixe (non résolu)

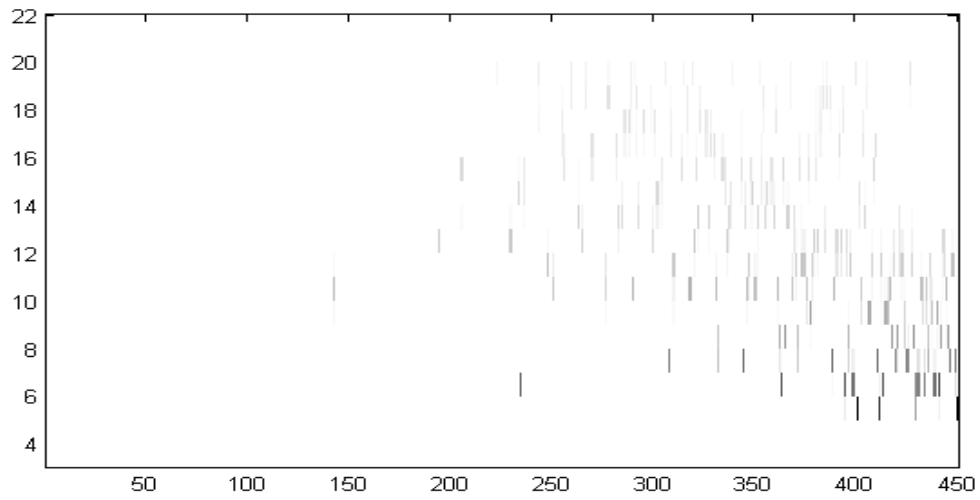


(b) Densité du graphe (résolu)

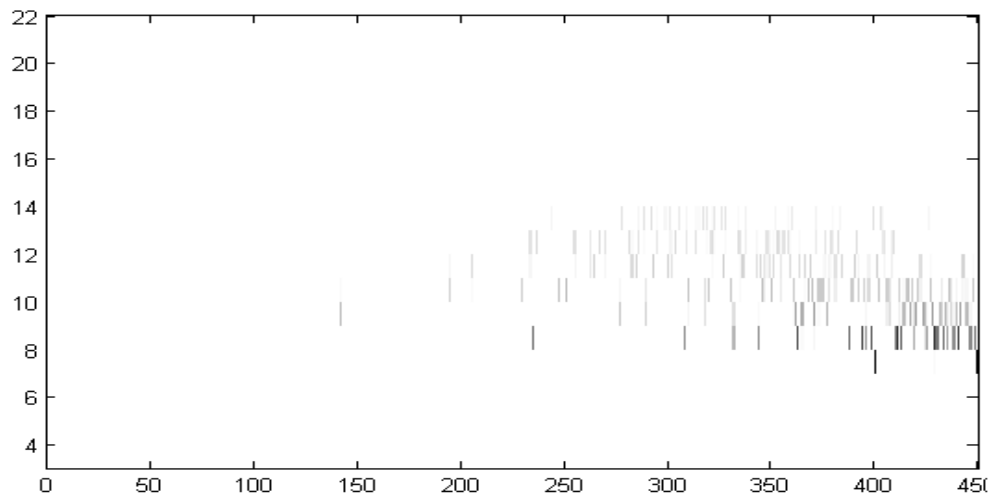


(c) Degré du nœud (résolu)

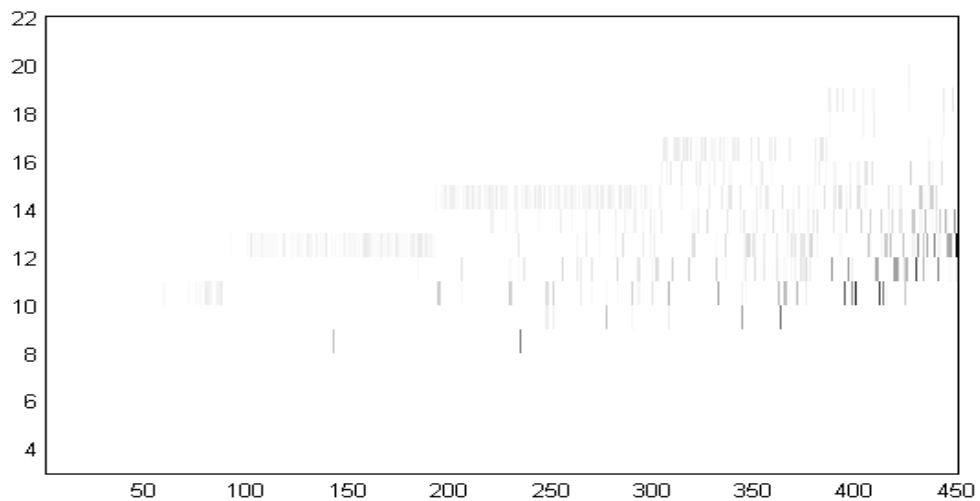
FIG. 2.13 – DSJC500.1 : Nombre d'occurrence par nœud pour les trois méthodes utilisant l'adaptation du paramètre de détection de boucle (instance parfois résolue)



(a) Intervalle fixe



(b) Densité du graphe



(c) Degré du nœud

FIG. 2.14 – le450_25d : Nombre d'occurrence par nœud pour les trois méthodes utilisant l'adaptation du paramètre de détection de boucle (instance non résolue)

méthode une période d'intensification autour de ces nœuds. Cette différence de fonctionnement entre ces méthodes se reflète sur le nombre de boucles détectées au cours de chaque exécution. Pour la première méthode, 22% des itérations sont de type "diversifiante", 20% pour la seconde et enfin 15% pour la dernière. Au final, la qualité de la solution obtenue avec la troisième méthode est meilleure que les deux autres ce que l'on peut attribuer au fait que l'algorithme passe plus de temps sur les nœuds identifiés "à problème" (ou conflictuels) avant de diversifier. Le comportement est véritablement adaptatif.

Pour conclure sur cette étude importante, pour les graphes structurés l'utilisation de la densité du graphe ou du degré des nœuds pour l'adaptation de la combinaison des méthodes de recherche a une très grande influence sur les résultats et sur le comportement de la méthode globale. Lorsque les bornes inférieure et supérieure sont définies individuellement pour chaque nœud l'adaptabilité de la méthode de combinaison est meilleure.

2.2.4.2.3 Étude de corrélation entre nombres de boucles et de visites

Tout comme pour l'adaptation de la durée Tabou, l'étude présentée ici examine la corrélation entre le nombre de boucles et le nombre de visites de chaque nœud suite à l'adaptation de la détection de boucle. Les résultats obtenus sur l'instance DSJC1000.5 avec l'adaptation du paramètre de détection de boucle ont été améliorés de façon significative en terme de nombre moyen de conflits restants. Cette amélioration est aussi analysée à travers le coefficient de corrélation. Les courbes des figures 2.15 et 2.16 présentent le nombre de visites et de boucles par nœud pour deux exécutions différentes de la méthode DBA+TA utilisant un intervalle fixe ([3; 20]).

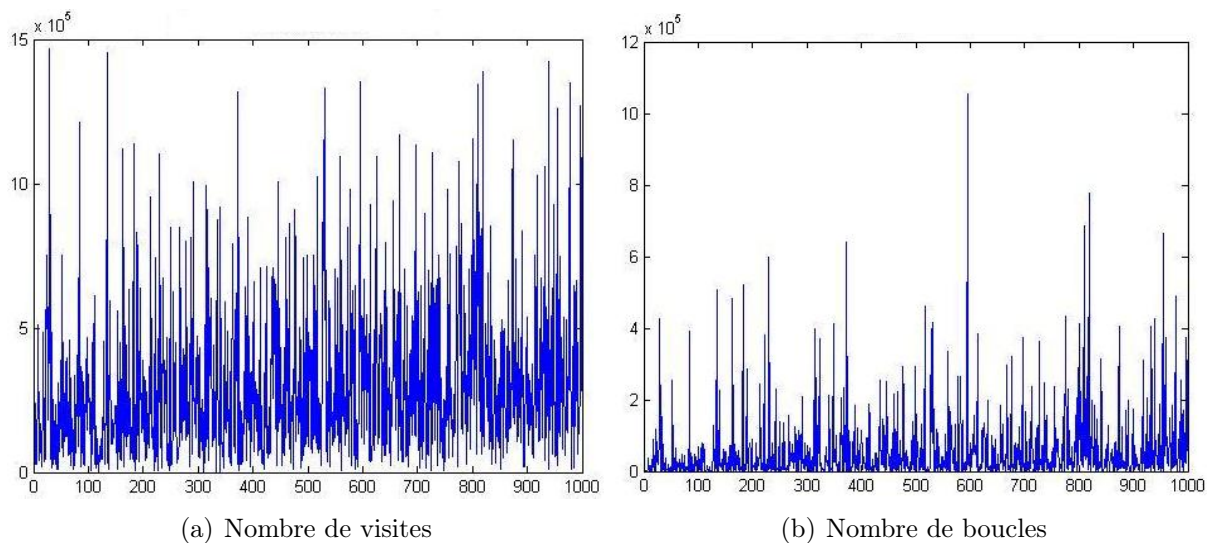


FIG. 2.15 – DSJC1000.5 : Corrélation entre nombre de visites et nombre de boucles par nœud par la méthode DBA+TA ($\text{corr}((a),(b)) = 0.8008$, évaluation=38)

La première exécution permet l'obtention d'une solution évaluée à 38 alors que la seconde permet d'obtenir une solution évaluée à 28. Le coefficient de corrélation est de 0.8008

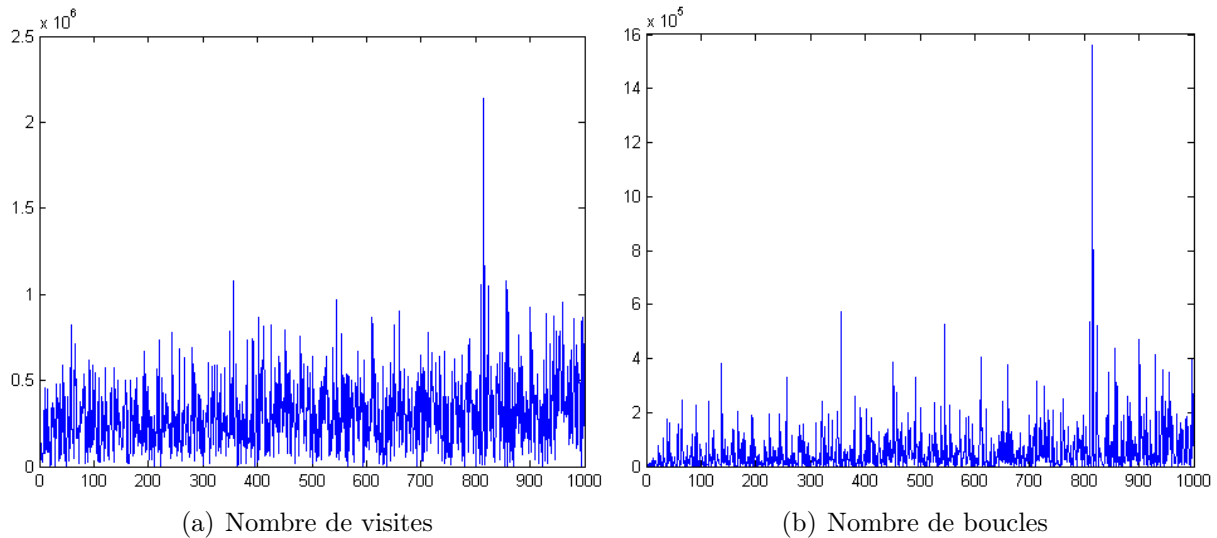


FIG. 2.16 – DSJC1000.5 : Corrélation entre nombre de visites et nombre de boucles par nœud par la méthode DBA+TA ($\text{corr}((a),(b)) = 0.7831$, évaluation=28)

pour la première et de 0.7831 pour la seconde. Cependant, nous observons visuellement que les courbes sont très différentes de celles des figures 2.9 et 2.11 pourtant proches en qualité de solutions (33 et 39 conflits). Cette observation révèle les limites de l'utilisation du coefficient de corrélation entre le nombre de visites et le nombre de boucles comme mesure de l'efficacité de la recherche. Bien que les courbes présentent une corrélation proche de 0.80 , on ne distingue pas le comportement de l'algorithme. Au contraire, il semble que le paramétrage adaptatif de la détection de boucle apporte un certain bruitage dans le comportement mais qui est bénéfique pour la recherche comme l'indique le tableau 2.13.

Une méthode de filtrage des données serait peut-être nécessaire avant d'étudier la corrélation pour éliminer les valeurs atypiques liées à certains nœuds, notamment les plus visités. Ce travail ouvre la voie à de nouvelles interrogations notamment sur les mécanismes de traitements de données et n'a pas été abordé durant la thèse.

2.2.4.2.4 Utilisation de la mémoire

Le coefficient de corrélation fournit uniquement une information sur les choix de nœuds effectués au cours de la recherche en analysant la relation entre le nombre de visites et le nombre de boucles par nœud. Analyser l'utilisation que fait la méthode de la mémoire à court terme qui contient l'ensemble des nœuds visités sur la fenêtre glissante pendant la recherche est instructif. Cette analyse permet de mieux comprendre l'influence de l'adaptation des deux paramètres de détection de boucle et de durée Tabou sur la recherche.

La figure 2.17 présente l'utilisation de la mémoire au cours d'exécutions non résolues sur l'instance DSJC1000.5. Les quatre figures représentent respectivement les quatre méthodes suivantes : la méthode de détection de boucle sans liste Tabou (courbe 2.17(a)),

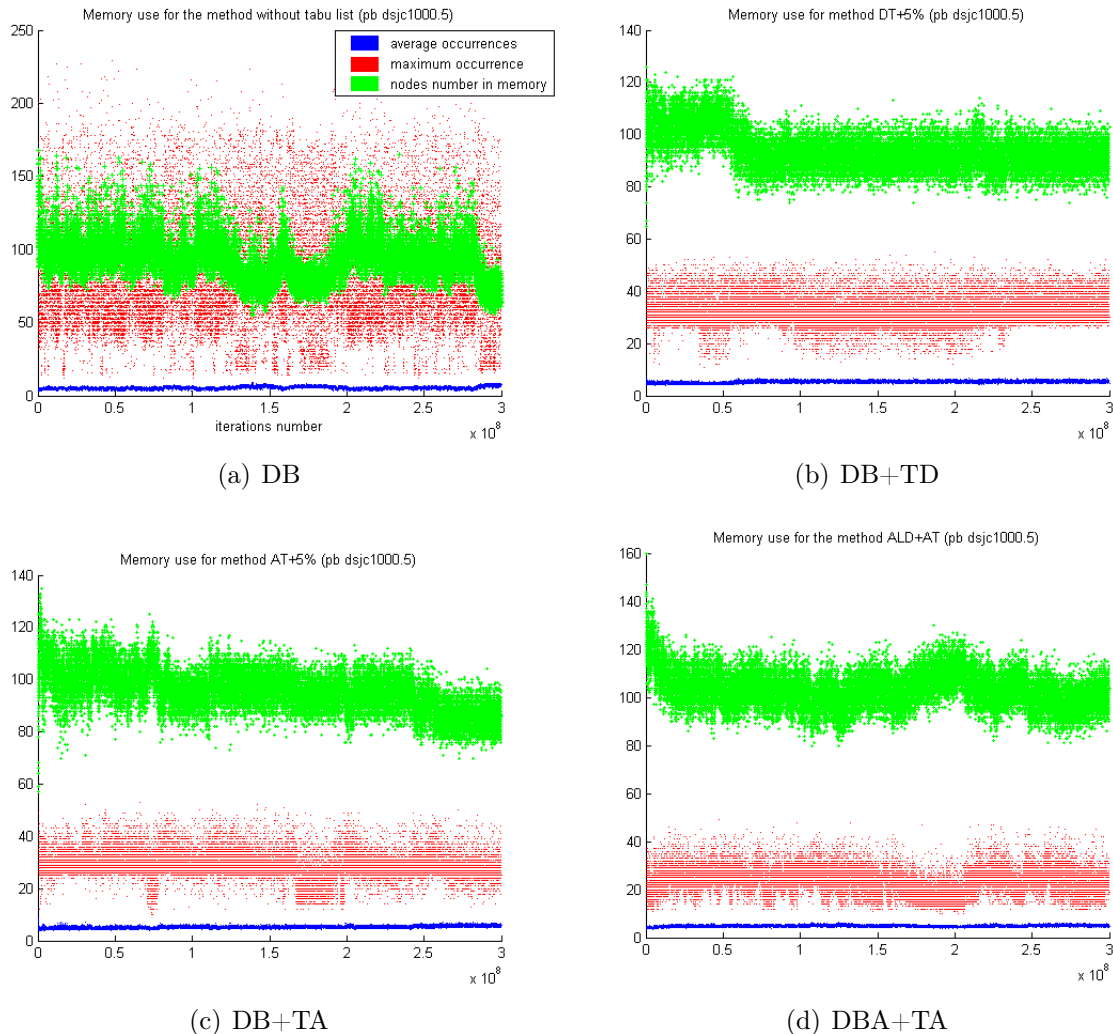


FIG. 2.17 – DSJC1000.5 : Utilisation de la mémoire par chacune des 4 méthodes

avec une liste Tabou dynamique (courbe 2.17(b)), avec une durée Tabou adaptative (courbe 2.17(c)) et enfin, la méthode utilisant les deux paramètres adaptatifs (courbe 2.17(d)). Sur ces figures, en abscisse est représenté le nombre d'itérations et l'évolution de trois critères est présentée en ordonnée ; ils correspondent au nombre moyen de visites de chaque nœud au sein de la fenêtre glissante (nuage de points du bas, couleur bleue), au nombre maximum de visites d'un nœud en mémoire (nuage de points du milieu, couleur rouge) et au nombre de nœuds présents dans la mémoire à chaque instant (nuage de points du haut, couleur verte).

Les variations du nombre maximum de visites d'un nœud dans la mémoire et du nombre de nœuds en mémoire sont liées. Lorsque le nombre de nœuds dans la mémoire diminue, le nombre d'apparitions du nœud le plus visité dans la fenêtre augmente. Avec la méthode sans liste Tabou (courbe 2.17(a)), quelques nœuds ont été visités plus de 200 fois lors des 500 dernières itérations, ce qui représente $2/5$ de la taille de la mémoire. En

même temps, il y a environ 100 nœuds différents dans la mémoire. La plupart d'entre eux sont visités très peu de fois. Le nœud le plus visité est le nœud qui a été le plus en conflit lors des dernières itérations. La détection de boucle n'empêche pas un nœud de revenir rapidement dans la fenêtre.

L'introduction d'une liste Tabou (courbes 2.17(b) et 2.17(c)) permet de réguler et de réduire considérablement le nombre de visites du nœud le plus visité. L'utilisation de la durée dynamique ou adaptative a la même influence sur le nombre de visites du nœud le plus visité. La différence entre ces deux types de durée Tabou est visible au niveau de l'évolution du nombre de nœuds en mémoire. Avec la durée dynamique, l'évolution de la recherche peut être divisée en deux parties. Lors de la première partie ($1/6$ de la recherche), le nombre de nœuds dans la mémoire varie entre 90 et 120 nœuds. Pendant la seconde partie, le nombre de nœuds décroît rapidement puis stagne entre 80 et 100 nœuds. Au contraire, dans le cas de l'utilisation de la durée adaptative, le nombre de nœuds décroît régulièrement au cours de la recherche. Il est de 120 nœuds environ au début de l'exécution, et autour de 80 vers la fin. Cette décroissance progressive du nombre de nœud en cours d'étude combinatoire profite à la recherche en référence aux tableaux 2.8 et 2.9 où la méthode DB+TA a amélioré tous les scores de DB+TD.

Avec l'adaptation des deux paramètres (courbe 2.17(d)), le nombre de visites du nœud le plus visité a diminué (on ne dépasse jamais 40) et ainsi, le nombre de nœuds en mémoire a augmenté. Les adaptations des paramètres de détection de boucle et de durée Tabou permettent à la méthode d'alterner continuellement les phases d'intensification et de diversification obtenues par la combinaison des deux recherches locales comme le montre l'oscillation des courbes. L'intensification se fait pendant une période de non détection de boucle plus longue pour les nœuds de plus fort degré ; dans ces périodes, la méthode utilise peu de nœuds mais les visite souvent. Puis ces nœuds deviennent tabous et la méthode diversifie sa recherche et augmente le nombre de nœuds visités tout en diminuant le nombre de visites par nœud. Ce comportement correspond à celui attendu.

2.2.4.2.5 Évolution des nœuds en conflit

Les figures précédentes montrent le fonctionnement global de la mémoire. Les figures 2.18 et 2.19 ci-après présentent l'évolution de l'ensemble des nœuds en conflit au cours des itérations pour les problèmes DSJC500.1 (exécution résolue) et DSJC500.5 (exécution non résolue). La méthode utilisée est la méthode DBA+TA basée sur la définition des bornes de l'intervalle $[OccMin; OccMax]$ en fonction du degré de chaque nœud. Chaque figure est composée de trois images correspondant à trois périodes différentes de la recherche : le début, le milieu et la fin. Chaque période est composée de 10 000 itérations. L'abscisse correspond aux itérations alors que l'ordonnée correspond aux différents nœuds classés de haut en bas par ordre décroissant de degré. Les nœuds correspondants aux valeurs les plus basses de l'axe des ordonnées sont les nœuds de plus faible degré. Chaque point visible de coordonnées (it, x) correspond à : le nœud x est en conflit à l'itération it .

Ces figures montrent que l'ensemble des nœuds en conflit varie au cours du temps.

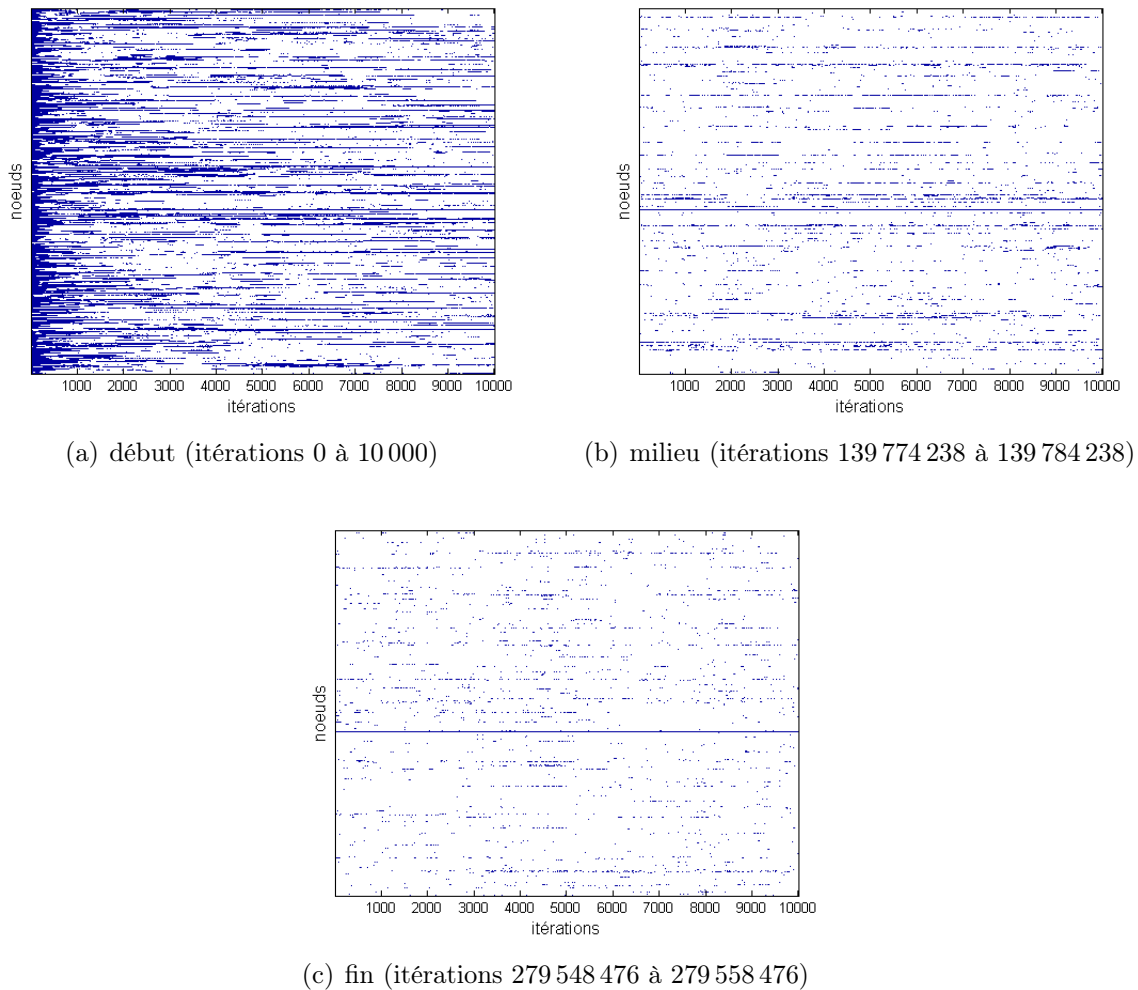


FIG. 2.18 – DSJC500.1 : Évolution des nœuds en conflit à différents stades de la recherche (instance résolue)

Pour les deux problèmes étudiés, la première période (figures 2.18(a) et 2.19(a)) est la période pendant laquelle il existe le plus de nœuds en conflit. Sur les figures 2.18(b) et 2.18(c), on observe que la plupart des nœuds en conflit le sont pour de courtes périodes. Seuls quelques-uns restent en conflit durablement. On observe un comportement différent sur les figures 2.19(b) et 2.19(c) où le nombre de nœuds durablement en conflit est plus grand, de l'ordre d'une vingtaine en fin d'exécution (5% des nœuds). L'ensemble des nœuds en conflit semble se figer au cours de l'exécution. Dans le cas DSJC500.1, la méthode a trouvé la solution optimale lorsqu'elle a résolu les conflits générés par le dernier nœud durablement en conflit. Dans le cas du problème DSJC500.5, le nombre de nœuds en conflit pour de longues périodes est plus important, la méthode basée sur des voisinages simples nécessiterait probablement des voisinages plus agressifs, c'est à dire de portée plus importante, pour casser des structures stabilisées de cette taille.

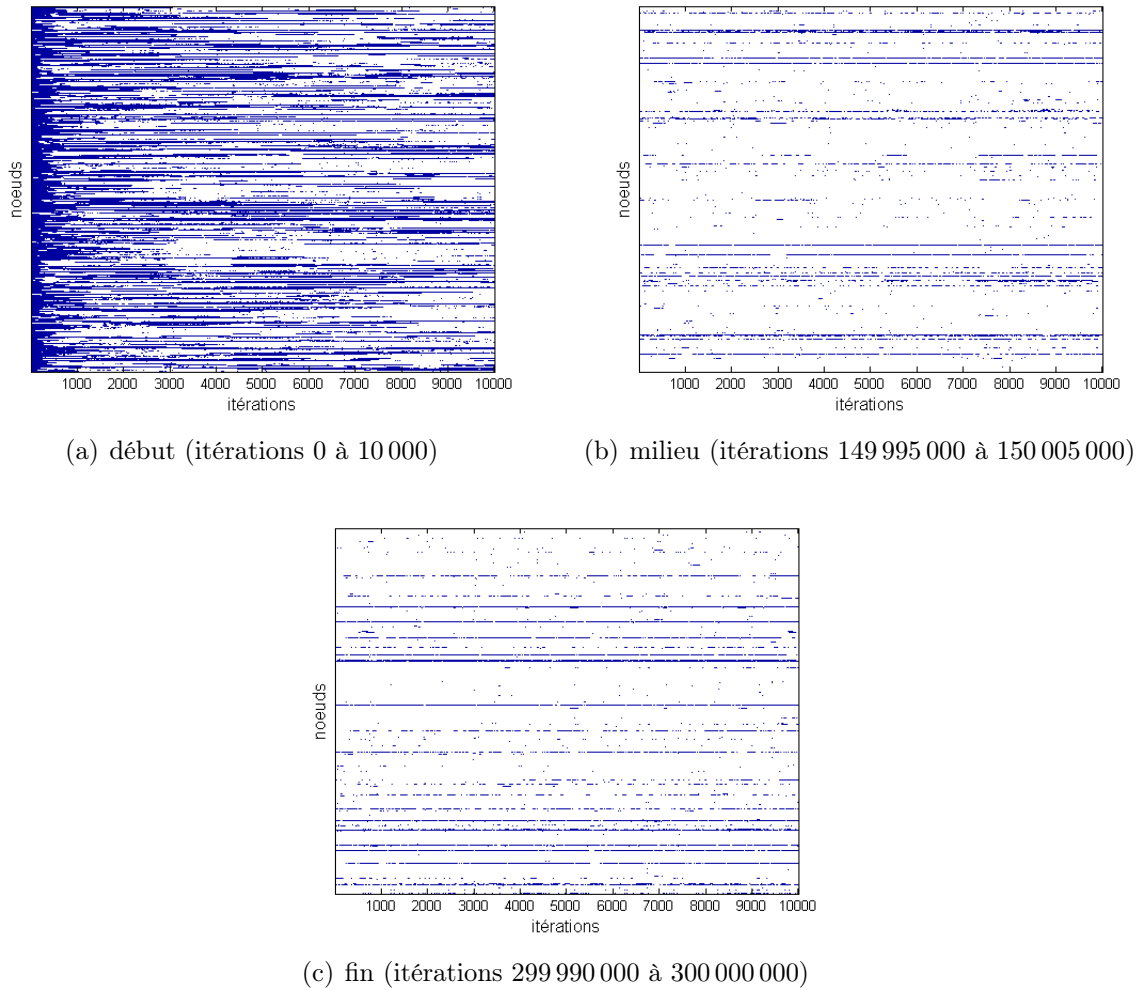


FIG. 2.19 – DSJC500.5 : Évolution des nœuds en conflit à différents stades de la recherche (instance non résolue)

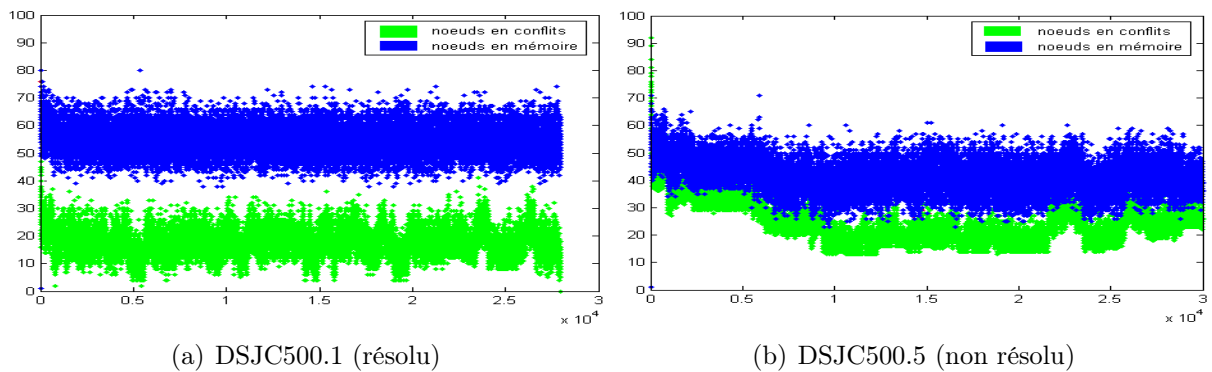


FIG. 2.20 – Évolution du nombre de nœuds en conflit par rapport au nombre de nœuds en mémoire au cours de la recherche

Afin d'étudier plus précisément les nœuds en conflit, les figures 2.20 présentent l'évolution du nombre de nœuds en conflit (nuage du bas) comparé au nombre de nœuds en mémoire (nuage du haut) lors d'une exécution résolue de l'instance DSJC500.1 (figure 2.20(a)) et pour une exécution non résolue de l'instance DSJC500.5 (figure 2.20(b)) de la méthode DBA+TA utilisant le degré du nœud. Les figures montrent que le nombre de nœuds en conflit est toujours plus petit que le nombre de nœuds en mémoire cependant dans des proportions différentes pour les deux problèmes. On rappelle que seuls les nœuds en conflit sont choisis et donc figurent en mémoire.

Un plus grand nombre de nœuds en conflit différents passe en mémoire pour DSJC500.1 (de 40 à 70) et en même temps le nombre de nœuds en conflit à chaque instant oscille entre 5 et 30. Pour DSJC500.5 le nombre de nœuds en conflit varie de 15 à 30, donc autant si ce n'est plus que pour l'autre problème, mais la quantité de ces nœuds est très stable d'une itération à l'autre (faible écart de l'intervalle) et la variété de ces nœuds est plus restreinte (30 à 55 nœuds différents en mémoire). Au final, il y a plus de nœuds en conflit dans l'exécution de DSJC500.5 et cet ensemble est relativement figé, contrairement à l'autre problème, ce qui est corrélé avec les observations des figures 2.18 et 2.19.

2.3 Comparaison des résultats

Cette partie compare notre méthode adaptative et quatre méthodes dédiées à la coloration de graphes publiées dans la littérature. Toutes ces méthodes sont basées sur l'utilisation de classes de couleur représentant l'ensemble des nœuds de même couleur et peuvent donc agir globalement sur un ensemble de nœuds. Le tableau 2.14 présente le nombre minimum de couleurs nécessaire à la résolution de chaque instance pour chacune des six méthodes présentées :

- La première méthode est la méthode AMACOL publiée par Galinier *et al.* [61]. Cette méthode représente un des meilleurs algorithmes dans la littérature.
- La seconde méthode est une recherche locale réitérée (*Iterated Local Search* ou ILS) présentée par Chiarandini *et al.* [29].
- La troisième méthode est nommée recherche locale guidée (*Guided Local Search* GLS) proposée par Chiarandini [27].
- La dernière méthode est la méthode de Recherche Tabou générique (*Generic Tabu Search* GTS) publiée par Dorne et Hao [46].
- Les deux dernières colonnes présentent nos méthodes : premièrement, la méthode sans adaptation utilisant une liste Tabou dynamique (DB+TD) avec $\alpha = 5\%$ et $f(N) = \sqrt{N}/2$ [113] et ensuite, la méthode combinant adaptation de la durée Tabou et de la détection de boucle utilisant le degré de chaque nœud (DBA+TA). Chaque méthode a été lancée 5 fois pour des exécutions de 300 millions d'itérations chacune.

TAB. 2.14 – Comparaison avec la littérature

problèmes	(χ, k^*)	AMACOL	ILS	GLS	GTS	DB+TD	DBA+TA
DSJC125.1	(-,5)	5	5	5	5	5	5
DSJC125.5	(-,17)	17	17	18	17	17	17
DSJC125.9	(-,30)	44	44	44	44	44	44
DSJC250.1	(-,8)	8	8	8	8	8	8
DSJC250.5	(-,22)	28	28	29	28	28	28
DSJC250.9	(-,72)	72	72	72	72	72	72
DSJC500.1	(-,12)	12	13	13	13	13	12
DSJC500.5	(-,48)	48	50	52	50	50	49
DSJC500.9	(-,126)	126	127	129	127	128	127
DSJC1000.1	(-,20)	20	21	21	21	21	21
DSJC1000.5	(-,83)	84	90	93	90	89	89
DSJC1000.9	(-,224)	224	227	233	226	230	226
flat300_20_0	(20,20)	20	20	20	20	20	20
flat300_26_0	(26,26)	26	26	33	26	26	26
flat300_28_0	(28,28)	31	31	33	31	31	31
flat1000_50_0	(50,50)	50	88	50	50	88	88
flat1000_60_0	(60,60)	60	89	90	60	88	87
flat1000_76_0	(76,83)	84	89	92	89	88	88
le450_5a	(5,5)	5	5	5	-	5	5
le450_5b	(5,5)	5	5	5	-	5	5
le450_5d	(5,5)	5	5	5	-	5	5
le450_15a	(15,15)	15	15	15	-	15	15
le450_15b	(15,15)	15	15	15	-	15	15
le450_15c	(15,15)	15	15	15	-	16	15
le450_15d	(15,15)	15	15	15	-	16	15
le450_25c	(25,25)	26	26	26	-	26	26
le450_25d	(25,25)	26	26	26	-	26	26

La deuxième colonne du tableau 2.14 indique le nombre chromatique χ lorsqu'il est connu et le meilleur résultat connu dans la littérature k^* pour chaque instance [29]. Toutes les instances présentées dans ce tableau sont des problèmes connus comme difficiles [27]. En ce qui concerne nos deux méthodes, la méthode DBA+TA obtient de meilleurs résul-

tats que la méthode DB+TD. L'adaptation de la détection de boucle et de la durée Tabou permet d'améliorer la qualité des résultats pour sept instances.

La méthode AMACOL est la meilleure méthode présentée ici. Les trois méthodes GTS, ILS et DBA+TA présentent des résultats identiques à la méthode AMACOL pour les six premiers problèmes et pour les neuf problèmes Leighton. En dépit du caractère générique de notre méthode, les résultats obtenus sont très compétitifs avec les quatre méthodes présentées. Sur l'ensemble des problèmes, la méthode DBA+TA se classe 2^e en dominant les 3 autres méthodes de recherche locale utilisant tout comme elle une seule solution et des mécanismes adaptatifs pour explorer l'espace de recherche et sortir des minimums locaux.

2.4 Conclusion

Le problème de coloration de graphe est un problème qui a énormément de références théoriques et applicatives et qui de ce fait constitue une bonne plate-forme de mise au point de méthode de recherche. Les diverses études sur la coloration ont donné lieu à la définition d'un grand nombre de variantes du problème et nous avons seulement travaillé sur la k -coloration où k est le nombre de couleurs imposé pour l'exercice. Dans un premier temps, notre travail a été mené sur les instances CNET, puis ayant résolu ces premières instances, nous nous sommes orientés vers les instances DIMACS plus difficiles et plus référencées. La première partie de ce chapitre présente l'ensemble de ces instances et donne des références de résultats avec plusieurs méthodes puis se focalise sur des structures de voisinage de base en recherche locale pour la coloration.

Dans la structure de voisinage basée sur les nœuds en conflit nous avons testé cinq types de recherche locale pour lesquelles nous avons montré qu'aucune ne résout tous les problèmes CNET mais sont assez complémentaire sur leurs performances. Suite à ce travail initial, nous avons proposé la conception d'une méthode générale et déterministe de combinaison de recherches locales dont l'objectif serait d'utiliser à bon escient les performances des deux méthodes selon les propriétés découvertes en cours de processus. Le problème qui se pose, et qui a été bien identifié dans l'état de l'art, est de définir les moyens de décider de l'utilisation d'une recherche locale ou d'une autre au cours de la recherche en fonction de l'historique de celle-ci. La méthode que nous avons mise au point s'appelle par la suite *Recherche Locale Adaptative*, notée RLA. Elle comprend trois éléments essentiels : deux recherches locales de base à combiner, une mémoire de gestion de boucle qui permettra de prendre les décisions de combinaison et une liste Tabou qui gèrera l'accès aux variables. Il y a deux hypothèses fortes dans cette conception mais qui pourront être levées par la suite : seulement deux méthodes sont combinées et elles sont toutes deux dans la même structure de voisinage. Elles définissent chacune une règle de restriction ou d'extension des voisins accessibles à l'intérieur cette structure.

La méthode de combinaison présentée repose principalement sur un mécanisme de détection de boucle qui a pour objectif de permettre l'adaptation de l'algorithme en com-

binant les deux recherches locales choisies. Nous avons défini une boucle comme étant la répétition excessive du choix d'une variable au cours de l'exécution. La détection de boucle ne permet pas donc pas de reconnaître des séquences de choix successifs. Cette définition de boucle est le résultat de l'analyse comportementale effectuée sur les méthodes séparées utilisées avec les problèmes CNET. En effet, lors de l'incapacité d'une méthode à atteindre une solution réalisable, le nombre de visites des nœuds devient très inégal d'un nœud à un autre. Cette inégalité est le résultat d'une mauvaise alternance des phases de diversification et d'intensification de la recherche. Lorsque une répétition est constatée, une méthode *diversifiante* qui élargie le voisinage est appliquée sinon on utilise une méthode *intensifiante* et *déterministe* car elle travaille toujours avec le nœud le plus en conflit. Cette combinaison de méthodes simples via la détection de boucle a amélioré tous les résultats obtenus séparément par les méthodes : tous les problèmes CNET ont été résolus et tous les problèmes DIMACS ont été améliorés que ce soit en nombre de résolution ou en nombre de conflits restants.

Nous avons observé que les variables détectées en boucle l'étaient durablement malgré les changements de voisinage opérés par la méthode. Nous avons donc introduit une liste Tabou ; celle-ci permet d'empêcher la méthode de choisir trop rapidement un nœud ayant déclenché une boucle. Le statut Tabou a aussitôt permis de réduire le nombre de boucles détectées, cependant la difficulté de l'ajout de la liste Tabou a été le paramétrage général de la durée Tabou pour obtenir de bonnes performances sur les problèmes. Pour cela, des durées statiques et dynamiques ont été essayées de façon empirique avec des alternatives sur le statut Tabou : toutes les variables utilisées sont mises Tabou ou seules les variables détectées en boucle sont mises Tabou. En travaillant sur les problèmes DIMACS nous avons clairement identifié un net avantage pour la mise en statut Tabou des seuls nœuds détectés en boucle. On a donc un apport de la combinaison des deux principes généraux : identifier les variables en boucles pour les interdire durablement et en même temps changer de recherche locale.

Suite à ces travaux sur l'adaptation des mécanismes de la recherche, nous avons introduit l'adaptation automatique des paramètres internes utilisées séparément par la détection de boucle et la liste Tabou. Il s'agit des principaux défauts identifiés dans la littérature où les valeurs de ces paramètres sont fixées au cas par cas. Nous avons opté pour une adaptation continue effectuée grâce aux informations collectées pendant la recherche. Les paramètres calculés au fur et à mesure sont la durée du statut Tabou pour chaque variable en boucle et le nombre d'occurrence à rechercher pour chaque variable pour détecter une boucle. Pour ce travail, nous avons analysé en profondeur le comportement de la méthode RLA avec des outils statistiques. Des spectrogrammes ont permis l'analyse des répétitions des durées Tabou utilisées variable par variable sur plusieurs exécutions complètes. Nous avons corrélé ces données avec le suivi du nombre de visites et du nombre de boucles par variables selon leur degré. A ce stade l'adaptation est alors effectuée en fonction d'historiques de recherche propres à chaque variable de décision. Ces analyses ont été faites sur plusieurs catégories de problèmes DIMACS (Leighton, DSJC, Flat) afin de définir un comportement général. Ce travail nous a permis de paramétrer dynamique-

ment les phases d'intensification et de diversification de la recherche par problème et pour chaque variable. Tous les résultats que nous avons obtenus auparavant sur les instances DIMACS ont été égalés ou améliorés en éliminant le paramétrage spécifique. En dernier lieu nous avons comparé nos scores avec quatre autres méthodes de la littérature ayant publié sur ces problèmes dont trois recherches locales adaptatives.

Bien que l'adaptation ait grandement amélioré les performances de la méthode, une étude de l'évolution de l'ensemble des nœuds en conflit sur une instance résolue et une autre non résolue a montré qu'il existe, dans les deux cas, un sous-ensemble de nœuds stratégiques composé des nœuds en conflit pour de longues périodes. Lorsque ce sous-ensemble comporte un grand nombre de nœuds, la portée de la structure de voisinage simple que nous avons utilisée semble mal adaptée. Dans les perspectives, nous envisageons d'étudier l'intérêt d'utiliser des structures de voisinage de portée plus importante avec la détection de boucle. Le second point qui mérite de nouvelles études est la taille de la fenêtre d'observation. Nous avons défini cette fenêtre en fonction du nombre de nœuds. Cependant, pour les problèmes de grande taille, de l'ordre de 1000 nœuds, la taille de la fenêtre semble inappropriée car trop grande et conduit à une dispersion des nœuds en conflit.

Chapitre 3

Recherche Locale Adaptative et affectation de fréquences

Le problème d'affectation de fréquences pour les systèmes de radiocommunications fonctionnant en évation de fréquences est un nouveau problème exposé en 2006 lors de la conférence nationale française sur la recherche opérationnelle et l'aide à la décision [36]. Dans un premier temps, la formulation détaillée du problème est décrite en première partie. Ensuite, les quelques méthodes proposées dans la littérature concernant la résolution de problèmes d'affectation de listes de couleurs ou de fréquences sont détaillées en seconde partie. Dans la troisième partie, nous présentons la méthode de recherche locale adaptative utilisée pour résoudre ce problème. Puis, une quatrième partie de résultats et d'analyses permet de comparer notre méthode aux résultats de deux autres équipes dans le cadre d'un contrat de recherche avec le CELAR. Nous donnons aussi des résultats sur des instances publiques plus récentes. Une partie de ce travail a été publiée dans la conférence *INOC 2006*.

Sommaire

3.1	Formalisation du problème	102
3.1.1	Description physique du problème	102
3.1.2	Les contraintes du problème	105
3.1.3	Les objectifs à optimiser	111
3.1.4	Les instances	113
3.2	Modèles de référence et méthodes	117
3.3	Application de la méthode RLA	120
3.3.1	Schéma général de la méthode	120
3.3.2	La solution initiale	122
3.3.3	Le processus itératif	127
3.4	Résultats et analyse sur les scénarios	146
3.4.1	Scénarios publics	147
3.4.2	Scénarios privés	148
3.5	Conclusion	150

3.1 Formalisation du problème

Les postes radio militaires fonctionnant en saut de fréquence posent de nouveaux problèmes d'allocation de fréquences qui doivent prendre en compte la taille du déploiement, les ressources limitées en fréquence mais aussi des phénomènes d'interférence qui mettent en jeu des listes de fréquences par émetteur.

Cette partie présente en détail le problème d'affectation de plans de fréquences avec évitement de fréquences tel qu'il a été défini par le CELAR¹. Elle s'appuie sur un rapport interne du projet et décrit plus succinctement dans [36]. Pour commencer, une description physique du problème est présentée. Ensuite, les contraintes du problème sont détaillées suivies des différents objectifs à optimiser. Finalement, les différentes caractéristiques des instances privées et publiques que nous avons utilisées sont analysées.

3.1.1 Description physique du problème

Pour les besoins militaires, des communications doivent être transmises entre différents groupes de véhicules transportant des émetteur-récepteurs aussi appelés "postes de radiocommunications". Ces postes sont regroupés en réseaux et sont reliés par un même canal de fréquence appelé "canal radio". Le schéma 3.1 illustre un exemple de déploiement de véhicules équipés de 1 à 3 postes. Cette figure présente un déploiement de 5 réseaux, notés r_i , et chaque réseau dispose de plusieurs postes; par exemple, le réseau r_1 possède 3 postes dont l'un est co-localisé sur un véhicule avec le réseau r_3 , un autre est co-localisé sur un véhicule avec les réseaux r_2 et r_4 et le 3^e est seul.

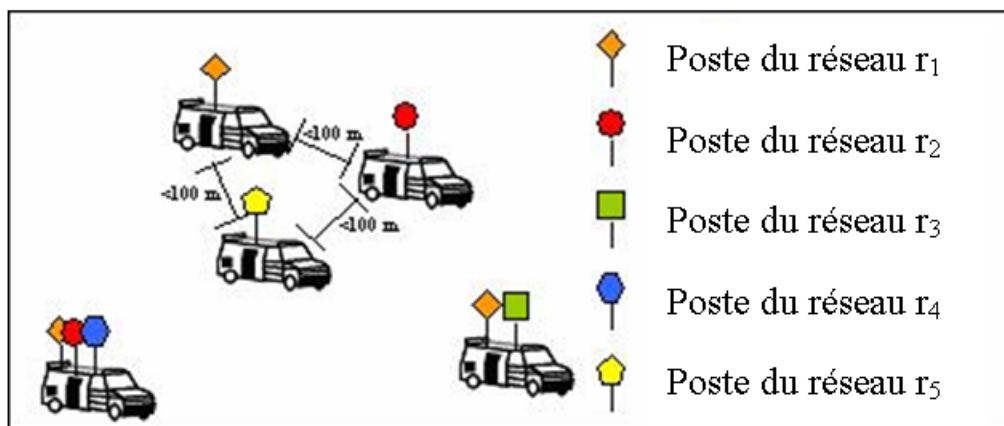


FIG. 3.1 – Exemple de déploiement de cinq réseaux

Plusieurs types de communications sont transmis sur ces postes. Le premier est de type "conférence" : les communications émises par un poste sont transmises à l'ensemble des postes appartenant au même réseau. Le second est de type "point à point" : seul un

¹Centre électronique de l'armement/Délégation générale de l'armement

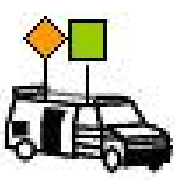

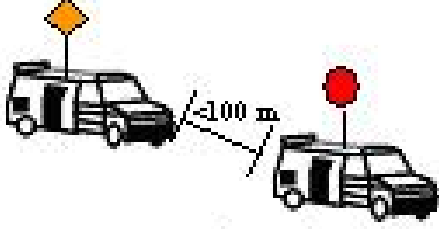
poste destinataire reçoit la communication du poste émetteur.

La qualité des communications est alors dictée par les différentes situations d'interférences produites par les postes radio. Nous distinguons deux types de situations :

- Les perturbations "co-véhicule" naissent de la présence de plusieurs émetteurs-récepteurs sur le même véhicule. Deux cas sont possibles :
 - perturbation "biposte" où seulement deux émetteurs-récepteurs sont situés sur le même véhicule,
 - perturbation "triposte" où trois émetteurs-récepteurs de réseaux différents sont installés sur un même véhicule.
- Les perturbations "co-site" sont générées par la présence de deux postes l'un à proximité de l'autre (par exemple, distants de moins de 100 mètres).

Pour chaque type de perturbation, on définit l'ensemble des postes concernés par la notion de "lien". Un lien est donc un ensemble de postes appartenant à différents réseaux susceptibles de se brouiller. Le tableau 3.1 donne une représentation graphique des trois types de liens définis dans le problème. Un risque d'interférence biposte, triposte et cosite entre les postes radios est modélisé par un lien entre les réseaux auxquels appartiennent les différents postes.

TAB. 3.1 – Représentation graphique des perturbations et des liens associés

perturbation "co-véhicule"		perturbation "co-site"
lien "biposte"	lien "triposte"	lien "cosite"
		

Ces perturbations nécessitent d'établir une contrainte de compatibilité électromagnétique entre les fréquences allouées aux réseaux auxquels les postes radio appartiennent de façon à éviter les brouillages. D'un problème constitué de véhicules, de postes et de réseaux, on passe ainsi à un problème d'optimisation composé de contraintes co-véhicule et co-site entre les réseaux auxquels il faut allouer des plans de fréquences en minimisant une fonction de calcul d'interférence.

La particularité de ce système est que les postes fonctionnent en "évasion de fréquences" (EVF). Une communication en évasion de fréquences n'est pas transmise sur une

seule fréquence mais sur une liste de fréquences, appelée "plan de fréquences". Chaque fréquence est utilisée pendant une très courte période et au cours d'une communication l'ensemble des fréquences de la liste est utilisée. Dans le domaine des radiocommunications, ce principe est généralement appelé "saut de fréquences" ou "frequency hopping" [96]. Le changement de fréquences se fait de façon pseudo-aléatoire par un algorithme propriétaire au système dans l'ensemble de fréquences alloué au réseau. Le rythme de changement de fréquences est en général assez élevé (de plusieurs dizaines à plusieurs centaines de fois par seconde) permettant à chaque fréquence de ne transmettre qu'une petite partie de l'information (quelques dizaines à quelques centaines de bits). Ainsi, tous les postes d'un même réseau changent périodiquement de fréquence et de façon synchronisée de façon à suivre la même séquence de fréquences. La figure 3.2 schématise le fonctionnement du saut de fréquences. Sur la période affichée, la séquence de saut est définie par les fréquences $F_4, F_6, F_5, F_2, F_3 \dots$

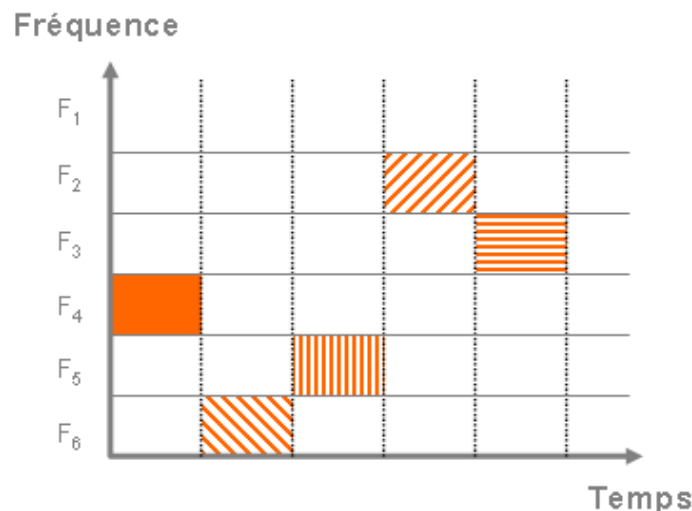


FIG. 3.2 – Changement de fréquences au cours du temps

Du fait de l'affectation d'une liste de fréquences par réseau, par rapport à un problème en fréquence fixe la combinatoire du problème est donc fortement amplifiée. Par ailleurs, les interférences entre deux réseaux ne s'expriment plus en terme d'écart entre fréquences comme dans les systèmes à fréquences fixes, mais elles sont le résultat de l'ensemble des fréquences utilisées par chaque réseau. Ainsi, on définira la qualité des communications non pas par le respect des contraintes d'écart entre fréquences mais par un niveau d'erreur binaire (noté TEB pour *Taux d'Erreur Binaire* ou BER pour *Bit Error Rate*) qui résulte notamment du nombre de fréquences communes entre les listes attribuées aux réseaux appartenant aux mêmes liens. Ce taux est utilisé pour quantifier le nombre d'erreurs observés à la réception d'une transmission numérique. En d'autres termes, il mesure le nombre de bits erronés du message reçu par rapport au message émis.

Par la suite, étant donné le fonctionnement en évation de fréquences, les contraintes

co-véhicule et co-site sur un réseau sont évaluées par un niveau d'interférence calculé sur l'ensemble des liens du réseau en question. Ce niveau d'interférence ne doit pas dépasser un seuil maximal d'interférence autorisé par réseau et exprimé en TEB. Au niveau de chaque lien, les interférences sont évaluées par la moyenne des brouillages calculée pour l'ensemble des couples (ou triplets pour le cas triposte) formés de chaque fréquence du réseau en question et de chaque fréquence du ou des réseaux brouilleurs.

D'autre part pour des aspects de résistance au brouillage et à l'écoute, les plans de fréquences doivent aussi être construits en fonction de critères qui leur sont propres tels qu'utiliser un maximum de fréquences du spectre et favoriser la réutilisation de fréquences entre les plans. Ils sont aussi assujettis à des critères utilisés en fréquences fixes tel que le fait que chaque réseau ne peut utiliser qu'un domaine restreint du spectre et que les couples de fréquences peuvent être soumis à des contraintes d'écartement en plus des contraintes de TEB.

3.1.2 Les contraintes du problème

Les contraintes du problème peuvent être regroupées en trois ensembles fonctionnels : les contraintes liées aux réseaux, les contraintes sur les plans de fréquences et celles liées aux interférences.

3.1.2.1 Les contraintes liées aux réseaux

Il existe deux types de réseaux : des réseaux dont les plans de fréquences sont déjà alloués et des réseaux dont les plans de fréquences doivent être alloués. Les réseaux déjà alloués ne peuvent être modifiés, ils sont appelés "réseaux hors-cible", noté R^{hc} . Seuls les plans de fréquences non alloués doivent être alloués et les réseaux concernés sont appelés "réseaux cible", noté R^c . Tous les réseaux, cibles et hors-cibles, sont pris en compte pour l'évaluation des interférences du problème, ce qui rend les problèmes plus contraints puisque certaines variables sont déjà affectées.

Chaque réseau $r \in R = R^c \cup R^{hc}$ est caractérisé par différentes données :

- un niveau hiérarchique h_r représentant son niveau de priorité,
- la taille minimale NF_r^{min} du plan de fréquences à allouer à ce réseau ; la taille d'un plan de fréquences est le nombre de fréquences utilisées par ce plan.
- le seuil maximal de TEB du réseau, noté TEB_r^{max} , qui définit le taux d'erreur binaire maximal autorisé pour le réseau r .
- et un domaine de ressource D_r , définissant l'ensemble des fréquences disponibles pour ce réseau. Chaque fréquence allouée à un réseau doit obligatoirement être dans le domaine associé. De plus, le domaine associé à chaque réseau doit être inclus dans

le domaine global de ressources D_g .

Les domaines de ressources associés à chaque réseau peuvent être différents comme le montrent les figures 3.3, 3.4 et 3.5.

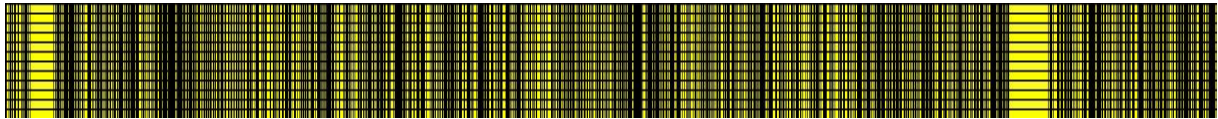


FIG. 3.3 – Domaines de ressource du scénario PUB02

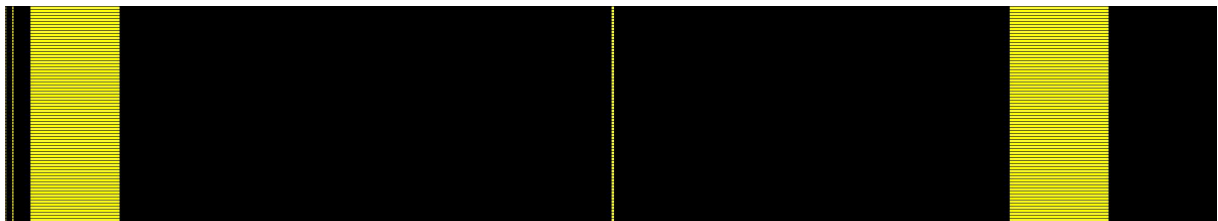


FIG. 3.4 – Domaines de ressource du scénario PUB03

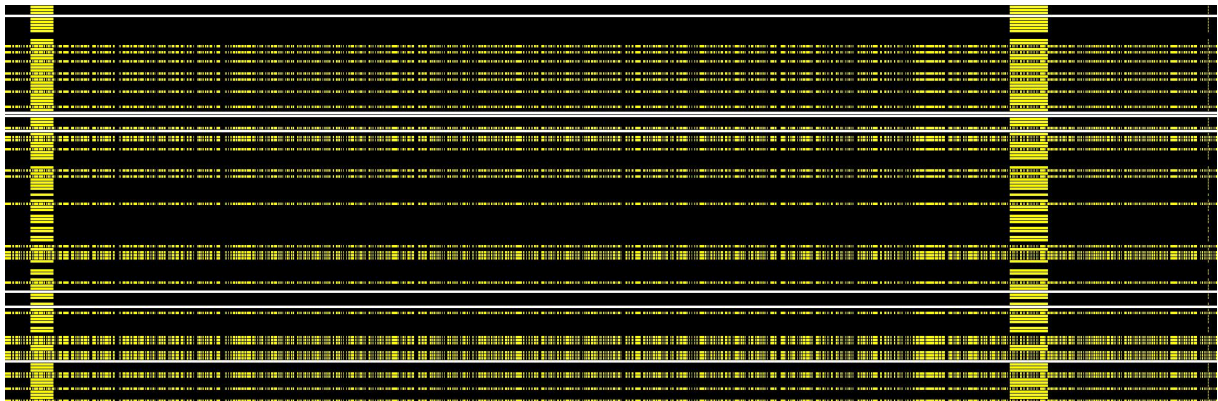


FIG. 3.5 – Domaines de ressource du scénario PUB07

Sur ces figures, l'axe des abscisses représente le spectre de fréquences (de 0 à 1999) et l'axe des ordonnées se réfère aux différents réseaux (de 1 au nombre de réseaux du problème donné). Une ligne représente le domaine associé à un réseau : les fréquences interdites sont de couleur jaune (grise) et celles autorisées sont en noir. Ces figures représentent 3 des 10 benchmarks publics² aussi appelés scénarios. Les figures 3.3, 3.4 et 3.5 représentent les domaines de ressources associés respectivement aux scénarios PUB02, PUB03 et PUB07.

²Disponible sur http://set.utbm.fr/info/projets/label/national/CELAR_PUBLIC.zip

Les domaines interdits peuvent être composés de sous-bandes interdites ou encore de fréquences isolées interdites. La figure 3.3 montre un domaine interdit composé de sous-bandes de différents pas ; les fréquences interdites sont régulièrement espacées dans le spectre. Le domaine interdit peut aussi être identique pour tous les réseaux comme le montre la figure 3.4 représentant le scénario PUB03. Pour chaque réseau de ce scénario, le domaine interdit est composé de 2 sous-bandes de pas 1 et de 2 fréquences isolées. Contrairement aux deux autres scénarios, les réseaux du scénario PUB07 utilisent plusieurs domaines interdits différents (figure 3.5).

TAB. 3.2 – Notations utilisées sur les réseaux

Notations utilisées	
R	ensemble des réseaux
R^c	réseaux cibles à allouer
R^{hc}	réseaux hors-cibles interdits de modifier
r	un réseau
h_r	niveau hiérarchique du réseau r
NF_r^{min}	nombre de fréquences minimum à allouer au réseau r
TEB_r^{max}	seuil maximum d'interférence autorisé pour le réseau r
D_r	domaine de ressources du réseau r
D_g	domaine global de ressources

3.1.2.2 Les contraintes liées aux plans de fréquences

La planification de fréquences consiste à allouer à chaque réseau une liste de fréquences appelée "plan de fréquences". Ce plan est un ensemble de fréquences structurées en intervalles appelés "sous-bandes". Un plan regroupe un nombre maximum SB_{max} de sous-bandes. Une sous-bande sb est définie par :

- une fréquence inférieure f_{min} ,
- une fréquence supérieure f_{max} ,
- un pas δ_f .

Une sous-bande sb est composée de l'ensemble des fréquences incluses dans l'intervalle $[f_{min}; f_{max}]$, respectant le pas δ_f , en considérant f_{min} comme fréquence initiale de l'échantillonnage. En d'autres termes, une sous-bande est un peigne de fréquences comprises entre une fréquence inférieure f_{min} et fréquence supérieure f_{max} et respectant un écart δ_f constant. On a donc :

$$sb = \{f_i \text{ tel que } f_i = f_{min} + i \times \delta_f \text{ et } f_{min} \leq f_i \leq f_{max}, i \in \mathbb{N}\} \quad (3.1)$$

Les sous-bandes d'un même plan de fréquences ne peuvent se chevaucher et leur nombre est limité. De plus, le plan de fréquences d'un réseau ne peut utiliser que les fréquences appartenant au domaine qui lui est associé.

On obtient la définition suivante du plan de fréquences pdf_r du réseau r :

soit SB , l'ensemble des sous-bandes sb ,

$$\left\{ \begin{array}{l} pdf_r \in \mathcal{P}(SB) \text{ tel que } \forall sb \in pdf_r, sb \subset D_r \\ \forall (sb_1, sb_2) \in pdf_r^2, sb_1 \neq sb_2 \Rightarrow (max(sb_1) < min(sb_2)) \text{ ou } (max(sb_2) < min(sb_1)) \\ |pdf_r|_{sb} \leq SB_{max} \\ |pdf_r|_f \geq NF_r^{min} \end{array} \right. \quad (3.2)$$

avec

- $r \in R$ un réseau,
- pdf_r le plan de fréquences du réseau r ,
- $\mathcal{P}(SB)$ l'ensemble des parties de SB ,
- sb_1, sb_2 deux sous-bandes,
- $min(sb), max(sb)$ fréquences inférieure et supérieure de la sous-bande sb ,
- D_r le domaine de ressource du réseau r ,
- $|pdf_r|_{sb}$ le nombre de sous-bandes du plan de fréquences pdf_r ,
- $|pdf_r|_f$ le nombre de fréquences du plan de fréquences pdf_r du réseau r ,
- NF_r^{min} le nombre minimum de fréquences que le réseau r doit utiliser.

Dans le cadre des problèmes publics que nous avons traités, le nombre maximal de sous-bandes SB_{max} est fixé à 10, le pas d'une sous-bande δ_f peut prendre trois valeurs fixées à 1, 2 et 4, et l'ensemble des domaines associés aux différents réseaux est inclus dans le domaine global noté D_g , regroupant 2000 fréquences, numérotées de 0 à 1999.

TAB. 3.3 – Notations utilisées sur les plans de fréquences

Notations utilisées	
pdf_r	plan de fréquence du réseau r
$ pdf_r _f$	nombre de fréquences du plan de fréquences du réseau r
$ pdf_r _{sb}$	nombre de sous-bandes du plan de fréquences du réseau r
SB_{max}	nombre maximum de sous-bandes dans un plan de fréquences
sb	une sous-bande
f_{min}	borne inférieure d'une sous-bande
f_{max}	borne supérieure d'une sous-bande
δ_f	pas d'une sous-bande

3.1.2.3 Les contraintes liées aux interférences

L'importance de l'interférence dépend essentiellement du type de perturbation électromagnétique caractérisé dans le problème par le type de lien. Les trois types de lien sont :

- les liens bipostes et tripostes liés à l'utilisation respective de deux ou de trois postes dans un même véhicule, et donc associées à une perturbation co-véhicule,

- et les liens cosites associées à une perturbation entre deux véhicules proches.

Pour chacun des liens l , le niveau d'interférence est mesuré par le taux d'erreur binaire élémentaire, noté $TEB_l(f_{r_r}; f_{r_{b1}} [, f_{r_{b2}}])$, qui correspond au niveau d'interférence généré par la fréquence $f_{r_{b1}}$ (et éventuellement $f_{r_{b2}}$ dans le cas triposte) sur la fréquence f_{r_r} , avec f_{r_r} la fréquence utilisée par le réseau récepteur r_r , et $f_{r_{b1}}$, $f_{r_{b2}}$ les fréquences utilisées par les réseaux brouilleurs r_{b1} et r_{b2} . La notation *lien* représente le type du lien :

- *BI* dans le cas *biposte* : l'interférence est calculée entre deux fréquences associées à deux réseaux, l'un récepteur et l'autre brouilleur.
- *CO* dans le cas *cosite* : l'interférence est calculée entre deux fréquences associées à deux réseaux, l'un récepteur et l'autre brouilleur.
- ou *TRI* dans le cas *triposte* : l'interférence est calculée entre trois fréquences associées à trois réseaux, l'un récepteur et les deux autres brouilleurs.

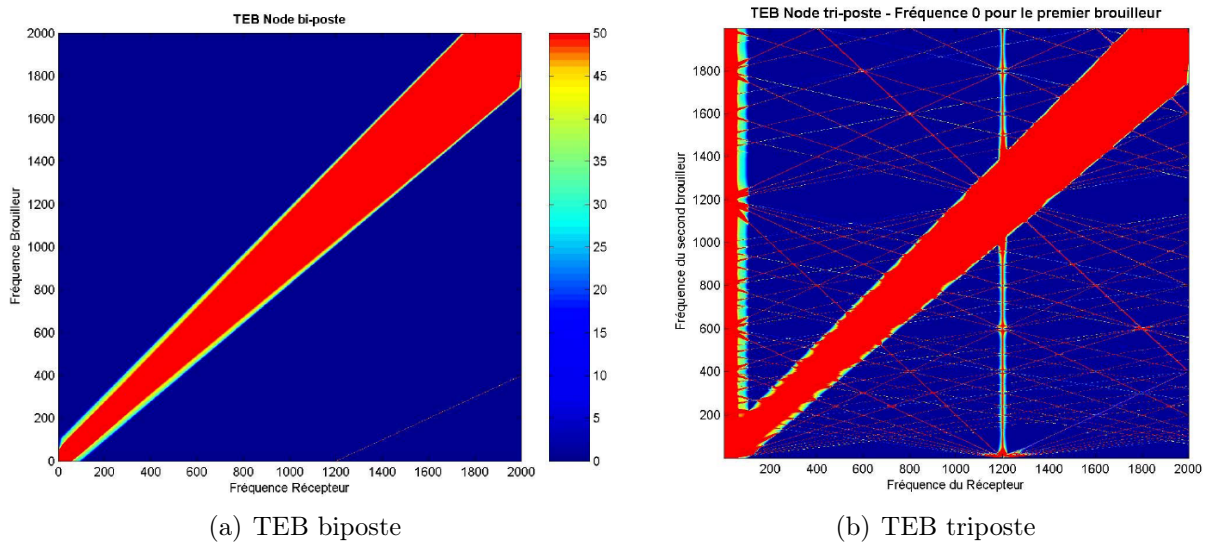


FIG. 3.6 – TEB élémentaire

Les valeurs de TEB élémentaires, $TEB_l(f_{r_r}; f_{r_{b1}} [, f_{r_{b2}}])$, sont pré-calculées par un simulateur. La figure 3.6 représente les valeurs élémentaires de TEB biposte (image (a)) et un plan de TEB triposte (image (b)). La couleur rouge (ou grise) représente les valeurs maximales de taux d'erreur binaire tandis que le bleu (ou les zones les plus sombres) correspond aux plus petites valeurs. Les calculs de TEB intègrent les phénomènes de perturbation les plus significatifs comme le bruit large bande, les raies et accords parasites, les phénomènes liés aux harmoniques et à l'intermodulation [17]. Les données de TEB triposte, utilisées lors de l'allocation des plans de fréquences, représentent plusieurs dizaines de Go. La figure 3.6(b) représente une partie des valeurs du taux d'erreur binaire triposte. Sur cette figure, la porteuse est en abscisse et la 2^e brouilleuse en ordonnée. La fréquence de la 1^{ère} brouilleuse est fixée à la fréquence 0. L'ensemble des TEB triposte correspond donc à un cube où chaque dimension correspond au spectre de fréquence.

Le niveau d'interférence entre les plans de fréquences est calculé à partir des valeurs élémentaires d'interférence entre les fréquences de chaque plan. Soit pdf_r le plan de fréquences du réseau r , le taux d'erreur binaire du réseau sera la valeur maximale de TEB générée par ce réseau en tant que récepteur pour tous les liens auquel il participe. En d'autres termes, pour chacun des liens auquel le réseau r participe, le niveau de taux d'erreur binaire est calculé en considérant le réseau r comme récepteur. La valeur de TEB référence pour chaque réseau est donc le pire cas parmi l'ensemble de ses liens. Le niveau courant d'interférence du réseau, noté $TEB^c(r)$, est calculé de la façon suivante :

$$TEB^c(r) = \max_{l(r_r; r_{b1}, r_{b2}) \in L} \left(TEB_l(pdf_{r_r}; pdf_{r_{b1}}, pdf_{r_{b2}}) \right) \quad (3.3)$$

avec $TEB_l(pdf_{r_r}; pdf_{r_{b1}}, pdf_{r_{b2}})$, le TEB provoqué par le lien l entre les réseaux r_{b1} , r_{b2} et r_r , plus précisément, par les deux plans de fréquences brouilleurs $pdf_{r_{b1}}$ et $pdf_{r_{b2}}$ sur le plan de fréquence du récepteur pdf_{r_r} . Il est calculé de la façon suivante :

$$TEB_l(pdf_{r_r}; pdf_{r_{b1}}, pdf_{r_{b2}}) = \sum_{\substack{f_{r_r} \in pdf_{r_r} \\ f_{r_{b1}} \in pdf_{r_{b1}} \\ f_{r_{b2}} \in pdf_{r_{b2}}}} \frac{TEB_l(f_{r_r}; f_{r_{b1}}, f_{r_{b2}})}{|pdf_{r_r}|_f \times |pdf_{r_{b1}}|_f \times |pdf_{r_{b2}}|_f} \quad (3.4)$$

avec $TEB_l(f_{r_r}; f_{r_{b1}}, f_{r_{b2}})$, le TEB élémentaire généré par l'utilisation des trois fréquences f_{r_r} , $f_{r_{b1}}$ et $f_{r_{b2}}$. La taille du plan de fréquences pdf_r du réseau r est notée $|pdf_r|_f$. Ainsi, le TEB d'un lien entre plusieurs réseaux est la somme des TEB élémentaires générés par chaque triplet de fréquences, pondérés par le produit des tailles de plans de fréquences. Il s'agit donc d'un TEB moyen par lien.

Pour chaque réseau r , on note TEB_r^{max} le niveau de qualité radio à respecter. Le niveau courant $TEB^c(r)$ du réseau r ne doit pas dépasser la valeur seuil TEB_r^{max} . Cette contrainte peut s'écrire de la façon suivante :

$$\forall r \in R, TEB^c(r) \leq TEB_r^{max} \quad (3.5)$$

TAB. 3.4 – Notations utilisées sur les interférences

Notations utilisées	
L	ensemble des liens
BI	ensemble des liens bipostes
CO	ensemble des liens cosites
TRI	ensemble des liens tripistes
r_r, r_{b1}, r_{b2}	réseaux récepteur et brouilleurs pour le calcul du TEB
TEB_l	Taux d'Erreur Binaire du lien l
$TEB^c(r)$	niveau d'interférence généré par le plan de fréquences du réseau r
TEB_r^{max}	seuil maximum d'interférence autorisé pour le réseau r

3.1.3 Les objectifs à optimiser

La fonction d'évaluation utilise deux types d'informations pour estimer la qualité des solutions. Le premier se rapporte au niveau d'interférence et le second représente la réutilisation des fréquences.

3.1.3.1 Les interférences

Le premier problème à traiter consiste à minimiser les interférences issues des plans de fréquences. Il comprend deux objectifs :

- minimiser la plus grande différence entre le seuil de TEB courant d'un réseau et la valeur seuil de ce même réseau,
- minimiser la somme de ces différences pour tous les réseaux pondérée par la hiérarchie de chacun d'eux.

Les deux objectifs sur les interférences sont agrégés dans une fonction unique à l'aide de deux poids α et α' . Ainsi, la fonction qui comprend les critères d'interférences, notée $F_{\text{TEB}}(S)$, est formulée par cette expression :

$$F_{\text{TEB}}(S) = \alpha \times \max_{r \in R} (\text{TEB}^c(r) - \text{TEB}_r^{\text{max}})^+ + \alpha' \times \sum_{r \in R} (h_r \times (\text{TEB}^c(r) - \text{TEB}_r^{\text{max}})^+) \quad (3.6)$$

avec

- R est l'ensemble des réseaux,
- $\text{TEB}^c(r)$ est le TEB courant du réseau r ,
- $\text{TEB}_r^{\text{max}}$ est la valeur maximale de TEB autorisée pour le réseau r ,
- $(x)^+$ est égal à x si $x > 0$, 0 sinon,
- α et α' sont des poids fixés pour chaque instance de problème.

3.1.3.2 L'utilisation des fréquences

En plus de la mesure des interférences, la qualité d'une solution est aussi mesurée en terme d'utilisation de fréquences. Trois composantes sont définies dans l'optique de favoriser la constitution de plans de fréquences de grande taille.

1. La taille minimale de l'ensemble des fréquences NF_r^{min} de chaque réseau r : l'objectif est de minimiser le nombre de réseaux dont le plan de fréquences ne respecte pas la taille minimale fixée définie par la fonction $F_{\text{taille}}(S)$:

$$F_{\text{taille}}(S) = \beta \times |\{\forall r \in R, |pdf_r|_f < NF_r^{\text{min}}\}| \quad (3.7)$$

avec

- r un réseau,
- $|pdf_r|_f$ est la taille courante du plan de fréquences du réseau r ,

- NF_r^{min} est la taille minimale à respecter par le plan de fréquences du réseau r ,
 - $|\{\forall r \in R, |pdf_r|_f < NF_r^{min}\}|$ est le nombre de plan de fréquences dont la taille est inférieure à la taille minimale exigée,
 - β est un paramètre du problème.
2. La taille du plus petit plan de fréquences : l'objectif est de maximiser la taille du plus petit plan de fréquences définie par la fonction $F_{min}(S)$:

$$F_{min}(S) = \gamma \times \min_{r \in R}(|pdf_r|_f) \quad (3.8)$$

avec

- r un réseau,
 - $|pdf_r|_f$ est la taille courante du plan de fréquences du réseau r ,
 - γ un paramètre du problème.
3. La réutilisation des fréquences : l'objectif est de maximiser la somme pondérée des tailles de tous les plans de fréquences définie par la fonction $F_{som}(S)$. Pour cela, le niveau de hiérarchie h_r est utilisé.

$$F_{som}(S) = \gamma' \times \sum_{r \in R} (h_r \times |pdf_r|_f) \quad (3.9)$$

avec

- r un réseau,
- $|pdf_r|_f$ la taille courante du plan de fréquences du réseau r ,
- h_r le niveau hiérarchique du réseau r ,
- γ' un paramètre du problème.

3.1.3.3 La fonction d'évaluation

Pour résumer, l'affectation de plans de fréquences en éviation de fréquences consiste à associer une liste de fréquences à chaque réseau du déploiement en respectant les contraintes du nombre maximum de sous-bandes SB_{max} et de non recouvrement des sous-bandes d'un même plan de fréquences, d'une part en minimisant le nombre de réseaux ayant un plan de fréquences trop petit $F_{taille}(S)$ et les interférences entre les plans de fréquences $F_{TEB}(S)$, et d'autre part en maximisant la taille du plus petit plan de fréquences $F_{min}(S)$ et la réutilisation des fréquences $F_{som}(S)$. Toutes ces fonctions sont combinées en une unique fonction de coût $F(S)$ à minimiser, formulée de la façon suivante :

$$F(S) = F_{TEB}(S) + F_{taille}(S) - (F_{min}(S) + F_{som}(S)) \quad (3.10)$$

avec

- $F_{TEB}(S)$: objectif lié aux interférences,
- $F_{taille}(S)$: objectif lié au nombre de réseaux de plans de fréquences trop petits,
- $F_{min}(S)$: objectif lié à la taille du plus petit plan de fréquences,
- $F_{som}(S)$: objectif lié à la réutilisation des fréquences.

NB : Les objectifs de taille de plans de fréquences, de nombre de réseaux et de qualité radio des plans de fréquences n'étant pas dans les mêmes unités, la fonction d'évaluation n'a pas de réalité physique. Par ailleurs les valeurs affectées aux paramètres pondérant chaque critère sont prépondérantes pour définir la typologie des solutions de chaque problème. Un modèle multi-objectif permettrait de trouver des solutions de compromis entre les critères en dissociant les critères d'unités distinctes, voire en considérant les quatre critères séparément.

3.1.3.4 Réalisabilité

L'objectif du problème est de minimiser la fonction de coût $F(S)$. Un problème est dit réalisable lorsqu'il existe au moins une solution réalisable à ce problème. Une solution s sera dite réalisable si tous les plans de fréquences respectent les contraintes d'interférence maximale et de taille minimale de plan de fréquences décrites en (3.5) et (3.2), soit :

$$s \text{ est réalisable} \iff \begin{cases} \forall r \in R, \text{TEB}^c(r) \leq \text{TEB}_r^{\max} \\ \text{et } |pdf_r|_f \geq NF_r^{\min} \end{cases} \quad (3.11)$$

avec

- r un réseau,
- $\text{TEB}^c(r)$ le TEB courant du réseau r et TEB_r^{\max} son TEB maximal,
- $|pdf_r|_f$ la taille courante du réseau r et NF_r^{\min} sa taille minimale.

NB : Une solution non réalisable peut être de meilleure qualité qu'une solution réalisable. Atteindre la réalisabilité n'est pas une fin en soi, cependant elle représente une caractéristique importante pour la résolution du problème. En effet, certains poids de la fonction de coût parmi α et α' pour le TEB, et β pour la taille, peuvent être très importants suivant les scénarios. Dans ce cas, une solution réalisable sera aussi une solution de bonne qualité au vu de la fonction d'évaluation.

3.1.4 Les instances

Deux groupes de jeux de tests fournis par le CELAR ont été utilisés dans notre travail, l'un *privé*, l'autre *public*. Les détails de ces instances sont présentés dans cette partie. En ce qui concerne les jeux privés, nous ne donneront que certaines caractéristiques correspondant aux propriétés des instances publiques. Les instances privées correspondent à une définition de problème plus complexe et comprennent des paramètres, contraintes ou objectifs confidentiels.

Chaque jeu de test est appelé un scénario. Ils sont décrits dans les tableaux 3.6 et 3.7. Les principales caractéristiques d'un scénario résumées dans le tableau 3.5 sont :

- Le temps d'exécution : un temps limite (en heure) est donné pour chaque instance, il est spécifié par la variable *Temps*.

- Le nombre de réseaux : le problème est composé d'un nombre R de réseaux dont R^c réseaux cibles et R^{hc} hors cibles ; ces réseaux déjà alloués ne peuvent être modifiés, mais doivent être pris en compte lors du calcul du niveau d'interférence.
- Les liens TEB : chaque problème est composé de $|L|$ liens TEB dont $|TRI|$ liens tripistes, $|BI|$ bipistes et $|CO|$ cosites.
- Les contraintes sur les plans de fréquences : le nombre de contraintes d'interférence noté $|TEB_r^{max}|$ et de taille minimale noté $|NF_r^{min}|$ ainsi que les valeurs minimales et maximales des contraintes TEB_r^{max} et NF_r^{min} utilisé par les différents réseaux du problème. Par exemple, 30 – 90 signifie que les valeurs de cette contrainte sont comprises entre 30 et 90 pour tous les réseaux du problème. Le nombre de domaines $|D|$ utilisé pour l'ensemble des réseaux est aussi précisé.
- Les paramètres de la fonction d'évaluation (α , α' , β , γ et γ') qui pondèrent les différents critères : α et α' pour les interférences, β , γ et γ' pour la taille des plans de fréquences.

Le tableau 3.5 récapitule les différentes caractéristiques définissant un scénario. Les valeurs de ces caractéristiques sont données dans le tableau 3.6 pour les instances privées et dans le tableau 3.7 pour les instances publiques.

TAB. 3.5 – Notations utilisées pour les instances

Notations utilisées	
SCx	scénario privé
$PUBx$	scénario public
R, R^c, R^{hc}	ensemble des réseaux, réseaux cibles (à allouer) et hors cible
L	ensemble des liens
BI, CO, TRI	ensemble des liens bipistes, cosites et tripistes
TEB_r^{max}	niveau maximum d'interférence autorisé pour le réseau r
NF_r^{min}	nombre minimum de fréquences du plan de fréquences associé au réseau r
D	ensemble des domaines utilisés par les différents réseaux
$\alpha, \alpha', \beta, \gamma, \gamma'$	paramètres de la fonction de coût

3.1.4.1 Les instances privées

Les 20 instances privées seront notées SCx , avec x une valeur entière comprise entre 01 et 20. Elles sont présentées dans le tableau 3.6.

Les scénarios privés sont très différents les uns des autres. Parmi eux, le scénario SC11 se démarque par le nombre important de réseaux hors-cibles (66) par rapport au nombre de réseaux cibles (6). Pour la plupart des autres scénarios, le nombre de réseaux hors-cibles est très faible voire nul pour 13 d'entre eux. D'autres scénarios se différencient par

TAB. 3.6 – Description des 20 scénarios privés

Scenario	SC01	SC02	SC03	SC04	SC05	SC06	SC07	SC08	SC09	SC10
Temps (h)	1	0.5	5	5	5	4	5	8	4	10
$ R $	10	12	72	96	142	123	130	249	249	400
$ R^c $	10	12	72	94	142	123	130	249	249	400
$ R^{hc} $	0	0	0	2	0	0	0	0	0	0
$ L $	26	0	1057	22 656	0	533	0	4 073	0	28 837
$ TRI $	3	0	6	0	0	19	0	106	0	39
$ BI $	8	0	63	178	0	130	0	282	0	595
$ CO $	15	0	988	22 478	0	384	0	3 685	0	28 203
$ TEB_r^{max} $	10	12	72	94	142	123	130	249	249	400
TEB_r^{max}	7	0	7	7-50	0	7	0	7	0	7
$ NF_r^{min} $	10	12	72	94	142	123	130	249	249	400
NF_r^{min}	30	30-90	30-50	30	30	30	30	30	30	30
$ D $	3	1	1	1	1	1	2	1	2	1
α	100000	100000	100000	100000	100000	100000	100000	100000	100000	100000
α'	10000	4166	695	532	353	407	385	201	201	125
β	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000
γ	-0.833	-0.7199	-0.1399	-0.1174	-0.1081	-0.1233	-0.0868	-0.0725	-0.0452	-0.029
γ'	-0.1666	-0.1439	-0.0279	-0.0234	-0.0216	-0.0246	-0.0173	-.0145	-0.0045	-0.0058
Scenario	SC11	SC12	SC13	SC14	SC15	SC16	SC17	SC18	SC19	SC20
Temps (h)	0.5	1	1	1	2.5	2.5	1	10	4	4
$ R $	72	72	130	73	249	250	124	249	400	400
$ R^c $	6	72	130	73	227	209	99	206	385	400
$ R^{hc} $	66	0	0	0	22	41	24	43	15	0
$ L $	53	1051	7 369	1 123	3 411	3 493	411	3 248	24 921	25 902
$ TRI $	1	6	25	6	75	66	10	71	21	22
$ BI $	2	61	241	85	238	255	105	216	512	517
$ CO $	50	984	7103	1 032	3320	3172	296	2 961	24 388	25 363
$ TEB_r^{max} $	21	72	130	73	166	166	86	161	260	260
TEB_r^{max}	6-50	6	6-7	7-57	6-50	5-50	6-50	6-50	6-50	1-50
$ NF_r^{min} $	6	72	130	73	227	209	99	206	385	400
NF_r^{min}	0-40	30-40	30-40	30-250	0-40	0-40	0-30	0-30	0-30	30
$ D $	2	2	2	2	2	2	2	2	2	2
α	100000	100000	100000	100000	100000	100000	100000	100000	100000	100000
α'	8334	695	385	685	221	240	506	243	130	125
β	10000	10000	10000	10000	10000	10000	10000	10000	10000	10000
γ	-2.3741	-0.1428	-0.3008	-0.143	-0.051	-0.0508	-0.1531	-0.1824	-0.0495	-0.0952
γ'	-0.4748	-0.0285	-0.0601	-0.0286	-0.0102	-0.0101	-0.0306	-0.0364	-0.0495	-0.019

l'utilisation d'une taille minimale de plan de fréquences très importante (de 30 à 90 pour le scénario SC2) ou très petites (de 0 à 30 pour les scénarios SC18 et SC19). Les scénarios SC08, SC15, SC16 et SC17 utilisent un nombre important de liens tripostes (de 66 à 106), ce qui implique une lenteur d'évaluation de la fonction de coût mais aussi une plus grande complexité de résolution. Aucune information n'était disponible sur la difficulté de ces instances lorsque nous les avons utilisées. Les scénarios privés SC02, SC05, SC07 et SC09 ont un nombre de liens TEB nul et leur seuil maximal de TEB défini pour l'ensemble des réseaux est nul aussi.

3.1.4.2 Les instances publiques

Les principales caractéristiques des 10 instances publiques sont présentées dans le tableau 3.7. Les instances publiques sont notées $PUBx$, avec x compris entre 01 et 10.

TAB. 3.7 – Description des 10 scénarios publics

Scenario	PUB01	PUB02	PUB03	PUB04	PUB05	PUB06	PUB07	PUB08	PUB09	PUB10
Temps (h)	0.5	0.5	1	1	1	1	1	1	1	1
$ R $	10	12	72	96	143	124	131	249	250	400
$ R^c $	10	12	72	86	134	110	124	238	213	386
$ R^{hc} $	0	0	0	10	9	14	7	11	37	14
$ L $	26	33	1 057	21 923	2 488	607	7 484	4 047	4 054	28 764
$ TRI $	3	5	3	0	11	22	25	104	103	43
$ BI $	8	8	83	274	282	170	272	289	361	676
$ CO $	15	20	971	21 649	2 195	415	7 187	3 654	3 590	28 045
$ TEB_r^{max} $	10	12	72	96	143	124	131	248	250	400
TEB_r^{max}	6-12	6-12	6-12	6-12	6-12	6-12	6-12	6-12	6-12	6-12
$ NF_r^{min} $	10	12	72	86	134	110	124	238	213	386
NF_r^{min}	30-70	30-40	30-70	30-70	30-250	30-70	30-70	30-70	30-250	30-70
$ D $	2	2	2	2	3	2	3	2	2	2
α	100 000	100 000	100 000	100 000	100 000	100 000	1	1	100 000	100 000
α'	50 000	41 700	6 950	5 900	3 800	4 550	131	248	2 350	1 300
β	30 000	30 000	50 000	35 000	30 000	30 000	124	238	31 000	30 000
γ	-11	-16,8	-12	-11,9	-10,5	-13,4	-124	-238	-10,8	-11
γ'	-0,2	-0,25	-0,03	-0,03	-0,01	-0,02	-1	-1	-0,007	-0,004

Les 10 scénarios publics représentent un panel de problèmes très différents. Le nombre de réseaux varie de 10 à 400, avec un nombre de liens variant de 26 à 28 764. De plus, les valeurs des paramètres sont très différentes : par exemple, le paramètre α prend les valeurs 1 et 100 000. Cela permet d'accentuer l'importance relative d'un ou de plusieurs critère(s) donné(s).

3.1.4.3 Étude de la combinatoire des instances

Cette partie présente une étude combinatoire sur les instances publiques. Dans un premier temps nous considérons un pas unique. Une sous-bande est définie par deux fréquences : une fréquence inférieure et une fréquence supérieure. Allouer 10 sous-bandes dans le spectre de fréquences revient à sélectionner 20 fréquences différentes dans l'ensemble des fréquences. Soit, pour 1 réseau, 1 pas unique par sous-bande et 10 sous-bandes par plan de fréquences, la combinatoire est $C_{2000}^{20} > 10^{60}$, ce qui correspond à un tirage de 20 fréquences sans remise.

Ensuite, considérons les 3 pas possibles de chaque sous-bande. On obtient donc le nombre de solutions possibles en multipliant la combinatoire précédente par 3^{10} .

Enfin, pour N réseaux, on obtient $(10^{60} \times 3^{10})^N$ solutions possibles. Prenons l'exemple d'un scénario composé de 400 réseaux, nous obtenons ainsi une combinatoire supérieure

à $10^{60 \times 400} \times 3^{10 \times 400} > 10^{26\,000}$.

Il faut faire un tirage avec remise pour considérer le cas où le nombre de sous-bandes varie de 1 à 10. On peut ainsi avoir une unique sous-bande au final si la même fréquence est toujours tirée. On obtient pour un réseau unique : $2\,000^{20} \approx 10^{66}$ possibilités, soit un accroissement de $10^{6 \times N}$ de la combinatoire. Pour un problème à 400 réseaux, nous obtenons ainsi au moins $10^{28\,400}$ solutions possibles.

Pour les déploiements de grande taille, la combinatoire associée est très importante. La taille de ce problème est proche de celle du problème d'allocation de fréquences en réseaux GSM qui pour un réseau urbain de très grande taille avoisine les $10^{26\,000}$ possibilités.

Par contre, il est difficile de se prononcer sur la difficulté du problème et notamment sur la réalisabilité avec les données fournies. En effet, celle-ci dépend du nombre de liens et de la structure des matrices de contraintes entre fréquences, ces matrices pouvant représenter soit des TEB soit des contraintes d'écart en fréquences.

3.2 Modèles de référence et méthodes

Hormis les méthodes proposées par les deux autres équipes du projet présentées en 3.4.2, il n'existe pas d'autres méthodes publiées traitant le problème d'affectation de listes de fréquences en éviation de fréquences défini par [36]. Cependant, dans la littérature il existe des problèmes qui s'en approchent : les problèmes d'affectation de fréquences [1, 84] et plus généralement les problèmes de T-coloration.

Il existe trois grands types de problèmes de coloration. Le premier type de problème de coloration a été défini dans le chapitre précédent. Dans cette partie, nous définirons uniquement les problèmes de T-coloration et de T-coloration avec ensembles qui couvrent une partie du problème d'affectation de listes de fréquences avec éviation de fréquences.

- **Problème de T-coloration :** Étant donné un graphe $G = (V, E)$ non-orienté, l'objectif du *problème de T-coloration* est de trouver une coloration tel que les couleurs affectées à des nœuds adjacents doivent vérifier la relation suivante :

$$\forall e_{i,j} \in E, x_i, x_j \in V, |c(x_i) - c(x_j)| \notin T_{i,j} \quad (3.12)$$

avec $T_{i,j}$ un ensemble d'entiers positifs définissant les séparations de couleur interdites entre les nœuds x_i et x_j .

NB : Le *problème de T-coloration restreinte* est une variante du problème de *T-coloration* pour laquelle la contrainte 3.12 devient :

$$\forall e_{i,j} \in E, x_i, x_j \in V, |c(x_i) - c(x_j)| \geq t_{i,j} \quad (3.13)$$

avec $T_{i,j} = \{0, 1, \dots, t_{i,j} - 1\}$ un ensemble d'entiers consécutifs.

- **Problème de T-coloration avec ensembles :** Étant donné un graphe $G = (V, E)$ non-orienté, soit B un ensemble de besoins avec $B = \{b_1, \dots, b_N \mid b_i \in \mathbb{N}\}$ correspondant aux besoins respectifs de chaque nœud x_i , l'objectif du *problème de T-coloration avec ensembles* est d'allouer un ensemble de b_i couleurs à chaque nœud x_i en respectant les contraintes suivantes :

– *Contrainte co-nœud :*

$$\forall c_{i,m}, c_{i,n} \in c(x_i), m \neq n, |c_{i,m} - c_{i,n}| \notin T_{i,i} \quad (3.14)$$

avec $c(x_i) = \{c_{i,1}, \dots, c_{i,b_i}\}$.

– *Contrainte entre nœuds adjacents :*

$$\forall (x_i, x_j) \in E, \forall c_{i,m} \in c(x_i), \forall c_{j,n} \in c(x_j), |c_{i,m} - c_{j,n}| \notin T_{i,j} \quad (3.15)$$

Les problèmes de T-coloration et de T-coloration avec ensembles ont été très largement étudiés dans la littérature [32, 43, 46], cependant ils ne couvrent pas les problèmes de découpage en bandes et de taille des plans de fréquences de notre problème.

Du côté des systèmes de radiocommunications, les problèmes d'affectation de fréquences³ sont des cas particuliers des problèmes de coloration [1, 55]. Les différences entre les problèmes d'affectation de fréquences sont liées aux différents systèmes de radiocommunication. Que ce soit en téléphonie mobile, pour les réseaux sans fil ou encore pour les liaisons point-à-point, les systèmes de radiocommunications les plus modernes reposent sur l'utilisation de liste de fréquences. Nous nous intéressons à trois systèmes utilisant différemment les fréquences qui leur sont affectées.

- Le système GSM (*Global System for Mobile Communications*) est une norme de la seconde génération de téléphonie mobile établie en 1982 par le CEPT (*Conférence des Administrations Européennes des Postes et Télécommunications*). A chaque antenne est attribuée un ensemble N composé au maximum de 62 canaux de fréquences (spectre disponible le plus souvent en GSM). Ce système utilise un saut de fréquences lent (*Slow Frequency Hopping*, SFH) qui permet de changer périodiquement de fréquences au cours de l'émission d'une communication après chaque burst. Un burst correspond à une fraction de la communication à transmettre. Pour éviter les brouillages, les fréquences utilisées entre antennes adjacentes doivent être différentes à chaque saut. L'affectation des fréquences ne peut se faire que par l'analyse du taux d'effacement des trames consécutif au brouillage, ce qui est proche des critères de TEB que nous utilisons.

Avec ou sans saut de fréquences, le problème se ramène à un problème de T-coloration avec ensembles, cependant la très grande majorité des travaux dans la littérature a été réalisée sans prendre en compte le saut de fréquences avec des

³Pour plus d'informations sur ces problèmes, se référer au site <http://fap.zib.de>.

méthodes à population [43, 44, 71, 110, 133], des méthodes de recherche locale [105] comme la Recherche Tabou [72, 98, 99] ou encore des méthodes de programmation par contraintes [52, 116, 138].

Bien que le problème d'affectation de fréquences ait été largement étudié dans la littérature, peu de travaux ont été réalisés sur l'affectation de listes de fréquences dans un réseau utilisant le saut de fréquences. Moon *et al.* [100] se sont intéressés à ce problème. La méthode utilisée décompose le problème en deux parties : la pré-génération de listes de fréquences puis la modification de ces listes. Pour cela, une méthode de recuit simulé a été développée. Comme pour le problème d'affectation de plans de fréquences, un domaine de fréquences disponibles correspond à chaque variable à allouer (ici, ce sont les porteuses). Les interférences générées sont calculées en fonction de l'écart entre les fréquences allouées. Mais les listes à allouer ne sont pas structurées en sous-bandes comme doivent l'être les plans de fréquences que nous utilisons.

- Parmi les réseaux sans fil, la norme IEEE 802.11, plus connue sous le label de Wi-Fi, utilise deux modes de fonctionnement. Le mode de fonctionnement implanté actuellement s'appelle le DSSS (en anglais, *Direct-Sequence Spread Spectrum*). Seuls treize canaux de fréquences sont disponibles. Chaque antenne utilise un seul et unique canal qui ne varie pas au cours du temps. Ce mécanisme peut être modélisé par un problème de T-coloration, les fréquences allouées aux antennes adjacentes doivent respecter un écart minimum afin de limiter les interférences. Différents travaux ont été réalisés récemment sur ce sujet [56, 89, 111]. Le second mode de fonctionnement, nommé FHSS (*Frequency-Hopping Spread Spectrum*) est basé sur le saut de fréquences et utilise donc des listes de fréquences. De la même façon que notre problème, la qualité s'évalue par un taux d'erreur binaire sur l'ensemble des fréquences. 70 canaux de fréquences sont disponibles et il s'agit aussi de sélectionner les bonnes listes de fréquences par antenne. Cependant cette technologie n'est pas implantée dans le système actuel et on ne trouve pas de travaux d'optimisation dessus.
- Pour finir, le dernier système de radiocommunication auquel nous nous sommes intéressés est l'OFDM (*Orthogonal Frequency-Division Multiplexing*) [88, 127]. Le principe est le suivant : une communication est transmise simultanément sur plusieurs fréquences. Les fréquences utilisées doivent être orthogonales entre elles afin de limiter les interférences. En utilisant six fréquences simultanément, on obtient ainsi un débit six fois supérieur à un système sans OFDM. Cette technologie est utilisée en réseaux sans fil comme par exemple pour le WiMax (*Worldwide Interoperability for Microwave Access*) basé sur le standard IEEE 802.16, et en liaisons point-à-point pour les fibres optiques. Sur ce système non plus, on ne dispose pas actuellement de référence ni même de scénarios de tests.

A travers cette recherche, on constate donc que l'état de l'art sur le problème qui nous est présenté est très peu fourni. Une des spécificités de ce problème est le rassemblement de

composants jusqu'alors traités séparément : liste de fréquences à allouer, liste de critères très différents à prendre en compte et contraintes dont la vérification est calculée par un simulateur. La notion de sous-bandes est par contre très spécifique. Tout cela dans un contexte où la combinatoire est gigantesque et les ressources machines et le temps de calcul limités. Notre démarche a donc été guidée par la mise au point d'une méthode issue de la coloration, demandant peu de ressources, capable de fournir des solutions entières et assez robuste pour prendre en compte les variations des poids sur les différents objectifs et contraintes. Nos études sur la coloration nous ont paru être le meilleur point de départ.

3.3 Application de la méthode RLA

La méthode employée pour traiter ces problèmes est une méthode de recherche locale basée sur l'utilisation de mécanisme d'extension et de restriction du voisinage selon les principes mis au point dans le chapitre sur la k -coloration [39, 40]. L'exploration de voisinage est basée sur deux mécanismes complémentaires : la détection de boucle et la liste Tabou employées respectivement pour étendre et restreindre le voisinage de la solution courante. Le mécanisme de détection de boucle utilise un historique de la recherche afin de diversifier la recherche en étendant le voisinage en cas de redondance du choix des variables à modifier. La liste Tabou limite le voisinage de la solution courante pour empêcher des choix répétitifs de variables pendant les prochaines itérations. Dans les deux cas, la notion de variable se rapporte au plan de fréquences d'un réseau cible. Nous ferons par la suite l'analogie entre un plan de fréquences et son réseau associé.

Dans cette partie, nous verrons tous les éléments qui se rapportent à la méthode mise en oeuvre : le schéma général de la méthode, la génération de la solution initiale utilisée ainsi que les adaptations de la méthode RLA pour le problème d'affectation de listes de fréquences. Le problème étant spécifique, plusieurs opérateurs particuliers ont été assemblés au sein de la méthode générale que nous avons établie.

3.3.1 Schéma général de la méthode

Dans un premier temps, le diagramme 3.7 présente la structure générale de la méthode. Sur ce diagramme, nous avons distingué 5 blocs indépendants.

1. Le premier bloc correspond à l'initialisation. Ce bloc regroupe la génération de la solution initiale détaillée dans la section suivante ainsi que l'évaluation globale de cette solution et l'initialisation de la liste Tabou. L'évaluation globale est appelée une seule fois durant la recherche. Cette évaluation consiste à calculer la fonction de coût de la solution sans utiliser des valeurs précédemment calculées. Dans un premier temps, l'évaluation de chaque lien est réalisée selon l'équation (3.4) suivi de l'évaluation de chaque plan de fréquences en utilisant l'équation (3.3). Enfin, le calcul de la fonction de coût de la solution est effectué.
2. Le deuxième bloc permet de sauvegarder une solution réalisable rencontrée par la méthode. L'obtention ou non d'une solution réalisable pour chaque problème est un

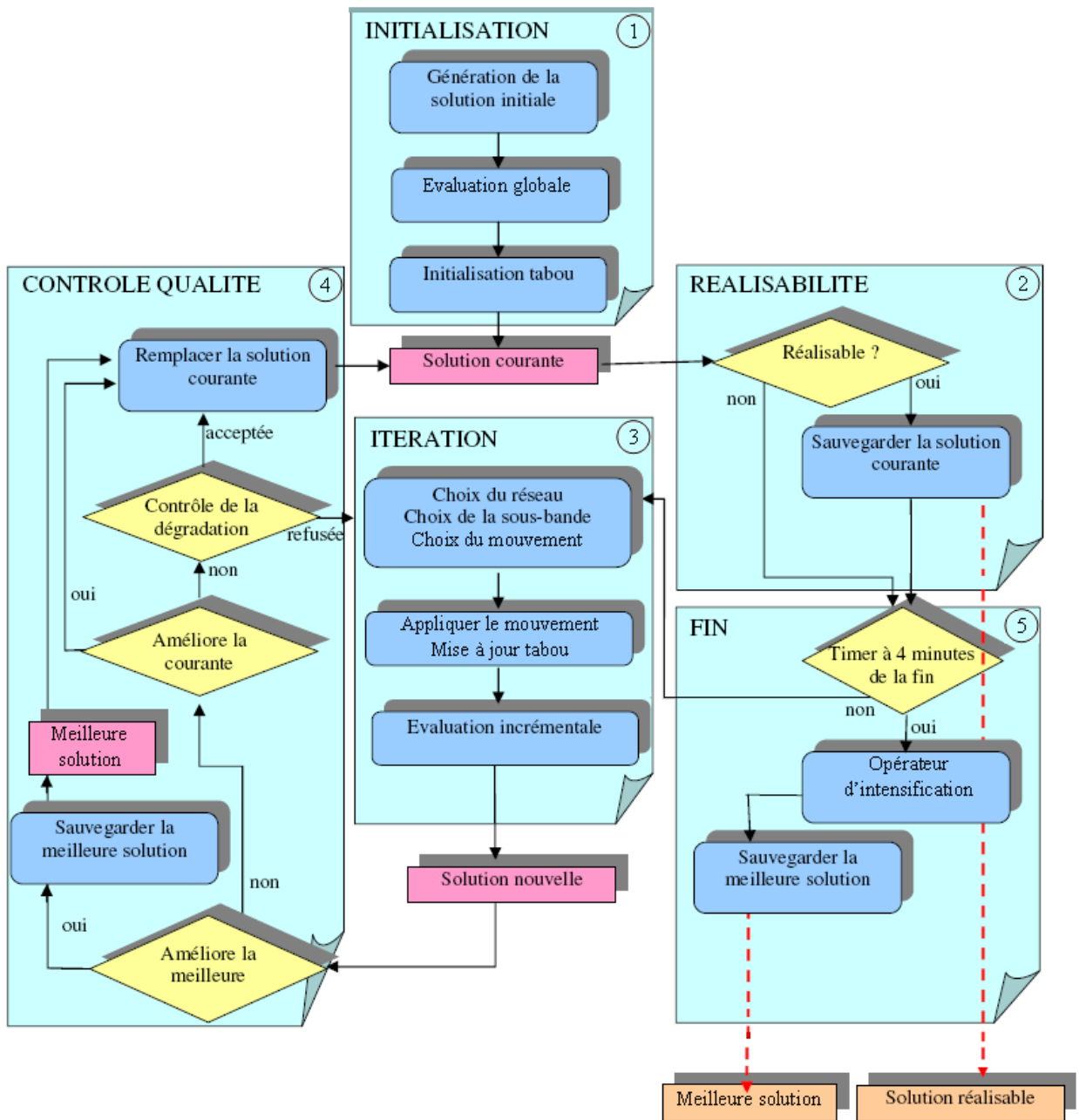


FIG. 3.7 – Diagramme du fonctionnement de la méthode

critère d'évaluation globale de la méthode. Il faut donc fournir une solution réalisable, peu importe sa qualité, en sortie du processus.

3. Le troisième bloc correspond au cycle itératif. Au cours d'une itération, un réseau à modifier est choisi puis une sous-bande de ce réseau et enfin un mouvement à appliquer. Les mécanismes utilisés pour choisir le réseau, la sous-bande et le mouvement à appliquer sont expliqués dans la partie 3.3.3. Une fois le mouvement choisi, celui-ci est appliqué à la sous-bande du réseau sélectionné. Ensuite, la solution modifiée est évaluée via une fonction d'évaluation incrémentale. Cette fonction remplit le rôle d'une évaluation partielle de la solution en se basant sur les valeurs déjà calculées et non impactées par la modification touchant à la sous-bande du réseau donné. Étant donné le caractère local de la modification touchant uniquement une sous-bande d'un plan de fréquences, seule l'évaluation des liens auxquels participe le réseau modifié est recalculée.
4. Le quatrième bloc correspond au contrôle de la qualité de la solution modifiée. La solution nouvelle remplace toujours la solution courante si elle l'améliore. Si elle ne l'améliore pas, la méthode utilise un contrôle de la dégradation détaillé en 3.3.3.4.
5. Enfin, le cinquième et dernier bloc regroupe les opérations finales à effectuer. La méthode d'intensification qui est appliquée sur la meilleure solution obtenue au cours de l'optimisation est une procédure itérative qui a pour objectif de chercher un optimum local autour de cette solution. Cette procédure opère sur la totalité de la meilleure solution et entraîne soit l'amélioration de la solution soit son maintien. Il s'agit de parcourir l'ensemble des sous-bandes des plans de fréquences cibles dans un ordre quelconque et à chaque sous bande est appliquée un mouvement pour trouver le meilleur voisin. Si l'opération sur une sous-bande n'améliore pas la solution, on passe à une autre sous-bande sans modifier la solution courante, sinon la solution améliorée est acceptée au fur et à mesure. Les voisines de chaque sous-bande ne sont donc parcourues qu'une seule fois.

3.3.2 La solution initiale

Dans cette section, la génération de la solution initiale est d'abord décrite puis les différentes options sont analysées par le biais de résultats expérimentaux.

3.3.2.1 Description globale

L'objectif de la procédure de génération de la solution initiale est de favoriser la création d'une solution initiale *réalisable* ou proche de la réalisabilité. Pour cela, un plan de fréquences de taille minimale respectant le 2^e critère de réalisabilité exprimé par l'équation (3.11) est associé à chaque réseau. Pour construire ce plan, on choisira de préférence les fréquences minimisant le TEB du réseau choisi et des réseaux voisins selon le 1^{er} critère de réalisabilité de l'équation (3.11). La procédure de génération de la solution initiale suit un

processus itératif basé sur une construction gloutonne. La figure 3.8 présente le schéma global de la méthode de génération de la solution initiale. La génération de la solution initiale est donc basée sur le choix du réseau dans un premier temps et ensuite de la sous-bande.

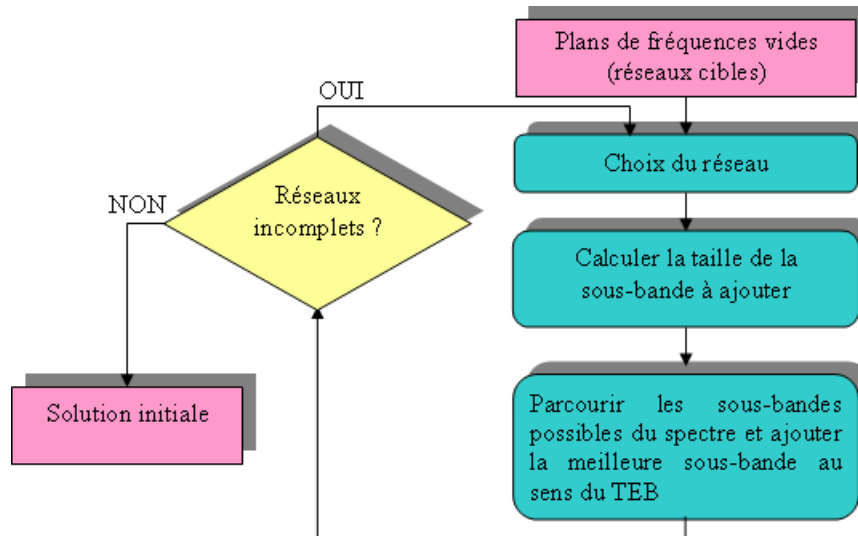


FIG. 3.8 – Schéma de construction de la solution initiale

Le choix du réseau lors de chaque itération dépend de plusieurs facteurs comme la taille minimale requise, le niveau d'interférence courant, etc. Ces facteurs sont détaillés dans la section suivante. Une fois le réseau choisi, un ensemble de sous-bandes de taille fixe, régulièrement réparties sur le spectre, sont testées et la meilleure est sélectionnée. La procédure est expliquée en deuxième sous-partie. A chaque itération, une seule sous-bande est ajoutée au plan de fréquences du réseau sélectionné. Une fois ajoutée, une sous-bande ne peut pas être supprimée du plan de fréquences courant au cours de la procédure de génération de la solution initiale.

3.3.2.1.1 Choix du réseau

Un poids définissant le niveau de priorité est associé à chaque réseau. Le poids $P_{init}(r)$ de chaque réseau r est calculé dynamiquement au cours de la construction de la solution initiale. Les facteurs déterminant la priorité d'un réseau sont : le nombre de fréquences manquantes pour atteindre la taille minimale NF_r^{min} , le niveau d'interférence selon les critères de réalisabilité et l'avancement des plans de fréquences des réseaux voisins. Si tous les réseaux voisins d'un réseau donné sont complets, alors il sera plus difficile de remplir le plan de fréquences du réseau du fait des critères d'interférences. Lors de chaque itération de la phase de construction de la solution initiale, le réseau de poids le plus

important est sélectionné. Le poids se calcule de la façon suivante :

$$\begin{aligned}
 P_{init}(r) &= (NF_r^{min} - |pdf_r|_f)^+ \\
 &\times \left(1 + \frac{TEB^c(r)}{1 + TEB_r^{max}} \right) \\
 &\times \left(\sum_{i \in Voisins(r)} |pdf_i|_f + 1 \right)
 \end{aligned} \tag{3.16}$$

avec

- r un réseau,
- $|pdf_r|_f$ la taille courante et NF_r^{min} la taille minimale du plan de fréquences du réseau r ,
- $TEB^c(r)$ le niveau courant de TEB et TEB_r^{max} le TEB maximal du réseau r ,
- $Voisins(r)$ est l'ensemble des réseaux liés au réseau r par un lien.

Un plan de fréquences est considéré complet lorsque le poids de son réseau est nul soit lorsque la taille minimale de son plan de fréquences est atteinte.

3.3.2.1.2 Choix de la sous-bande

Chaque plan de fréquences est limité à un maximum de SB_{max} sous-bandes. Quand un réseau est choisi pendant la phase de construction de la solution initiale, l'algorithme calcule le nombre de fréquences manquantes par rapport au nombre minimum de fréquences requises. Puis, la taille des sous-bandes à ajouter est calculée en fonction du nombre de fréquences manquantes et du nombre de sous-bandes non allouées. Ainsi, les sous-bandes à ajouter seront toutes petites et de même taille. De trop grandes sous-bandes pourraient être plus difficiles à allouer à cause des domaines de ressources de chaque réseau or notre procédure ne revient pas en arrière.

Des sous-bandes du domaine de ressource sont examinées dans l'ordre à partir de la première fréquence du domaine ou à partir d'une fréquence choisie aléatoirement dans le domaine. Pour chaque fréquence inférieure, tous les pas possibles sont examinés. Si la sous-bande courante respecte le critère de réalisabilité sur les TEB, la sous-bande est conservée. Dans le cas contraire, la recherche continue en explorant les autres sous-bandes du domaine. La meilleure sous-bande examinée, produisant la solution partielle de moindre coût, est conservée.

La procédure de génération de la solution initiale peut être un procédé très coûteux en terme de temps de calcul. Le contrôle du temps est donc nécessaire pour éviter que la construction de la solution initiale n'entame trop le délai imposé pour traiter le problème. Le procédé de génération de la solution réalisable est limité à 30% du temps accordé pour chaque problème. Quand le temps est dépassé, une sous-bande choisie aléatoirement dans le domaine est ajoutée à chaque plan de fréquence vide car dans la solution initiale, tous les réseaux doivent contenir au minimum une sous-bande afin de pouvoir être évalués.

3.3.2.2 Résultats et analyse

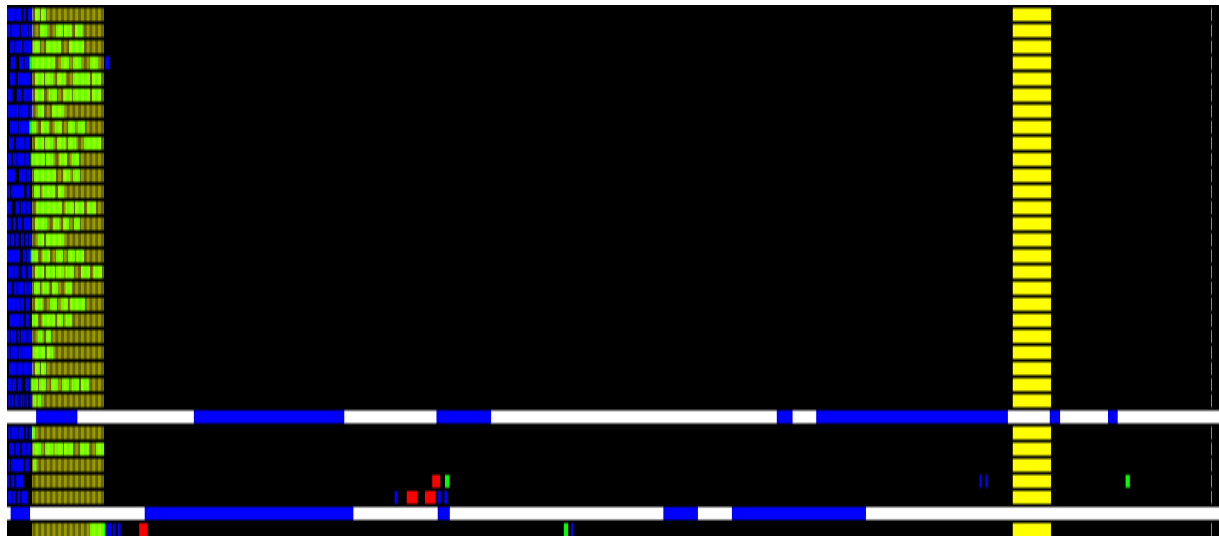
Cette section compare deux méthodes de génération de la solution initiale dans le tableau 3.8 en évaluant la fonction de coût (équation (3.10)) de la solution initiale et de la meilleure solution obtenue après optimisation. L'optimisation a été faite avec la méthode de recherche locale que nous avons définie. Les deux procédures de génération de la solution initiale utilisent la méthode gloutonne expliquée précédemment. La seule différence entre ces deux méthodes est le point de départ de la méthode gloutonne. Dans le cas d'un départ "aléatoire", la première sous-bande testée a une fréquence inférieure choisie aléatoirement dans le domaine du réseau. Dans le cas "ordonné", la première sous-bande testée correspond à la sous-bande dont la fréquence inférieure est la première fréquence disponible dans le domaine.

TAB. 3.8 – Comparaison des solutions initiales

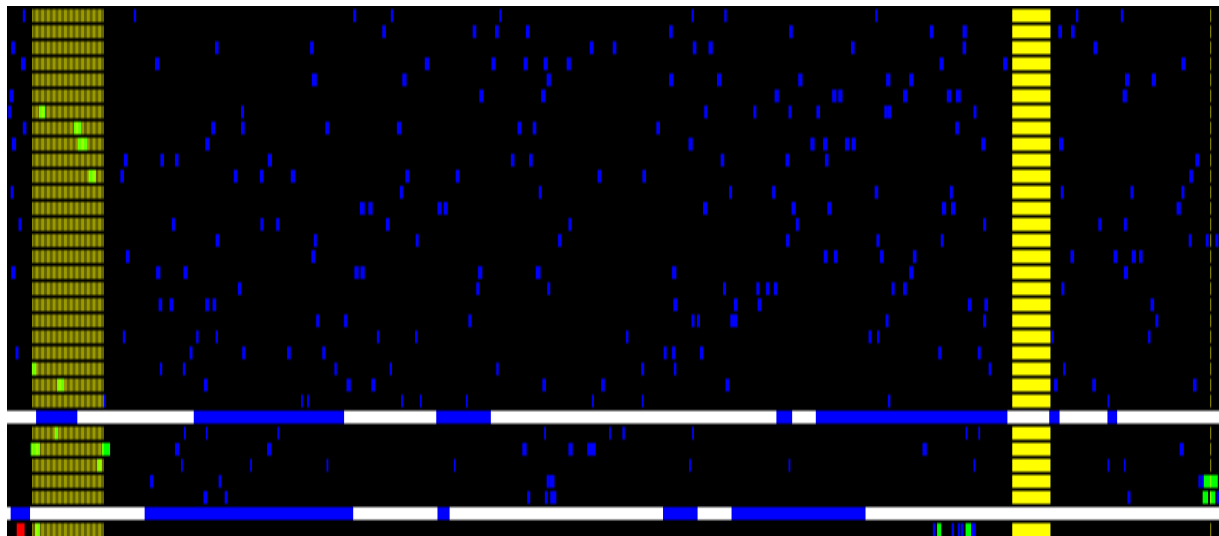
problèmes	"aléatoire"		"ordonné"	
	initiale	meilleure	initiale	meilleure
PUB01	-840	-15 803	-840	-15 291
PUB02	-896	-1 436	525 223	-1 445
PUB03	-734	-13 242	22 951	-14 879
PUB04	-808	-27 857	-807	-27 857
PUB05	-566	-9 565	749 700	-9 724
PUB06	727 949	196 222	2 078 590	73 084
PUB07	10 836	-473 840	-1 398	-478 566
PUB08	43 572	-796 187	60 945	-840 513
PUB09	1 414 131	769 544	3 184 736	690 294
PUB10	631 099	-7 769	4 098 085	-10 409

Bien que la solution initiale obtenue dans le cas "aléatoire" soit meilleure que celle obtenue dans le cas "ordonné" pour 9 des 10 scénarios examinés, la solution finale obtenue après optimisation est meilleure à partir d'une solution initiale générée par la méthode ordonnée. Pour 9 des 10 scénarios, la qualité de la solution après optimisation est meilleure dans le cas "ordonné". Commencer par une fréquence aléatoire permet d'améliorer la qualité de la solution initiale, mais en conclusion, la qualité de la solution après optimisation est plus mauvaise.

Les figures 3.9 représentent graphiquement une coupe des solutions initiales obtenues avec les méthodes "ordonnée" et "aléatoire" pour le scénario PUB10. Sur ces figures, l'ordonnée représente les fréquences (de 0 à 1999) et l'abscisse un sous-ensemble de réseaux du problème. Le scénario PUB10 utilise 400 réseaux, ces figures représentent les plans de fréquences de 30 d'entre eux. On observe que les solutions initiales sont très différentes. En effet, les sous-bandes de la solution initiale obtenue avec la méthode "ordonnée" sont très regroupées au début du spectre contrairement à celles de la méthode "aléatoire". Ce regroupement en début de spectre peut être un avantage car le reste du spectre est disponible pour ajouter plus facilement des fréquences par la suite. Les solutions finales



(a) méthode "ordonnée"



(b) méthode "aléatoire"

FIG. 3.9 – Représentation graphique des solutions initiales sur le scénario public PUB10

obtenues à partir de ces deux solutions sont toutes deux réalisables. La différence entre les solutions finales est le nombre de fréquences utilisées : dans le cas "ordonné", 609 261 fréquences sont utilisées avec le plus petit plan de fréquences composé de 158 fréquences contre 477 756 fréquences au total et 99 fréquences au minimum dans le cas de la méthode "aléatoire".

L'utilisation systématique de la partie inférieure des domaines permet de mieux gérer les interférences : la réutilisation du spectre est favorisée pour les réseaux qui ne se brouillent pas, laissant ainsi disponibles les autres fréquences pour les réseaux à problème. Nous avons donc retenu la méthode gloutonne ordonnée pour la suite de notre travail.

3.3.3 Le processus itératif

Une fois la solution initiale construite, la méthode de recherche locale peut être conçue. La méthode de recherche locale est un processus itératif. À chaque itération, un mouvement est effectué sur la solution courante menant à une nouvelle solution, voisine de la solution courante, et pouvant être acceptée ou rejetée selon un mécanisme de contrôle de la dégradation.

Une itération est définie par des opérations successives dont la finalité est de modifier la liste des fréquences qui constitue le plan de fréquences d'un réseau :

1. le choix du réseau dépendant de la détection de boucle et de la liste Tabou,
2. le choix de la sous-bande,
3. et finalement, le choix de la modification à effectuer sur cette sous-bande.

Ces trois étapes sont décrites en détail dans cette section. Lorsqu'une nouvelle solution a été produite, un mécanisme de contrôle de dégradation permettra de décider si la solution sera maintenue pour l'itération suivante.

3.3.3.1 Détection de boucle et liste Tabou

Les mécanismes de détection de boucle et de liste Tabou s'appliquent uniquement sur le choix du réseau. Le choix de la sous-bande et du mouvement ne dépendent pas de ces deux mécanismes. Ces deux mécanismes et leur utilisation pour le problème traité sont expliqués dans cette partie.

Afin de sélectionner le réseau (ou le plan de fréquences) à modifier, une fonction de poids $P_c(r)$ est employée pour chaque réseau r estimant la qualité du plan de fréquences associé. Ce poids montre l'influence de chaque plan de fréquences sur l'évaluation de la solution complète en fonction du niveau de TEB atteint par le réseau et de son déficit en fréquences par rapport à la largeur minimale demandée. La formule (3.17) utilisée pour définir ce poids prend en compte les paramètres de la fonction d'évaluation de la solution.

$$\begin{aligned}
 P_c(r) = h_r \times & (\alpha' \times (\text{TEB}^c(r) - \text{TEB}_r^{\text{max}})^+ \\
 & + \beta \times (NF_r^{\text{min}} - |pdf_r|_f)^+ \\
 & + \gamma' \times |pdf_r|_f)
 \end{aligned} \tag{3.17}$$

avec

- r un réseau,
- $|pdf_r|_f$ la taille courante et NF_r^{min} la taille minimale du réseau r ,
- $\text{TEB}^c(r)$ le niveau courant de TEB et $\text{TEB}_r^{\text{max}}$ le TEB maximal du réseau r ,
- h_r le niveau hiérarchique du réseau r ,
- α' , β et γ' les paramètres de la fonction de coût.

Dans un premier temps, nous avons défini une recherche locale basée sur la modification du plan de fréquences de pire qualité. Cette méthode a été utilisée avec et sans

liste Tabou sur le choix du réseau. Le réseau de poids le plus important est choisi. En cas d'égalité, le réseau est choisi aléatoirement parmi les réseaux de poids les plus importants. Ce choix *déterministe* du réseau est réalisé à chaque itération, il est un facteur important de l'algorithme. Cependant, le caractère purement déterministe peut entraîner la convergence prématurée vers un optimum local. L'option avec liste Tabou force l'algorithme à choisir le réseau non tabou de poids le plus important.

Le réseau choisi devient tabou pour une durée T choisie aléatoirement dans l'intervalle DT défini en équation (3.18).

$$T = rand(DT) \text{ avec } DT = \left[0.5 \times \frac{\sqrt{|R|}}{2}; 1.5 \times \frac{\sqrt{|R|}}{2} \right] \quad (3.18)$$

Il s'agit là d'un algorithme de recherche locale avec liste Tabou selon un modèle très fréquemment utilisé en littérature : l'algorithme travaille sur les variables qui contribuent le plus à la dégradation de la fonction de coût. Cependant, la Recherche Tabou avec exploration totale du voisinage est inapplicable du fait du très grand nombre de voisins pour chaque solution.

Comme nous l'avons fait sur le problème de k -coloration, une deuxième méthode basée sur la diversification dans le choix du réseau est ajoutée par l'utilisation d'un mécanisme de détection de boucle. En effet, l'objectif de ce procédé est de détecter les redondances dans le choix des réseaux au cours des itérations. La détection de boucle est basée sur la liste des derniers réseaux visités qui représente une fenêtre temporelle mise à jour après chaque itération. L'observation de la répétition du choix d'un réseau permet de réagir en modifiant le comportement de la méthode pendant la recherche. Pendant une itération, le choix du réseau ne se fera plus de façon *déterministe* selon le poids le plus fort. Le réseau sera choisi aléatoirement parmi l'ensemble des réseaux non tabous dont le poids n'est pas nul.

Contrairement au problème de k -coloration, tous les réseaux peuvent être sélectionnés. En coloration, nous avons distingué les variables générant le plus de conflits par rapport aux variables en conflits. Cependant, il existe un troisième type de variables, les variables ne générant pas de conflit. Par contre, pour ce problème, tous les réseaux peuvent être sélectionnés puisque même les réseaux réalisables qui satisfont les contraintes de réalisabilité citées par l'équation (3.11) peuvent contribuer à l'amélioration de la fonction de coût via le critère de taille de plan de fréquences.

Les paramètres de la détection de boucle sont identiques à ceux de la méthode appliquée en coloration. Lorsque le nombre de visites d'un réseau r est supérieur à une valeur seuil représentant un nombre d'occurrence autorisé occ lors des m dernières itérations, alors une boucle de répétition est détectée et une itération de type *diversifiante* est effectuée. La taille de la mémoire m des dernières itérations observées est définie en fonction du nombre de variables du problème. Ici, le nombre de réseaux total (cibles et hors-cibles)

du problème est utilisé, comme l'indique l'équation (3.19).

$$m = \frac{|R|}{2} \quad (3.19)$$

Le nombre d'occurrences autorisé occ est défini par un pourcentage de la taille de la mémoire fixé empiriquement à 5%.

$$occ = \lceil 0.05 \times m \rceil \quad (3.20)$$

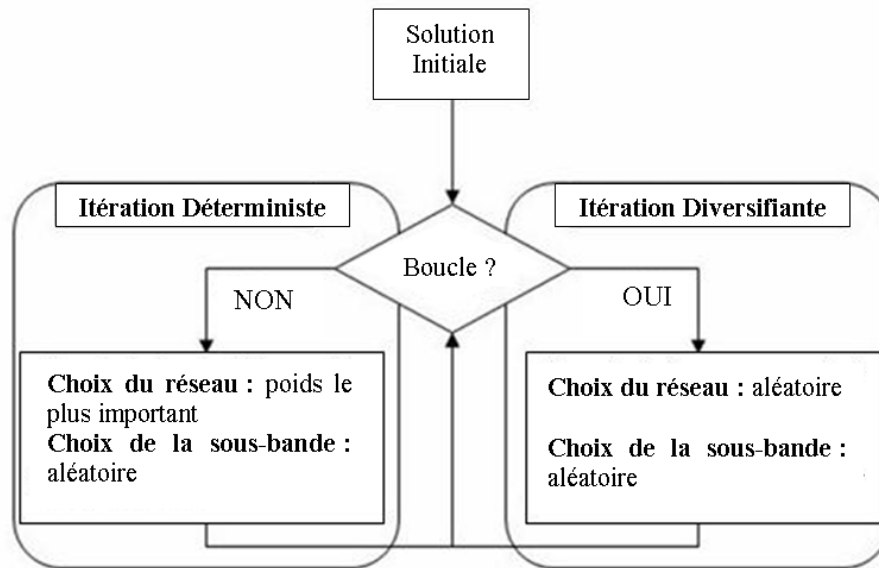


FIG. 3.10 – Mécanisme de détection de boucle

La figure 3.10 représente le fonctionnement de la méthode RLA appliquée au problème d'affectation de listes de fréquences. Les deux types d'itérations utilisés par la méthode diffèrent par la sélection du réseau à modifier. Pendant une itération déterministe, le choix de réseau est déterministe car le réseau de poids le plus élevé est choisi. Dans l'autre cas, le réseau est choisi aléatoirement parmi l'ensemble des réseaux.

Dans la méthode combinant la liste Tabou à la détection de boucle, seuls les réseaux non tabous peuvent être choisis quel que soit le type d'itération. Dans ce cas, deux options sont considérées pour la gestion du statut tabou : tous les réseaux choisis deviennent tabous ou seuls les réseaux détectés en boucle deviennent tabous.

Pour résumer, nous avons donc quatre méthodes concurrentes pour gérer le processus itératif :

- recherche locale basée sur le poids des réseaux,
- recherche locale avec liste Tabou,
- recherche locale avec détection de boucle,
- recherche locale avec détection de boucle et liste Tabou.

3.3.3.2 Choix de la sous-bande

Le choix de la sous-bande est identique pour toutes les méthodes de recherche locale énoncées. Cependant, plusieurs mécanismes de choix sont possibles. Cette sous-partie décrit puis analyse chacun d'eux.

3.3.3.2.1 Description globale

Le choix de la sous-bande est fait en fonction de leur contribution à la qualité de la solution. La contribution est calculée en fonction du TEB généré via l'équation (3.3) en considérant le plan de fréquences pdf_r , composé uniquement de la sous-bande sb .

Les critères de taille se rapportent à l'ensemble du plan de fréquences et ne peuvent donc pas être utilisés au niveau des sous-bandes. En effet, une sous-bande courte ou large ne constitue pas un critère de choix directement associé à une amélioration possible vis-à-vis du spectre. On peut tout aussi bien gagner en qualité à agrandir les unes ou les autres, cela dépend de la distribution globale des sous-bandes. Cependant, on distingue les cas d'une sous-bande vide (contribution à -1) du cas d'une sous-bande ne générant pas de conflit TEB (contribution à 0).

Trois procédés sont utilisés par la suite pour le choix de la sous-bande :

- choix aléatoire de la sous-bande parmi les sous-bandes non vides du plan de fréquences,
- choix proportionnel par rapport à la contribution des sous-bandes du plan de fréquences,
- choix de la sous-bande ayant la contribution maximale pour le plan de fréquences.

3.3.3.2.2 Influence sur la qualité de la solution finale

L'objet de cette étude est d'observer la qualité des solutions obtenues en fonction du mode de sélection des sous-bandes. Pour cela, différents tests ont été effectués sur les scénarios privés SC01, SC02, SC03, SC07 et SC08 pour une durée d'exécution d'une heure.

Le tableau 3.9 présente les résultats obtenus. Toutes les dégradations sont acceptées. Les solutions initiales sont générées par la méthode gloutonne "ordonnée" et seuls les mouvements simples sont utilisés (cf. partie 3.3.3.3.1). Les meilleurs scores sont en gras.

A partir de ces résultats, plusieurs remarques peuvent être formulées mais sans appor-

TAB. 3.9 – Influence du type de choix de la sous-bande

Scénario	SC01	SC02	SC03	SC07	SC08
Solution initiale	-1110	-2942	2,2E+06	24990	75,3E+06
Choix aléatoire	-4428	-2966	2,2E+06	22631	61,9E+06
Choix proportionnel	-3142	-2942	1,5E+06	24990	59,3E+06
Choix max.	-2237	-2950	2,2E+06	24143	66,4E+06

ter de conclusion définitive sur le meilleur principe.

1. Quel que soit le scénario, le choix de la sous-bande de contribution maximale n'a pas de meilleurs résultats que les deux autres choix. Ce principe ne permet pas de diversifier suffisamment la structure de la solution courante.
2. Les résultats obtenus par les deux autres choix de sous-bande sont très partagés. Par exemple, pour le scénario SC01, le choix de la sous-bande aléatoire donne une solution réalisable évaluée à -4428, mais pour le scénario SC03, le choix proportionnel donne un résultat sensiblement meilleur.

Nous avons poursuivi l'étude en analysant la structure des solutions obtenues en fonction de la méthode.

3.3.3.2.3 Impact sur l'occupation du spectre

Le niveau de TEB d'un plan est fortement dépendant du taux de recouvrement entre ses sous-bandes et les sous-bandes des autres plans de fréquences avec lesquels il partage des liens. Pour les trois types de choix de sous-bandes, nous avons comparé les plans de fréquences des solutions finales avec ceux de la solution initiale pour plusieurs scénarios afin de mesurer le degré de dispersion des sous-bandes dans le spectre. Les images suivantes représentent la solution initiale (figure 3.11) et la meilleure solution obtenue pour chaque choix de sous-bande (figures 3.12, 3.13 et 3.14) pour le scénario privé SC01. L'évaluation de ces solutions est donnée dans le tableau 3.9. En terme d'évaluation, la solution aléatoire est donc la meilleure. Chacune de ces figures présente une solution complète ; chaque ligne représente le plan de fréquences d'un réseau. L'abscisse représente le spectre de fréquences et l'ordonnée correspond aux différents réseaux. Les couleurs utilisées correspondent aux fréquences interdites en jaune (couleur claire), aux fréquences inutilisées en noir et les autres couleurs correspondent aux sous-bandes de pas différents (une teinte différente est donnée par pas).



FIG. 3.11 – Solution initiale pour la méthode gloutonne "ordonnée"



FIG. 3.12 – Choix de sous-bande aléatoire

On peut remarquer qu'il existe de grandes différences entre ces solutions. En ce qui concerne la réalisabilité, trois solutions sont réalisables, seule la solution obtenue par le

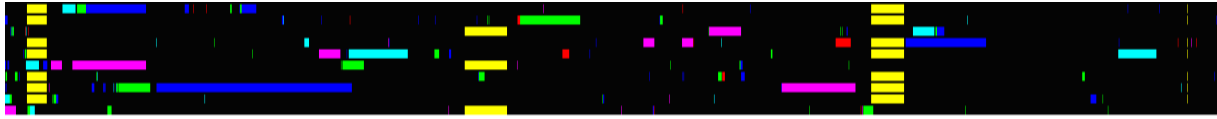


FIG. 3.13 – Choix de sous-bande proportionnel

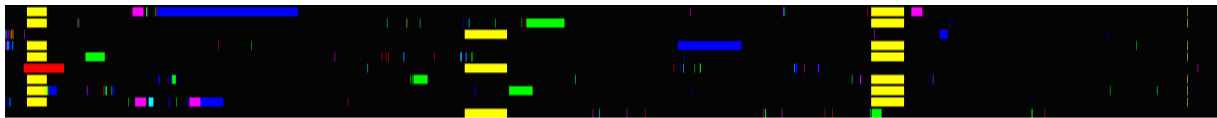


FIG. 3.14 – Choix de sous-bande de contribution maximale

choix de la sous-bande de contribution maximale (figure 3.14) ne l'est pas. En effet, un des réseaux ne respecte pas la taille minimale de 30 fréquences, il utilise uniquement 25 fréquences. En ce qui concerne le niveau d'interférence, toutes les solutions respectent les seuils de TEB maximal. La principale différence entre ces solutions est le nombre de fréquences utilisées. La solution initiale est la solution utilisant le moins de fréquences (300). La solution obtenue avec le choix aléatoire de la sous-bande utilise 1436 fréquences contre 1114 pour le choix proportionnel et 752 pour le choix de la sous-bande de contribution maximale. La solution utilisant le plus de fréquences est donc la meilleure.

En terme de dispersion des sous-bandes sur le spectre, on remarque que la plupart des sous-bandes de la solution initiale sont positionnées dans la première moitié du spectre de fréquences ce qui correspond à la méthode de génération de solution initiale utilisée. En découpant le spectre en 5 parties limitées par les fréquences interdites, on observe la répartition suivante : 19.3% des sous-bandes se situent dans la première partie, 53.4% pour la seconde, 7.3% pour la troisième, 20% pour la quatrième et 0% pour la dernière que nous noterons 19.3|53.4|7.3|20|0.

Pour les autres solutions, les répartitions en pourcentage du nombre de sous-bandes sont 17.5|39.1|28.7|13.3|1.4 pour la solution obtenue avec le choix aléatoire, 17.4|38.4|29.7|11.6|2.9 pour le choix proportionnel et enfin 16.6|44.1|26.9|11.7|0.7 pour le choix de contribution maximale. Les répartitions des solutions de contribution proportionnelle et du choix aléatoire sont plus équilibrées que celles de la solution initiale et de la contribution maximale.

Enfin, on observe que 37% des sous-bandes de la solution obtenue par le choix aléatoire sont composées d'une unique fréquence contre 55% pour le choix proportionnel et 66% pour le choix de contribution maximale. Au début de l'algorithme, les sous-bandes de fortes contributions sont pénalisées par leur niveau de TEB. La sélection de ces sous-bandes conduit donc l'algorithme à les réduire au maximum pour améliorer la qualité plutôt que de chercher à faire baisser le TEB en augmentant le nombre de fréquences de leurs brouilleuses. L'algorithme construit ainsi des solutions peu performantes sur le critère de réutilisation par les méthodes utilisant la contribution maximale ou proportionnelle.

Le nombre de fréquences identiques entre la solution finale et la solution initiale est un autre critère d'analyse intéressant. Les plans de fréquences des solutions basées sur le choix aléatoire, proportionnel et de contribution maximale utilisent respectivement 24, 22 et 19 fréquences communes avec la solution initiale ce qui représente environ 7% à 8% de fréquences communes quelle que soit la méthode. Toutes les méthodes sont donc relativement équivalentes sur ce critère. On remarque par contre que deux sous-bandes de la solution initiale se retrouvent à l'identique dans la solution obtenue par le choix de la contribution maximale. Ainsi, certaines sous-bandes de faible contribution ont été très rarement choisies par la méthode du choix de la sous-bande de contribution maximale mais aussi par celle utilisant le choix proportionnel.

3.3.3.2.4 Synthèse

Le choix de la sous-bande de contribution maximale est exclu du fait des mauvais résultats globaux après optimisation. Une analyse de la méthode et des résultats du choix proportionnel nous permet d'expliquer ces phénomènes. La contribution des sous-bandes définie à partir des valeurs de TEB n'est pas forcément la meilleure solution pour guider les choix. Les sous-bandes de faible contribution ne sont pas choisies alors qu'elles peuvent aussi améliorer globalement la solution. Par exemple, le fait d'agrandir une sous-bande de taille quelconque mais de faible contribution en TEB permet, en augmentant la taille du plan de fréquences, de contribuer à la satisfaction du critère sur le nombre minimum de fréquences du réseau et sur la réutilisation du spectre.

En plus, le fait d'ajouter des fréquences à un plan de fréquences peut, quelle que soit la sous-bande, faire baisser le TEB courant du réseau et des réseaux qu'il brouille du fait de l'évaluation basée sur la moyenne. En effet, le niveau de TEB d'un réseau est la valeur maximale de TEB générée par chacun des liens auquel il participe, dont la qualité est évaluée par la moyenne des valeurs de TEB élémentaires entre chacune des fréquences des plans des réseaux concernés (le récepteur et les brouilleurs). Par conséquent, toutes les sous-bandes peuvent donc contribuer à l'amélioration de la solution globale, soit en agissant directement sur la résolution d'un conflit de TEB, comme le font les méthodes de choix de sous-bande envisagées, soit en ajoutant des fréquences au plan indépendamment du TEB local de la sous-bande.

En conclusion, le choix aléatoire de la sous-bande est la meilleure méthode pour faire progresser les solutions sur les critères d'interférences et de taille. Chacune des sous-bandes d'un réseau ayant la même probabilité d'être choisie, des modifications peuvent donc être appliquées à chacune des sous-bandes et conduire à une meilleure occupation du spectre de fréquences.

3.3.3.3 Les opérateurs de mouvement

Le nombre et la variation des paramètres qui conditionnent la configuration des solutions sont importants : nombre de sous-bandes, tailles des sous-bandes, pas des sous-

bandes et fréquence inférieure des sous-bandes. L'étude combinatoire des scénarios a montré que l'espace des solutions est immense. Pour aborder cet espace, nous avons privilégié l'utilisation combinée d'opérateurs de mouvements qui définissent des voisinages très différents entre les solutions et donc qui autorisent différents types de déplacements à partir de la solution courante. Ils se limitent tous aux modifications sur une sous-bande de la solution courante. Ces voisinages autorisent des modifications parfois mineures (une fréquence d'une sous-bande change) et parfois importantes (toutes les fréquences d'une sous-bande changent) de la solution courante. Du fait de l'existence de ces multiples voisins, il y a plusieurs chemins de longueurs différentes reliant entre elles les mêmes solutions. L'utilisation d'un chemin particulier est induite par le choix de l'opérateur de mouvement correspondant.

Cette partie présente les différents types de mouvements. Elle est divisée en deux sous-parties : les mouvements simples dans un premier temps, puis les mouvements complexes combinant plusieurs mouvements simples. La définition des mouvements est en général guidée par le problème ; ils correspondent à des heuristiques dont l'objectif est de guider la recherche vers des solutions dont la nature nous semble plus favorable au problème.

3.3.3.1 Mouvements simples

Tous les mouvements utilisés peuvent être regroupés par catégorie en fonction de leur influence sur la taille du plan de fréquences. En effet, certains mouvements ne modifient pas la taille des plans de fréquences et d'autres ajoutent ou suppriment des fréquences. Le décalage d'une sous-bande dans le spectre ne modifie pas la taille du plan alors que l'agrandissement de la sous-bande et l'ajout d'une sous-bande augmentent le nombre de fréquences utilisées. Au contraire, la réduction et la suppression d'une sous-bande diminuent le nombre de fréquences du plan. Le changement de pas au sein d'une sous-bande peut quant à lui augmenter ou diminuer la taille du plan. La présentation de chaque catégorie de mouvements est suivie d'une analyse de ceux-ci par rapport au type de choix de sous-bande effectué.

a. Stagnation de la taille du plan

Le mouvement ne modifiant pas la taille de la sous-bande consiste à décaler une sous-bande dans le spectre de fréquences. Par la suite, il est noté "décalage". L'objectif de ce mouvement est de modifier la position de la sous-bande dans le domaine sans modifier ni le pas ni la taille de la sous-bande. Les décalages vers le bas ou vers le haut du spectre sont testés, et la meilleure possibilité est retenue. Ce mouvement ne peut être appliqué que lorsqu'il existe des fréquences libres au-dessus ou au-dessous de la sous-bande sélectionnée.

b. Augmentation de la taille du plan

- **Ajout de sous-bande :** Ce mouvement est effectué uniquement lorsque le numéro de sous-bande choisi aléatoirement correspond à un index de sous-bande libre. La nouvelle sous-bande doit obligatoirement être dans le domaine de ressource du ré-

seau choisi et ne pas chevaucher les sous-bandes existantes du plan. Le pas et la fréquence inférieure de la nouvelle sous-bande sont choisis aléatoirement. Différentes valeurs de fréquence supérieure sont testées et la meilleure est retenue.

- **Agrandissement de sous-bande :** Ce mouvement permet de modifier la taille de la sous-bande en décalant la fréquence inférieure ou la fréquence supérieure de la sous-bande jusqu'à atteindre la fréquence supérieure ou inférieure des autres sous-bandes du plan ou buter sur une fréquence interdite. L'agrandissement est fait de manière progressive en ajoutant une à une les nouvelles fréquences. Les fréquences ajoutées doivent être dans le domaine du réseau et respecter le pas de la sous-bande. Le meilleur agrandissement est conservé. La figure 3.15 schématise le fonctionnement de la méthode d'agrandissement à droite. Sur ce schéma, la sous-bande sélectionnée pour être modifiée se trouve entre une autre sous-bande du plan de fréquences et une sous-bande interdite. Une fréquence est ajoutée sur la partie supérieure de la sous-bande à modifier puis l'ajout est évalué. Ce procédé est répété tant qu'il est possible d'ajouter une fréquence à la sous-bande. Sur ce schéma, seules deux fréquences sont ajoutables à droite.

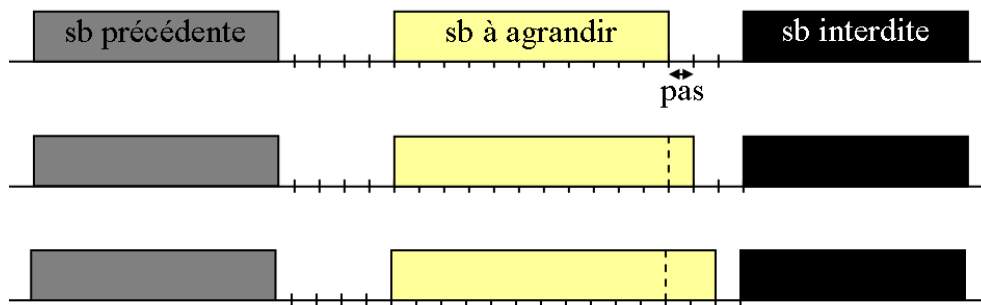


FIG. 3.15 – Les possibilités d'agrandissement d'une sous-bande

- **Réduction du pas de sous-bande :** Le mouvement de changement de pas permet de modifier le pas de la sous-bande sans modifier la fréquence inférieure. Tous les pas possibles strictement inférieurs au pas courant sont testés et le meilleur est sélectionné. Le changement de pas conserve la borne inférieure de la sous-bande et ajuste la borne supérieure en fonction du nouveau pas en restant à l'intérieur de la sous-bande.

c. Diminution de la taille du plan

- **Réduction de taille de sous-bande :** Comme pour le mouvement d'agrandissement, le mouvement de réduction choisit la meilleure réduction possible en modifiant soit la fréquence inférieure soit la fréquence supérieure de la sous-bande sélectionnée fréquence par fréquence. Ce mouvement peut aboutir à la suppression de la sous-bande sélectionnée. La suppression est possible uniquement lorsque le plan de fréquences du réseau sélectionné contient d'autres sous-bandes. En effet, un plan de

fréquences ne peut pas être vide.

- **Augmentation du pas de sous-bande** : Comme pour le mouvement de réduction du pas, le mouvement d'augmentation du pas ne modifie pas la fréquence inférieure et ajuste la fréquence supérieure à l'intérieur de l'intervalle de la sous-bande en fonction du nouveau pas. Tous les pas possibles strictement supérieurs au pas courant sont testés et le meilleur est sélectionné.

3.3.3.3.2 Mouvements combinés

Nous avons observé des particularités des domaines de certains problèmes tels que les peignes de fréquences interdites qui rendent difficiles les modifications de sous-bande avec les mouvements simples. Ces propriétés demandent la mise au point de voisinages spécifiques qui correspondent à des combinaisons de mouvements simples. Ces combinaisons peuvent être atteintes par l'algorithme par l'enchaînement de deux procédures de voisinage adéquat mais leur probabilité d'être utilisée successivement est faible : la sous-bande doit être choisie deux fois. Nous avons donc créé des mouvements combinés particuliers : le "changement de pas avec découpage" et le "changement de pas avec agrandissement".

a. Changement de pas avec découpage

Différents scénarios utilisent des domaines avec des fréquences interdites isolées. A plusieurs reprises nous avons observé des sous-bandes chevauchant ces fréquences interdites ; la figure 3.16 montre une solution obtenue pour le scénario SC02 et un agrandissement sur le spectre avec des fréquences interdites isolées.

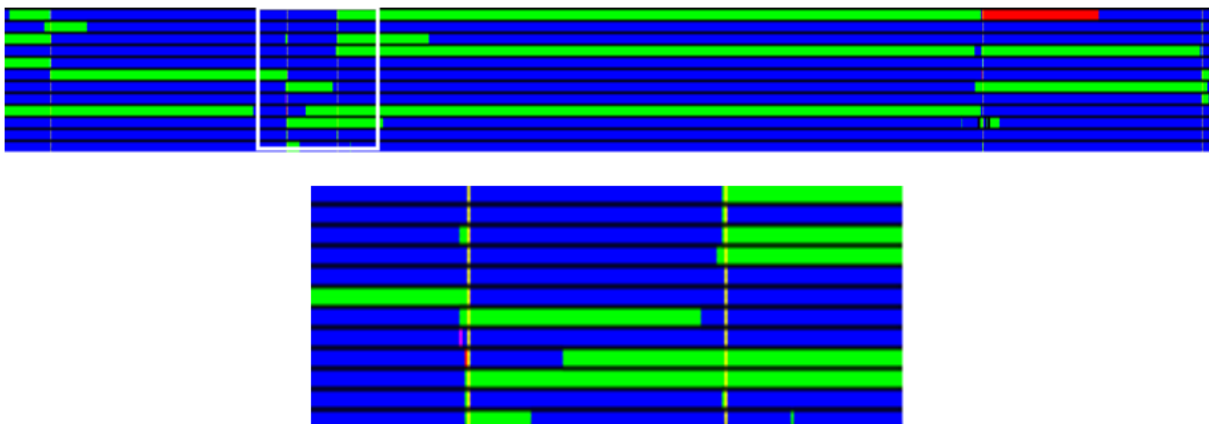


FIG. 3.16 – Chevauchement de fréquences interdites dans le scénario SC02

Les sous-bandes bleues (en gris foncé) correspondent à un pas de 1, en vert (en gris clair) à un pas de 2, et en rouge à un pas de 4. On remarque bien que chacune des sous-bandes vertes chevauche une fréquence interdite.

Si on utilise un changement de pas sur ce type de sous-bande, la fréquence interdite conduit très souvent à une augmentation du pas et donc à une diminution du nombre de fréquences de la sous-bande. Le mouvement conduit donc à diminuer le taux de réutilisation du spectre et dégrade systématiquement la fonction de coût de ce type de solution.

Afin de limiter la perte de fréquences, un nouveau mouvement (noté par la suite CD pour changement de pas avec découpage) permet de découper une sous-bande en 2 lorsque celle-ci chevauche une fréquence interdite. Ensuite un changement de pas selon la procédure définie auparavant est effectué sur une des deux parties de cette sous-bande. Le changement de pas se fait aléatoirement sur un des deux côtés.

TAB. 3.10 – Mouvement de changement de pas avec découpage

	SC01	SC02	SC03	SC04	SC05	SC10
Sans CD	-42 486	-100 010	-904	-2,47E+06	-380 698	26E+06
Avec CD	-41 125	-108 288	-140 518	-2,46E+06	-383 317	28E+06

Ce mouvement permet d'éviter les fréquences interdites isolées et améliore le score sur les scénarios concernés comme indiqué au tableau 3.10 mais il n'est pas adapté au problème des peignes de fréquences interdites.

NB : La méthode d'intensification employée une fois à la fin de l'algorithme (cf. section 3.3.1) utilise deux types de mouvements : un changement de pas avec découpage si le nombre de sous bandes dans le plan le permet ($|pdf_r|_{sb} < SB_{max}$) ou un simple changement de pas. Contrairement à l'application des mouvements lors de l'optimisation, en fin de processus un changement de pas (avec ou sans découpage) d'une sous-bande n'est accepté que s'il entraîne l'amélioration de la meilleure solution. Dans le cas contraire, la solution précédente est maintenue et on passe à une autre sous-bande.

b. Changement de pas avec agrandissement

Les peignes de fréquences interdites peuvent conduire à fabriquer des petites sous-bandes imbriquées entre les fréquences interdites. Sur l'image de la solution obtenue pour le scénario SC03 (en figure 3.17), on remarque bien les petites sous-bandes à l'intérieur du peigne. La plupart d'entre elles ont un pas de 1 et, à cause du peigne et de leur proximité, elles ne peuvent être ni agrandies ni décalées. Un changement pour un pas plus grand sur ces sous-bandes serait utile mais il entraîne une réduction du nombre de fréquences et, très probablement, une forte dégradation de la solution.

De plus, ce type de solution présente beaucoup d'espace inutilisé. Pour la plupart des réseaux, le nombre maximal de sous-bande est déjà atteint du fait du peigne et ainsi aucune sous-bande ne peut être ajoutée dans les espaces libres. Le plan de fréquences devient rapidement composé d'un grand nombre de sous-bandes de petites tailles ce qui le rend peu attractif du point de vue des critères de taille.

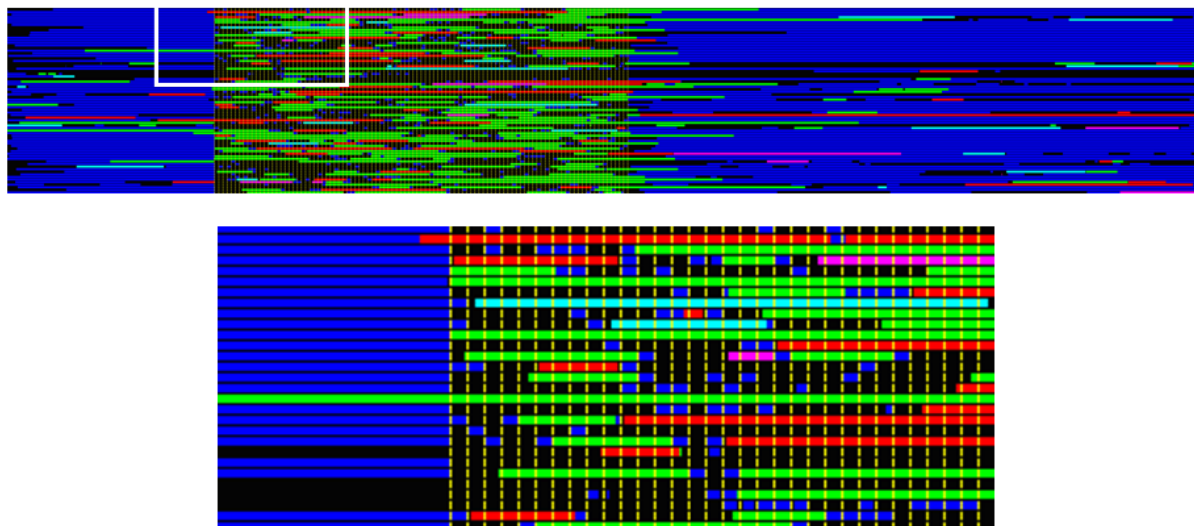


FIG. 3.17 – Présence d'un grand peigne dans le scénario SC03

Le mouvement combiné "changement de pas avec agrandissement" (noté par la suite CA) est une combinaison du mouvement de changement de pas et de celui d'agrandissement à droite. Le changement de pas peut permettre d'enjamber les peignes de fréquences interdites et pour chaque pas différent du pas d'origine, un agrandissement à droite est effectué de façon à ajouter des fréquences à la solution et donc améliorer la fonction de coût sur ce critère. Toutes les fréquences pouvant être ajoutées à la sous-bande sont testées comme le montre le schéma 3.15. La meilleure solution est conservée. Ainsi, les petites sous-bandes coincées entre deux fréquences du peigne peuvent être modifiées par l'action combinée des deux mouvements. Les résultats obtenus par l'ajout de ce nouveau mouvement sont présentés dans le tableau suivant.

TAB. 3.11 – Mouvement de changement de pas avec agrandissement

	SC01	SC02	SC03	SC04	SC05	SC10
Sans CA	-41 328	-108 959	-509 083	-2,41E+06	-389 489	20,8E+06
Avec CA	-41 896	-106 099	-998 652	-2,44E+06	-425 777	15,5E+06

Ce mouvement apporte un gain pour la plupart des scénarios. Dans le cas du scénario SC02, les deux évaluations sont quasiment identiques.

3.3.3.3 Impact du choix des mouvements

Le but de cette partie est d'étudier chacun des mouvements simples que nous avons défini. Pour cela, nous analyserons le comportement de chaque mouvement par rapport à la méthode de choix de la sous-bande précédemment étudié dans la section 3.3.3.2. Dans cette étude, les mouvements sont choisis au hasard de façon équiprobable à chaque itération. Nous nous sommes intéressés à l'effet des mouvements sur la solution courante en terme d'amélioration ou de dégradation de la qualité de celle-ci. Deux scénarios privés

SC01 et SC03 ont été utilisés. Ces deux scénarios sont très différents. Le scénario SC01 est réalisable alors que le scénario SC03 ne l'est pas.

Les résultats sont présentés en figures 3.18 et 3.19. Deux critères d'analyse sont représentés dans les histogrammes :

- En (a), la moyenne des différences de fonction de coût entre la solution courante et la solution précédente par rapport au type de mouvement effectuée intitulée "gain moyen". Une différence positive indique une dégradation et une valeur négative une amélioration.
- En (b), pour chaque mouvement, la proportion d'itérations de ce mouvement ayant conduit à une amélioration de la fonction de coût de la solution courante intitulée "proportion d'amélioration".

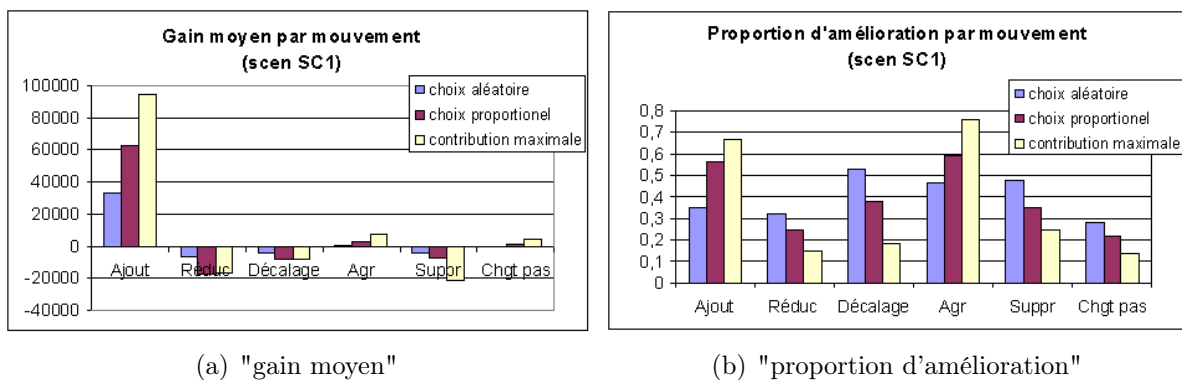


FIG. 3.18 – Impact des mouvements sur la qualité sur le scénario SC01

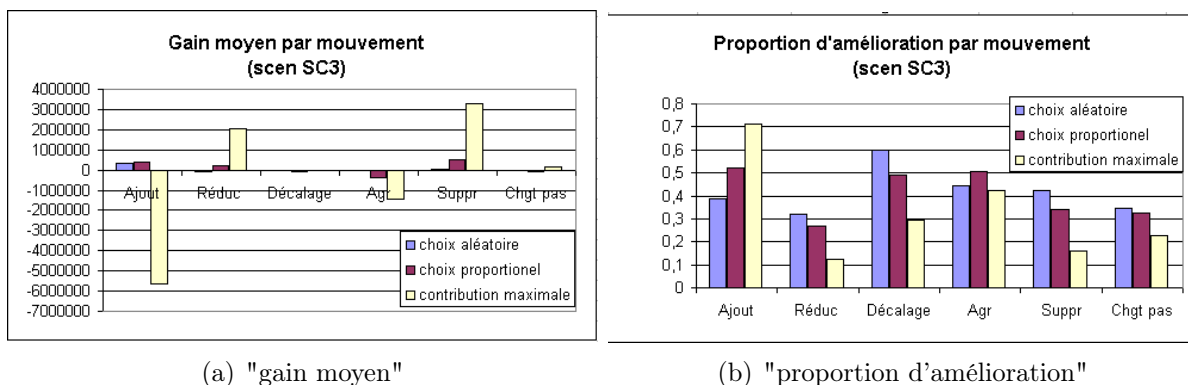


FIG. 3.19 – Impact des mouvements sur la qualité sur le scénario SC03

A partir de cette étude, on remarque plusieurs points pour chaque type de mouvement :

- **Mouvements conservant la taille (décalage) :** Ce type de mouvement améliore en moyenne la solution courante quel que soit la méthode de choix de la sous-bande. Cependant, les choix de sous-bande de contribution proportionnelle et de contribution maximale paraissent donner de meilleurs résultats que le choix aléatoire sur le scénario SC01. Il n'y a pas d'effet sur le scénario SC03.
- **Mouvements augmentant la taille (ajout, agrandissement) :** Les méthodes d'ajout et d'agrandissement ne se comportent pas de la même façon pour le scénario SC01 et pour le scénario SC03. Pour le scénario SC01, l'ajout d'une sous-bande améliore plus souvent la solution courante lorsque la sous-bande à modifier est choisie en fonction de la contribution maximale bien que ce choix de la sous-bande donne en moyenne de plus grandes dégradations en terme d'amplitude. Pour ce scénario, le choix aléatoire obtient le meilleur gain moyen pour les mouvements d'ajout et d'agrandissement, bien qu'en proportion le nombre d'ajout ou d'agrandissement ayant amélioré la solution courante soit le plus faible. Pour le scénario SC03, on observe que l'ajout et l'agrandissement lors du choix de la sous-bande de contribution maximale permettent d'obtenir le meilleur gain moyen. Au final, d'un scénario à l'autre l'intérêt de ce type de mouvement est complètement opposé.
- **Mouvements réduisant la taille (réduction, suppression) :** La réduction et la suppression ont aussi des comportements bien différents selon le scénario. Pour le scénario SC01, ces deux types de mouvements ont tendance à améliorer en moyenne la valeur de la solution courante. Pourtant, peu de mouvements de ces deux types apportent des améliorations (autour de 25% en moyenne). On observe que pour le choix de sous-bande aléatoire, plus de 30% des mouvements de réduction apportent une amélioration pour une amélioration moyenne sur tous les mouvements de réduction de $-10\,000$. Pour la réduction, le choix de la sous-bande proportionnellement à la contribution paraît donner de meilleurs résultats. Ce choix améliore dans 20% des cas pour une amélioration moyenne de $-20\,000$ environ. Le fait de pouvoir supprimer la sous-bande de contribution maximale donne de très bons résultats en terme d'amélioration de l'évaluation totale de la solution courante.

Sur le scénario SC03, les observations sont totalement opposées. En effet, les mouvements de réduction et de suppression aboutissent à une dégradation moyenne très importante de la solution courante : de l'ordre de $2\,000\,000$ pour le mouvement de réduction et de $3\,000\,000$ pour la suppression lorsque la sous-bande de contribution maximale est choisie. Ces deux mouvements obtiennent un gain moyen meilleur lorsque la sous-bande est choisie aléatoirement. Pour ce type de mouvement non plus, il n'y a pas de résultat valable pour tous les scénarios.

- **Le changement de pas (chgt pas) :** Cette méthode améliore la solution courante au maximum dans 35% des cas. L'amélioration obtenue est en moyenne très faible sur les deux scénarios étudiés.

Globalement, pour les 2 scénarios la proportion d'amélioration sur l'ensemble des mouvements est supérieure avec le choix de sous-bande aléatoire. Par exemple pour le scénario SC01, 40% des mouvements améliorent la solution courante avec le choix de sous-bande aléatoire contre 37% pour le choix proportionnel et 32% pour le choix de contribution maximale. Au niveau du gain moyen, il y a beaucoup de disparité entre les mouvements et les problèmes mais le choix aléatoire de sous-bande présente les plus faibles amplitudes de dégradation et donc constitue le choix de sous-bande le moins risqué. En conclusion sur le choix de la sous-bande, cette nouvelle étude conforte le fait que le choix aléatoire semble le plus adéquat pour un ensemble de problèmes.

En ce qui concerne les mouvements, il est plus difficile de conclure. Sur ces deux scénarios, les mouvements donnent des résultats totalement opposés. En moyenne, chacun apporte un gain ou une dégradation selon le scénario. A noter que dans le cadre du choix aléatoire, on observe en figures (b) que chacun des mouvements permet d'améliorer la solution courante au minimum 1 fois sur 4. En conclusion, chacun des mouvements a son importance et le choix des meilleurs mouvements ne peut se faire que dans le cadre d'un scénario particulier. Dans le paragraphe qui suit, nous avons étudié la possibilité d'une sélection automatique des mouvements en fonction de leur performance pour chaque scénario.

3.3.3.3.4 Choix du mouvement adaptatif versus aléatoire

Étant donné le nombre important de voisinages définis, nous nous sommes intéressés à la fréquence d'appel de chacun des mouvements. Le choix du mouvement doit-il être aléatoire ? Ou faut-il l'adapter au cours de la recherche ? C'est pour répondre à cette question qu'une étude sur une procédure de choix de mouvements adaptatifs a été réalisée.

Pour les hyper-heuristiques, Cowling *et al.* [33–35] ont proposé l'utilisation d'une fonction de choix basée sur les performances précédentes de différentes heuristiques (pour plus de détails, cf. section 1.2.2.4). Nous avons aussi choisi d'utiliser les performances des différents mouvements effectués par un système de pénalités et de récompenses. Dans un premier temps, tous les mouvements ont la même probabilité d'être appelés. Puis, si le mouvement appliqué améliore la solution courante alors sa probabilité d'être à nouveau appelé augmente. Inversement, si le mouvement cause une dégradation sa probabilité diminue. Comme tous les mouvements ne peuvent pas forcément être appliqués à une sous-bande donnée, la probabilité qu'un mouvement soit utilisé dépend aussi du nombre de mouvements applicables sur la sous-bande sélectionnée.

Le poids affecté à chacun des mouvements est initialisé à une valeur entière p , et décrémenté ou incrémenté de 1 en fonction de l'évaluation de la solution après application du mouvement. De plus, une borne inférieure et une borne supérieure ont été fixées afin d'éviter qu'un mouvement ait un poids nul, ou inversement, qu'un seul mouvement de l'ensemble soit utilisé. Les tests présentés dans le tableau 3.12 correspondent aux résultats obtenus pour les scénarios privés. Pour le choix de mouvement adaptatif, le poids p est initialisée à 25 pour chacun des mouvements. Les bornes inférieures et supérieures

sont fixées à 1 et 50. Ces valeurs ont été fixées arbitrairement après quelques essais sur des intervalles plus ou moins grands.

TAB. 3.12 – Choix adaptatif et choix aléatoire des mouvements

	SC01	SC02	SC03	SC04	SC05
aléatoire	-39 035	-28 539	-566 848	-1,6E+06	-385 876
adaptatif	-17 586	-30 229	503 519	-1,2E+06	-139 275
	SC06	SC07	SC08	SC09	SC10
aléatoire	-94 932	-191 610	26,7E+06	138 515	14E+06
adaptatif	508 353	-7 138	43,6E+06	263 603	30E+06

Le choix adaptatif donne de très mauvais résultats pour presque tous les scénarios. De façon générale, la méthode adaptative a favorisé les mouvements d'agrandissements parce que ceux-ci apportent de nombreuses améliorations successives au début de l'exécution. La méthode d'agrandissement a alors été appelée plus souvent que les autres méthodes. Cependant les améliorations progressives sont faibles et au bout d'un moment, les sous-bandes agrandies sont difficilement manipulables par les autres mouvements. Les solutions se retrouvent alors bloquées dans les optimums locaux malgré les différentes règles de voisinage. D'autres paramètres ont été testés en réduisant l'intervalle des poids possibles, par exemple, entre [1 ; 5]. Les tests effectués montrent des résultats semblables.

En conclusion, bien que le choix de mouvements adaptatifs paraisse intéressant en théorie sur ce problème réel, les résultats obtenus ne permettent pas de le confirmer avec une méthode de pénalité/récompense adaptative. D'autres études seraient nécessaires sur ce problème.

3.3.3.4 Le contrôle de dégradation

Le problème d'affectation de listes de fréquences utilise une fonction mono-objectif agrégeant plusieurs critères où les poids associés à chaque critère sont définis pour chaque instance et peuvent être très importants. A cause de cette formulation, certains mouvements peuvent dégrader énormément la qualité de la solution courante. La figure 3.20 montre l'évolution de la qualité de la solution courante au cours des itérations. La méthode présentée n'utilise pas de contrôle de dégradation. Cette courbe montre beaucoup de brusques et importantes dégradations. Après chaque dégradation, la recherche nécessite souvent un grand nombre d'itérations pour retrouver une solution de qualité similaire. Un mécanisme de contrôle de la dégradation permettrait de conditionner l'acceptation des solutions dégradées. Plusieurs types de contrôle de dégradation présents dans la littérature sont introduits puis comparés sur un ensemble de scénarios.

3.3.3.4.1 Description des méthodes

Les mécanismes de contrôle de dégradation sont de différents types : le contrôle du nombre de dégradations acceptées, de l'amplitude de la dégradation acceptable et un

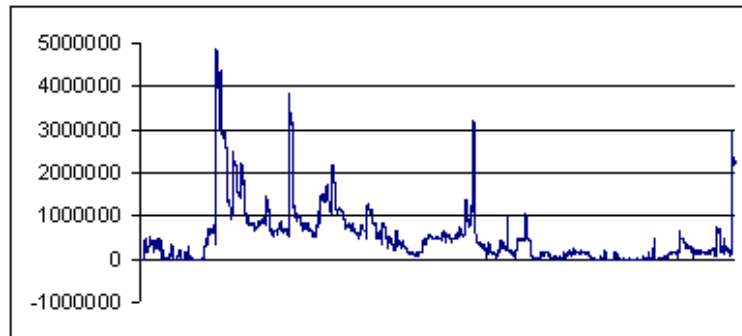


FIG. 3.20 – Variation de la fonction de coût sans contrôle de dégradation (scénario SC01)

contrôle de dégradation combinant le nombre et l'amplitude des dégradations.

a. Le contrôle en nombre de dégradation

Le contrôle est mis en application par une probabilité, $p=1/n$, d'acceptation de chaque dégradation ou par une acceptation systématique d'une dégradation après chaque n dégradations. Le contrôle des dégradations en nombre permet de contrôler le nombre de dégradations acceptées sans prendre en compte l'amplitude. C'est la procédure la plus simple ; elle conduit à s'éloigner d'un optimum local à une fréquence régulière.

b. Le contrôle en amplitude de dégradation

Après une itération dégradante, l'amplitude de la dégradation est mesurée et la probabilité d'accepter cette dégradation est attribuée à la nouvelle solution en fonction de l'amplitude de la dégradation. Plus la dégradation est importante, plus la probabilité de l'accepter est petite. Deux fonctions différentes ont été utilisées : une fonction linéaire $p(x) = a \times x + b$, avec $a < 0$, et une fonction exponentielle $p(x) = a \times \exp(\frac{-x}{b})$, avec x l'amplitude de la dégradation de la solution modifiée par rapport à la solution courante. La seconde formulation reprend le procédé de simulation de Metropolis utilisé dans le recuit simulé [58, 83].

c. Le contrôle par exploration partielle du voisinage

L'exploration partielle du voisinage signifie qu'un sous-ensemble de solutions voisines de la solution courante est évalué lors de chaque itération. Une solution voisine sera choisie en fonction de sa qualité et de celle des autres voisines évaluées. Deux sortes d'approche d'exploration partielle ont été évaluées. La première consiste à choisir la meilleure solution parmi les n voisins visités (noté VPn). Dans la seconde, la solution est choisie aléatoirement parmi les $(n - d)$ meilleurs voisins (noté $VPn-d$). Ce mécanisme de contrôle de dégradation est utilisé par la méthode IDWalk [102], proposée par Neveu *et al.* Il s'agit donc d'un mécanisme qui tient compte à la fois de la valeur des voisines et du nombre de solutions évaluées.

d. Le contrôle de dégradation en nombre et en amplitude

En plus de réduire le nombre de dégradations admises, cette technique fixe un seuil de dégradation correspondant à l'amplitude maximale de la dégradation tolérée par rapport à la solution précédente ou à la meilleure solution. Quand la dégradation est au-dessous de ce seuil, la solution est acceptée selon une probabilité donnée comme dans la première méthode. Il s'agit donc d'une combinaison des deux premières méthodes.

Le mécanisme utilisé de contrôle de la dégradation est basé sur la qualité de la solution courante avant la modification s_c , de la solution modifiée s_m et enfin de la meilleure solution s^* . La probabilité de choisir une solution dégradante est calculée de la façon suivante :

$$s_m \text{ acceptée ssi } \begin{cases} F(s_m) - F(s^*) < a \times (F(s_c) - F(s^*)) + c \\ \text{et} \\ rand(100) < p \end{cases} \quad (3.21)$$

Les paramètres a et c indiquent les paramètres du contrôle de l'amplitude de la dégradation. La valeur de c correspond au niveau de dégradation autorisé lorsque la solution courante est la meilleure solution. L'amplitude de la dégradation maximale acceptée augmente proportionnellement via le paramètre a avec la différence d'évaluation entre la solution courante et la meilleure. Ainsi, on empêche tout éloignement soudain de la meilleure solution rencontrée comme l'indiquait la figure 3.20. Enfin, le paramètre p représente le pourcentage de dégradations acceptées respectant la contrainte d'amplitude.

La figure 3.21 schématise le fonctionnement du contrôle de dégradation en amplitude de cette méthode en fonction de la valeur du paramètre a . Sur ces images, les droites $F(s^*)$, $F(s^*) + c$ et $F(s^*) + 2c$ servent de repère afin d'estimer l'amplitude des futures dégradations au fur et à mesure que l'évaluation de la solution courante s_m s'éloigne de celle de la meilleure solution s^* . Lorsque $a = 0$ (première courbe), la solution s_m doit vérifier l'équation $F(s_m) - F(s^*) < c$ pour être acceptée. En d'autres termes, le contrôle de la dégradation se fait en fonction de la meilleure solution s^* . Lorsque $a \neq 0$ (deuxième et troisième courbes), le cumul des dégradations est autorisé. Lorsque $a < 1$, au fur et à mesure que l'on accepte des dégradations, l'amplitude des dégradations acceptables diminue. A l'inverse, lorsque $a > 1$, plus on accepte des dégradations, plus l'amplitude des dégradations acceptables augmentent.

Le paramètre c correspondant à l'amplitude maximale acceptée peut soit être fixé pour toute la recherche, soit varier au cours du temps pour intensifier la recherche à la fin de la recherche. Quatre types de méthodes ont été testés :

- amplitude fixe notée $FIX[c]$
- évolution linéaire notée $LIN[c_{max} - c_{min}] : c_t = c_{max} - (c_{max} - c_{min}) \times t$
- évolution cubique notée $CUB[c_{max} - c_{min}] : c_t = c_{max} - (c_{max} - c_{min}) \times t^3$
- évolution racine cubique notée $RAC[c_{max} - c_{min}] : c_t = c_{max} - (c_{max} - c_{min}) \times \sqrt[3]{t}$

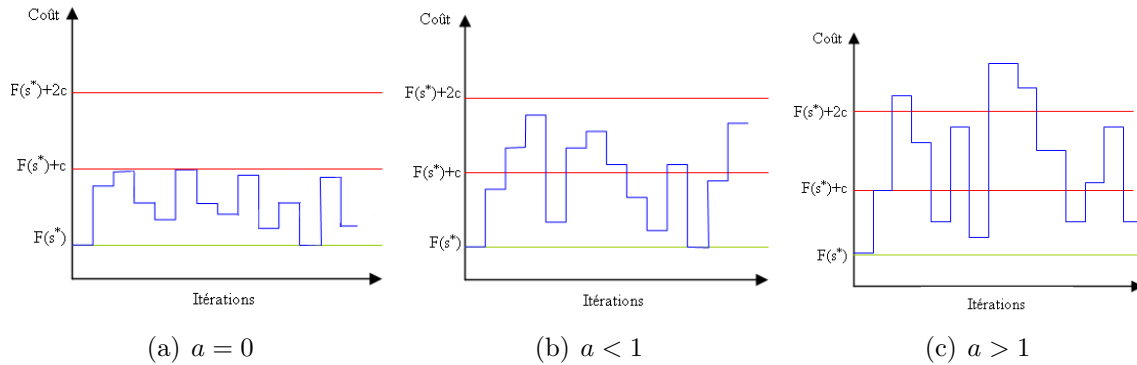


FIG. 3.21 – Influence du paramètre a sur l'évolution de l'amplitude des dégradations acceptées

Avec c_t qui est le seuil de dégradation accepté à l'instant t et la variable t qui représente l'évolution du temps au cours de la recherche. La variable t est initialisée à 0 au début de l'exécution et prend la valeur 1 à la fin. L'amplitude c_t varie dans un intervalle $[c_{min}, c_{max}]$.

3.3.3.4.2 Résultat et analyse

Dans cette section, nous présentons les tests réalisés sur des instances privées. Le tableau 3.13 présente sur les deux scénarios SC01 et SC03 un classement des quatre options de la méthode de contrôle de dégradation en nombre et en amplitude (contrôle (d)). La meilleure méthode est en haut du tableau. L'option avec évolution linéaire n'est jamais classée dans les dix premiers donc elle n'est pas présentée dans le tableau. L'option la plus performante et qui a été retenue pour être comparée avec les autres méthodes de contrôle de la dégradation est la méthode FIX.

TAB. 3.13 – Classement des méthodes de contrôle de dégradation

SC01		SC03	
méthode	score	méthode	score
FIX[1000]	-43534	FIX[1000]	-1172070
CUB[30000-2000]	-42327	CUB[30000-2000]	-1027311
CUB[30000-10000]	-42297	FIX[10000]	-789820
RAC[30000-5000]	-42285	RAC[30000-10000]	-718672
RAC[30000-2000]	-42169	RAC[10000-2000]	-679757
CUB[30000-5000]	-42119	CUB[30000-10000]	-674997
RAC[30000-10000]	-41729	RAC[30000-5000]	-631194
RAC[10000-1000]	-41630	FIX[0]	-628425
RAC[10000-2000]	-41050	RAC[10000-1000]	-617749
FIX[10000]	-40517	CUB[30000-5000]	-563173
FIX[0]	-38236	RAC[30000-2000]	10271

Le tableau 3.14 présente l'évaluation de la meilleure solution obtenue au cours d'une

unique exécution avec les différents mécanismes de contrôle de dégradation indiqués en première colonne et notés de (a) à (d).

TAB. 3.14 – Procédures de contrôle de dégradation sur les instances privées

Méthode	Paramètres	SC01	SC02	SC03	SC04	SC05	SC10
Sans contrôle		-10525	-23459	14e+06	-462548	-175919	24e+06
En nombre (a)	$p=1\%$	-39 397	-98 422	-935 907	-2,48e+06	-377 634	20e+06
En amplitude fct linéaire (b)	$b=e/100$ $a=-b/100000$	-41 563	-112 969	-882 070	-2,45e+06	-402 937	13,6e+06
En amplitude fct exp (b)	$a=e/100$ $b=35 000$	-39 466	-114 368	-620 653	-2,40e+06	-458 365	6,9e+06
Voisinage partiel VPn (c)	VP10	-22 952	-20 284	-496 308	-1,89e+06	-476 718	25,4e+06
Voisinage partiel VPn-d (c)	VP10-2	-39 238	-37 996	-927 408	-2,00e+06	-375 661	18,6e+06
En nombre et en amplitude (d)	$p = 1\%, a = 1.4$ FIX[30000]	-40 836	-114 368	-863 194	-2,31e+06	-466 370	19e+06

Pour chaque scénario, les 4 meilleures méthodes sont classées : plus le résultat est bon, plus la couleur est foncée. La méthode sans contrôle de dégradation ne permet en aucun cas d'obtenir une solution classée dans les 4 premières. Le contrôle de la dégradation améliore nettement les résultats. Chaque mécanisme donne des résultats intéressants cependant il n'y a aucune méthode qui surpasse toutes les autres pour toutes les instances testées, et parfois les écarts relatifs sont minces. Cette observation peut être expliquée par la sensibilité de la méthode aux paramètres. En d'autres termes, pour les 6 instances étudiés, il est difficile de trouver une valeur de paramètre qui soit adéquat à toutes les situations. Par exemple, la méthode VPn permet d'obtenir de bons résultats pour le scénario SC05 tandis que l'intensification agressive exécutée par cette méthode n'est pas adaptée au scénario SC02. De la même manière, une méthode basée sur l'exploration comme la méthode basée sur le contrôle en nombre et en amplitude (dernière ligne) donne des résultats très bons pour les scénarios 1, 2 et 5 mais obtient de moins bons résultats pour le scénario 10.

Parmi les deux mécanismes toujours classés dans les quatre premiers scores, le contrôle en amplitude avec fonction linéaire (ligne 3) et le contrôle en nombre et en amplitude (dernière ligne), nous avons arbitrairement choisi ce dernier car elle nous paraît offrir le contrôle le plus adaptatif.

3.4 Résultats et analyse sur les scénarios

Cette partie est divisée en deux : tout d'abord les résultats sur les scénarios publics sont présentés suivis de ceux obtenus à la fin du contrat sur les scénarios privés.

3.4.1 Scénarios publics

Cette section compare les différentes méthodes de recherche locale avec les mécanismes de détection de boucle et de liste Tabou. Toutes les méthodes ont la même architecture globale décrite en figure 3.7.

Le tableau 3.15 présente tout d'abord l'évaluation des meilleures solutions obtenues sur une seule exécution pour les 10 scénarios publics. Les deux premières méthodes en colonne n'emploient pas de mécanisme de détection de boucle, ce sont des méthodes basées sur l'utilisation d'itérations de type recherche locale basée sur la résolution de conflits. Lors de chaque itération, le réseau choisi est toujours celui de poids le plus fort (cf. section 3.3.3.1) et la sous-bande est choisie aléatoirement. La différence entre les deux méthodes repose sur l'utilisation ou non d'une liste Tabou. La durée Tabou employée est définie dynamiquement selon la formule (3.18) qui correspond au standard de la littérature. Les trois autres méthodes sont basées sur le mécanisme de détection de boucle avec la combinaison des deux méthodes illustrée en figure 3.10 : sans liste Tabou sur les réseaux (4^e colonne), avec une liste Tabou sur tous les réseaux ayant déclenché une boucle (5^e colonne) et avec une liste Tabou sur tous les réseaux visités (6^e colonne).

TAB. 3.15 – Influence de la détection de boucle et de la liste Tabou

problèmes	plus fort conflit		détection de boucle		
	sans tabou	avec tabou	sans tabou	avec tabou	
				les réseaux bouclants	tous les réseaux
PUB01	-10 426	-10 827	-13 764	-13 892	-12 111
PUB02	-928	-961	-1 353	-1 436	-1 447
PUB03	-8 110	-9 516	-13 932	-13 952	-14 574
PUB04	-15 843	-17 515	-27 866	-27 837	-27 786
PUB05	-7 309	-9 654	-10 720	-12 144	-10 912
PUB06	515 417	95 906	71 738	96 314	70 360
PUB07	-23 554	-53 777	-478 471	-476 693	-484 297
PUB08	-194 435	-336 741	-887 354	-869 292	-878 640
PUB09	1 667 459	962 109	876 409	755 753	684 699
PUB10	-5 672	-7 541	-10 731	-10 624	-10 680

Une première comparaison peut être faite entre les deux méthodes sans mécanisme de détection de boucle. Pour tous les scénarios, l'ajout de la liste Tabou dynamique permet d'améliorer les résultats précédents, l'utilisation de la diversification à moyen terme par restriction du voisinage est donc efficace. Puis, une comparaison peut être faite avec la détection de boucle. L'utilisation du mécanisme de détection de boucle sans liste Tabou améliore considérablement les résultats sur tous les scénarios. La détection de boucle pour diversifier par extension du voisinage est donc plus efficace que la liste Tabou. Enfin, combiner la liste Tabou à la détection de boucle permet d'améliorer les meilleurs résultats pour 7 cas sur 10. Globalement, la meilleure exécution pour chaque exemple (en gras) a été obtenue avec une des trois méthodes en utilisant le mécanisme de détection de boucle.

TAB. 3.16 – Combinaison de la détection de boucle et de la liste Tabou (sur 5 exécutions)

problèmes	détection de boucle		
	sans	avec tabou	
	tabou	les réseaux bouclants	tous les réseaux
PUB01	-13 981	-14 359	-14 828
PUB02	-1 538	-1 507	-1 543
PUB03	-13 466	-13 905	-13 625
PUB04	-32 485	-32 839	-32 512
PUB05	-9 669	-10 025	-9 562
PUB06	240 937	141 303	145 587
PUB07	-474 565	-474 072	-472 030
PUB08	-863 941	-828 613	-819 019
PUB09	895 716	810 793	826 828
PUB10	-10 139	-9 207	-10 017

Les résultats précédents ne concernent qu'une seule exécution. Pour les confirmer, le tableau 3.16 présente des résultats plus globaux pour toutes les méthodes avec détection de boucle. Les scores donnés correspondent à la moyenne sur 5 exécutions.

Une classification des meilleurs résultats est effectuée grâce à l'utilisation de différentes couleurs de cellules. La cellule la plus foncée correspond au meilleur résultat obtenu pour le problème. Ainsi, on observe facilement que la première méthode (sans liste Tabou) obtient le meilleur résultat à trois reprises, elle est classée seconde pour deux problèmes et dernière pour les cinq autres. Sur ce même principe, la deuxième méthode est classée première cinq fois, seconde trois fois et dernière pour uniquement deux instances. La dernière méthode est la meilleure pour deux instances, seconde pour cinq autres et dernière pour trois problèmes. Globalement, on observe donc que, sur les cinq exécutions, la deuxième méthode obtient les meilleures performances. La combinaison des méthodes de recherche locale via la détection de boucle couplée avec une liste Tabou sur les réseaux bouclants offre les meilleures performances sur cette application. Les scénarios privés ci-dessous permettent de la comparer à d'autres méthodes développées par d'autres équipes.

3.4.2 Scénarios privés

Cette partie présente les résultats finaux de la méthode livrée à la fin du contrat sur les 20 scénarios privés dans leurs versions finales (tableau 3.17). Les résultats ne doivent pas être comparés avec les résultats des études intermédiaires donnés dans les parties précédentes car les coefficients des composantes de la fonction de coût sont différents. Ces résultats sont comparés à ceux obtenus par les deux autres méthodes utilisées dans le cadre du contrat et référencées dans [36]. Tous les tests ont été effectués une seule fois sur la même machine par le CELAR. La deuxième colonne de ce tableau permet d'identifier

les scénarios réalisables, ils sont notés R. Les autres colonnes rapportent les résultats par méthode.

- **Une méthode basée sur le recuit simulé RS :** Le recuit simulé est une méthode de recherche locale qui s'inspire d'un processus utilisé en métallurgie alternant des périodes de refroidissement lent et de réchauffage de matériaux (recuit) afin d'obtenir des structures stables. La méthode d'optimisation repose sur un paramètre de température qui décroît au cours du temps. La température permet de contrôler l'amplitude des dégradations acceptables au cours de l'exécution et permet ainsi à la recherche locale de s'extirper des minima locaux. De très nombreuses applications ont été traitées avec cette méthode [79, 80, 83, 126].
- **Une méthode basée sur CN-Tabu :** (*Tabu Search on a Consistent Neighborhood* [51, 52]) Cette méthode est une métaheuristique hybride qui inclut des mécanismes de propagation de consistance au sein d'une Recherche Tabou qui utilise des solutions partielles consistantes. En d'autres termes, les solutions partielles explorées ne génèrent aucun conflit. Un ensemble de mécanismes de réparation est utilisé afin de permettre le maintien de la consistance après chaque mouvement. Le principe général de cette méthode a été appliqué avec succès sur plusieurs problèmes de référence [50, 130].
- **Une méthode basée sur la recherche locale adaptative RLA :** La méthode est basée sur l'architecture référencée en figure 3.7. L'adaptation de la recherche locale est réalisée en alternant les phases d'intensification et de diversification de la recherche en fonction de l'historique et de la solution courante; elle a été étudiée en k -coloration et elle est référencée en figure 3.10. Les paramètres d'adaptation automatique sur la détection de boucle et sur la durée Tabou ne sont pas utilisés car ils ont été étudiés après le contrat et les scénarios privés ne sont plus exploitables.

La couleur des cellules utilisées permet de classer les résultats obtenus sur chacun des scénarios. La couleur la plus foncée représente les meilleurs résultats, le blanc représentant les moins bons. Ces couleurs permettent de classer chacune des méthodes mais aussi de montrer que parmi ces trois méthodes distinctes aucune ne permet d'obtenir les meilleurs résultats pour tous les scénarios présentés. Globalement la méthode CN-Tabu est meilleure sur les problèmes réalisables et la méthode RLA sur les problèmes non réalisables. En effet, la méthode CN-Tabu est meilleure pour 10 des 14 scénarios réalisables mais jamais sur les scénarios non réalisables. En revanche, la méthode RLA est classée première sur 5 des 6 scénarios non réalisables et sur 3 scénarios réalisables.

Pour l'étude des scénarios privés, le cahier des charges prévoyait de classer les méthodes selon leur performance sur l'ensemble des scénarios en fonction de l'écart entre le résultat de chaque méthode et le meilleur résultat des trois méthodes. Le classement repose donc sur une mesure de robustesse sur un ensemble de problèmes différents. La note globale sur 1000 est donnée en dernière ligne du tableau. La méthode RLA a été clas-

TAB. 3.17 – Résultat du projet sur les scénarios privés

problèmes	R ?	RS	CN-Tabu	RLA
SC01	R	-4903, 70	-5865, 32	-5026, 93
SC02	R	-2271, 29	-1786, 77	-1087, 62
SC03	R	-5680, 55	-7587, 97	-4735, 31
SC04	R	-8173, 75	-8173, 75	-7637, 34
SC05	R	-1837, 21	-5274, 89	-4762, 54
SC06	R	-3672, 75	-5573, 63	-2608, 28
SC07	R	4321, 29	-5440, 36	-4482, 21
SC08	-	1019254, 80	671524, 75	317127, 40
SC09	R	-172, 47	-2013, 55	-109, 87
SC10	-	724305, 49	206459, 76	61213, 81
SC11	R	-3093, 81	-4501, 13	-3781, 08
SC12	R	-6586, 75	-7540, 83	1269, 22
SC13	R	584008, 22	1289372, 41	-5019, 41
SC14	R	-7263, 64	-8100, 92	-6926, 51
SC15	-	340482, 82	386792, 56	174057, 10
SC16	-	354724, 25	367615, 99	178064, 58
SC17	-	24274, 67	187405, 72	89038, 82
SC18	-	579799, 75	667298, 18	354530, 81
SC19	R	38278, 62	26709, 68	-2507, 75
SC20	R	57 877,48	698 554,63	8 768,60
Note globale		988	985	999

sée première. Après clôture du contrat, [36] a présenté en référence les meilleurs résultats connus sur chaque problème sans contrainte de temps de calcul. Pour quatre scénarios, les performances obtenues par la méthode RLA durant le contrat, avec contrainte de temps de calcul, étaient les meilleures connues.

3.5 Conclusion

Ce chapitre a présenté les études sur la recherche locale adaptative mise en oeuvre pour le problème d'affectation de listes de fréquences en éviation de fréquences pour des réseaux de radiocommunications militaires. Il est composé de quatre parties dont la présentation de la formalisation du problème et des instances, des modèles théoriques et applicatifs de référence, puis l'application de la méthode, suivie des résultats qui ont pu être comparés avec d'autres méthodes. Après le travail assez théorique sur la k -coloration, l'affectation de fréquences a constitué une plate-forme réelle d'expérimentation pour notre méthode tout en constituant un problème qui s'appuie sur des modèles théoriques proches tels la T-coloration avec ensembles. Nous tenons à souligner qu'une partie de ce travail a été réalisée dans le cadre d'un contrat de recherche avec le CELAR.

Dans la première partie le problème d'affectation de listes de fréquences en éviation de fréquences est expliqué. Il s'agit d'un problème très spécifique dont l'objectif est d'allouer à chaque réseau un plan de fréquences qui est une liste structurée en minimisant une fonction de coût qui agrège plusieurs critères. La structure d'un plan de fréquences repose sur la notion de sous-bande qui est un ensemble de fréquences défini dans le spectre par un intervalle et un pas d'échantillonnage. La fonction de coût utilisée combine quatre objectifs différents via une somme pondérée : minimiser le niveau d'interférence, minimiser le nombre de plans de fréquences de taille inférieure à un seuil, maximiser la taille du plus petit plan de fréquences et maximiser la réutilisation des fréquences du spectre. Le choix du modèle de fonction de coût agrégée est imposé. Ce problème d'optimisation est soumis à des contraintes de réalisabilité des solutions qui sont le non chevauchement des sous-bandes, des fréquences interdites et imposées, une taille minimale de plan et un niveau d'interférence maximal par plan.

Nous avons étudié dans la littérature les problèmes qui se rapprochent le plus de ce nouveau problème. Les modèles de T-coloration et de T-coloration avec ensembles ont pour objectif d'allouer une valeur ou un ensemble de valeurs à des variables en respectant une contrainte d'écart entre les valeurs allouées. Ces types de problèmes modélisent différentes applications d'affectation de fréquences liés à différents systèmes de radiocommunications. Ce sont les modèles les plus proches de notre problème cependant celui-ci a trois grandes particularités. D'une part les interférences ne se traduisent pas en contraintes d'écartement de fréquences comme en T-coloration car elles sont calculées par une moyenne de Taux d'Erreur Binaire sur toutes les fréquences d'un plan. D'autre part les fréquences sont organisées en sous-bandes du spectre ce qui ne permet pas d'allouer n'importe quel ensemble du domaine de définition à une variable. Enfin la fonction de coût est composée de critères agrégés et pondérés qui ne correspondent pas à une lecture de la satisfaction de contraintes d'écart.

La troisième partie de ce chapitre contient notre contribution sur ce problème. Elle explique d'abord comment la méthode RLA a été appliquée. Le schéma général de la méthode est resté identique à celui proposée en k -coloration et nous y avons ajouté des heuristiques spécifiques au problème posé comme cela se fait pour toute méthode générale. Ces heuristiques ont été utilisées dans les phases de génération de la solution initiale et d'itération. La procédure de détection de boucle sur les réseaux et de choix adaptatif de la recherche locale intensive ou diversifiante a encadré le fonctionnement des heuristiques. Pour la détection de boucle nous avons défini une fonction de pesage des réseaux afin de les classer en fonction de leur conflit pour les phases d'intensification. Nous avons aussi mené de front une version de la méthode avec et sans liste Tabou pour évaluer l'apport de cette combinaison dans ce contexte applicatif. Le paramétrage de la fenêtre de recherche d'occurrence a été le même que pour la k -coloration.

En ce qui concerne les heuristiques, nous avons étudié différentes méthodes gloutonnes de génération de la solution initiale. Elles avaient pour objectif de créer si possible une

solution réalisable. Nous nous sommes inspirés de la méthode DSATUR en favorisant l'utilisation de la partie inférieure du spectre pour laisser le plus de fréquences disponibles aux réseaux en conflit. Nous avons aussi créé plusieurs structures de voisinages basées sur la manipulation des sous-bandes. Aucun voisinage n'a été définie directement sur les fréquences à cause de la structuration des plans qui interdit un très grand nombre de fréquences du domaine de définition en fonction de l'état courant des plans. Chaque opérateur de mouvement a donné lieu a une étude approfondie sur diverses instances. Enfin, nous avons introduit un contrôle de la dégradation qui mesure l'écart à la meilleure solution trouvée car la fonction de coût utilise des poids parfois très importants qui modifient considérablement l'évaluation d'une solution à une voisine. Ces proportions de changement d'évaluation n'existaient pas dans la k -coloration.

Le chapitre se conclut par une quatrième partie sur les résultats obtenus par la méthode sur des scénarios privés définis uniquement dans le cadre du contrat et publics qui sont disponibles. Tout comme en k -coloration la méthode RLA basée sur la combinaison de deux recherches locales par la détection de boucle a donné de meilleurs scores qu'une méthode intensive avec liste Tabou. Ensuite sur les jeux publics nous avons constaté que l'association de la détection avec un statut Tabou sur les nœuds détectés en boucle est l'option la plus performante. Le bon comportement de cette méthode a donc été confirmé sur ce travail. Pour l'étude des scénarios privés la comparaison de trois méthodes différentes venant de trois équipes distinctes a été faite par le CELAR. La méthode RLA a été classée première à huit reprises et a obtenu la meilleure note globale sur les 20 scénarios.

Conclusions et Perspectives

Dans la littérature de plus en plus de méthodes d'optimisation utilisent un mécanisme d'adaptation. Un mécanisme adaptatif permet de modifier le comportement d'une méthode en fonction d'un historique de recherche. Dans cette thèse, nous avons travaillé sur la mise au point d'une méthode de recherche locale adaptative pour les problèmes d'optimisation combinatoire. Dans le premier chapitre, nous avons défini différents mécanismes d'adaptation en fonction du type de mémoire utilisé, de la composition de l'historique, des informations mémorisées et enfin de leur influence sur le comportement de la méthode. Deux types de mémoire sont utilisés en optimisation : la mémoire explicite qui retient les événements passés et la mémoire implicite qui ne retient pas l'expérience du passé. Ces deux types de mémoire sont utilisés dans les méthodes adaptatives pour sauvegarder des informations liées à la fréquence d'apparition, à la récence, à la qualité ou encore à l'influence du choix d'une valeur, d'une variable ou d'un mouvement. Afin de caractériser l'influence de ces informations mémorisées sur le comportement des méthodes de recherche locale, nous avons distingué les méthodes utilisant une structure de voisinage des méthodes qui en utilisent plusieurs. Deux types d'influence ont été identifiés. Le premier est le choix de la structure de voisinage dans les méthodes qui en utilisent plusieurs. Le second est l'utilisation de mécanismes d'extension ou de restriction de l'ensemble des solutions accessibles au sein d'une même structure de voisinage. Les méthodes adaptatives utilisent souvent un seul mécanisme d'adaptation. Nous avons proposé une méthode de recherche locale combinant plusieurs mécanismes d'adaptation définissant ainsi des procédés adaptatifs d'extension et de restriction du voisinage.

Dans le second chapitre, la méthode adaptative proposée a été mise au point sur le problème de k -coloration de graphe. Ce problème a été très utilisé dans la littérature et constitue une bonne plate-forme d'étude. La première partie de ce chapitre est consacrée à la description du problème, des instances utilisées ainsi que de différentes méthodes de recherche locale proposées dans la littérature. Toutes les études ont été effectuées sur des instances très connues et très utilisées de ce problème : les instances DIMACS. La seconde partie présente le travail mené sur la méthode de recherche locale adaptative à partir de différentes études comportementales. Dans un premier temps, suite à l'observation de répétitions dans les choix de la méthode de recherche locale, nous avons créé un mécanisme de détection de boucle. Une boucle est la répétition d'un choix effectué par la méthode et non une séquence de répétitions de ces choix. La combinaison de deux méthodes différentes de recherche locale via la détection de boucles a permis de résoudre l'ensemble des instances CNET et d'améliorer les résultats obtenus sur les instances DIMACS par

rapport aux recherches locales utilisées isolément. L'observation d'un nombre important de boucles détectées par les mêmes nœuds de façon successive nous a conduit à utiliser une liste Tabou pour gérer l'accès aux variables. Deux options ont été étudiées : l'utilisation d'une liste Tabou de toutes les variables ou uniquement des variables bouclantes. L'ajout d'une liste Tabou nécessite de définir une durée Tabou. Nous avons comparé une durée statique avec une durée choisie dans un intervalle, toutes deux étant très utilisées dans la littérature. L'emploi d'une liste Tabou de variables bouclantes pour une durée dynamique s'est avérée plus intéressante. La combinaison de la détection de boucles à une liste Tabou a permis d'éviter la détection successive de boucles générées par une même variable. Cependant, la performance de la méthode était trop dépendante de la valeur des paramètres utilisés. C'est pour cette raison que nous avons étudié l'adaptation des deux paramètres liés aux deux mécanismes utilisés : la détection de boucle et la liste Tabou régulant respectivement la restriction et l'extension du voisinage. L'adaptation de ces paramètres en fonction d'historique propre à chaque variable a été la plus performante. La méthode a permis d'obtenir des résultats comparables à d'autres méthodes publiées dans la littérature. Ce travail s'est appuyé sur des outils d'analyse de données comme les spectrogrammes qui mesurent la fréquence d'apparition d'évènements et des coefficients de corrélation entre différentes courbes.

Le troisième chapitre présente l'application de la méthode à un problème réel, l'affectation de listes de fréquences en évitement de fréquences pour des réseaux de radiocommunications. Ce problème est récent, il a été rendu public en 2006 à la suite d'un contrat mettant en compétition trois équipes de recherche. Le problème traité a pour objectif d'allouer une liste de fréquences à un ensemble de réseaux en respectant une structure très spécifique des listes sous forme de sous-bande, des contraintes d'interférences sur les plans ainsi que des contraintes de réutilisation des fréquences. Les contraintes d'interférence ne dépendent pas d'un écart entre fréquences mais d'une valeur de Taux d'Erreur Binaire nécessitant un pré-calcul par un simulateur. Nous avons étudié la combinatoire de ce problème, pour les instances de grande taille elle est comparable à celle du problème d'affectation de fréquences pour les plus grands réseaux urbains en GSM. Étant donné qu'aucun travail n'avait été publié sur le nouveau problème qui nous a été posé, nous avons recherché dans la littérature les problèmes les plus semblables. Les problèmes d'affectation de fréquences pour les systèmes civils présentent quelques similitudes. Cependant il existe aussi de nombreuses différences liées au mode de calcul de l'interférence et à la structure des listes de fréquences. Nous avons appliqué la méthode de *Recherche Locale Adaptative* à ce problème. Afin d'ajuster la méthode RLA à ses spécificités, plusieurs études ont été menées : une étude du choix de la sous-bande, une étude sur les mouvements et une étude sur les mécanismes de contrôle de la dégradation. Des mouvements avec des structures de voisinage différentes ont été définis sur les sous-bandes pour prendre en compte la structure des listes de fréquences. La méthode RLA utilisant ces heuristiques a été comparée aux deux autres méthodes basées sur *CN-Tabu* et sur un recuit simulé conçues par les équipes concurrentes lors du contrat. Lors de la clôture du contrat, la recherche locale adaptative s'est montrée la plus robuste sur les vingt jeux de tests privés.

Au cours de ces études, notre méthode a néanmoins montré plusieurs faiblesses qui ouvrent la voie à de nouvelles recherches. En guise de perspective, nous envisageons d'utiliser plusieurs structures de voisinage. Nous nous sommes intéressés plus particulièrement à l'extension et la restriction d'une seule structure de voisinage basée sur la modification d'une unique variable en conflit. Cependant, suite à l'adaptation des paramètres d'extension et de restriction du voisinage sur le problème de k -coloration, nous avons observé l'existence de sous-ensembles stables de variables durablement en conflit composés parfois de plusieurs dizaines de variables. Les différents travaux identifiés dans l'état de l'art sur l'utilisation de plusieurs structures de voisinage et sur l'adaptation du choix entre elles ont en général permis d'améliorer les performances globales des méthodes. Nous envisageons de modifier plusieurs variables parmi les variables durablement en conflit en utilisant le procédé utilisé dans la méthode *Ruin and Recreate* qui a aussi été utilisée par les méthodes de recherche à voisinage large ainsi que *CN-Tabu*. Nous pensons adapter la portée de la structure de voisinage utilisée en fonction de l'historique de la recherche propre à chaque variable. Cet historique pourrait regrouper des informations liées à sa fréquence d'apparition parmi l'ensemble des variables en conflit au cours des dernières itérations.

La seconde perspective envisagée est l'étude du contrôle de la dégradation. Nous avons utilisé différents mécanismes sur le problème d'affectation de listes de fréquences et observé que la définition d'un mécanisme dépendant de paramètres fixés nécessite une étape de calibrage très longue. Nous avons utilisé un contrôle de dégradation défini en fonction des amplitudes importantes des dégradations subies par la solution courante au cours des itérations. En coloration, l'amplitude des dégradations est très faible et le contrôle de la dégradation n'était pas nécessaire. Cette différence de fonctionnement est due au calcul de l'évaluation de la solution. La fonction d'évaluation utilisée en affectation de fréquences utilisait des poids très importants entraînant d'importantes différences d'évaluation entre deux solutions voisines. Une étude de la possibilité d'adapter ce contrôle de la dégradation au problème est nécessaire pour généraliser la méthode. Nous envisageons de généraliser le contrôle de dégradation adaptatif que nous avons défini pour ce problème. Le paramétrage évoluerait tout au long de l'exécution en fonction de la meilleure solution trouvée et de la solution courante selon l'évolution de l'algorithme et pas seulement lors d'une période de démarrage de celui-ci comme pour le recuit simulé.

Enfin, nous avons arbitrairement fixé la taille de la mémoire par rapport au nombre de variables du problème tout au long de cette thèse. En troisième perspective, une étude de l'importance de ce paramètre et de sa définition est proposée. L'utilisation d'une mémoire trop petite peut être négative en empêchant l'algorithme de détecter les boucles principalement lorsque le nombre de variables en conflit est très important. A l'inverse une valeur trop grande prendrait en compte des itérations très anciennes et nous avons vu que le nombre de variables en conflit varie au cours du temps. Le nombre de variables en conflit à un instant donné est plus faible que le nombre de variables en mémoire comme nous l'avons vu au cours de notre étude statistique sur l'utilisation de la mémoire sur le problème de k -coloration. Utiliser une taille trop grande revient à détecter des boucles sur des nœuds ayant été visités beaucoup plus tôt. Ajuster la taille de la mémoire est

donc très important et peut avoir une très grande influence sur le comportement de la méthode pour les instances de grande taille. En particulier en k -coloration, nous avons noté que la taille actuelle de la mémoire ne semble pas adaptée au nombre important de variables en conflit pour de longues périodes sur les problèmes aléatoires à 1000 nœuds. La troisième perspective est donc l'étude du paramétrage adaptatif de la taille de la mémoire en fonction de son influence sur la performance de la méthode.

Publications

Journaux Internationaux

I. Devarenne, H. Mabed, A. Caminada, (2007). *Intelligent neighborhood exploration in local search heuristics*, IEEE International Journal on Artificial Intelligence Tools, accepté, à paraître en 2008.

Conférences Internationales avec Impact Factor

I. Devarenne, H. Mabed, A. Caminada, (2006). *Intelligent neighborhood exploration in local search heuristics*, ICTAI'06 : Proceedings of the 18th IEEE International Conference on Tools with Artificial Intelligence, p. 144-150, Washington D.C., USA, November 2006.

Conférences Internationales avec comité de lecture

H. Mabed, I. Devarenne, A. Caminada, (2007), *Frequency Planning for Slow Frequency Hopping System*, International Network Optimization Conference, Spa, Belgique, April 2007.

I. Devarenne, H. Mabed, A. Caminada, (2006). *Self-adaptive neighborhood exploration parameters in a local search*, 7th EU/MEeting on Adaptive, Self-Adaptive, and Multi-Level Metaheuristics, Malaga, Espagne, Novembre 2006.

I. Devarenne, H. Mabed, A. Caminada, (2006). *Optimization by extension-restriction neighborhood in local search : application to graph coloring problem*, 20th European Simulation and Modeling Conference, Toulouse France, Octobre 2006.

I. Devarenne, A. Caminada, H. Mabed, (2005). *Analysis of Adaptive Local Search for the Graph Coloring Problem*, 6th Metaheuristics International Conference - MIC 2005, Vienne Autriche, August 2005.

Communications Internationales Francophones

I. Devarenne, H. Mabed, A. Caminada, (2006). *Hybridization of adaptive local search and tabu list*, META'2006, Hammamet, Tunisie, November 2006.

H. Mabed, I. Devarenne, A. Caminada, (2006). *Recherche Locale et contrôle de dégradation*, Septième Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision - ROADEF 2006, Lille, France, Février 2006.

I. Devarenne, H. Mabed, A. Caminada, (2006). *Hybridation des mécanismes de détection de boucles et de liste tabou pour la recherche locale*, Septième Congrès de la Société

Française de Recherche Opérationnelle et d'Aide à la Décision - ROADEF 2006, Lille, France, Février 2006.

I. Devarenne, A. Caminada, H. Mabed. (2005). *Gestion de Boucles en Recherche Locale pour la Coloration de Graphes*, Sixième Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision - ROADEF 2005, Tours, France, Février 2005.

Mémoire

I. Devarenne (2004), *Études d'heuristiques à mémoire pour l'affectation de fréquences*. Mémoire de DEA en Informatique Automatique et Productique, Université de Technologie Belfort Montbéliard, Belfort (90), France, Septembre 2004.

Bibliographie

- [1] K. I. Aardal, S. P. M. van Hoesel, A. M. C. A. Koster, C. Mannino, and A. Sassano. Models and solution techniques for frequency assignment problems. ZIB-report 01–40, Konrad-Zuse-Zentrum für Informationstechnik Berlin, Germany, 2001.
- [2] F. F. Ali, Z. Nakao, R. B. Tan, and Y-W. Chen. An evolutionary approach for graph coloring. In *Systems, Man, and Cybernetics, 1999. IEEE SMC '99 Conference Proceedings. 1999 IEEE International Conference on*, number 5, pages 527–532, 1999.
- [3] S. Amin and J.L. Fernandez-Villacaiias. Dynamic local search. *Genetic Algorithms in Engineering Systems : Innovations and Applications, GALEZIA 97. Second International Conference On (Conf. Publ. No. 446)*, pages 129–132, 1997.
- [4] E. J. Anderson, C. A. Glass, and C. N Potts. *Local search in combinatorial optimization*, chapter Machine scheduling, pages 361–415. John Wiley & Sons, 1997.
- [5] H. Arntzen and A. Løkketangen. A tabu search heuristic for a university timetabling problem. In *Proceedings of the Fifth Metaheuristics International Conference*, 2003.
- [6] C. Artigues, M. Gendreau, and L.-M. Rousseau. A flexible model and a hybrid exact method for integrated employee timetabling and production scheduling. In *CORS / Optimization Days 2006 Joint Conference*, pages 64–81, Montréal, Canada, 2006.
- [7] C. Avanthay, A. Hertz, and N. Zufferey. A variable neighborhood search for graph coloring. *European Journal of Operational Research*, 151 :379–388, 2003.
- [8] V. Bachelet and E-G. Talbi. Cosearch : a co-evolutionary metaheuristic. In *Congress on Evolutionary Computation CEC'2000*, pages 1550 – 1557, 2000.
- [9] R. Battiti. *Reactive Search : Toward Self-Tuning Heuristics*, chapter 4, pages 61–83. John Wiley and Sons Ltd, 1996. Modern heuristic search methods.
- [10] R. Battiti and A. Bertossi. Greedy, prohibition and reactive heuristics for graph partitioning. In *IEEE transactions on computers*, volume 48, 1999.
- [11] R. Battiti and M. Protasi. Reactive search, a history-based heuristic for max-sat. *ACM Journal of Experimental Algorithms*, 2, 1997.
- [12] R. Battiti and G. Tecchiolli. The reactive tabu search. *ORSA Journal on Computing*, 6(2) :126–140, 1994.
- [13] N. Belacel, M. Cuperlovic-Culf, R. Ouellette, and M.R. Boulassel. The variable neighborhood search metaheuristic for fuzzy clustering cDNA microarray gene expression data. In *Artificial Intelligence and Applications AIA 2004*, 2004.

- [14] W. ben Ameer. Computing the initial temperature of simulated annealing. *Computational Optimization and Applications*, 29(3) :369–385, 2004.
- [15] A. Billionnet. *Optimisation discrète De la modélisation à la résolution par des logiciels de programmation mathématique*. Dunod, 2007.
- [16] I. Blöchliger. *Suboptimal colorings and solution of large chromatic scheduling problems*. PhD thesis, Ecole polytechnique fédérale de Lausanne EPFL, 2005.
- [17] L. Boithias. *Propagation des ondes radioélectriques dans l’environnement terrestre*. Collection Techniques et Scientifique des Télécommunications. Dunod, 1983.
- [18] O. Braysy. A reactive vns for the vehicle routing problem with time windows. *INFORMS Journal on Computing*, 15(4) :347368, 2003.
- [19] D. Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4) :251–256, 1979.
- [20] W. J. Brown. An iterated local search with adaptive memory applied to the snake in the box problem. Master’s thesis, B.S. Mathematics, Georgia Institute of Technology, 2001.
- [21] E. Burke, G. Kendall, J. Newall, E. Hart, P. Ross, and S. Schulenburg. *Handbook of metaheuristics*, chapter Hyper-heuristics : an emerging direction in modern search technology, pages 457–474. Kluwer Academic Publishers, 2003.
- [22] E. K. Burke and G. Kendall. *Search Methodologies : Introduction tutorials in optimization and decision support techniques*. Springer, 2005.
- [23] P. Calegari, G. Coray, A. Hertz, D. Kobler, and P. Kuonen. A taxonomy of evolutionary algorithms in combinatorial optimization. *Journal of Heuristics*, 5 :145–158, 1999.
- [24] A. Caminada, J-K. Hao, J-L. Lutton, and V. Martin. *Recherche opérationnelle et réseaux*, chapter Réseaux de communications. Gerd Finke, 2002.
- [25] A. Capone and M. Trubian. Channel assignment problem in cellular systems : A new model and a tabu search algorithm. *IEEE Transactions on Vehicular Technology*, 48(4) :1252–1260, 1999.
- [26] T. Carchrae and J.C. Beck. Cost-based large neighborhood search. In *Workshop on the Combination of Metaheuristic and Local Search with Constraint Programming Techniques*, 2005.
- [27] M. Chiarandini. *Stochastic Local Search Methods for Highly Constrained Combinatorial Optimisation Problems*. PhD thesis, Computer Science Department, Darmstadt University of Technology, Darmstadt, Germany, 2005.
- [28] M. Chiarandini, I. Dumitrescu, and T. Stützle. Local search for the graph colouring problem. a computational study. Technical report, AIDA-03-01, Intellectics Group, Computer Science Department, Darmstadt University of Technology, Darmstadt, Germany, January 2003.
- [29] M. Chiarandini, I. Dumitrescu, and T. Stützle. Stochastic local search algorithms for the graph colouring problem. Technical report, AIDA-05-03, Intellectics Group, Computer Science Department, Darmstadt University of Technology, Darmstadt, Germany, 2005.

- [30] M. Chiarandini and T. Stützle. An application of iterated local search to graph coloring problem. In D. S. Johnson, A. Mehrotra, and M. Trick, editors, *Proceedings of the Computational Symposium on Graph Coloring and its Generalizations*, pages 112–125, Ithaca, New York, USA, September 2002.
- [31] Y. Collette and P. Siarry. *Optimisation multiobjectif*. Eyrolles, 2002.
- [32] F. Comellas and J. Ozón. Graph coloring algorithms for assignment problems in radio networks. In J. Alspector, R. Goodman, and T. X. Brown, editors, *Applications of Neural Networks to Telecommunications 2*, pages 49–56. Lawrence Erlbaum Ass., Inc., Publis., Hillsdale, NJ, 1995.
- [33] P. Cowling, G. Kendall, and E. Soubeiga. A parameter-free hyperheuristic for scheduling a sales summit. In *Proceedings of the 4th Metaheuristic International Conference, MIC 2001*, pages 127–131, 2001.
- [34] P. Cowling, G. Kendall, and E. Soubeiga. Hyperheuristics : A robust optimisation method applied to nurse scheduling. In *Parallel Problem Solving from Nature - PPSN VII : 7th International Conference*, volume Volume 2439/2002 of *Lecture Notes in Computer Science*, pages 851–860, Granada, Spain, 2002. Springer Berlin / Heidelberg.
- [35] P. Cowling, G. Kendall, and E. Soubeiga. Hyperheuristics : A tool for rapid prototyping in scheduling and optimisation. In *Proceedings of the second workshop on Evolutionary Computation in Combinatorial Optimization (EvoCOP2002)*, volume Volume 2279/2002 of *Lecture Notes in Computer Science*. Springer LNCS, 2002. ISSN0302-9743 (Print) 1611-3349 (Online).
- [36] T. Defaix. SFH-FP "slow frequency hopping - frequency planning". In *Septième Congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision - ROADEF 2006*, Lille, France, 2006.
- [37] M. den Besten and T. stutzle. Neighborhoods revisited : an experimental investigation into the effectiveness of variable neighborhood descent for scheduling. In *Proceedings of MIC 2001*, pages 545–550, Porto, Portugal, July 2001. 2001.
- [38] M. den Besten, T. Stützle, and M. Dorigo. Design of iterated local search algorithms. In *Proceedings of the EvoWorkshops on Applications of Evolutionary Computing*, pages 441–451, London, UK, 2001. Springer-Verlag.
- [39] I. Devarenne, A. Caminada, and H. Mabed. Analysis of adaptive local search for the graph coloring problem. In *6th Metaheuristics International Conference - MIC 2005*, Vienne, Autriche, 2005.
- [40] I. Devarenne, H. Mabed, and A. Caminada. Intelligent neighborhood exploration in local search heuristics. In *18th IEEE International Conference on Tools with Artificial Intelligence*, Washington D.C., USA, November 2006.
- [41] L. di Gaspero, M. Chiarandini, and A. Schaerf. A study on the short-term prohibition mechanisms in tabu search. In A. Perini P. Traverso G. Brewka, S. Coradeschi, editor, *Proc. of the 17th European Conference on Artificial Intelligence (ECAI2006)*, pages 83–87. IOS Press, 2006.

- [42] L. di Gaspero and A. Schaerf. A tabu search approach to the traveling tournament problem. In *MIC2005 : The 6th Metaheuristics International Conference*, Viena, Austria, 2005.
- [43] R. Dorne. *Étude des méthodes heuristiques pour la coloration, la T-coloration et l'affectation de fréquences*. PhD thesis, Université de montpellier II, mai 1998.
- [44] R. Dorne and J-K. Hao. An evolutionary approach for frequency assignment in cellular radio networks. In *IEEE Int. Conference on Evolutionary Computing*, pages 539–544, Perth, Australia, 1995.
- [45] R Dorne and J-K. Hao. A new genetic local search algorithm for graph coloring. In *Lecture Notes in Computer Science*, 1498, pages 745–754. Springer-Verlag, 1998.
- [46] R. Dorne and J-K. Hao. Tabu Search for graph coloring, T-coloring and Set T-colorings. In I.H. Osman S. Voss, S. Martello and C. Roucairol, editors, *Metaheuristics : Advances and Trends in Local Search Paradigms for Optimization*, chapter 6, pages 77–92. Kluwer, 1998.
- [47] C. Duhamel. *Un Cadre Formel pour les Méthodes par Amélioration Itérative - Application à deux problèmes d'Optimisation dans les Réseaux -*. PhD thesis, Université Blaise Pascal - Clermont-Ferrand II, 2001.
- [48] I. Dumitrescu and T. Stützle. Combinations of local search and exact algorithms. *Applications of Evolutionary Computing LNCS*, 2611 :211–223, 2003.
- [49] I. Dumitrescu and T. Stützle. A survey of methods that combine local search and exact algorithms. Technical report, AIDA-03-07, FG Intellektik, FB Informatik, TU, Darmstadt, Germany, 2003.
- [50] A. Dupont. *Étude d'une métaheuristique hybride pour l'affectation de fréquences dans les réseaux tactiques évolutifs*. PhD thesis, Université Montpellier II, 2005.
- [51] A. Dupont and M. Vasquez. Solving the dynamic frequency assignment problem. In *MIC2005 : The Sixth Metaheuristics International Conference*, Vienna, Austria, 2005.
- [52] A. Dupont, M. Vasquez, and D. Habet. Consistent neighbourhood in a tabu search. *Workshop on Combination of Metaheuristic and Local Search with Constraint Programming techniques, University of Nantes, France*, 2005.
- [53] D. Duvivier, P. Preux, C. Fonlupt, D. Robilliard, and E-G. Talbi. The fitness function and its impact on local search methods. In *Proc. Conf. IEEE System, Man, and Cybernetics*, pages 2478–2483, San Diego, USA, october 1998. 1998.
- [54] A.E. Eiben, J.I. van Hemert, E. Marchiori, and A.G. Steenbeek. Solving binary constraint satisfaction problems using evolutionary algorithms with an adaptative fitness function. *Lecture Notes in Computer Science*, 1498, 1998.
- [55] A. Eisenblätter, M. Grötschel, and A. Koster. Frequency Planning and Ramifications of Coloring. *Discussiones Mathematicae, Graph Theory*, (22) :51–88, 2002.
- [56] A. Eisenblätter, H-F. Geerdes, and I. Siomina. Integrated access point placement and channel assignment for wireless lans in an indoor office environment. In *Proc. of the 8th IEEE Intl. Symposium on a World of Wireless, Mobile and Multimedia Networks*, June 2007.

- [57] C. Fleurent and J.A. Ferland. Algorithmes génétiques hybrides pour l'optimisation combinatoire. *RAIRO*, 30 :373–388, 1996.
- [58] D. A. Fotakis, S. D. Likothanassis, and S. K. Stefanakos. An evolutionary annealing approach to graph coloring. In Egbert J. W. Boers, Stefano Cagnoni, Jens Gottlieb, Emma Hart, Pier Luca Lanzi, Gunther Raidl, Robert E. Smith, and Harald Tijink, editors, *Applications of Evolutionary Computing. EvoWorkshops2001 : EvoCOP, EvoFlight, EvoIASP, EvoLearn, and EvoSTIM. Proceedings*, volume 2037, pages 120–129, Como, Italy, 18-19 2001. Springer-Verlag.
- [59] P. Galinier and J-K. Hao. Hybrid evolutionary algorithms for graph coloring. *Journal of Combinatorial Optimization*, 3 :379–397, 1999.
- [60] P. Galinier and A. Hertz. A survey of local search methods for graph coloring. *Comput. Oper. Res.*, 33(9) :2547–2562, 2006.
- [61] P. Galinier, A. Hertz, and N. Zufferey. An adaptive memory algorithm for the k-colouring problem. Technical report, Les cahiers de Gerad G-2003-35, 2004.
- [62] F. Glover. Tabu Search - part I. *ORSA Journal on Computing*, 1(3) :190–206, 1989.
- [63] F. Glover. Tabu Search - part II. *ORSA Journal on Computing*, 2 :4–32, 1990.
- [64] F. Glover, J. P. Kelly, and M. Laguna. Genetic algorithms and tabu search : hybrids for optimization. *Comput. Oper. Res.*, 22(1) :111–134, 1995.
- [65] D. Godard, P. Laborie, and W. Nuijten. Randomized large neighborhood search for cumulative scheduling. In *Proceedings ICAPS-05*, Monterey, California, USA, 2005.
- [66] P. Hansen and B. Jaumard. Algorithms for the maximum satisfiability problem. *Computing*, 44(4) :279–303, 1990.
- [67] P. Hansen and N. Mladenovic. Recherche à voisinage variable. Technical report, Les cahiers de Gerad G-2000-08, 2000.
- [68] P. Hansen and N. Mladenovic. Variable Neighbourhood Search : Principles and applications. *European Journal of Operational Research*, 130 :449–467, 2001.
- [69] P. Hansen and N. Mladenovic. VNS, a chapter of "handbook of metaheuristics". Technical report, Les cahiers de GERAD G-2001-41, 2001.
- [70] P. Hansen, N. Mladenovic, and D Urosevic. Variable Neighborhood Search and local branching. *Computers and Operations Research*, 33(10) :3034–3045, 2006.
- [71] J-K. Hao and R. Dorne. Study of genetic search for the frequency assignment problem. *Lecture Notes in Computer Science*, 1063 :333–344, 1996.
- [72] J-K. Hao, R. Dorne, and P. Galinier. Tabu search for frequency assignment in mobile radio networks. *Journal of Heuristics*, 4(1) :47–62, 1998.
- [73] J-K. Hao and P. Galinier. Tabu search for maximal constraint satisfaction problems. *Lecture Notes in Computer Science*, 1330 :196–208, 1997.
- [74] A. Hertz and D. de Werra. Using tabu search techniques for graph coloring. *Computing*, 39(4) :345–351, 1987.
- [75] A. Hertz, E. Taillard, and D. De Werra. A tutorial on tabu search. In *Proc. of Giornate di Lavoro AIRO'95 (Enterprise Systems : Management of Technological Organizational Changes)*, pages 13–24, Italy, 1995.

- [76] A. Hertz and M. Widmer. Guidelines for the use of meta-heuristics in combinatorial optimization. *European Journal of Operational Research*, 151 :247–252, 2003.
- [77] L. M. Hvattum, A. Løkketangen, and F. Glover. *Integer Programming : Theory and Practice*, chapter New Heuristics and Adaptive Memory Procedures for Boolean Optimization Problems, pages 1–18. CRC Press, Boca Raton, FL, 2005.
- [78] T. R. Jensen and B. Toft. *Graph Coloring Problems*. Wiley Interscience, 1995.
- [79] D. S. Johnson, C. R. Aragon, L. A. Mcgeoch, and C. Schevon. Optimization by simulated annealing : An experimental evaluation, part I, graph partitioning. *Operations research*, 37(6) :865–892, 1989.
- [80] D. S. Johnson, C. R. Aragon, L. A. Mcgeoch, and C. Schevon. Optimization by simulated annealing : An experimental evaluation, part II, graph coloring and number partitioning. *Operations research*, 39(3) :378–406, 1991.
- [81] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning : A survey. *Journal of Artificial Intelligence Research*, (4) :237–285, 1996.
- [82] P. Kilby, P. Prosser, and P. Shaw. Guided local search for the vehicle routing problem. In *Proceedings of the 2nd International Conference on Meta-heuristics*, 1997.
- [83] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598) :671–680, 13 May 1983.
- [84] A. Koster. *Frequency Assignment : Models and Algorithms*. PhD thesis, Universiteit Maastricht, 1999.
- [85] P. Laborie and D. Godard. Self-adapting large neighborhood search : Application to single-mode scheduling problems. In *MISTA-07*, Paris (France), August 2007.
- [86] M. Laguna. *Global Optimization And Meta-Heuristics*. Eolss Publishers, University of Colorado at Boulder, USA, 2002.
- [87] F.T. Leighton. A graph coloring algorithm for large scheduling problems. *Journal of research of the national bureau of standards*, (84) :489–505, 1979.
- [88] C. Lengoumbi, P. Godlewski, and P. Martins. An efficient subcarrier assignment algorithm for downlink ofdma. *Vehicular Technology Conference, 2006. VTC-2006 Fall. 2006 IEEE 64th*, 2006.
- [89] K. K. Leung and B.-J. Kim. Frequency assignment for IEEE 802.11 wireless networks. In *Proceedings of VTC 2003-Fall*, Orlando, FL, 2003.
- [90] A. Lim and F. Wang. Meta-heuristics for robust graph coloring problem. In *Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence (ICTAI 2004)*, 2004.
- [91] G. Lin, D. Xu, Z.-Z. Chen, T. Jiang, J. Wen, and Y. Xu. An efficient branch-and-bound algorithm for the assignment of protein backbone nmr peaks. In *Proceedings of the IEEE Computer Society Bioinformatics Conference (CSB'02)*, 2002.
- [92] H. Lourenço, O. Martin, and T. stutzle. A beginner's introduction to iterated local search. In *Proceedings of MIC 2001*, pages 1–6, Porto, Portugal, July 2001.

- [93] H. Lourenço, O. Martin, and T. stutzle. A gentle introduction to iterated local search. In *Metaheuristics International Conference*, volume 1, 2001.
- [94] H. Lourenço, O. Martin, and T. stutzle. *Iterated Local Search*, chapter Iterated Local Search, pages 321–353. Kluwer Academic Publishers, Norwell, MA, 2002.
- [95] I. Méndez-Díaz and P. Zabala. A branch-and-cut algorithm for graph coloring. *Discrete Appl. Math.*, 154(5) :826–847, 2006.
- [96] L. B. Milstein and M.K. Simon. Spread sepctrum communications. In IEEE Press, editor, *the mobile communications handbook*, pages 152–165, 1996.
- [97] N. Mladenovic and P. Hansen. Variable Neighborhood Search. *Comps. in Opns. Res.*, 24 :1097–1100, 1997.
- [98] R. Montemanni, J. N.J. Moon, and D. H. Smith. An improved tabu search algorithm for the fixed-spectrum frequency assignment problem. *IEEE Transactions on Vehicular Technology*, 52(4) :891–901, 2003.
- [99] R. Montemanni and D. H. Smith. A tabu search algorithm with a dynamic tabu list for the frequency assignment problem. *Technical Report, University of Glamorgan*, 2001.
- [100] J. N. J. Moon, L. A. Hughes, and D. H. Smith. Assignment of frequency lists in frequency hopping networks. *IEEE Transactions on Vehicular Technology*, 54(3) :1147–1159, 2005.
- [101] B Neveu and G Trombettoni. Hybridation de GWW avec de la recherche locale. In *9èmes Journées Nationales sur la Résolution Pratique de Problèmes NP-Complets, JNPC’03*, Amiens, France, 2003.
- [102] B Neveu, G Trombettoni, and F Glover. Idwalk : a candidate list strategy with a simple diversification device. In Springer, editor, *Principles and Practice of Constraint Programming, CP’04, LNCS*, LNCS, 2004.
- [103] J. B. Orlin, A. P. Punnen, and A. S. Schulz. Approximate local search in combinatorial optimization. In *SODA ’04 : Proceedings of the fifteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 587–596, Philadelphia, PA, USA, 2004. Society for Industrial and Applied Mathematics.
- [104] M. Palpant. *Recherche exacte et approchée en optimisation combinatoire : schémas d’intégration et applications*. PhD thesis, Laboratoire Informatique d’Avignon, 2005.
- [105] M Palpant, C Artigues, and P Michelon. a Large Neighborhood Search method for solving the frequency assignment problem. Technical Report 385, LIA report, 2003.
- [106] L. Paquete and T. Stützle. An experimental investigation of iterated local search for coloring graphs. *Applications of Evolutionary Computing*, 2279 :122–131, 2002.
- [107] M. Pirlot. Métaheuristiques pour l’optimisation combinatoire : un aperçu général. In *Optimisation approchée en recherche opérationnelle*. Lavoisier, 2002.
- [108] J. Puchinger and G. R. Raidl. Relaxation guided variable neighborhood search. In *Proceedings of the XVIII Mini EURO Conference on VNS*, Tenerife, Spain, 2005.

- [109] R. Qu, E. Burke, and B. McCollum. A survey of search methodologies and automated approaches for examination timetabling. Technical report, Computer Science Technical Report No. NOTTCS-TR-2006-4, 2006.
- [110] D. Renaud and A. Caminada. Evolutionary methods and operators for frequency assignment problem. *SpeedUp Journal*, 11(2) :27–32, 1997.
- [111] J. Riihijärvi, M. Petrova, and P. Mähönen. Frequency allocation for WLANs using graph colouring techniques. In *Proceedings of WONS'05*, St. Moritz, Switzerland, 2005.
- [112] S. Ropke and D. Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4) :455–472, 2006.
- [113] P. Ross, D. Corne, and H. Fang. Improving evolutionary timetabling with delta evaluation and directed mutation. In *Parallel Problem Solving From Nature III*, number 866 in Springer-Verlag Lecture Notes in Computer Science, pages 556–565, 1994.
- [114] K-K. Sadan, B. A. Norman, D. W. Coit, and A. E. Smith. Exploiting tabu search memory in constrained problems. *INFORMS Journal on Computing*, 16(3) :241–254, 2004.
- [115] G. Schrimpf, J. Schneider, H. Stamm-Wilbrandt, and G. Dueck. Record breaking optimization results using the ruin and recreate principle. *Journal of Computational Physics*, 159 :139171, 2000.
- [116] M. Schulz. *Solving Frequency Assignment Problem with Constraint Programming*. PhD thesis, Technische Universität Berlin, Institut für Mathematik, 2003.
- [117] M. Sevaux and P. Thomin. Heuristics and metaheuristics for a parallel machine scheduling problem : a computational evaluation. In *4^e Metaheuristic International Conference MIC2001*, Porto, Portugal, July 2001.
- [118] M. Sevaux and P. Thomin. Recherche taboue améliorée pour l’ordonnancement sur machines parallèles. In *3^e conférence Francophone de MODélisation et SIMulation Conception, Analyse et Gestion des Systèmes industriels MOSIM'01*, Troyes, France, avril 2001.
- [119] P. Shaw. A new local search algorithm providing high quality solutions to vehicle routing problems group. Technical report, APES, 1998.
- [120] T Stützle. Applying iterated local search to the permutation flow shop problem. Technical report, AIDA-98-04 FG Intellektik, Darmstadt University of Technology, Computer Science Department, Intellectics Group., 1998. Darmstadt University of Technology, Computer Science Department, Intellectics Group. (submitted).
- [121] T. Stützle. *Local Search Algorithms for Combinatorial Problems : Analysis, Improvements, and New Applications*. PhD thesis, Am Fachbereich Informatik der Technischen Universität Darmstadt vorgelegte, 1998.
- [122] T Stützle. Iterated local search for the quadratic assignment problem. Technical report, AIDA-99-03, 1999.

- [123] E. D. Taillard. Robust taboo search for the quadratic assignment problem. *Parallel Computing*, 17 :443–455, 1991.
- [124] E. D. Taillard, L.-M. Gambardella, M. Gendreau, and J.-Y. Potvin. Adaptive memory programming : A unified view of meta-heuristics. Technical Report IDSIA-19-98, 10, 1998.
- [125] E.G. Talbi. A taxonomy of hybrid metaheursitics. *Journal of Combinatorial Optimization*, pages 1–45, 1999.
- [126] EG Talbi and T Muntean. Hill climbing, simulated annealing and genetic algorithm : a comparative study and application to the mapping problem. In *Proc. of the Twenty-Sixth Hawaii International Conference on System Sciences*, volume 2, pages 565–573, 1993.
- [127] L-K. Tee, C. Van Rensburg, and Y. Ding. Performance of a multi-channel ofdm system under the effect of adjacent channel interference. *IEEE Vehicular Technology Conference*, 1(62) :397–401, 2005.
- [128] D. Thierens. Population-based iterated local search : Restricting neighborhood search by crossover. In K. Deb, R. Poli, W. Banzhaf, H-G. Beyer, E. Burke, P. Darwen, D. Dasgupta, D. Floreano, J. Foster, M. Harman, O. Holland, P L. Lanzi, L. Spector, A. Tettamanzi, D. Thierens, and A. Tyrrell, editors, *Genetic and Evolutionary Computation – GECCO-2004, Part II*, volume 3103 of *Lecture Notes in Computer Science*, pages 234–245, Seattle, WA, USA, 26-30 June 2004. Springer-Verlag.
- [129] C. Touzet. *Les réseaux de neurones artificiels, introduction au connexionnisme*. Paris, juillet 1992.
- [130] M. Vasquez. *Résolution en variables 0-1 de problèmes combinatoires de grande taille par la méthode tabou*. PhD thesis, Université d’Angers, 2000.
- [131] C. Voudouris and E.P.K. Tsang. Guided local search and its application to the travelling salesman problem. *European Journal of Operational Research, Anbar Publishing.*, 113(2) :469–499, March 1999.
- [132] B. Weinberg. *Analyse et résolution approchée de problèmes d’optimisation combinatoire : application au problème de coloration de graphes*. PhD thesis, Université des sciences et technologies de lille, 2004.
- [133] B. Weinberg, V. Bachelet, and E-G. Talbi. A co-evolutionist meta-heuristic for the frequency assignment problem. In *Frequency Assignment Workshop*, London, England, July 2000.
- [134] B. Weinberg, V. Bachelet, and E-G. Talbi. A co-evolutionist meta-heuristic for the assignment of the frequencies in cellular networks. *Lecture Notes in Computer Science LNCS No.2037, Lake Como, Italy*, pages 140–149, april 2001. First European Workshop on Evolutionary Computation in Combinatorial Optimization EvoCop.
- [135] K. L. Weldon. A simplified introduction to correlation and regression. *Journal of Statistics Education*, 8(3), 2000.
- [136] M. Widmer, A. Hertz, and D. Costa. *Ordonnancement de la Production*, chapter Les Métaheuristiques. HERMES science publications, 2000.

- [137] T. Wong, P. Bigras, and B. de Kelper. A multi-neighborhood and multi-operator strategy for the uncapacitated exam proximity problem. *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, 4 :3810– 3816, 2005.
- [138] M. Yokoo and K. Hirayama. Frequency assignment for cellular mobile systems using constraint satisfaction techniques. In *Proceedings of the IEEE Annual Vehicular Technology Conference (VTC2000-Spring)*, 2000.