FACULTÉ DES SCIENCES
UNIVERSITÉ D'ANGERS

Loria
Laboratoire lorrain de recherche en informatique et ses applications

Rapport de stage : Hybridation de méthodes exactes et approchées pour la coloration de graphe pondéré

MASTER 2 INFORMATIQUE
PARCOURS INTELLIGENCE ARTIFICIELLE

Étudiant :
Habib KHTEIRA

Encadrant :
Cyril GRELIER
Ayse AKBALIK
Nicolas JOZEFOWIEZ

24 août 2026

Remerciements

Je tiens à exprimer ma profonde gratitude à mes encadrants, Monsieur
Cyril Grelier, Madame Ayse Akbalik et Monsieur Nicolas Jozefowiez, pour leur accompagnement, leur disponibilité et leurs précieux conseils tout au long de mon stage. Leurs remarques constructives, leur expertise et la qualité de nos échanges ont grandement contribué à l'avancement de ce travail. Ils m'ont permis d'approfondir mes connaissances en optimisation combinatoire, de mieux comprendre les exigences d'un travail de recherche et de développer progressivement mon autonomie scientifique et technique.

Je remercie également l'ensemble des membres de l'équipe OPTIMIST pour leur accueil chaleureux et pour les échanges enrichissants que j'ai pu avoir avec eux. Je remercie plus largement le Loria de m'avoir accueilli dans un environnement de travail stimulant et de m'avoir offert des conditions particulièrement favorables à la réalisation de ce stage.

J'adresse enfin mes remerciements aux enseignants du Master Informatique de l'Université d'Angers pour les connaissances acquises au cours de ma formation, ainsi qu'à mes proches pour leur soutien et leurs encouragements tout au long de cette année.

Cette expérience a constitué une étape particulièrement enrichissante de mon parcours, tant sur le plan scientifique et professionnel que personnel.

Table des matières

Contexte

Dans le cadre du Master 2 Informatique, parcours Intelligence Artificielle, de l'Université d'Angers, j'effectue un stage de recherche au Loria, à Nancy. Ce stage se déroule du 3 mars au 27 août 2026. Le Loria est une unité mixte de recherche consacrée à la recherche fondamentale et appliquée en informatique. Il réunit le CNRS, l'Université de Lorraine, CentraleSupélec et Inria [1]. J'ai été accueilli au sein de l'équipe OPTIMIST. Ce travail est encadré par Cyril Grelier, Ayse Akbalik et Nicolas Jozefowiez.

Le sujet porte sur l'hybridation de méthodes exactes et approchées pour le problème de coloration de graphe pondéré. Une première étape a consisté à étudier les principales formulations exactes et les méthodes approchées proposées dans la littérature. Le travail s'est ensuite concentré sur l'adaptation à la réparation de solutions partielles des modèles M2 [Malaguti *et al.*, 2009] et Dual [Cornaz *et al.*, 2017], deux modèles mathématiques de type programmation linéaire en nombres entiers (PLNE) issus de la littérature. Il a conduit à la conception et à l'implémentation d'une ALNS hybride. Enfin, une campagne expérimentale a permis d'évaluer cette approche et de la comparer à RedLS [Wang *et al.*, 2020] et à ILSTS [Nogueira *et al.*, 2021], deux méthodes approchées efficaces existantes dans la littérature.

Ce rapport est organisé comme suit. La section 1 introduit le WVCP, ses principales difficultés de résolution et deux de ses cas d'application. La section 2 présente les principales méthodes exactes et approchées proposées dans la littérature. La section 3 décrit l'ALNS développée, notamment ses opérateurs de destruction et de réparation, son mécanisme d'intensification et ses stratégies d'adaptation. La section 4 présente le protocole expérimental et analyse les résultats obtenus. Enfin, la section 5 résume les principales conclusions de ce travail et présente plusieurs perspectives.

1. https://www.loria.fr/fr/presentation/

1 Introduction

Dans ce chapitre, nous présentons d'abord le problème de coloration de graphe pondéré ainsi que l'objectif associé. Nous décrivons ensuite les principales difficultés de sa résolution. Enfin, nous illustrons son intérêt pratique à travers deux applications : les communications par satellite et l'ordonnancement d'examens.

1.1 Problème de coloration de graphe pondéré

Le problème de coloration de graphe pondéré (*Weighted Vertex Coloring Problem*, WVCP) généralise le problème de coloration de graphe standard (*Vertex Coloring Problem*, VCP). Dans les deux problèmes, chaque sommet reçoit une couleur et deux sommets adjacents doivent recevoir des couleurs différentes.

La différence entre les deux problèmes réside dans la fonction objectif. Dans le VCP, l'objectif est de minimiser le nombre total de couleurs utilisées afin de colorier tous les sommets du graphe. Dans le WVCP, chaque sommet possède un poids et le coût d'une couleur est défini comme le poids maximal parmi les sommets auxquels cette couleur est attribuée. L'objectif est alors de minimiser la somme des coûts des différentes couleurs utilisées.

Le VCP peut être considéré comme un cas particulier du WVCP dans lequel tous les sommets possèdent un poids égal à 1. Dans cette situation, le coût de chaque couleur vaut également 1 et minimiser la somme des coûts revient simplement à minimiser le nombre de couleurs utilisées.

Une instance du WVCP est définie par un graphe pondéré $G = (V, E, w)$, où V est l'ensemble des sommets, E l'ensemble des arêtes et $w(v) > 0$ le poids du sommet v.

Pour une coloration réalisable $S = \{C_1, \ldots, C_k\}$, son score est

$$f(S) = \sum_{j=1}^{k} W(C_j), \qquad W(C_j) = \max_{v \in C_j} w(v). \tag{1}$$

Le WVCP consiste à trouver une coloration réalisable S^* qui minimise $f(S)$.

1.2 Difficultés de résolution du WVCP

Le WVCP est NP-difficile, puisque le VCP en constitue le cas particulier où tous les poids valent 1. Les méthodes exactes deviennent donc rapidement coûteuses lorsque la taille ou la densité du graphe augmente. Les résultats présentés dans la sous-section 4.2 confirment cette difficulté : sous la limite de temps fixée, aucune formulation ne prouve l'optimalité sur la totalité des 188 instances.

Une difficulté supplémentaire provient de la structure de la fonction objectif. Contrairement au VCP, le coût d'une couleur dépend uniquement du sommet de poids maximal qui lui est affecté. Ainsi, le déplacement d'un sommet dont le poids est inférieur au maximum de sa classe de couleur n'a généralement aucun impact sur la valeur de la solution.

Par conséquent, une très grande partie des mouvements réalisés au cours de la recherche sont des mouvements neutres, c'est-à-dire qu'ils modifient la répartition des sommets entre les couleurs sans changer le coût total de la solution. De nombreuses colorations différentes possèdent ainsi exactement la même valeur objective et forment de vastes plateaux dans l'espace de recherche.

L'objectif de ce travail est de proposer une approche hybride capable d'améliorer plus régulièrement les solutions du WVCP. Elle associe une recherche adaptative à grands voisinages, des modèles exacts de réparation et une recherche locale. Cette combinaison vise à mieux traverser les plateaux de l'espace de recherche.

1.3 Cas d'application du WVCP

1.3.1 Communication par satellite SS/TDMA

Le WVCP trouve également une application dans les systèmes de communication par satellite de type SS/TDMA (Satellite Switched Time Division Multiple Access) [Ribeiro *et al.*, 1989]. Dans ce type de réseau, plusieurs stations au sol communiquent par l'intermédiaire d'un satellite. Les demandes de communication doivent être réparties sur plusieurs créneaux temporels de manière à respecter les limitations matérielles du système tout en minimisant la durée totale des transmissions.

Le trafic à transmettre est représenté par une matrice carrée

$$T = (t_{ij}),$$

où chaque élément t_{ij} désigne le volume de données que la station i doit transmettre à la station j. Une valeur nulle indique qu'aucune communication n'est requise entre les deux stations.

Par exemple, pour quatre stations, la matrice de trafic est la suivante :

$$T = \begin{pmatrix} 0 & 90 & 40 & 0 \\ 70 & 0 & 20 & 50 \\ 80 & 30 & 0 & 60 \\ 0 & 45 & 75 & 0 \end{pmatrix}$$

Ce problème peut être modélisé sous la forme d'un WVCP. À partir de la matrice de trafic, on construit un graphe dans lequel :

- chaque communication non nulle t_{ij} est représentée par un sommet ;
- le poids du sommet est égal au volume de données t_{ij} ;
- deux sommets sont reliés par une arête lorsque les communications correspondantes utilisent une même station, c'est-à-dire lorsqu'elles possèdent la même station d'émission ou la même station de réception.

Pour la matrice précédente, on obtient les sommets :

$$t_{12}(90),\ t_{13}(40),\ t_{21}(70),\ t_{23}(20),\ t_{24}(50),\ t_{31}(80),\ t_{32}(30),\ t_{34}(60),\ t_{42}(45),\ t_{43}(75).$$

Les incompatibilités entre transmissions conduisent notamment aux arêtes suivantes :

$$(t_{12}, t_{13}), (t_{21}, t_{23}), (t_{21}, t_{24}), (t_{23}, t_{24}), (t_{31}, t_{32}), (t_{31}, t_{34}), (t_{32}, t_{34}), (t_{42}, t_{43}),$$
$$(t_{21}, t_{31}), (t_{12}, t_{32}), (t_{12}, t_{42}), (t_{32}, t_{42}), (t_{13}, t_{23}), (t_{13}, t_{43}), (t_{23}, t_{43}), (t_{24}, t_{34}).$$

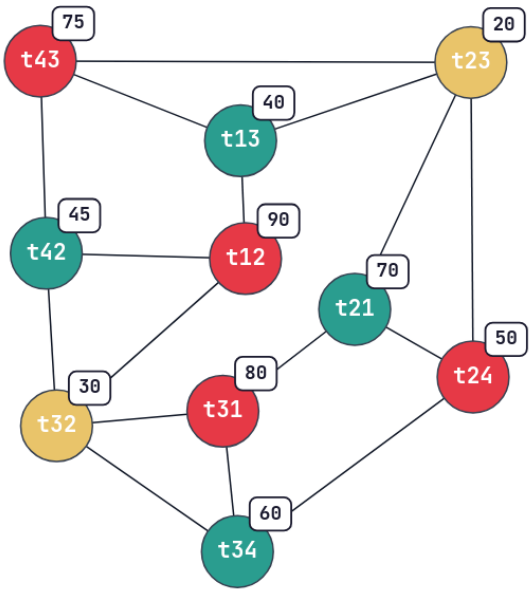


FIGURE 1 – Coloration WVCP du graphe d'incompatibilité associé à la matrice de trafic. Une arête relie deux transmissions ayant la même station d'émission ou la même station de réception. Les trois couleurs représentent une planification réalisable en trois créneaux.

1.3.2 Ordonnancement des examens

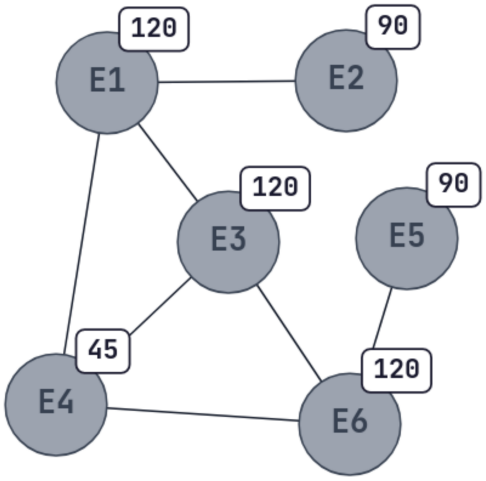
L'ordonnancement des examens est l'une des applications classiques des problèmes de coloration de graphe [Leighton, 1979]. Une université doit répartir un ensemble d'examens dans différents créneaux horaires tout en respectant plusieurs contraintes. La contrainte principale est qu'un étudiant ne peut pas passer deux examens simultanément. Deux examens ayant au moins un étudiant en commun ne peuvent donc pas être programmés au même moment.

Dans sa version classique, ce problème est modélisé comme un problème de coloration de graphe standard dont l'objectif est de minimiser le nombre de créneaux horaires nécessaires. Il est cependant possible d'y ajouter une dimension supplémentaire en considérant la durée des examens.

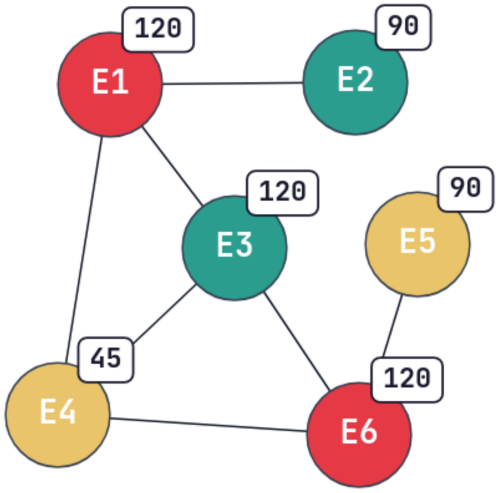
Dans ce contexte, chaque examen possède une durée qui peut être interprétée comme le poids associé à un sommet. La durée d'un créneau horaire correspond alors à la durée du plus long examen qui lui est affecté. L'objectif devient ainsi de minimiser la somme des durées des différents créneaux utilisés, ce qui conduit naturellement à une modélisation sous la forme d'un WVCP.

Examen	Durée	Participants
E1	120	P1, P2, P7
E2	90	P1, P3
E3	120	P2, P4, P8
E4	45	P4, P5, P7
E5	90	P6, P8
E6	120	P4, P6

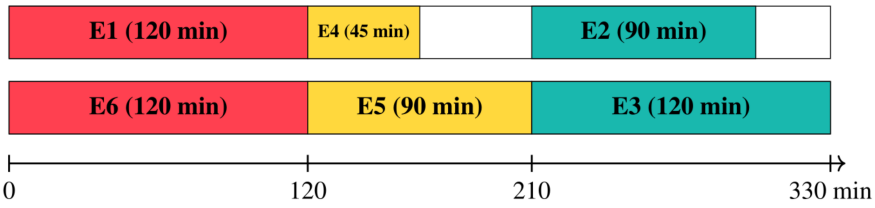
1 - Données d'entrée : examens, durées et participants associées



2 - Représentation du problème par un graphe: chaque examens est représenté par un sommet et deux sommets sont adjacents si les examens correspondant partagent au moins un participant.



3 - Résolution du problème WVCP : les examens partageant la même couleur ne sont pas en conflit et sont planifiés dans le même créneau horaire



4 - Ordonnancement finale des créneaux

FIGURE 2 – Planification des examens par coloration de graphe pondéré.

2 État de l'art

Dans ce chapitre, nous faisons une synthèse des principales méthodes proposées pour résoudre le WVCP. Nous distinguons les méthodes exactes, les heuristiques constructives et les métaheuristiques. Cette synthèse permet également de situer les méthodes retenues dans notre algorithme.

2.1 Méthodes exactes

Les méthodes exactes abordent le WVCP à l'aide de formulations en programmation linéaire en nombres entiers, de la génération de colonnes ou de la programmation par contraintes. [Malaguti *et al.*, 2009] proposent plusieurs formulations fondées sur l'affectation des sommets aux couleurs ou sur la couverture d'ensembles. Parmi elles, le modèle M2 réduit les symétries en associant chaque couleur à son sommet dominant. [Furini and Malaguti, 2012] représentent ensuite chaque couleur par un ensemble stable dont le coût est le poids maximal. Comme le nombre de ces ensembles peut être exponentiel, ils intègrent une génération de colonnes à une procédure de séparation et évaluation afin de prouver l'optimalité.

Dans une autre représentation, [Cornaz *et al.*, 2017] associent chaque classe de couleur à une étoile orientée, enracinée en son sommet le plus lourd. Le WVCP devient alors un problème d'ensemble stable de poids maximal dans un graphe auxiliaire. Cette reformulation est appelée modèle Dual dans la suite. Plus récemment, [Goudet *et al.*, 2023] proposent des bornes sur l'objectif et sur le nombre de couleurs, ainsi que des règles de réduction. Ils présentent également des modèles primal, dual et joint en programmation par contraintes. Les formulations M2 et Dual et leur adaptation à la réparation sont détaillées respectivement dans les sections 3.3.1 et 3.3.2.

2.2 Méthodes approchées

Les méthodes approchées recherchent rapidement des colorations de bonne qualité sans garantir leur optimalité. Nous présentons d'abord deux heuristiques constructives, puis les principales métaheuristiques étudiées pour le WVCP.

2.2.1 Heuristiques

Les heuristiques constructives colorent progressivement les sommets selon un ordre de priorité. Welsh–Powell pondéré [Welsh and Powell, 1967, Grelier, 2023] classe d'abord les sommets par poids décroissant, puis par degré, avant de placer chacun dans la première couleur compatible. DSatur pondéré [Brélaz, 1979, Grelier *et al.*, 2022] tient aussi compte des couleurs déjà présentes dans le voisinage. Il sélectionne en priorité le sommet de saturation maximale, combine cette mesure avec son poids et utilise le degré résiduel pour départager les égalités.

2.2.2 Métaheuristiques

Les métaheuristiques améliorent une ou plusieurs colorations en combinant recherche locale, perturbation et diversification. GRASP [Prais and Ribeiro, 2000] alterne une construction gloutonne aléatoire et une recherche locale, puis ajuste les niveaux de randomisation selon la

qualité des solutions. AFISA [Sun *et al.*, 2018] explore des colorations réalisables et non réalisables avec une recherche tabou. Une pénalité dynamique contrôle les conflits et une perturbation adaptative aide à quitter les optima locaux.

RedLS [Wang *et al.*, 2020] combine la résolution des conflits et l'amélioration du score. Sa liste tabou limite les retours immédiats, tandis qu'une pondération des arêtes difficiles oriente la recherche. ILSTS [Nogueira *et al.*, 2021] alterne également une recherche tabou et des perturbations, dont l'intensité augmente en cas de stagnation. Ses voisinages déplacent un sommet ou réorganisent localement les couleurs.

D'autres travaux explorent des mécanismes plus globaux. MCTS [Grelier *et al.*, 2022] construit un arbre de colorations partielles et utilise UCB pour équilibrer exploration et exploitation. DLMCOL [Goudet *et al.*, 2022] emploie un réseau de neurones pour guider une méthode mémétique dont les recherches locales sont parallélisées. Enfin, AHEAD [Grelier *et al.*, 2024] sélectionne dynamiquement un opérateur de croisement et une recherche locale selon les récompenses observées.

3 Recherche adaptative à grands voisinages (ALNS)

La recherche à grands voisinages (*Large Neighborhood Search*, LNS) repose sur un processus itératif de destruction et de réparation d'une solution courante [Shaw, 1998]. La destruction retire simultanément une partie des affectations. La réparation reconstruit ensuite une solution complète de manière exacte ou approchée. Cette exploration de voisinages étendus aide à quitter les optima locaux et les plateaux de recherche.

La recherche adaptative à grands voisinages (*Adaptive Large Neighborhood Search*, ALNS) est une extension des LNS qui adapte ses choix à partir des performances observées pendant la recherche [Ropke and Pisinger, 2006]. Dans notre méthode, elle sélectionne conjointement un opérateur de destruction et un modèle de réparation, puis ajuste le taux de sommets à détruire.

Dans ce travail, détruire une solution du WVCP revient à décolorer sélectivement un ensemble de sommets. La réparation, confiée au solveur Gurobi, réaffecte ces sommets à des couleurs valides tout en minimisant le coût global. Les sommets non détruits restent regroupés dans leurs classes de couleur. Le sous-problème décide si les sommets détruits rejoignent ces groupes ou forment de nouvelles couleurs. À notre connaissance, cette combinaison d'une destruction adaptative et de réparations exactes n'avait pas encore été proposée spécifiquement pour le WVCP.

La Figure 3 présente les principales étapes de la méthode. Les sections indiquées détaillent ensuite chacune d'elles. La vue algorithmique complète est donnée à la fin du chapitre.

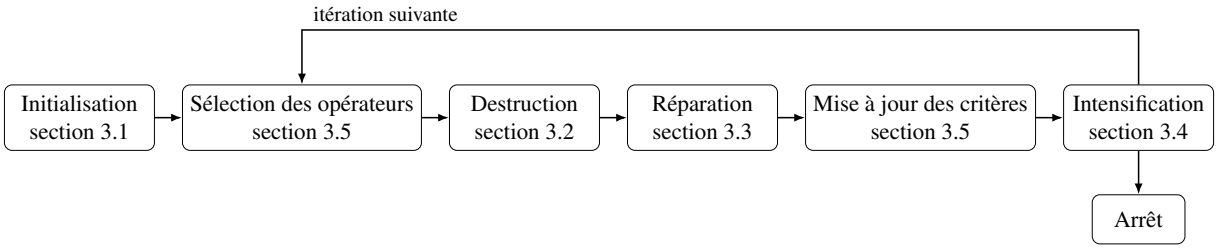


FIGURE 3 – Organisation générale de l'ALNS proposée pour le WVCP.

3.1 Initialisation de la solution

La solution initiale est construite par DSatur pondéré. À chaque étape, les sommets non colorés sont ordonnés en priorité selon leur degré de saturation, c'est-à-dire le nombre de couleurs distinctes présentes dans leur voisinage. Les égalités sont départagées en favorisant le sommet de poids le plus élevé, puis celui de plus grand degré résiduel. Le sommet retenu est placé dans la première couleur compatible ; une nouvelle couleur est créée si aucune couleur existante ne peut le recevoir. Cette initialisation fournit rapidement une coloration réalisable tenant compte à la fois des conflits et des poids.

3.2 Phase de destruction

La phase de destruction consiste à sélectionner un ensemble de sommets $U \subseteq V$ qui seront temporairement décolorés. La solution partielle obtenue après leur retrait est notée S^{-}. Le nombre cible de sommets à détruire est noté K.

La taille réellement transmise au modèle est toujours $k = |U|$. Elle peut être supérieure à K lorsque l'opérateur détruit entièrement une classe de couleur.

Trois opérateurs sont utilisés afin d'explorer différentes parties de l'espace de recherche. Ils s'appuient respectivement sur le poids des sommets, le coût des classes de couleur et la densité locale du graphe. Ils sont notés D_{heavy}, D_{color} et D_{dense}. L'ensemble des opérateurs de destruction est donc

$$\mathcal{D} = \{D_{\text{heavy}}, D_{\text{color}}, D_{\text{dense}}\}. \tag{2}$$

Un mécanisme tabou mémorise les sommets et les classes de couleur récemment détruits. Il évite ainsi de sélectionner les mêmes éléments à chaque itération et favorise l'exploration de différentes parties de la solution. Les éléments non tabous sont toujours examinés en priorité. Lorsque leur nombre ne permet pas d'atteindre la cible K, la restriction est progressivement assouplie afin de garantir la construction du sous-problème, sans modifier l'ordre propre à l'opérateur choisi.

3.2.1 Destruction des sommets les plus lourds (D_{heavy})

Cet opérateur cible les sommets de poids élevé. Dans le WVCP, ces sommets sont particulièrement importants, car ils sont susceptibles de déterminer le coût de leur classe de couleur. Leur retrait permet au mécanisme de réparation de remettre en cause leur affectation.

Les sommets des instances sont déjà ordonnés par poids décroissant. L'opérateur les parcourt donc directement dans cet ordre. Il retient d'abord les sommets autorisés par la mémoire taboue, puis complète l'ensemble si cela est nécessaire pour atteindre la taille cible. Les sommets détruits sont ensuite inscrits dans cette mémoire.

Algorithme 1 Opérateur D_{heavy} : destruction des sommets les plus lourds

Entrée : Solution courante S, cible K, mémoire taboue des sommets
Sortie : Solution partielle S^-, ensemble détruit U et mémoire mise à jour

$S^- \leftarrow S$
$U \leftarrow \varnothing$
pour chaque sommet v, par poids décroissant **faire**
    si $|U| = K$ **alors**
        sortir de la boucle
    si v est coloré et autorisé par la mémoire taboue **alors**
        Décolorer v dans S^- et l'ajouter à U
Compléter U selon le même ordre si $|U| < K$
Mettre à jour la mémoire taboue avec les sommets de U
retourner (S^-, U)

3.2.2 Destruction des couleurs les plus coûteuses (D_{color})

Cet opérateur cible les classes de couleur qui contribuent le plus à la fonction objectif. Au début de la destruction, les couleurs non vides sont ordonnées par coût décroissant. Lorsqu'une couleur admissible est sélectionnée, elle est entièrement détruite. Tous ses sommets sont décolorés et ajoutés à U, puis la mémoire taboue des couleurs est mise à jour. L'opérateur poursuit ce traitement jusqu'à ce que $|U| \geq K$. Il peut donc dépasser la taille cible, car une classe de couleur n'est jamais détruite partiellement.

Algorithme 2 Opérateur D_{color} : destruction des couleurs les plus coûteuses

Entrée : Solution courante S, cible K, mémoire taboue des couleurs
Sortie : Solution partielle S^-, ensemble détruit U et mémoire mise à jour

$S^- \leftarrow S$
$U \leftarrow \varnothing$
$C \leftarrow$ couleurs non vides de S^-, ordonnées par coût décroissant
pour chaque couleur $c \in C$, dans l'ordre défini **faire**
    si $|U| \geq K$ **alors**
        sortir de la boucle
    si c est autorisée par la mémoire taboue **alors**
        Décolorer tous les sommets de c et les ajouter à U
Continuer selon le même ordre, si nécessaire, jusqu'à $|U| \geq K$
Mettre à jour la mémoire taboue avec les couleurs détruites
retourner (S^-, U)

3.2.3 Destruction d'une zone dense (D_{dense})

Cet opérateur cherche à perturber une région fortement connectée du graphe. Pour chaque sommet v, il calcule la densité du sous-graphe induit par son voisinage ouvert $N(v)$:

$$\text{dens}(v) = \begin{cases} \dfrac{2|E(N(v))|}{d(v)(d(v)-1)}, & \text{si } d(v) \geq 2, \\ 0, & \text{sinon,} \end{cases} \tag{3}$$

où $d(v) = |N(v)|$. Le score d'un centre est ensuite défini par

$$\rho(v) = \text{dens}(v)\big(d(v)+1\big). \tag{4}$$

La multiplication par $d(v)+1$ favorise les voisinages à la fois denses et suffisamment grands. Les scores sont calculés une seule fois avant la boucle principale.

Cet opérateur partage la mémoire taboue des sommets avec la destruction des sommets lourds. Le statut tabou est vérifié uniquement pour le choix du centre. Les voisins d'un centre admissible peuvent être détruits quel que soit leur statut.

Les centres sont examinés par score décroissant. L'opérateur décolore d'abord le centre, puis ses voisins. Il s'arrête dès que K sommets ont été détruits et met à jour la mémoire taboue.

Algorithme 3 Opérateur D_{dense} : destruction d'une zone dense

Entrée : Solution courante S, cible K, mémoire taboue des sommets, scores $\rho(v)$
Sortie : Solution partielle S^-, ensemble détruit U et mémoire mise à jour

$S^- \leftarrow S$
$U \leftarrow \varnothing$
$C \leftarrow V$, ordonné par score ρ décroissant
tant que $|U| < K$ et $C \neq \varnothing$ **faire**
    Retirer de C le premier centre autorisé par la mémoire taboue
    Décolorer ce centre, puis ses voisins, jusqu'à atteindre K
Compléter la sélection selon le même ordre si $|U| < K$
Mettre à jour la mémoire taboue avec les sommets de U
retourner (S^-, U)

3.2.4 Données conservées pour la réparation

Matrice de similarité B Avant la destruction, la matrice booléenne B est construite à partir de la solution courante S :

$$B_{uv} = \begin{cases} 1, & \text{si } u \text{ et } v \text{ appartiennent à la même couleur dans } S, \\ 0, & \text{sinon.} \end{cases} \tag{5}$$

Après la destruction de U, les lignes et les colonnes associées aux sommets détruits sont annulées :

$$B^{-}_{uv} = \begin{cases} B_{uv}, & u \notin U \text{ et } v \notin U, \\ 0, & \text{sinon.} \end{cases} \tag{6}$$

Les relations restantes définissent les groupes conservés

$$\mathcal{G} = \{G_1, \ldots, G_q\}. \tag{7}$$

Chaque G_g contient tous les sommets non détruits d'une même couleur. Pour chaque groupe, le programme conserve :

— son ensemble de sommets G_g ;
— son poids maximal $W_g = \max_{v \in G_g} w_v$.

Ces sommets restent regroupés pendant toute la réparation.

Matrice de compatibilité A La matrice booléenne de compatibilité groupe–sommet, notée $A = (a_{ig})$, est définie par :

$$a_{ig} = 1 \quad \Longleftrightarrow \quad \forall v \in G_g,\ \{u_i, v\} \notin E. \tag{8}$$

Ainsi, $a_{ig} = 1$ signifie que u_i peut rejoindre le groupe G_g sans créer de conflit avec son noyau conservé. Cette matrice est calculée après chaque destruction, avant la construction du modèle de réparation. Dans M2, elle impose $x_{ig} \leq a_{ig}$ et interdit une affectation incompatible. Dans Dual, un arc entre u_i et G_g est créé uniquement lorsque $a_{ig} = 1$. Elle permet donc aux deux modèles de tenir compte des couleurs conservées sans réintroduire tous leurs sommets comme variables de décision.

3.3 Phase de réparation

La phase de réparation reçoit la solution partielle S^{-}, les groupes conservés $\mathcal{G}$ et les sommets détruits U. Elle doit reconstruire une coloration complète sans modifier l'intérieur des groupes conservés. Deux formulations complémentaires sont utilisées. M2 décrit directement l'affectation des sommets aux couleurs, tandis que Dual maximise les économies obtenues en regroupant des éléments compatibles. Dans les deux cas, la limite de cinq secondes impose de résoudre rapidement un sous-problème plus petit que l'instance complète.

Ces deux opérateurs de réparation sont respectivement notés R_{M2} et R_{D}. L'ensemble des opérateurs de réparation est donc défini par

$$\mathcal{R} = \{R_{\mathrm{M2}}, R_{\mathrm{D}}\}. \tag{9}$$

3.3.1 Réparation fondée sur le modèle M2 (R_{M2})

Modèle original. Soit $V = \{1, \dots, n\}$ l'ensemble des sommets, indexés par poids décroissant, avec $w_1 \geq w_2 \geq \cdots \geq w_n$. Pour $h \leq i$, la variable binaire y_{ih} vaut 1 si le sommet i reçoit la couleur représentée par le sommet dominant h. Le modèle original de [Malaguti *et al.*, 2009] s'écrit :

$$\min \quad \sum_{h=1}^{n} w_h y_{hh} \tag{M2-1}$$

$$\text{s.c.} \quad \sum_{h=1}^{i} y_{ih} = 1, \qquad \forall i \in V, \tag{M2-2}$$

$$y_{ih} + y_{jh} \leq y_{hh}, \qquad \forall \{i,j\} \in E, \quad h \leq \min(i,j), \tag{M2-3}$$

$$y_{ih} \leq y_{hh}, \qquad \forall i \in V, \quad h < i, \tag{M2-4}$$

$$y_{ih} \in \{0,1\}, \qquad \forall i \in V, \quad h \leq i. \tag{M2-5}$$

L'objectif (M2-1) minimise la somme des poids des sommets dominants. La contrainte (M2-2) affecte chaque sommet à une unique couleur. La contrainte (M2-3) interdit à deux sommets adjacents de partager une couleur. La contrainte (M2-4) exige que le sommet dominant initialise la couleur. Enfin, (M2-5) définit le domaine des variables.

Adaptation au sous-problème de l'ALNS. Le modèle adapté décide si chaque sommet détruit rejoint un groupe conservé ou une nouvelle couleur représentée par un sommet détruit. Les sommets de $U = \{u_1, \dots, u_k\}$ sont indexés par poids décroissant et $w_i = w(u_i)$. Les variables x_{ig} indiquent l'affectation de u_i au groupe conservé G_g. Les variables y_{ih} conservent le sens du modèle original pour les nouvelles couleurs. Enfin, Z_g représente le coût du groupe G_g après réparation.

$$\min \quad \sum_{g=1}^{q} Z_g + \sum_{i=1}^{k} w_i y_{ii} \tag{R_{M2}-1}$$

$$\text{s.c.} \quad \sum_{g=1}^{q} x_{ig} + \sum_{h=1}^{i} y_{ih} = 1, \qquad i = 1, \dots, k, \tag{R_{M2}-2}$$

$$y_{ih} \leq y_{hh}, \qquad 1 \leq h < i \leq k, \tag{R_{M2}-3}$$

$$x_{ig} \leq a_{ig}, \qquad i = 1, \dots, k, \quad g = 1, \dots, q, \tag{R_{M2}-4}$$

$$Z_g \geq W_g, \qquad g = 1, \dots, q, \tag{R_{M2}-5}$$

$$Z_g \geq w_i x_{ig}, \qquad i = 1, \dots, k, \quad g = 1, \dots, q, \tag{R_{M2}-6}$$

$$x_{ig} + x_{jg} \leq 1, \qquad \forall \{u_i, u_j\} \in E, \quad g = 1, \dots, q, \tag{R_{M2}-7}$$

$$y_{ih} + y_{jh} \leq y_{hh}, \qquad \forall \{u_i, u_j\} \in E, \quad h \leq \min(i,j), \tag{R_{M2}-8}$$

$$x_{ig} \in \{0,1\}, \qquad i = 1, \dots, k, \quad g = 1, \dots, q,$$

$$y_{ih} \in \{0,1\}, \qquad 1 \leq h \leq i \leq k,$$

$$Z_g \geq 0, \qquad g = 1, \dots, q. \tag{R_{M2}-9}$$

L'objectif (R_{M2}-1) additionne le coût des groupes conservés et celui des nouvelles couleurs. La contrainte (R_{M2}-2) affecte chaque sommet détruit à une seule destination. La contrainte (R_{M2}-3)

impose l'initialisation d'une nouvelle couleur par son sommet dominant. La contrainte (R_{M2}-4) interdit l'affectation d'un sommet à un groupe conservé incompatible.

Les contraintes (R_{M2}-5) et (R_{M2}-6) mettent à jour le coût de chaque groupe. À l'optimum, Z_g est le maximum entre W_g et les poids des sommets qui rejoignent G_g. Les contraintes (R_{M2}-7) et (R_{M2}-8) interdisent les conflits dans les groupes conservés et les nouvelles couleurs. Enfin, (R_{M2}-9) définit le domaine des variables.

Décodage et intégration dans la solution globale. La solution réparée est initialisée avec la solution partielle S^-, qui contient déjà les groupes conservés. Les variables sont ensuite décodées dans l'ordre $u_1, \ldots, u_k$:

— si $x_{ig} = 1$, u_i est ajouté à la couleur globale associée au groupe G_g ;
— si $y_{ii} = 1$, une nouvelle couleur globale est créée pour u_i ;
— si $y_{ih} = 1$ avec $h < i$, u_i est ajouté à la couleur globale précédemment créée par u_h.

La solution obtenue doit être complète et sans conflit. Dans le cas contraire, le décodage est déclaré invalide et la solution précédant la destruction est restaurée.

3.3.2 Réparation fondée sur le modèle Dual (R_{Dual})

Modèle original. Le modèle Dual de [Cornaz *et al.*, 2017] reformule le WVCP comme un problème d'ensemble stable de poids maximal. Une classe de couleur est un ensemble stable dans $G = (V, E)$ et devient une clique dans le graphe complémentaire $\bar{G} = (V, \bar{E})$.

Cette seconde représentation complète M2 et ne constitue donc pas un modèle introduit indépendamment du reste de l'approche. Les évaluations de la littérature montrent notamment que Dual est plus performant sur plusieurs graphes denses [Cornaz *et al.*, 2017, Goudet *et al.*, 2023]. Cette complémentarité motive son utilisation comme second opérateur de réparation.

Les sommets sont indexés par poids décroissant, $w(v_1) \geq \cdots \geq w(v_n)$. Chaque arête du complémentaire est orientée du sommet le plus lourd vers le plus léger :

$$v_i \to v_j \quad \text{si } i < j. \tag{D-1}$$

Dans chaque clique orientée, l'unique sommet sans arc entrant est le sommet le plus lourd et représente la couleur.

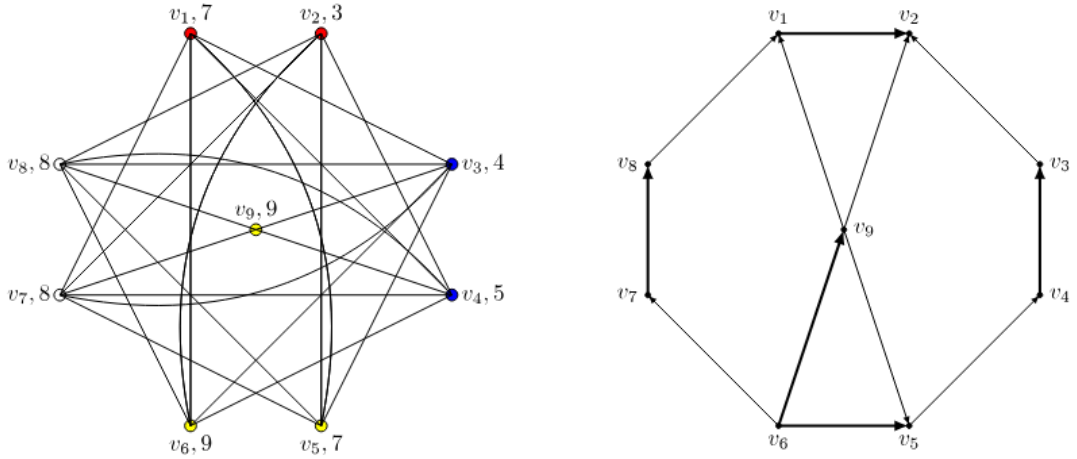


Fig. 1. Example: a graph G (left) and the orientation of its complement $\bar{G}$ (right).

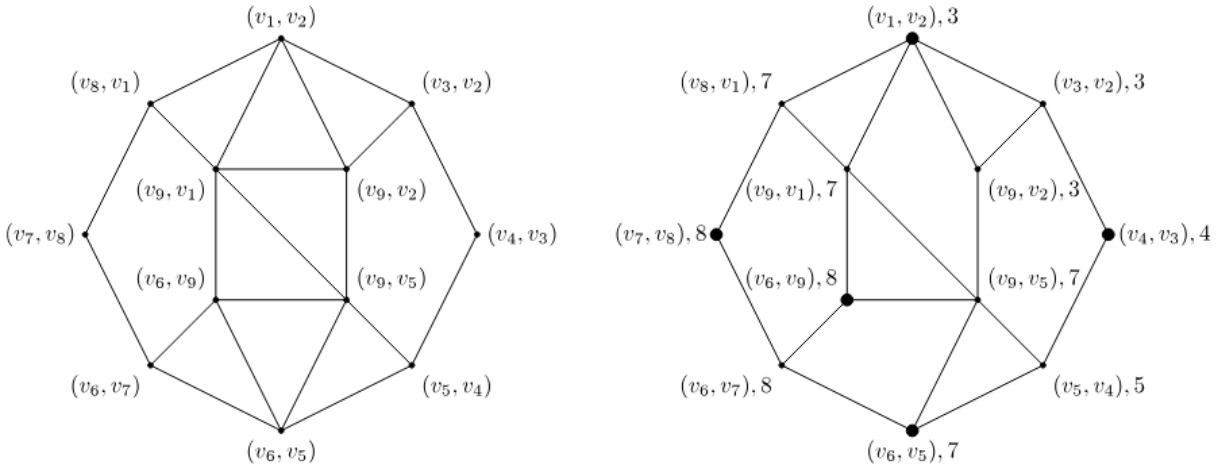


Fig. 2. Example: the line-graph $L(\overline{G})$ (left) and auxiliary graph $\hat{G}$ (right).

FIGURE 4 – Étapes de construction du graphe auxiliaire. En haut : graphe initial et complémentaire orienté. En bas : graphe ligne et graphe auxiliaire pondéré. [Cornaz *et al.*, 2017].

Chaque arc de $\vec{\bar{G}}$ devient un sommet du graphe auxiliaire $\hat{G}$:

$$V(\hat{G}) = \vec{E}. \tag{D-2}$$

Deux sommets de $\hat{G}$ sont adjacents lorsque leurs arcs sont incompatibles : ils ont la même origine avec des cibles adjacentes dans G, ils ont la même cible, ou la cible de l'un est l'origine de l'autre. Un ensemble stable de $\hat{G}$ regroupe donc des arcs compatibles appartenant aux étoiles orientées qui représentent les couleurs.

Chaque arc uv reçoit le poids de sa cible, $\hat{w}_{uv} = w(v)$. Le lien avec l'objectif du WVCP est

$$\alpha(\hat{G}, \hat{w}) + \chi_{\max}(G, w) = \sum_{v \in V} w(v), \tag{D-3}$$

où $\alpha(\hat{G}, \hat{w})$ désigne le poids maximal d'un ensemble stable de $\hat{G}$, tandis que $\chi_{\max}(G, w)$ désigne la valeur optimale du WVCP sur le graphe pondéré (G, w), c'est-à-dire le coût minimal d'une coloration propre.

Pour chaque arc $e \in \vec{E}$, la variable binaire x_e indique si le sommet correspondant de $\hat{G}$ appartient à l'ensemble stable :

$$\max \quad \sum_{e \in \vec{E}} \hat{w}_e x_e \tag{D1}$$

$$\text{s.c.} \quad x_{vu} + x_{vw} \leq 1, \qquad \substack{v,u,w \in V:\ vu, vw \in \vec{E},\\ uw, wu \notin \vec{E}}, \tag{D2}$$

$$x_e + x_f \leq 1, \qquad \forall v \in V, \quad e \in \delta^-(v), \quad f \in \delta^+(v), \tag{D3}$$

$$x_e + x_f \leq 1, \qquad \forall v \in V, \quad e, f \in \delta^-(v), \quad e \neq f, \tag{D4}$$

$$x_e \in \{0,1\}, \qquad \forall e \in \vec{E}. \tag{D5}$$

L'objectif (D1) maximise le poids des arcs sélectionnés. La contrainte (D2) interdit deux arcs de même origine lorsque leurs cibles ne peuvent pas partager une couleur. La contrainte (D3) interdit les chaînes orientées. La contrainte (D4) empêche un nœud d'avoir deux dominants. Enfin, (D5) définit le domaine des variables.

Adaptation au sous-problème de l'ALNS. Le modèle adapté reconstruit la solution à partir d'un graphe auxiliaire orienté. Ses nœuds correspondent aux groupes conservés $\mathcal{G} = \{G_1, \ldots, G_q\}$ et aux sommets détruits $U = \{u_1, \ldots, u_k\}$, avec $w_i = w(u_i)$. Un arc représente la possibilité de placer deux éléments compatibles dans une même couleur. Il est orienté de l'élément le plus lourd, appelé *dominant*, vers le plus léger, appelé *dominé*. Deux nœuds sont compatibles lorsque l'union des sommets qu'ils représentent est un ensemble stable dans le graphe initial. L'ensemble $\mathcal{I}$ contient les paires de nœuds incompatibles.

Pour un groupe et un sommet compatible, l'orientation dépend de la comparaison entre W_g et w_i. Entre deux sommets détruits non adjacents, l'arc est orienté du plus lourd vers le plus léger. Le modèle de réparation R_{D} complet s'écrit alors dans un seul bloc :

$$\vec{E} = \{(G_g, u_i) : a_{ig} = 1,\ W_g \geq w_i\} \cup \{(u_i, G_g) : a_{ig} = 1,\ w_i > W_g\} \cup \{(u_i, u_j) : i < j,\ \{u_i, u_j\} \notin E\}, \tag{R_{D}-1}$$

$$w(e) = \begin{cases} w_i, & e = (G_g, u_i), \\ W_g, & e = (u_i, G_g), \\ w_j, & e = (u_i, u_j), \end{cases} \tag{R_{D}-2}$$

$$\max \quad \sum_{e \in \vec{E}} w(e) x_e \tag{R_{D}-3}$$

$$\text{s.c.} \quad x_{(a,b)} + x_{(a,c)} \leq 1, \quad \forall (b,c) \in \mathcal{I}, (a,b), (a,c) \in \vec{E}, \tag{R_{D}-4}$$

$$x_{e_{\mathrm{in}}} + x_{e_{\mathrm{out}}} \leq 1, \quad \forall v \in \mathcal{G} \cup U, e_{\mathrm{in}} \in \delta^-(v),\ e_{\mathrm{out}} \in \delta^+(v), \tag{R_{D}-5}$$

$$\sum_{e \in \delta^-(v)} x_e \leq 1, \quad \forall v \in \mathcal{G} \cup U, \tag{R_{D}-6}$$

$$x_e \in \{0,1\}, \quad \forall e \in \vec{E}. \tag{R_{D}-7}$$

La relation (R_{D}-1) construit les arcs admissibles et (R_{D}-2) attribue à chaque arc le poids de son élément dominé. Ce poids représente le coût économisé lorsque la cible rejoint la couleur de l'origine ; maximiser (R_{D}-3) revient donc à minimiser le coût réparé. La contrainte (R_{D}-4) interdit de placer sous un même dominant deux éléments incompatibles. La contrainte (R_{D}-5)

empêche un élément dominé de devenir à son tour dominant, tandis que (R_{D}-6) lui impose au plus un dominant. Les arcs sélectionnés forment ainsi des étoiles disjointes représentant les couleurs réparées, et (R_{D}-7) définit le domaine des variables.

Décodage et intégration dans la solution globale. La solution réparée est initialisée avec la solution partielle S^{-}, qui contient déjà les groupes conservés. Les arcs sélectionnés sont ensuite utilisés pour reconstruire les couleurs :

- lorsqu'une étoile est centrée sur un groupe conservé G_g, les sommets détruits reliés à ce groupe sont ajoutés à la couleur globale associée à G_g ;
- lorsqu'une étoile est centrée sur un sommet détruit u_i et contient un groupe G_g, u_i ainsi que les autres sommets détruits de l'étoile sont ajoutés à la couleur globale associée à G_g ;
- lorsqu'une étoile contient uniquement des sommets détruits, une nouvelle couleur globale est créée pour sa racine, puis les autres sommets de l'étoile y sont ajoutés ;
- tout sommet détruit qui n'appartient à aucune étoile définie par les arcs sélectionnés est placé dans une nouvelle couleur singleton.

Les groupes conservés non concernés par un arc sélectionné restent inchangés. La solution obtenue doit être complète et de pénalité nulle. Dans le cas contraire, le décodage est déclaré invalide et la solution précédant la destruction est restaurée.

3.4 Intensification

Après chaque réparation valide, RedLS intensifie la solution obtenue pendant un nombre limité d'itérations [Wang *et al.*, 2020]. Cette recherche locale combine la résolution des conflits, l'amélioration du score et une mémoire taboue. Elle exploite ainsi rapidement le voisinage de la solution reconstruite avant l'itération suivante de l'ALNS.

ILSTS s'est montré plus stable et globalement plus performant lors des expériences préliminaires. RedLS converge toutefois plus rapidement. Cette vitesse le rend mieux adapté aux intensifications courtes et répétées intégrées à l'ALNS. La solution intensifiée n'est acceptée que si elle reste réalisable et ne dégrade pas la solution précédant la destruction.

3.5 Sélection des opérateurs

La sélection adaptative constitue une composante centrale de la méthode proposée [Ropke and Pisinger, 2006]. Un faible taux de destruction facilite la résolution du sous-problème, mais peut limiter la diversification. Un taux élevé explore un voisinage plus large, mais rend le modèle de réparation plus difficile. De même, aucun opérateur de destruction ou de réparation n'est systématiquement le meilleur pour toutes les instances et toutes les phases de la recherche.

L'ALNS proposée adapte donc :

- le taux de sommets à détruire, en fonction des performances de Gurobi ;
- le choix conjoint de l'opérateur de destruction et de l'opérateur de réparation, en fonction des améliorations déjà obtenues.

3.5.1 Sélection de couples destruction–réparation

Les trois opérateurs de destruction de $\mathcal{D}$ et les deux opérateurs de réparation de $\mathcal{R}$ forment l'ensemble des couples $\mathcal{O} = \mathcal{D} \times \mathcal{R}$. Le produit cartésien associe chaque opérateur de destruc-

tion à chaque opérateur de réparation. Il contient donc les six couples

$$\begin{aligned}\mathcal{O} = \mathcal{D} \times \mathcal{R} = \{&(D_{\text{heavy}}, R_{\text{M2}}), (D_{\text{heavy}}, R_{\text{Dual}}),\\ &(D_{\text{color}}, R_{\text{M2}}), (D_{\text{color}}, R_{\text{Dual}}),\\ &(D_{\text{dense}}, R_{\text{M2}}), (D_{\text{dense}}, R_{\text{Dual}})\}.\end{aligned} \tag{10}$$

Chaque couple $o \in \mathcal{O}$ possède un poids ω_o, initialisé à zéro. Ce poids règle sa probabilité de sélection : une amélioration augmente ω_o et rend le couple concerné plus susceptible d'être choisi lors des itérations suivantes. Soit S_t la solution avant destruction et S_{t+1} la solution à la fin de l'itération, après réparation, RedLS et contrôle de non-dégradation. Le gain est

$$\Delta_t = f(S_t) - f(S_{t+1}). \tag{11}$$

Une récompense est accordée uniquement si $\Delta_t > 0$:

$$\gamma_t = \Delta_t(1 + \eta t), \qquad \eta = 0{,}01. \tag{12}$$

Le poids du couple sélectionné o_t est alors mis à jour par

$$\omega_{o_t} \leftarrow \omega_{o_t} + \gamma_t. \tag{13}$$

Le facteur dépendant de l'itération donne davantage d'importance aux améliorations obtenues tardivement, lorsque la solution est généralement plus difficile à améliorer.

3.5.2 Mécanisme Roulette

Le mécanisme Roulette suit le principe de sélection adaptative d'opérateurs utilisé en ALNS par [Ropke and Pisinger, 2006]. Il transforme les poids en probabilités de sélection. La masse du couple $o \in \mathcal{O}$ est $1 + \omega_o$, d'où

$$\Pr_{\text{Roulette}}(o) = \frac{1 + \omega_o}{\sum_{o' \in \mathcal{O}} (1 + \omega_{o'})}. \tag{14}$$

Le terme 1 rend la première sélection uniforme lorsque tous les poids sont nuls. Il garantit également qu'un couple non récompensé conserve toujours une probabilité strictement positive. Roulette favorise donc les associations performantes sans supprimer complètement l'exploration des autres couples.

3.5.3 Mécanisme Deleter

Deleter complète le poids ω_o par une chance λ_o, initialisée à 1. La probabilité de sélection devient

$$\Pr_{\text{Deleter}}(o) = \frac{(1 + \omega_o)\lambda_o}{\sum_{o' \in \mathcal{O}} (1 + \omega_{o'})\lambda_{o'}}. \tag{15}$$

Après une amélioration, la chance du couple sélectionné est réinitialisée à 1. Sinon, elle est multipliée par $\beta = 0{,}50$. Le couple est supprimé si la nouvelle valeur devient inférieure à $\lambda_{\min} = 0{,}05$ et si un autre couple reste actif. Dans le cas contraire, sa chance est maintenue à $\lambda_{\min}$. Un couple supprimé n'est pas réactivé. Deleter élimine donc progressivement les couples qui échouent, alors que Roulette maintient leur exploration.

3.5.4 Adaptation du taux de destruction

Le taux commence à $d_0 = 0{,}20$. Après la réparation, il est adapté à partir du statut σ_t du modèle Gurobi et de son temps de résolution r_t. Le temps imparti à chaque réparation est fixé à cinq secondes :

$$d_{t+1} = \begin{cases} \max(0{,}05, d_t - 0{,}05), & \text{si } \sigma_t \neq \text{OPTIMAL}, \\ \min(1, d_t + 0{,}05), & \text{si } \sigma_t = \text{OPTIMAL et } r_t < 3 \text{ s}, \\ d_t, & \text{sinon.} \end{cases} \tag{16}$$

Si Gurobi ne prouve pas l'optimalité dans le temps imparti, le sous-problème est considéré comme trop difficile et le taux est diminué. Si l'optimalité est prouvée rapidement, le taux est augmenté afin d'explorer un voisinage plus large. Lorsque le modèle est optimal mais nécessite au moins trois secondes, le taux reste inchangé. Cette adaptation utilise la performance du modèle et non la seule amélioration du score.

3.6 Vue d'ensemble de l'algorithme

L'Algorithme 4 rassemble les étapes précédentes. La sélection et l'adaptation encadrent ainsi chaque cycle de destruction, réparation et intensification.

Algorithme 4 ALNS hybride proposée pour le WVCP

Entrée : Instance pondérée G, limite de temps, ensembles d'opérateurs $\mathcal{D}$ et $\mathcal{R}$
Sortie : Meilleure coloration $S^\star$

Construire l'ensemble des couples $\mathcal{O} \leftarrow \mathcal{D} \times \mathcal{R}$
Construire S avec DSatur pondéré et poser $S^\star \leftarrow S$
Initialiser les mémoires taboues, les poids ω_o pour $o \in \mathcal{O}$ et le taux d
Poser $t \leftarrow 1$
tant que la limite de temps n'est pas atteinte **faire**
    Sélectionner un couple $o_t = (D_t, R_t) \in \mathcal{O}$ ▷ section 3.5
    Calculer $K \leftarrow \lceil d\lvert V\rvert \rceil$ et construire B à partir de S
    Détruire S avec D_t pour obtenir U et S^- ▷ section 3.2
    Mettre à jour B, puis construire $\mathcal{G}$ et A ▷ section 3.2.4
    Réparer S^- avec R_t ▷ section 3.3
    Adapter le taux d à partir du statut et du temps de réparation
    si la réparation est valide **alors**
        Intensifier la solution réparée avec RedLS ▷ section 3.4
        si la solution est valide et non dégradée **alors**
            Accepter la solution et mettre à jour $S^\star$
        sinon
            Restaurer la solution précédant la destruction
    sinon
        Restaurer la solution précédant la destruction
    Mettre à jour les critères de sélection du couple o_t
    $t \leftarrow t + 1$
retourner $S^\star$

4 Analyse expérimentale

Dans ce chapitre, nous évaluons expérimentalement les différentes approches étudiées pour résoudre le WVCP. Nous comparons d'abord les formulations exactes, puis les mécanismes Roulette et Deleter de l'ALNS. La configuration retenue est ensuite confrontée aux méthodes de l'état de l'art sur un sous-ensemble de 55 instances difficiles. Enfin, une analyse statistique complète la comparaison des méthodes approchées.

4.1 Conditions expérimentales

Deux protocoles expérimentaux sont considérés. Le premier compare les modèles exacts sur 188 instances du WVCP. Une exécution est réalisée pour chaque couple modèle–instance. Le second compare les méthodes approchées sur un sous-ensemble de 55 instances. Ce sous-ensemble regroupe les instances considérées comme les plus difficiles du benchmark, notamment en raison de leur taille et de la difficulté rencontrée par les modèles exacts. Il a été fixé indépendamment des résultats de l'ALNS afin d'éviter de sélectionner a posteriori les instances qui lui sont favorables. Sa taille limite aussi le coût de la campagne, qui prévoit plusieurs exécutions d'une heure par méthode et par instance.

Les méthodes ont été implémentées en C++17 et compilées avec GCC 13.3.0. Les modèles exacts et les réparations de l'ALNS sont résolus avec Gurobi 13.0.0. Toutes les expérimentations ont été exécutées sur le cluster *grvingt* du site de Nancy de Grid'5000 [Balouek *et al.*, 2013]. Chaque nœud est équipé de deux processeurs Intel Xeon Gold 6130 à 2,10 GHz, comportant chacun 16 cœurs, et de 192 Gio de mémoire vive. Une exécution réserve un cœur et sa durée est limitée à 3 600 secondes. Pour chacune des 55 instances, ILSTS, RedLS et chacune des deux versions de l'ALNS disposent de 20 exécutions prévues, soit 1 100 exécutions par méthode ou mécanisme.

Les codes sources sont disponibles dans deux dépôts GitHub distincts : les modèles exacts à l'adresse https://github.com/HabibKHTEIRA/WVCP_modeles et la méthode ALNS à l'adresse https://github.com/HabibKHTEIRA/WVCP_ALNS.

Dans les tableaux de résultats, *Meill.* désigne le meilleur score obtenu, *Moy.* le score moyen et *Temps* le temps d'obtention du meilleur score, exprimé en secondes. La colonne BKS indique le meilleur score connu (*Best Known Score*). Une étoile signale que ce score est optimal. Dans le tableau des 55 instances, les valeurs en gras indiquent le meilleur score parmi les quatre méthodes ou la meilleure moyenne parmi les trois méthodes approchées. En cas d'égalité, toutes les valeurs correspondantes sont mises en gras. Joint ne possède pas de colonne moyenne, car une seule exécution exacte est disponible par instance. Dans ses colonnes, « TL » indique que la limite de temps a été atteinte sans solution exploitable.

4.2 Comparaison expérimentale des modèles exacts

Cette sous-section compare les formulations exactes étudiées pour le WVCP et évalue l'effet des bornes proposées par [Goudet *et al.*, 2023].

Les cinq configurations évaluées sont les suivantes :

— Dual, fondé sur la reformulation de [Cornaz *et al.*, 2017] ;
— Dual+B, le modèle Dual renforcé par les bornes de [Goudet *et al.*, 2023] ;
— M2, proposé par [Malaguti *et al.*, 2009] ;
— M2+B, le modèle M2 renforcé par les bornes de [Goudet *et al.*, 2023] ;

— Joint, lie les représentations primale et duale [Goudet *et al.*, 2023] ; dans nos expériences, il s'agit d'une formulation linéaire en nombres entiers implémentée avec Gurobi.

Dans toute la suite, le suffixe « +B » indique l'ajout des bornes au modèle correspondant. Les modèles sont comparés selon le nombre de preuves d'optimalité obtenues, le temps nécessaire pour établir ces preuves et le nombre de meilleurs scores connus atteints.

Le Tableau 1 regroupe les comparaisons par paires et le résumé global. Dans la matrice centrale, chaque case est présentée sous la forme O/T. La valeur O compte les instances dont l'optimalité est prouvée uniquement par le modèle indiqué en ligne. La valeur T compte les instances communes pour lesquelles ce modèle établit la preuve plus rapidement. Les deux dernières colonnes indiquent respectivement le nombre de meilleurs scores connus atteints et le nombre d'optimalités prouvées.

	Comparaisons par paires : O/T					Résumé global	
	Dual	Dual+B	M2	M2+B	Joint	# meilleurs scores connus	# optimalités prouvées
Dual	–	$0/43$	**11**/**105**	$0/$**120**	$0/$**99**	139	133
Dual+B	$0/$**57**	–	**11**/**104**	$0/$**120**	$0/$**91**	139	133
M2	$0/16$	$0/17$	–	$0/$**71**	$0/15$	133	122
M2+B	**3**/13	**3**/13	**14**/32	–	**1**/10	144	**136**
Joint	**2**/30	**2**/34	**13**/**107**	$0/$**125**	–	**152**	135

TABLE 1 – Comparaison par paires et résumé global des modèles exacts

Le Tableau 1 montre d'abord que les bornes n'apportent aucune preuve supplémentaire à Dual : Dual et Dual+B prouvent chacun 133 optimalités et atteignent 139 meilleurs scores connus. Elles améliorent néanmoins le temps de preuve sur certaines instances, puisque Dual+B est plus rapide que Dual dans 57 cas, contre 43 dans le sens inverse.

L'effet des bornes est plus marqué pour M2. M2+B prouve l'optimalité de 14 instances non résolues par M2, sans observer le cas inverse. Son total passe ainsi de 122 à 136 preuves. Cette amélioration du nombre de preuves ne s'accompagne cependant pas d'une réduction systématique du temps : M2 est plus rapide dans 71 comparaisons communes, contre 32 pour M2+B.

Joint obtient le plus grand nombre de meilleurs scores connus, avec 152, et prouve 135 optimalités. Il devance nettement M2 et M2+B sur le temps de preuve, avec respectivement 107 et 125 comparaisons favorables, contre 15 et 10 dans le sens inverse. Dual et Dual+B sont toutefois plus souvent rapides que Joint sur leurs instances communes, avec respectivement 99 et 91 comparaisons favorables.

Ces résultats confirment l'intérêt de Joint pour résoudre des instances complètes. Dans l'ALNS, chaque réparation doit cependant être répétée sous une limite de temps courte. Joint maintient simultanément les représentations primale et duale ainsi que leurs contraintes, ce qui augmente la taille du sous-problème. Son ajout ferait également passer de six à neuf le nombre de couples destruction–réparation à sélectionner et à évaluer. Pour conserver un mécanisme adaptatif plus simple, nous retenons donc M2 et Dual comme réparateurs séparés. Ils offrent deux représentations complémentaires et sont directement adaptables aux sous-problèmes produits par la destruction.

4.3 Comparaison des mécanismes Roulette et Deleter

Avant de comparer l'ALNS aux méthodes de l'état de l'art, nous évaluons les deux mécanismes de sélection adaptative étudiés : Roulette et Deleter. Roulette conserve tous les couples destruction–réparation et module leur probabilité de sélection selon leurs performances passées. Deleter ajoute à cette adaptation un coefficient qui pénalise plus fortement les couples peu performants. Toutes les autres composantes de l'algorithme sont identiques, notamment les opérateurs de destruction et de réparation, les paramètres, les graines et la limite de temps. Cette comparaison permet ainsi d'isoler l'effet du mécanisme de sélection. Le Tableau 2 présente les résultats obtenus sur l'ensemble des 55 instances. Pour chaque mécanisme, *Meill.* désigne le meilleur score obtenu, *Moy.* le score moyen et *Temps* le temps d'obtention du meilleur score, exprimé en secondes. Les valeurs en gras indiquent le meilleur score ou la meilleure moyenne entre les deux mécanismes ; en cas d'égalité, les deux valeurs sont mises en gras.

TABLE 2 – Comparaison de l'ALNS avec les mécanismes Roulette et Deleter sur les 55 instances de test.

Instance	BKS	ALNS–Roulette			ALNS–Deleter		
		Meill.	Moy.	Temps	Meill.	Moy.	Temps
C2000.5	2144	**2411**	**2449.3**	2231	2451	2451.8	957
C2000.9	5477	**5906**	**5989.9**	3361	6413	6425	120
DSJC1000.1	300	**317**	327.5	2965	**317**	**323.3**	2619.5
DSJC1000.5	1185	1344	1365.9	3598	**1337**	**1352.6**	2488
DSJC1000.9	2836	**2966**	**2999.8**	2521	2988	3061.8	909
DSJC125.1g	23*	**23**	**23**	44.9	**23**	**23**	34.9
DSJC125.1gb	90*	**90**	**90.2**	1126.9	**90**	113.8	81
DSJC125.5g	71	**71**	**72**	845.5	84	84	0
DSJC125.5gb	240	**240**	**241.5**	1694	286	286	0
DSJC125.9g	169*	**169**	**169**	2.4	**169**	172.4	1.9
DSJC125.9gb	604*	**604**	**604**	3.5	**604**	**604**	4
DSJC250.1	127	**129**	131.8	1113	**129**	**130.6**	1662
DSJC250.5	392	413	457.7	286	**401**	**409.4**	1227
DSJC250.9	934*	1078	1078	0	**935**	**939**	1395
DSJC500.1	184	224	224	0	**193**	**194.9**	2836
DSJC500.5	685	750	775.6	2196.5	**744**	**759**	1378
DSJC500.9	1662	**1691**	**1712.3**	1244	1698	1719.3	3346
DSJR500.1	169*	**169**	**169**	35.8	**169**	**169**	45
flat1000_50_0	924	1294	1326.6	2621	**1274**	**1308.4**	3474
flat1000_60_0	1162	1342	1362.8	1754	**1336**	**1351.2**	3523
flat1000_76_0	1165	**1295**	1344.2	3175	1315	**1339.4**	3483
latin_square_10	1480	1568	1596.5	1463	**1546**	**1580.8**	2711
le450_15a	212	**219**	**221.6**	2084.5	221	222.6	1857
le450_15b	216	**220**	**223.8**	691	222	224.2	1682
le450_15c	275	291	**295.6**	3187	**287**	295.7	3355
le450_15d	272	**285**	291.2	1250	287	**291.1**	2753.5
le450_25a	306	**306**	**306.1**	1121.8	**306**	307.1	1523.3
le450_25b	307*	**307**	**307.1**	620.2	**307**	307.8	1435.7
le450_25c	342	**355**	**363.8**	1356	361	366.2	3147
le450_25d	330	**347**	**352.7**	1534	**347**	355.4	3122
queen10_10	162	**162**	162.6	1117.5	**162**	**162.5**	823.1
queen10_10g	43*	**43**	43.1	1019.8	**43**	**43**	787.3
queen10_10gb	164	**165**	165.8	1463	**165**	**165.6**	1826.7
queen11_11	172	**173**	**175.4**	1338	174	175.9	3554
queen11_11g	47	**47**	48.2	278	48	**48.1**	1268.3
queen11_11gb	176	177	178.3	233.7	**176**	**177.7**	218
queen12_12	185	**187**	190.5	1540	188	**189.8**	2683.7

Suite à la page suivante

Table 2 – suite

Instance	BKS	ALNS–Roulette			ALNS–Deleter		
		Meill.	Moy.	Temps	Meill.	Moy.	Temps
queen12_12g	50	**51**	51.8	1141	**51**	**51.5**	1925.4
queen12_12gb	191	192	194.2	442.5	**191**	**193.9**	1747
queen13_13	194	**196**	201.2	674	198	**201**	1591
queen14_14	215	**218**	**222.2**	1055	220	223.1	1993
queen15_15	223	**231**	**234.9**	2336	**231**	235.1	3205
queen16_16	234	**244**	**248.6**	989	245	249.1	1874
queen8_8g	36*	**36**	**36**	135.7	**36**	**36**	32
queen8_8gb	132*	**132**	**132**	107.2	**132**	138.2	79.1
queen9_9g	41*	**41**	**41**	236.8	46	46	0
queen9_9gb	159	**159**	**159.8**	813.5	**159**	166.4	1706.2
wap01a	545	**589**	**601.5**	3499	605	607	2257
wap02a	538	**581**	590.8	3207	582	**583.5**	2841
wap03a	562	**657**	**657**	0	**657**	**657**	0
wap04a	563	**662**	**662**	0.4	**662**	**662**	0
wap05a	541	**551**	**557.6**	2691	554	561.9	3073
wap06a	516	**543**	553.4	3161	546	**552.8**	3573
wap07a	555	623	638.8	3307	**621**	**628.5**	3470
wap08a	529	**603**	611.1	3245	605	**606**	3467
BKS atteints			16/55			14/55	
Meilleurs scores entre les deux mécanismes			43/55			32/55	
Meilleurs scores moyens			30/55			31/55	
Exécutions atteignant le même score			360/1087			345/990	

Roulette obtient globalement les meilleurs résultats. Elle atteint 16 BKS, contre 14 pour Deleter, et fournit le meilleur score des deux mécanismes sur 43 instances, contre 32 pour Deleter. Deleter présente un léger avantage sur les scores moyens, avec la meilleure moyenne sur 31 instances contre 30 pour Roulette. Cet écart reste toutefois faible au regard de l'avantage de Roulette sur les meilleurs scores et les BKS atteints. Les résultats de l'ALNS avec Roulette sont donc retenus pour la comparaison avec les méthodes de l'état de l'art.

4.4 Comparaison avec les méthodes de l'état de l'art

Le Tableau 3 regroupe les résultats disponibles pour les 55 instances étudiées. ILSTS combine une recherche tabou itérée et des perturbations adaptatives [Nogueira *et al.*, 2021]. RedLS est une recherche locale avec réduction et mémoire taboue [Wang *et al.*, 2020]. Ces deux méthodes de la littérature sont comparées à l'ALNS hybride développée dans ce travail, utilisée ici avec le mécanisme Roulette retenu dans la sous-section précédente. Les résultats de Joint, issus de la campagne exacte sur 188 instances, sont ajoutés sur le même sous-ensemble afin de fournir un point de comparaison commun. Joint comporte uniquement le meilleur score et le temps de résolution. Les conventions utilisées dans ce tableau sont définies dans les conditions expérimentales.

TABLE 3 – Résultats obtenus sur les 55 instances de test avec l'ALNS–Roulette, ILSTS et RedLS, ainsi que les résultats de Joint sur le même sous-ensemble.

Instance	BKS	ILSTS			RedLS			ALNS (Roulette)			Joint	
		Meill.	Moy.	Temps	Meill.	Moy.	Temps	Meill.	Moy.	Temps	Meill.	Temps
C2000.5	2144	2247	2288.8	3460	**2180**	**2205.3**	1648	2411	2449.3	2231	–	TL
C2000.9	5477	6028	6072.8	3440	**5605**	**5658.8**	3463	5906	5989.9	3361	–	TL
DSJC1000.1	300	306	**308.5**	1869.5	**305**	**308.5**	2903	317	327.5	2965	–	TL
DSJC1000.5	1185	1240	1265.5	1101	**1202**	**1220.8**	1145	1344	1365.9	3598	–	TL
DSJC1000.9	2836	3046	3079.2	3563	**2855**	**2870.8**	3476	2966	2999.8	2521	–	TL
DSJC125.1g	23*	**23**	**23**	8.1	**23**	23.6	9.3	**23**	**23**	44.9	**23**	3600.5
DSJC125.1gb	90*	**90**	**90**	48.8	91	92.7	0	**90**	90.2	1126.9	91	3600.5
DSJC125.5g	71	**71**	**71**	228.2	72	72.3	327.2	**71**	72	845.5	76	3600.5
DSJC125.5gb	240	**240**	**240**	241.1	244	252.5	0	**240**	241.5	1694	263	3600.5
DSJC125.9g	169*	**169**	**169**	749.4	**169**	**169**	1.7	**169**	**169**	2.4	**169**	1.76
DSJC125.9gb	604*	**604**	**604**	391.3	**604**	606.2	4.7	**604**	**604**	3.5	**604**	1.42
DSJC250.1	127	**127**	**128.2**	120	129	134.9	6	129	131.8	1113	152	3603.9
DSJC250.5	392	**395**	**399.8**	1070	399	404.6	287	413	457.7	286	471	3604.04
DSJC250.9	934*	936	946.6	2160	935	**938.8**	1761	1078	1078	0	**934**	182.16
DSJC500.1	184	**188**	**189.9**	1964	189	196.9	1093.6	224	224	0	–	TL
DSJC500.5	685	727	738.7	3572	**708**	**715.6**	1298	750	775.6	2196.5	–	TL
DSJC500.9	1662	1733	1750.7	2900	**1675**	**1681.5**	2538	1691	1712.3	1244	–	TL
DSJR500.1	169*	**169**	**169**	0.5	**169**	171.8	2	**169**	**169**	35.8	**169**	1195.37
flat1000_50_0	924	1218	1233.4	1194	**1165**	**1180.6**	2510	1294	1326.6	2621	–	TL
flat1000_60_0	1162	1240	1265.4	2105	**1202**	**1217.7**	2770	1342	1362.8	1754	–	TL
flat1000_76_0	1165	1229	1245.4	653	**1186**	**1199.6**	3251	1295	1344.2	3175	–	TL
latin_square_10	1480	1564	1587.2	1777	**1517**	**1529.2**	1863.5	1568	1596.5	1463	–	TL
le450_15a	212	**213**	**215.8**	1241	214	217.8	21	219	221.6	2084.5	–	TL
le450_15b	216	**218**	**219.2**	2306	**218**	222.3	68	220	223.8	691	–	TL
le450_15c	275	285	287.5	1657	**282**	**286.7**	57	291	295.6	3187	–	TL
le450_15d	272	**278**	**283.4**	1791	280	283.8	223.8	285	291.2	1250	–	TL
le450_25a	306	**306**	**306**	753.8	**306**	307.4	602.5	**306**	306.1	1121.8	315	3611.82
le450_25b	307*	**307**	**307**	73.2	**307**	309.4	62.7	**307**	307.1	620.2	**307**	3611.17
le450_25c	342	**350**	**354.2**	1075	**350**	354.4	99	355	363.8	1356	–	TL
le450_25d	330	342	345.2	2008	**336**	**340.9**	76.5	347	352.7	1534	–	TL
queen10_10	162	**162**	**162**	31.3	**162**	163.4	1714	**162**	162.6	1117.5	165	3600.34
queen10_10g	43*	**43**	**43**	13.5	**43**	43.6	32.3	**43**	43.1	1019.8	44	3600.33
queen10_10gb	164	**164**	**164**	560.3	165	166.2	293	165	165.8	1463	167	3600.34
queen11_11	172	**172**	**172.8**	1422	174	177.7	57	173	175.4	1338	178	3600.57
queen11_11g	47	**47**	**47**	346.8	**47**	48.5	65	**47**	48.2	278	–	TL
queen11_11gb	176	**176**	**176.1**	797.4	177	178.9	105	177	178.3	233.7	–	TL
queen12_12	185	**185**	**186.5**	2975	188	190.4	4.3	187	190.5	1540	195	3601.05
queen12_12g	50	**50**	**50.4**	1187.1	51	51.1	156.3	51	51.8	1141	53	3601.04
queen12_12gb	191	**191**	**191.8**	1445	193	200	10	192	194.2	442.5	195	3601.08
queen13_13	194	**194**	**196.2**	2896	**194**	203.9	397	196	201.2	674	209	3601.68
queen14_14	215	**216**	**218.2**	2721	219	225.6	1	218	222.2	1055	–	TL
queen15_15	223	228	**229.2**	1848.2	**227**	231.2	163	231	234.9	2336	–	TL
queen16_16	234	**238**	**240.9**	2011	239	242.3	3	244	248.6	989	271	3604.22
queen8_8g	36*	**36**	**36**	1.4	**36**	36.5	78.9	**36**	**36**	135.7	**36**	88.82
queen8_8gb	132*	**132**	**132**	0.7	133	136.8	0	**132**	**132**	107.2	**132**	51.55
queen9_9g	41*	**41**	**41**	1.3	**41**	43	141.3	**41**	**41**	236.8	**41**	3600.1
queen9_9gb	159	**159**	**159**	190.3	**159**	161.2	4	**159**	159.8	813.5	**159**	3600.12
wap01a	545	**553**	**558.2**	3129	573	596.9	25	589	601.5	3499	–	TL
wap02a	538	**545**	**549.2**	2870.5	551	584.8	3391	581	590.8	3207	–	TL
wap03a	562	617	629.2	3582	**568**	**622.5**	2813	657	657	0	–	TL
wap04a	563	616	**624.1**	3389	**566**	631.8	1832	662	662	0.4	–	TL
wap05a	541	**543**	546.5	3085	544	**546**	814.4	551	557.6	2691	–	TL
wap06a	516	523	**527.2**	2205	**518**	**527.2**	2507	543	553.4	3161	–	TL
wap07a	555	572	578	3368	**557**	**571.7**	1845.5	623	638.8	3307	–	TL
wap08a	529	544	555.1	3197	**538**	**546.8**	1025	603	611.1	3245	–	TL

Suite à la page suivante

Table 3 – suite

Instance	BKS	ILSTS			RedLS			ALNS (Roulette)			Joint	
		Meill.	Moy.	Temps	Meill.	Moy.	Temps	Meill.	Moy.	Temps	Meill.	Temps
BKS atteints			24/55			13/55			16/55			10/55
Meilleurs scores parmi les quatre méthodes			35/55			34/55			16/55			10/55
Meilleurs scores moyens			38/55			20/55			7/55			–
Optimalités prouvées			0/55			0/55			0/55			6/55
Exécutions atteignant le même score			426/1100			200/1100			360/1087			–

4.5 Analyse des performances

Pour chaque paire de méthodes parmi ILSTS, RedLS et l'ALNS avec Roulette, et pour chaque instance, les scores obtenus avec les mêmes graines sont appariés. Lorsqu'une exécution de l'ALNS est manquante, seuls les couples de résultats disponibles pour une même graine sont conservés. Le test bilatéral non paramétrique des rangs signés de Wilcoxon vérifie l'hypothèse nulle selon laquelle la médiane des différences entre les deux séries est nulle. La valeur p mesure alors, sous cette hypothèse, la probabilité d'observer une statistique au moins aussi extrême que celle obtenue. Le seuil $p \leq 0{,}001$ fixe donc, pour chaque comparaison, un niveau de significativité de 0,1 %. Il ne représente pas la probabilité que l'hypothèse nulle soit vraie.

Une différence est retenue comme significative lorsque ce seuil est respecté. La direction est ensuite donnée par le score moyen : la méthode dont la moyenne est la plus faible est déclarée meilleure. Dans les autres cas, aucune différence significative n'est conclue. Ce test n'est pas appliqué aux méthodes exactes, car une seule exécution est disponible pour chaque couple modèle–instance. Les trois comparaisons par paires portent sur les 55 instances ; pour celles impliquant l'ALNS avec Roulette, la taille de l'échantillon apparié peut toutefois être inférieure à 20 lorsque des exécutions sont manquantes.

	ILSTS	RedLS	ALNS
ILSTS	–	$\mathbf{25}/19$	$\mathbf{38}/14$
RedLS	$11/19$	–	$\mathbf{25}/20$
ALNS	$3/14$	$10/20$	–

TABLE 4 – Comparaisons statistiques par paires sous la forme S/N : S est le nombre d'instances où la méthode en ligne est significativement meilleure et N le nombre d'instances sans différence significative. Le nombre de victoires le plus élevé pour chaque paire est indiqué en gras.

Le temps n'est comparé que lorsque deux méthodes atteignent le même score et que l'écart est d'au moins une seconde. Ces cas restent peu nombreux : RedLS est plus rapide qu'ILSTS sur une instance, contre deux dans le sens inverse ; l'ALNS avec Roulette est plus rapide qu'ILSTS sur deux instances, contre cinq dans le sens inverse. Aucun avantage temporel n'est relevé entre RedLS et cette version de l'ALNS dans cette situation. Le temps ne modifie donc pas les conclusions fondées sur les scores.

Globalement, l'ALNS avec Roulette est moins efficace que les deux méthodes de la littérature, RedLS [Wang *et al.*, 2020] et ILSTS [Nogueira *et al.*, 2021]. Le Tableau 4 confirme d'abord l'avantage d'ILSTS : cette méthode est meilleure que RedLS sur 25 instances, contre 11 dans le sens inverse. Son avantage est encore plus net face à l'ALNS avec Roulette, avec 38

comparaisons favorables contre 3. Ce résultat est cohérent avec le Tableau 3, dans lequel ILSTS atteint 24 BKS et obtient le meilleur score moyen sur 38 instances.

La comparaison avec RedLS reste également défavorable à l'ALNS avec Roulette. RedLS est significativement meilleure sur 25 instances, tandis que l'ALNS avec Roulette l'est sur 10 ; aucune différence significative n'est observée sur les 20 autres instances. Bien que l'ALNS avec Roulette atteigne 16 BKS, contre 13 pour RedLS, elle n'obtient le meilleur score moyen que sur 7 instances, contre 20 pour RedLS. Le seul nombre de BKS atteints ne suffit donc pas à conclure à sa supériorité : la distribution des résultats met en évidence une meilleure robustesse de RedLS.

L'ALNS avec Roulette fournit désormais un résultat pour chacune des 55 instances. La campagne reste néanmoins légèrement incomplète : 1 087 exécutions sont exploitables sur les 1 100 prévues, soit 13 exécutions manquantes. Les résultats ne montrent pas de supériorité globale de l'ALNS proposée avec Roulette. Ils permettent toutefois d'identifier ses principaux axes d'amélioration : mieux cibler les sommets détruits, augmenter l'efficacité de la réparation et réduire le nombre d'exécutions qui ne produisent pas de résultat exploitable.

5 Conclusion et perspectives

Conclusion scientifique. Dans ce travail, nous avons étudié le problème de coloration de graphe pondéré par des approches exactes et approchées. Nous avons conçu une ALNS hybride combinant trois opérateurs de destruction, deux réparations fondées sur M2 et Dual, puis une intensification par RedLS [Wang *et al.*, 2020]. Les sous-problèmes de réparation ont été formulés en programmation linéaire en nombres entiers, implémentés en C++ avec Gurobi et intégrés à des mécanismes adaptant le couple d'opérateurs et le taux de destruction. Les expériences exactes confirment la complémentarité des formulations : Joint atteint le plus grand nombre de meilleurs scores connus, tandis que M2 renforcé par les bornes obtient le plus de preuves d'optimalité. Parmi les méthodes approchées, ILSTS [Nogueira *et al.*, 2021] est globalement la plus performante et RedLS reste plus robuste que l'ALNS proposée. Cette dernière atteint plusieurs meilleurs scores connus, mais les tests statistiques ne montrent pas de supériorité globale. L'hybridation avec une réparation exacte est donc réalisable, mais une résolution répétée limitée à cinq secondes ne suffit pas à la rendre compétitive sur l'ensemble des instances difficiles. Les améliorations prioritaires concernent ainsi le ciblage de la destruction et le coût de la réparation.

Le ciblage de la destruction pourrait s'appuyer sur une analyse des solutions produites par les méthodes de la littérature. Il serait notamment intéressant d'observer, lors de chaque amélioration, les sommets dont la couleur change et leurs caractéristiques : poids, rôle de sommet dominant, contribution au coût d'une couleur, degré, conflits ou fréquence de déplacement. Ces informations pourraient conduire à la définition d'un score d'impact, éventuellement appris, afin de sélectionner plus souvent les sommets les plus influents dans la solution courante.

L'efficacité des opérateurs de réparation pourrait également être améliorée en optimisant le passage des données entre la solution courante et les modèles exacts. Cela comprendrait le précalcul des compatibilités, la réutilisation des structures communes aux sous-problèmes successifs et la mise à jour des seules variables et contraintes affectées par la destruction. Le recours à une stratégie de réparation moins coûteuse constitue aussi une piste intéressante. Il permettrait de réduire le temps consacré à chaque réparation et d'effectuer davantage d'itérations de l'ALNS dans une même limite de temps.

À plus long terme, une approche d'optimisation multi-niveaux pourrait également être envisagée pour traiter le WVCP de manière plus générale. Elle consisterait à réduire progressivement le graphe en regroupant des sommets compatibles, à résoudre le problème sur cette représentation agrégée, puis à reconstruire et à raffiner la coloration sur le graphe initial. À notre connaissance, cette famille de méthodes n'a pas encore été étudiée spécifiquement pour le WVCP et pourrait constituer une nouvelle direction de recherche.

Conclusion du stage. Au-delà des résultats scientifiques, ce stage m'a apporté une expérience complète du développement d'une méthode de recherche. J'ai appris à partir d'une question ouverte, à formuler des hypothèses, à les traduire en modèles et algorithmes, puis à analyser objectivement des résultats qui ne confirment pas toujours les attentes initiales. Le développement en C++, l'utilisation de Gurobi, la préparation des campagnes sur Grid'5000 et l'exploitation des résultats ont renforcé mon autonomie technique et ma capacité à conduire des expériences reproductibles. Le travail au Loria, les échanges avec mes encadrants et les discussions avec les chercheurs m'ont aussi permis de découvrir le fonctionnement quotidien d'un laboratoire et les exigences de la recherche opérationnelle. Après ce stage, je souhaite m'orienter vers des activités de recherche et développement. Cette expérience renforce mon intérêt pour la préparation ultérieure d'une thèse à l'interface entre intelligence artificielle et optimisation combinatoire.

Références

[Balouek *et al.*, 2013] Daniel Balouek, Alexandra Carpen Amarie, Ghislain Charrier, Frédéric Desprez, Emmanuel Jeannot, Emmanuel Jeanvoine, Adrien Lèbre, David Margery, Nicolas Niclausse, Lucas Nussbaum, Olivier Richard, Christian Pérez, Flavien Quesnel, Cyril Rohr, and Luc Sarzyniec. Adding virtualization capabilities to the Grid'5000 testbed. In Ivan I. Ivanov, Marten van Sinderen, Frank Leymann, and Tony Shan, editors, *Cloud Computing and Services Science*, volume 367 of *Communications in Computer and Information Science*, pages 3–20. Springer International Publishing, 2013.

[Brélaz, 1979] Daniel Brélaz. New methods to color the vertices of a graph. *Communications of the ACM*, 22(4) :251–256, 1979.

[Cornaz *et al.*, 2017] Denis Cornaz, Fabio Furini, and Enrico Malaguti. Solving vertex coloring problems as maximum weight stable set problems. *Discrete Applied Mathematics*, 217 :151–162, 2017.

[Furini and Malaguti, 2012] Fabio Furini and Enrico Malaguti. Exact weighted vertex coloring via branch-and-price. *Discrete Optimization*, 9(2) :130–136, 2012.

[Goudet *et al.*, 2022] Olivier Goudet, Cyril Grelier, and Jin-Kao Hao. A deep learning guided memetic framework for graph coloring problems. *Knowledge-Based Systems*, 258 :109986, 2022.

[Goudet *et al.*, 2023] Olivier Goudet, Cyril Grelier, and David Lesaint. New bounds and constraint programming models for the weighted vertex coloring problem. In *IJCAI*, pages 1927–1934, 2023.

[Grelier *et al.*, 2022] Cyril Grelier, Olivier Goudet, and Jin-Kao Hao. On monte carlo tree search for weighted vertex coloring. In *European Conference on Evolutionary Computation in Combinatorial Optimization (Part of EvoStar)*, pages 1–16. Springer, 2022.

[Grelier *et al.*, 2024] Cyril Grelier, Olivier Goudet, and Jin-Kao Hao. A memetic algorithm with adaptive operator selection for graph coloring. In *European conference on evolutionary computation in combinatorial optimization (part of EvoStar)*, pages 65–80. Springer, 2024.

[Grelier, 2023] Cyril Grelier. *Métaheuristiques Guidées par l'Apprentissage pour la Coloration de Graphe*. Theses, Université d'Angers, December 2023.

[Leighton, 1979] F. T. Leighton. A graph coloring algorithm for large scheduling problems. *Journal of Research of the National Bureau of Standards*, 84(6) :489–503, 1979.

[Malaguti *et al.*, 2009] Enrico Malaguti, Michele Monaci, and Paolo Toth. Models and heuristic algorithms for a weighted vertex coloring problem. *Journal of Heuristics*, 15(5) :503–526, 2009.

[Nogueira *et al.*, 2021] Bruno Nogueira, Eduardo Tavares, and Paulo Maciel. Iterated local search with tabu search for the weighted vertex coloring problem. *Computers & Operations Research*, 125 :105087, 2021.

[Prais and Ribeiro, 2000] Marcelo Prais and Celso C Ribeiro. Reactive grasp : An application to a matrix decomposition problem in tdma traffic assignment. *INFORMS Journal on Computing*, 12(3) :164–176, 2000.

[Ribeiro *et al.*, 1989] Celso Carneiro Ribeiro, Michel Minoux, and Manoel Camillo Penna. An optimal column-generation-with-ranking algorithm for very large scale set partitioning problems in traffic assignment. *European Journal of Operational Research*, 41(2) :232–239, 1989.

[Ropke and Pisinger, 2006] Stefan Ropke and David Pisinger. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. *Transportation Science*, 40(4) :455–472, 2006.

[Shaw, 1998] Paul Shaw. Using constraint programming and local search methods to solve vehicle routing problems. In *Principles and Practice of Constraint Programming – CP98*, volume 1520 of *Lecture Notes in Computer Science*, pages 417–431. Springer, 1998.

[Sun *et al.*, 2018] Wen Sun, Jin-Kao Hao, Xiangjing Lai, and Qinghua Wu. Adaptive feasible and infeasible tabu search for weighted vertex coloring. *Information Sciences*, 466 :203–219, 2018.

[Wang *et al.*, 2020] Yiyuan Wang, Shaowei Cai, Shiwei Pan, Ximing Li, and Monghao Yin. Reduction and local search for weighted graph coloring problem. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 2433–2441, 2020.

[Welsh and Powell, 1967] D. J. A. Welsh and M. B. Powell. An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal*, 10(1) :85–86, 1967.