THÈSE

en co-tutelle

Présentée pour l'obtention du titre de Docteur

en **Sciences de Gestion**
des
FACULTÉS UNIVERSITAIRES CATHOLIQUES DE MONS

et

en **Optimisation et Sûreté des Systèmes**
de l'
UNIVERSITÉ DE TECHNOLOGIE DE TROYES

Présentée par

HONGYING FEI

Soutenance le 17 Mars 2006

Vers un outil d'aide à la planification et à l'ordonnancement des blocs opératoires

---

Jury :

Membres belges

M.	**Rudy De Winne**	Facultés Universitaires Catholiques de Mons	**Président**
M.	**El-Houssaine Aghezzaf**	Université de Gand	**Rapporteur**
M.	**Fouad Riane**	Facultés Universitaires Catholiques de Mons	**Invité**
Mme	**Nadine Meskens**	Facultés Universitaires Catholiques de Mons	**Directrice de thèse**

Membres français :

M.	**Michel Gourgand**	Université Blaise Pascal	**Rapporteur**
M.	**Christian Tahon**	Université de Valenciennes et Hainaut-Cambrésis	**Invité**
Mlle	**Catherine Combes**	Université de Jean Monnet (Saint-Etienne)	**Examinateur**
M.	**Chengbin Chu**	Université de Technologie de Troyes	**Directeur de thèse**

Vers un outil d'aide à la planification et à l'ordonnancement des blocs opératoires

RÉSUMÉ
Dans cette thèse, nous étudions la gestion des blocs opératoires, et plus particulièrement la planification et l'ordonnancement de ces blocs. Le choix d'études de ce secteur hospitalier est lié au fait qu'il est réputé comme un lieu hautement stratégique dans un établissement hospitalier, surtout en terme de coûts. Il est dès lors utile de s'intéresser à l'optimisation de l'utilisation des ressources hospitalières. Etant donné que l'optimisation du fonctionnement des blocs opératoires est un problème vaste et complexe, nous nous focalisons sur deux sous-problèmes déjà réputés difficiles : la planification et l'ordonnancement des interventions chirurgicales. Ce sont des problèmes centrés sur la programmation opératoire et dont l'objectif est d'obtenir un programme opératoire réalisable et efficace du bloc.

Les problèmes de planification sont premièrement formalisés comme des modèles mathématiques en nombres entiers et puis résolus par des procédures heuristiques et par un « Branch-and-Price » basés sur la génération de colonnes qui est utilisée pour résoudre les relaxations linéaires des modèles concernés afin de trouver les bornes inférieures. Les modèles d'ordonnancement sont traités comme des variantes de modèles de « *flow shop* » hybride à deux étages et résolus par des algorithmes génétiques hybrides. Finalement, nos modèles ont été testés et validés sur un cas réel.

Mots-clés : Gestion des blocs opératoires, programmation opératoire, génération de colonnes, Branch-and-Price, algorithme génétique hybride

Towards a decision-aiding tool for the operating theatre planning and scheduling

ABSTRACT
This thesis presents our studies on the operating theatre management, especially on the operating theatre planning and scheduling problem in case that this surgical sector is always regarded as the kernel in a hospital in terms of the expenditure. Therefore, it's necessary to optimize the assignment of hospital resources in the operating theatre. Since this type of problem is extremely complex, we concentrate on just two sub-problems that are also considered difficult ones by researchers: the surgical cases planning and scheduling problem, which aim to make a surgical cases programming with an objective of obtaining a realizable and efficient surgical cases operating schedule. At first the weekly planning problems of surgical cases are formulated as a mathematical integer programming, and then they are solved by heuristic procedures and Branch-and-Price one. All these procedures are based on the column generation procedure, which is used to solve the linear relaxation of each model for a lower bound. The scheduling models are treated as variants of hybrid two-stage flow shop problem and are solved by proposed hybrid genetic algorithms. Finally, our models are tested and validated in a real case.

Keywords: Operating theatre management, surgical cases programming, column generation, Branch-and-Price, hybrid genetic algorithm

Remerciements

Le travail présenté dans cette thèse a été réalisé dans le cadre d'un projet « programme Pôles d'attraction interuniversitaires - Etat belge –Services fédéraux des affaires scientifiques techniques et culturelles ».

Les années consacrées à la réalisation de ce travail de recherche ont été l'occasion de rencontres, de travaux et de réflexions en collaboration avec de nombreuses personnes que je désire remercier aujourd'hui.

En premier lieu, je voudrais exprimer ma profonde gratitude à ma directrice et mon directeur de thèse, Madame Nadine Meskens et Monsieur Chengbin Chu, pour leur disponibilité, et pour toute la confiance qu'ils m'ont accordée au cours de ces années de thèse. Surtout, je voudrais remercier plus particulièrement Madame Nadine Meskens, pour sa patience en corrigeant cette thèse car la relecture d'une thèse écrite par une non francophone, comme moi, est vraiment très difficile et lui a nécessité de nombreuses heures de travail. Sans elle, je n'aurais jamais pu y arriver.

J'exprime également toute ma reconnaissance à Monsieur Abdelhakim Artiba, qui était mon premier directeur de thèse de la FUCaM. Bien qu'il ne dirige plus ma thèse depuis deux ans, je le remercie pour son aide et sa disponibilité pendant les deux premières années de thèse.

Mes remerciements vont aussi aux membres du jury :

À Monsieur Rudy De Winne, qui m'a fait l'honneur de présider ce jury ;

À Messieurs El-Houssaine Aghezzaf et Michel Gourgand pour avoir accepté de lire et de rapporter cette thèse malgré leur importante charge de travail ;

À Mademoiselle Catherine Combes, qui m'a beaucoup aidé dans mon travail et qui a accepté d'être examinateur de cette thèse ;

À Messieurs Fouad Riane et Christian Tahon pour avoir accepté de faire partie de ce jury.

Je remercie également mes amis David Duvivier, Olivier Roux, Jean-Philippe Vandamme, Guillaume Fleurquin, Benoît Roland, Valérie Dhaevers … pour leurs conseils avisés et leur aide lors de la rédaction de ma thèse. Je tiens également à témoigner ma reconnaissance à tous les membres, actuels et anciens, du CREGI, du département GPO et de l'OSI que j'ai côtoyé durant ma thèse. Je les remercie pour leurs accueils, leurs encouragements et leurs aides et pour m'avoir permis de travailler dans une ambiance chaleureuse et conviviale tant en France qu'en Belgique.

Je remercie mes proches pour leur soutien. Particulièrement, je remercie mon mari, Ye Wang, qui m'a soutenu et encouragé tout au long de la réalisation de cette thèse, même dans les moments les plus difficiles.

Enfin, je tiens à rassurer ceux qui, par manque d'espace, ne sont pas nommés dans ces remerciements. Je ne vous oublie pas, vous avez d'ores et déjà votre place dans ma mémoire.

Table des Matières

Liste des Tableaux

Liste des Figures

Introduction Générale

Avec l'accroissement des préoccupations de la population européenne, de plus en plus vieillissante, relatives à son état de santé, les besoins en santé augmentent régulièrement et entraînent donc une hausse de l'activité hospitalière. Face à l'exigence accrue des patients sur le niveau de service, à des contraintes législatives de plus en plus strictes sur le temps de travail du personnel hospitalier et aux soucis d'économie des pouvoirs publics, l'utilisation optimale des ressources hospitalières est plus que jamais nécessaire. Dans ce contexte, les recherches dans le domaine de la gestion hospitalière sont en pleine expansion.

Dans cette thèse, nous étudions la gestion des blocs opératoires, et plus particulièrement la planification et l'ordonnancement de ces blocs. Le choix d'études de ce secteur hospitalier est lié au fait qu'il est réputé comme un lieu hautement stratégique dans un établissement hospitalier, surtout en terme de coûts. Il est dès lors intéressant de s'intéresser à l'optimisation de l'utilisation des ressources hospitalières.

L'optimisation du fonctionnement des blocs opératoires est un problème extrêmement complexe. Cette complexité est accentuée par le caractère aléatoire intrinsèque du problème (à titre d'exemple, il est quasi impossible de connaître de manière certaine la durée d'une intervention chirurgicale). De multiples contraintes telles que l'emploi du temps des chirurgiens, leurs compétences spécifiques, les contraintes législatives sur le temps de travail des équipes d'infirmières, le matériel médical spécialisé, la disponibilité des lits, etc. doivent être prises en compte. Les critères qui servent à comparer les solutions entre elles sont également nombreuses : les temps d'attente des patients, l'utilisation des appareils médicaux et les occupations des lits. Par conséquent, nous allons nous concentrer sur deux sous-problèmes déjà réputés difficiles : la planification et l'ordonnancement des interventions chirurgicales. Ce sont des problèmes centrés sur la programmation opératoire et dont l'objectif est d'obtenir un programme opératoire réalisable et efficace du bloc.

Les problèmes de planification sont premièrement formalisés comme des modèles mathématiques en nombres entiers, puis résolus au moyen d'heuristiques et de méthodes exactes basées sur la génération de colonnes qui est utilisée pour résoudre les relaxations linéaires des modèles concernés afin de trouver les bornes inférieures. Les modèles d'ordonnancement sont traités comme les variantes des modèles de « *flow shop* » hybride à deux étages et résolus par des algorithmes génétiques hybrides.

Cette thèse se compose de six parties :

Dans la première partie, nous décrivons le contexte et la problématique de notre sujet de recherche.

Un état de l'art en programmation opératoire est présenté dans la seconde partie ainsi que la relation entre la gestion des blocs opératoires et la gestion de production. Cela

justifie l'adaptation de méthodes et outils issus de la gestion industrielle aux systèmes hospitaliers.

Ensuite, deux modèles de planification des blocs opératoires, un modèle de programmation par allocation préalable de plages (« *block scheduling* ») et un modèle de programmation ouverte (« *open scheduling* ») ainsi que les méthodes de résolution sont étudiés dans la troisième partie.

Puis, dans la quatrième partie, deux modèles d'ordonnancement sont présentés pour ordonnancer les interventions affectées au préalable par les modèles de planification aux salles d'opération pour une journée et ensuite sont résolus par les algorithmes génétiques hybrides.

Dans la cinquième partie, tous les résultats des expérimentations numériques portant sur des données générées aléatoirement sont présentés.

Une résolution d'un cas réel de planification et ordonnancement d'interventions dans un service d'endoscopie est exposée dans la partie VI. Ce cas est particulier car il n'y a que 2 salles d'opération ce qui nous permet pour l'ordonnancement d'utiliser des méthodes spécifiques pour ce cas. La planification des actes a été effectuée par la procédure heuristique basée sur la génération de colonnes et l'ordonnancement par l'algorithme de Gonzalez-Sahni.

La dernière partie conclut notre travail et présente les perspectives de nos recherches.

Partie I :

Gestion des blocs opératoires : contexte et problématique

1. Introduction

L'environnement des systèmes de soins de santé est en constante évolution. Dans ce contexte, les gestionnaires d'hôpitaux se doivent de rendre leur établissement plus productif. Selon plusieurs études [Gordon *et al.*, 1988][Macario *et al.*, 1995][Clergue, 1999], le bloc opératoire constitue près de 10% du budget de fonctionnement d'un hôpital du fait qu'il implique différentes ressources humaines et matérielles coûteuses et rares telles que les chirurgiens, anesthésistes, infirmières, etc.. Donc, de plus en plus de chercheurs s'intéressent à la gestion des blocs opératoires, tentant d'améliorer le fonctionnement de ce secteur pour maximiser son efficacité ainsi que la satisfaction du patient tout en comptant sur une enveloppe budgétaire figée.

En effet, la gestion du fonctionnement de ce secteur est un problème extrêmement complexe parce qu'il faut prendre en compte de multiples contraintes telles que l'emploi du temps des chirurgiens, leurs compétences spécifiques, les contraintes législatives sur le temps de travail des équipes d'infirmières, la disponibilité des lits, ….

Dans cette première partie, nous allons présenter le contexte de notre étude ainsi que notre problématique de recherche.

2. Contexte

Avant d'expliquer les différents modes d'organisation des blocs opératoires, nous allons d'abord décrire leur environnement à savoir les différents types d'établissements hospitaliers et ce dans les contextes belge et français.

2.1 Etablissements hospitaliers

Les établissements hospitaliers sont des établissements de soins de santé où des examens et/ou des traitements spécifiques de médecine spécialisée, relevant de la médecine, de la chirurgie et éventuellement de l'obstétrique, peuvent être effectués ou appliqués à tout moment dans un contexte pluridisciplinaire.

2.1.1 Etablissements hospitaliers en Belgique

Les établissements hospitaliers sont classés selon des modes d'organisation, de financement et de fonctionnement. En Belgique, concernant le fonctionnement des hôpitaux, nous pouvons classifier les établissements hospitaliers en deux groupes : les hôpitaux généraux et les hôpitaux spéciaux. De plus, il y a plusieurs types d'établissements hospitaliers spéciaux en Belgique[1] :

- Hôpitaux psychiatriques : hôpitaux exclusivement destinés à des patients psychiatriques ;
- Hôpitaux universitaires : les hôpitaux qui, eu égard à leur fonction propre dans le domaine des soins, de l'enseignement et de la recherche scientifique appliquée, répondent aux conditions fixées par le Roi et sont désignés comme tels par lui sur proposition des autorités académiques d'une université belge ;
- Etablissements médico-sociaux : n'étant pas considérés comme hôpitaux, ce sont des établissements psychiatriques fermés destinés au simple hébergement de personnes âgées ou d'enfants ;
- Habitations protégées et homes de séjour provisoire : après avis du Conseil National des établissements hospitaliers, ce sont des établissements pour les patients psychiatriques et d'autres groupes désignés par le Roi par un arrêté délibéré en Conseil des Ministres ;
- Petits hôpitaux : les hôpitaux qui disposent d'un nombre très limité de services et/ou de lits, et où un nombre très limité de médecins hospitaliers sont en fonction.

Selon De Troyer et Krzeslo [2004], en 1980, la Belgique disposait de 521 hôpitaux pour 92436 lits (68254 lits dans 444 hôpitaux généraux et 24182 lits dans 77 hôpitaux psychiatriques) ; en 1998, elle possédait 239 hôpitaux pour 73201 lits (56553 lits dans 169 hôpitaux généraux et 16468 lits dans 70 hôpitaux psychiatriques). Au niveau national, les chiffres d'activité globale des hôpitaux montrent que le séjour moyen par patient est passé de 19,5 jours en 1980 à 11,0 jours en 1998.

Le financement de la plupart des établissements hospitaliers se fait sur la base d'une dotation de l'autorité fédérale. Jusqu'au début des années 1980, l'autorité fédérale belge, comme d'autres pays, comptait un nombre élevé d'hôpitaux de petite taille.

Avant 2002, le financement de l'activité hospitalière provenait de cinq sources (UNMS[2] 2001) :

- le budget « prix de journée », destiné à financer les frais d'exploitation hors hôpital de jour ;
- les forfaits par journée finançant les frais d'exploitation de l'hôpital de jour ;
- les honoraires médicaux fixés par les accords médico-mutualistes ;
- le chiffre d'affaire des spécialités pharmaceutiques ;
- la fourniture de prothèses et d'implants et la participation du patient.

[1] La loi sur les hôpitaux Belges, 6ème édition complètement révisée, par Ceuterick G. et Duvillier, 1998

[2] Données UNMS : Union Nationale des Mutualités Socialistes.

Depuis 2002, une réforme du financement hospitalier a été mise en œuvre. Depuis, le financement de base ne couvre plus qu'une partie des coûts découlant des normes d'encadrement dans les services hospitaliers, le solde devant être assuré par l'activité (justifiée) réelle (UNMS 2001) [De Troyer et Krzeslo, 2004]. Cette nouvelle réforme renforce donc le poids de l'activité dans le financement et réduit celui de la structure de l'offre dans le calcul du budget de l'hôpital.

En fait, le budget des établissements hospitaliers, présenté dans la Figure 1, n'a cessé d'augmenter durant cette dernière décennie en Belgique.

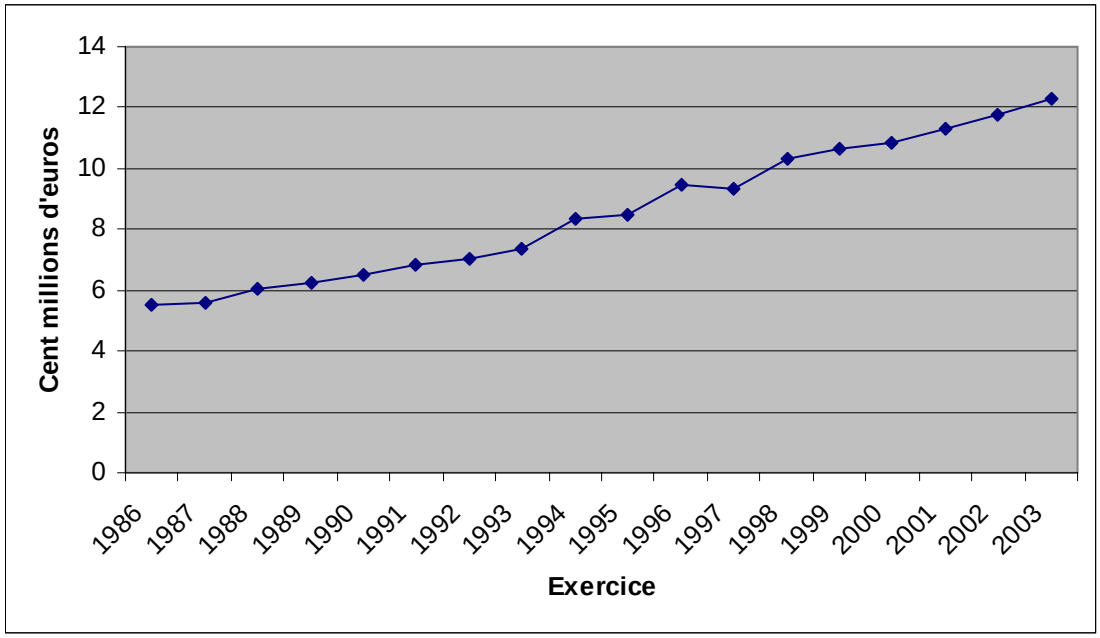

Figure 1 : **Budget des moyens financiers des établissements hospitaliers en Belgique**[3]

2.1.2 Etablissements hospitaliers en France

Le secteur hospitalier français présente un paysage varié. Il fait cohabiter différents établissements combinant des modes d'organisation et de gestion, de financement et de régulation, de participation aux missions de services publics très différents. Les statuts des personnels travaillant dans ces établissements sont également variés. Comme présenté dans [DREES, 1999] et [Dutheil, 2005], il y a principalement trois types de statuts juridiques des établissements hospitaliers : les établissements publics, les établissements privés à but non lucratif et les établissements privés à but lucratif.

A. Etablissements publics, sont des structures locales relevant de l'autorité de l'Etat qu'il exerce à travers les agences régionales de l'hospitalisation (ARH), et qui peuvent être à vocation générale ou spécialisée, de taille variable. Ils sont soumis à des règles de droit public pour leur fonctionnement tandis que leur gestion budgétaire et financière s'inscrit dans le cadre général de la comptabilité publique ;

[3] Crédit budgétaire :
http://www.health.fgov.be/vesalius/devnew/FR/prof/finance/Ziekenhuizen/budget.pdf

B. Etablissements privés à but non lucratif, sont en général des associations relevant de la loi 1901, des réalisations sanitaires et sociales mutualistes ou encore des fondations. La plupart sont reconnus d'utilité publique. La plupart de ceux qui ont signé une convention de participation au service public hospitalier sont, de ce fait, soumis aux obligations du service public ;

C. Etablissements privés à but lucratif, sont des sociétés commerciales. Leur financement est assuré par une dotation budgétaire allouée par l'agence régionale de l'hospitalisation dont ils dépendent pour les forfaits d'hospitalisation. Leur fonctionnement est assuré par le biais de contrats d'exercice liant l'établissement aux médecins exerçant à titre libéral.

Plus précisément, selon le bilan de l'année 2004, il existe en France 3836 établissements hospitaliers dont la répartition est décrite dans le Tableau 1[Dutheil, 2005].

Etablissements publics	**Etablissements privés à but non lucratif**	**Etablissements privés à but lucratif**
1035 établissements 314576 lits	1425 établissements (dont 582 participent au service public hospitalier) 68698 lits	1376 établissements 94924 lits

Tableau 1 : **Répartition des établissements hospitaliers**

Quel que soit leur statut, les établissements hospitaliers sont sous tutelle des agences régionales d'hospitalisation, qui exercent un contrôle sur leur fonctionnement, l'application des règles de sécurité sanitaire et la répartition territoriale. L'ensemble des établissements de santé accueillent plus de 13 millions de patients chaque année (dont près de 8 millions dans le seul secteur public), ce qui représente 143 millions de journées d'hospitalisation au total (dont environ 95 millions pour le seul secteur public) [Dutheil, 2005].

Qu'ils soient publics ou privés, l'ensemble des établissements hospitaliers sont presque tous financés par la sécurité sociale ce qui a pour conséquence d'assurer le remboursement des soins au patient même si celui-ci choisit le secteur privé à but lucratif dans le cadre d'une prestation de santé.

Sur le plan budgétaire, une enveloppe nationale, financée par la branche assurance maladie de la sécurité sociale, est votée chaque année par le Parlement en France. Celle-ci est répartie en dotations régionales allouées aux agences régionales de l'hospitalisation qui fixent la dotation budgétaire des établissements hospitaliers placés sous leur autorité. Les dotations régionales tiennent compte des enjeux nationaux et régionaux de santé publique et des besoins de la population.

Selon la source DREES[4], les dépenses totales de santé en France sont de 158 milliards d'euros. Les dépenses hospitalières s'élèvent à 61 milliards d'euros.

[4] Source DREES, Direction de la Recherche, des Etudes, de l'Evaluation et des Statistiques

2.1.3 Part des dépenses de santé dans le PIB[5]

En général, les établissements de santé représentent toujours une part importante des dépenses de santé dans le PIB, montrée par la comparaison internationale ci-dessous :

Part des dépenses de santé dans le PIB - 2001 - Comparaison internationale (source OCDE[6])

Etats-Unis	13,9%
Suisse	10,9%
Allemagne	10,7%
Canada	9,7%
France	9,5%
Grèce	9,4%
Portugal	9,2%
Belgique	9%
Pays-Bas	8,9%
Suède	8,7%
Danemark	8,6%
Italie	8,4%
Norvège	8,3%
Autriche	7,7%
Royaume-Uni	7,6%
Espagne	7,5%

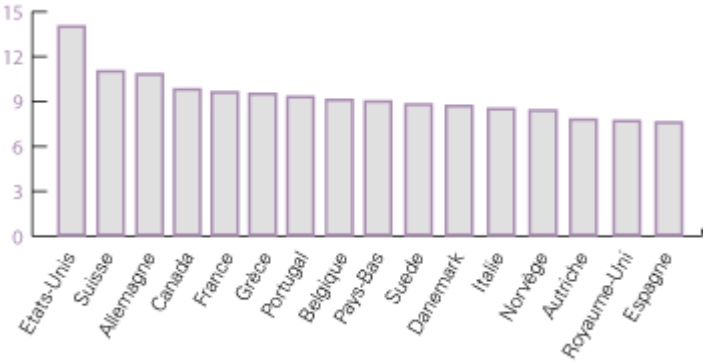

Tableau 2 : **Comparaison internationale de la part des dépenses de santé dans le PIB**

Dans Tableau 2, nous pouvons trouver que c'est presque la même situation pour la France que pour la Belgique, où la part des dépenses de santé compose environ 9% du PIB.

2.2 Organisation interne de l'hôpital

De manière générale, le système hospitalier est une organisation hiérarchique. Il est gouverné par un système central de gestion et d'approvisionnement et est géré par un conseil d'administration et un directeur général. La direction est chargée de différentes unités : les unités de soins (secteur de production de soins) et les unités administratives. Ces unités sont subdivisées en services classés selon leurs activités.

[5] PIB : Produit intérieur brut

[6] Organisation de Coopération et de Développement Economiques

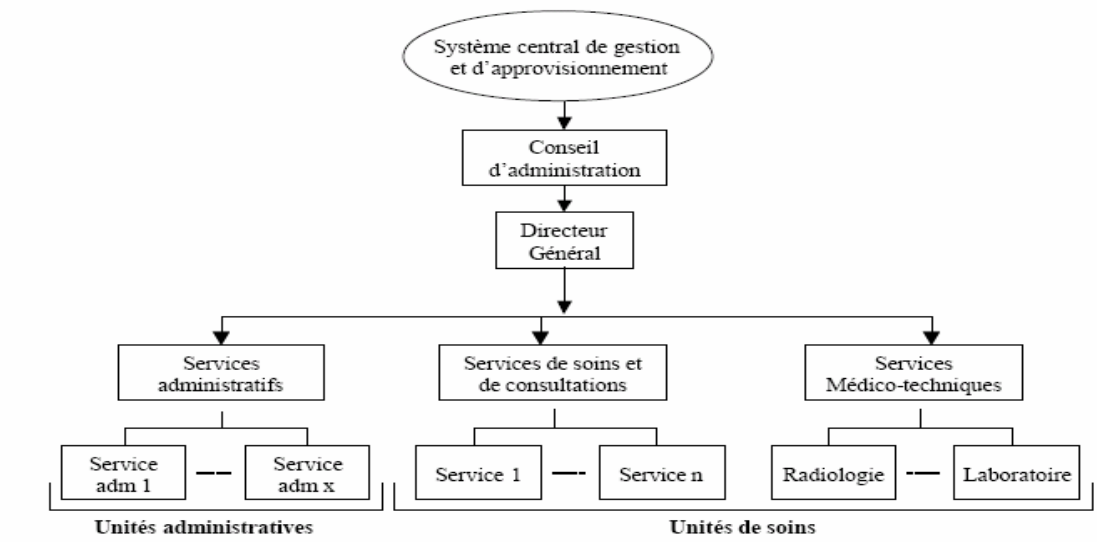

Figure 2 : Vision structurelle d'un système hospitalier

Comme montré dans la Figure 2, un système hospitalier se divise principalement en un ensemble de centres de responsabilités : le système central de gestion et d'approvisionnement, le conseil d'administration, les services administratifs, les services de soins et de consultations et les services médico-techniques. Chaque centre de responsabilité est lui-même divisé en unités fonctionnelles qui réalisent un type d'activité et qui sont rattachées à un service. De manière similaire, Moidon et Tonneau [1999] ont aussi étudié le système d'un établissement hospitalier et classifié la grande variété d'unités en cinq secteurs :

1. Les services cliniques : c'est le lieu d'hébergement des patients. L'équipement de ces services est généralement faible (essentiellement des lits), sauf pour des unités dites de soins intensifs (réanimation par exemple). L'organisation du travail est liée aux pratiques des médecins : chaque service est dirigé par un médecin spécialisé ;
2. Les consultations : elles sont souvent incluses dans des services cliniques, au point d'être souvent difficilement séparables au niveau des ressources, et servent de filtre entre la demande du public et l'hôpital. Dans ce secteur, le service d'urgence a une place à part, puisque les malades viennent sans rendez-vous et relèvent en principe de soins immédiats. Comme les services cliniques, les consultations sont des unités peu technologiques sauf pour certains services d'urgence ;
3. Le plateau technique : il inclut la chirurgie, l'obstétrique, la pédiatrie, l'anesthésie réanimation, les explorations fonctionnelles et les disciplines interventionnelles, l'imagerie et la biologie dans une perspective d'accès à des soins de qualité 24 heures sur 24 pour la population du territoire qu'il dessert [Revel *et al.*, 2003, p8]. Là aussi, les chefs de service sont des professionnels de la santé (médecins, pharmaciens, chirurgiens, etc.), ayant suivi un cursus de formation dans la spécialité correspondante. Ces unités travaillent pour les services cliniques et les consultations, il s'agit d'une concentration et d'une collectivisation des équipements due aux grands investissements induits par la technicisation de la médecine ;
4. Le secteur logistique : il prend en charge l'hébergement des malades et effectue un certain nombre de tâches à destination des unités précédemment décrites. Il s'agit

essentiellement de la restauration et de la blanchisserie, activités pouvant parfois être sous-traitées, mais aussi du transport des malades, de consommables, d'examens, etc. Il est possible de classer dans ce secteur la pharmacie, lieu de stockage, de préparation et de distribution des produits médicamenteux qui pourrait appartenir à d'autres unités ;

5. Le secteur administratif : il est composé de la direction générale, du service financier, de celui du personnel, de l'économat, de l'informatique, etc. Le directeur de l'hôpital appartient à un corps d'administrateurs spécialisés. Ce n'est donc pas un médecin. Dans un centre hospitalier régional (CHR), l'équipe de direction est de quelques dizaines de personnes, nombre systématiquement faible par comparaison avec les entreprises industrielles de même importance.

De plus, Gabel *et al.* [1999] présentent une structure traditionnelle d'organisation pour les hôpitaux, liée au bloc opératoire (Figure 3) :

Dans cette structure d'organisation, toutes les entités ont été créées pour faciliter les activités opérationnelles.

En général, dans n'importe quel système d'organisation utilisé, le bloc opératoire représente une partie très importante. Donc, dans la section suivante, nous allons nous concentrer sur le secteur du bloc opératoire.

3. Gestion du bloc opératoire

La gestion des blocs opératoires est actuellement l'objet de multiples questionnements, aussi bien sur le court terme que sur le long terme. Elle se heurte en effet aux problèmes difficiles d'emploi des ressources aussi bien humaines que matérielles, qui sont aujourd'hui toutes deux disponibles en capacité finie.

Le bloc opératoire, consistant normalement en salles d'opération et en salles de réveil (dites aussi Salles de Surveillance Post-Interventionnelle, SSPI), représente un univers composite dans lequel s'exprime une multitude de professions et de cultures différentes dont la finalité devrait être identique : le soin aux patients [Cuviller, 1997a, 1997b].

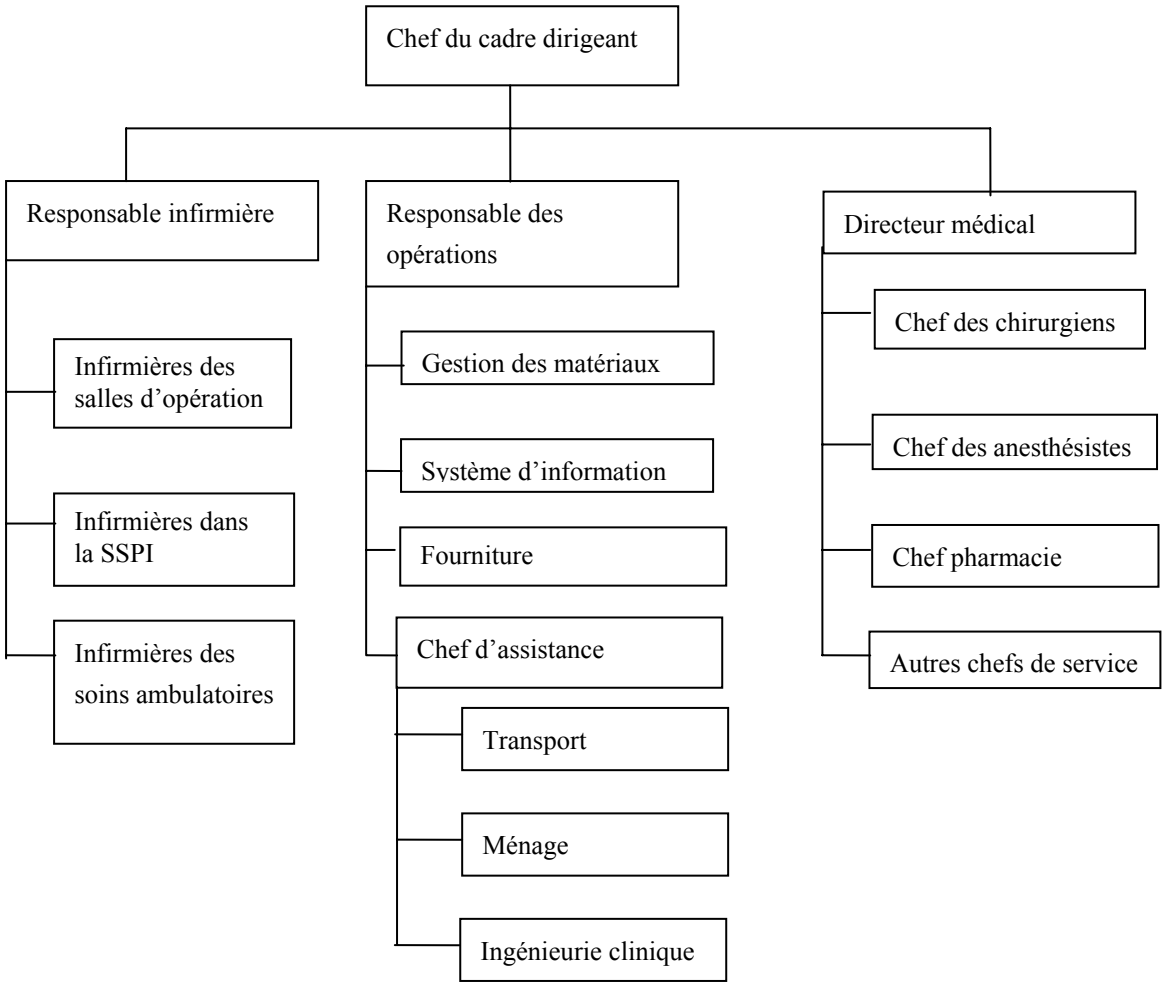

Figure 3 : **Structure traditionnelle d'une organisation hospitalière liée au bloc opératoire**

Il est un lieu hautement stratégique du plateau technique parce qu'il implique presque toutes les activités des secteurs présentes dans ce plateau technique comme souligné par Revel *et al.* [2003] : « Le bloc opératoire est à l'interface de nombreuses activités : chirurgie, obstétrique, anesthésie, explorations fonctionnelles, radiologie et biologie. Une organisation plus efficiente des blocs opératoires est nécessaire pour mieux utiliser les moyens humains et techniques disponibles et améliorer leur productivité. La gestion des blocs opératoires devrait pouvoir évoluer rapidement vers des centres de responsabilité dans les établissements publics. ».

La Figure 4 montre l'interrelation entre le bloc opératoire et les autres services.

Par conséquent, nous allons nous concentrer sur le nœud central, le bloc opératoire, en nous focalisant sur la programmation opératoire afin de rendre ce secteur plus efficace et plus productif.

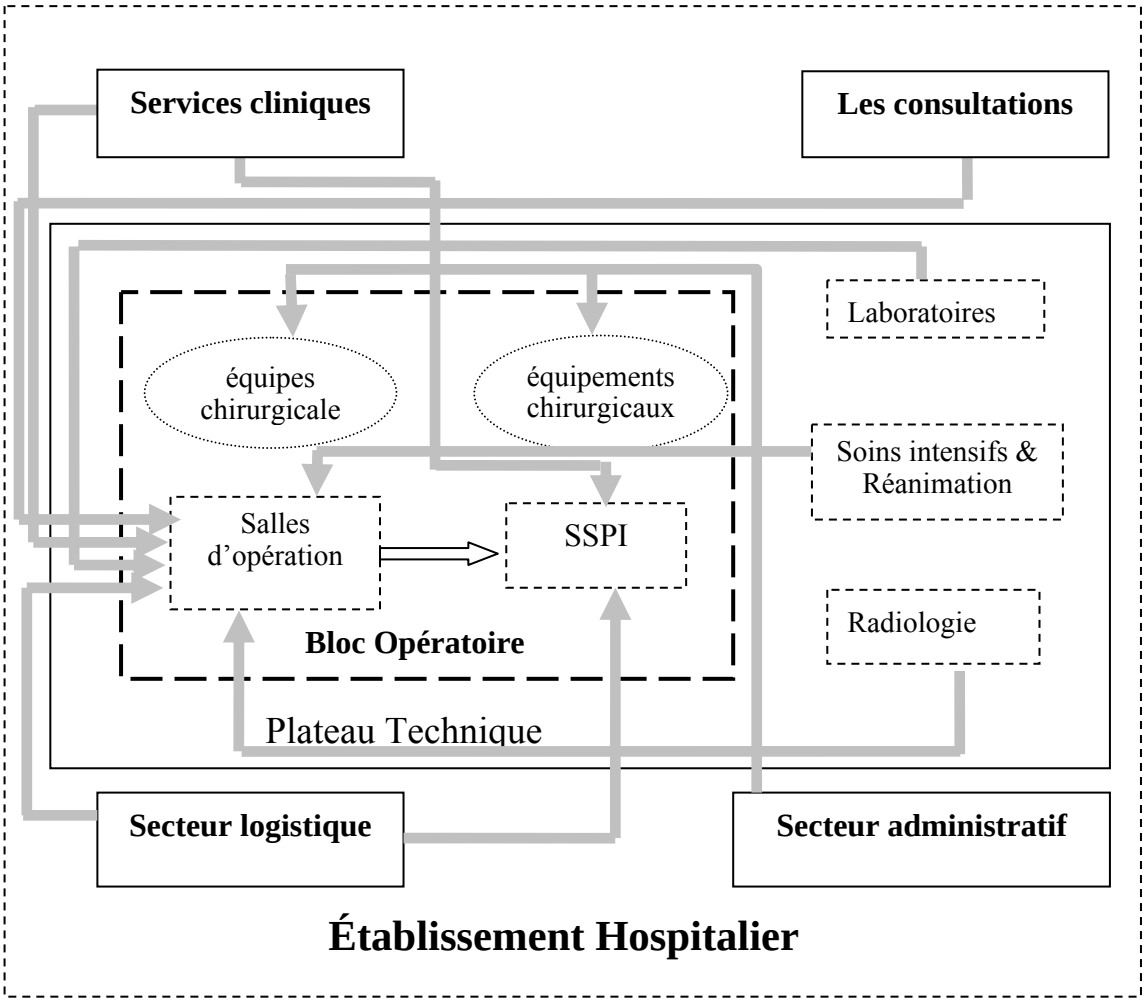

Figure 4 : Interrelation entre le bloc opératoire et les autres services

3.1 Processus opératoire

Avant d'entamer la discussion sur la programmation opératoire, décrivons le processus opératoire dans l'établissement hospitalier, surtout dans un bloc opératoire.

Les processus opératoires se décomposent en trois phases :

1. La phase pré-opératoire durant laquelle le patient subit des consultations chirurgicales et anesthésiques. Elle s'étend de la prise en charge du patient jusqu'à la veille de l'intervention. Pendant cette phase, une date « provisoire » d'intervention sera proposée au patient. Cette date peut être modifiable ou non selon la politique du bloc opératoire ;
2. La phase per-opératoire proprement dite qui s'étend de la préparation psychique du patient, avant l'intervention, jusqu'à ce qu'il se réveille et quitte la SSPI. Cette phase a lieu le jour de l'intervention. Ce jour-là, les patients sont d'abord anesthésiés et transférés par les brancardiers aux salles d'opération, où les équipes chirurgicales vont les opérer. Après avoir été opérés, ils vont être transférés à la SSPI et y restent jusqu'au moment où les anesthésistes les autorisent à retourner dans leurs services d'hospitalisation ou dans les secteurs de soins intensifs et de réanimation. Cette phase est la partie la plus importante du processus opératoire ;

3. La phase post-opératoire : après son réveil, le patient sera transféré vers son service d'hospitalisation. Cette phase recouvre l'ensemble des soins nécessaires suite à l'intervention. Dans le cas où l'état du patient serait jugé critique, il sera plutôt conduit vers les soins intensifs et de réanimation.

Dans cette recherche, nous allons essayer de rendre ce processus opératoire plus efficace en trouvant une bonne solution pour la planification des interventions dans la phase pré-opératoire, sur plusieurs semaines ou mois, ainsi que pour l'ordonnancement des interventions dans la phase per-opératoire, pour une journée. Nous ne tiendrons pas compte de la phase post-opératoire.

3.2 Programmation opératoire

Le bloc opératoire constitue non seulement le secteur le plus important et le plus coûteux dans la majorité des hôpitaux, mais il est également le service le plus complexe à gérer et à planifier, étant donnés les nombreux aléas, le nombre élevé d'acteurs, la difficulté de standardisation et de coordination des interventions chirurgicales.

De façon générale, nous pouvons dire que l'activité de planification opératoire consiste à organiser au mieux l'activité des différentes salles d'opération de sorte à utiliser les blocs opératoires le plus efficacement possible (minimisation des périodes de temps mort et du recours aux heures supplémentaires), à minimiser les coûts d'utilisation, à satisfaire les besoins des chirurgiens, à satisfaire les besoins des patients, à satisfaire les besoins des autres membres de l'équipe opératoire (personnel infirmier et anesthésistes), à utiliser efficacement les salles de réveil et à atteindre un faible niveau d'annulation.

A la lecture de ces objectifs, il apparaît évident que bon nombre d'entre eux ne peuvent être pris en considération simultanément. A titre d'exemple, la règle qui veut que l'nous plaçons en premier les cas supposés les plus longs afin de limiter le recours aux heures supplémentaires est contraire à une bonne organisation des salles de réveil.

Le nombre de facteurs à prendre en considération lors de la planification de l'activité opératoire est important. Dans leur article, Hamilton et Breslawski [1994] nous dressent un inventaire assez complet des éléments potentiellement importants à prendre en compte lors de la programmation opératoire. Les facteurs recensés sont de plusieurs types. Nous y trouvons des contraintes fortes telles que le nombre de lits disponibles en salle de réveil, la disponibilité en nombre de salles d'opération ou en équipements, le nombre d'anesthésistes disponibles, etc. Nous y recensons également des facteurs à optimiser tels que le nombre de cas planifiés, l'heure de début de l'intervention désirée, la séquence opératoire souhaitée, etc. Enfin certains facteurs relèvent plutôt d'une meilleure gestion de l'environnement du bloc opératoire, citons ici, les arrivées tardives du personnel, la mise à disposition de dossiers patients incomplets, l'absence de fournitures,…. Une bonne gestion implique donc la résolution des problèmes organisationnels, une prise en compte de l'ensemble des contraintes dites fortes et une optimisation des contraintes dites faibles.

La programmation opératoire, appelée « wait list management» ou « surgical process management » dans la littérature anglo-saxonne, consiste à construire un planning prévisionnel des interventions chirurgicales à réaliser pendant une période, généralement une semaine, à partir des demandes émanant des services chirurgicaux et de prescripteurs externes.

Comme présenté dans [Magerlein et Martin, 1978], la programmation opératoire se décompose de deux sous-problèmes séquentiels :

1. **Planification à l'avance (« advance scheduling »)** qui consiste à affecter une date d'opération aux patients dans l'avenir. D'après Margerlein et Martin [1978], le problème de la planification à l'avance peut se séparer en deux catégories de problèmes selon le nombre et le type de contraintes de ressources prises en compte : dans une catégorie, nous ne prenons en compte que la durée d'ouverture des salles d'opération; dans l'autre catégorie, nous prenons en compte la durée d'ouverture des salles d'opération ainsi que la disponibilité des lits d'hospitalisation. De plus, d'autres chercheurs subdivisent la planification à l'avance, selon les techniques utilisées pour affecter des plages horaires aux chirurgiens, en deux types : « les plages horaires réservées» et « premier arrivé premier servi » ;
2. **Ordonnancement d'affectation (« allocation scheduling »)** qui consiste à déterminer l'ordre des interventions dans un bloc opératoire pour une journée. D'après Margerlein et Martin [1978], nous pouvons aussi diviser l'ordonnancement d'interventions en deux catégories : dans une première catégorie, nous visons à estimer au mieux la durée d'opération des interventions afin de diminuer le risque de non faisabilité du programme opératoire; dans l'autre catégorie, les interventions sont ordonnancées dans les salles d'opération pour une journée.

Hamilton et Breslawki [1994] listent les dix éléments les plus importants à tenir en compte dans l'élaboration d'un programme opératoire : nombre des salles d'opération; nombre des équipements chirurgicaux spécialisés; disponibilité des plages horaires réservées aux chirurgiens; politique de la programmation opératoire dans un hôpital ; durées opératoires estimées; temps normal d'ouverture et fermeture des blocs opératoires; disponibilité d'anesthésistes; limitation des salles liées aux équipements chirurgicaux; heure de départ assignée aux chirurgiens; les heures supplémentaires possibles des salles d'opération.

Blake et Carter [1997a, 1997b] ont rajouté un troisième sous-problème : l'ordonnancement des ressources extérieures. Celles-ci représentent toutes les ressources nécessaires pour les interventions mais fournies par des fournisseurs externes à l'hôpital, comme les équipements chirurgicaux et les autres matériels médicaux. Selon leur définition, l'ordonnancement des ressources extérieures vise à identifier et réserver toutes les ressources extérieures aux blocs opératoires pour assurer leurs services aux patients.

La construction du programme opératoire est donc très complexe et son processus est très variable d'un hôpital à un autre. Toutefois, trois modèles ont pu être dégagés : la programmation par allocation préalable de plages horaires (« block scheduling »), la

programmation ouverte (« open scheduling »), et la programmation par allocation et ajustement de plages (« modified block scheduling »). [Patterson, 1996][Kharraja, 2003]

3.2.1 Programmation par allocation préalable de plages – « block scheduling »

La programmation par allocation préalable des plages horaires est une programmation opératoire correspondant à une affectation d'une salle d'opération durant une période fixée à un et un seul chirurgien.

La problématique à traiter pour ce type de programmation opératoire se compose de deux parties :

- L'allocation préalable de plages horaires (« Master Surgical Schedule ») à chaque chirurgien (ou unités chirurgicales) selon leurs disponibilités et préférences [Magerlein et Martin, 1978] [Przasnyski, 1986] [Kennedy, 1992] et [Blake et Carter, 1997a, 1997b] ;
- Remplissage des plages horaires : planification et l'ordonnancement des interventions selon le chirurgien dans leurs plages horaires réservées.

Durant cette période, seuls les chirurgiens en question peuvent planifier et ordonnancer leurs interventions dans les plages horaires des salles qui leurs ont été affectées.

Dès que ces plages horaires sont affectées à un chirurgien pour une période donnée, elles ne peuvent plus être modifiées excepté lors des négociations et des redéploiements des plages horaires par la direction de l'établissement hospitalier (par exemple tous les trois mois ou annuellement). La Figure 5 présente un exemple de planning d'affectation de plages horaires de lundi à mardi pour trois salles d'opération et cinq chirurgiens.

<table>
<tr><td></td><td colspan="6">LUNDI</td><td colspan="6">MARDI</td></tr>
<tr><td></td><td>8</td><td>10</td><td>12</td><td>14</td><td>16</td><td>18</td><td>10</td><td>12</td><td>14</td><td>16</td><td>18</td></tr>
<tr><td>Chirurgien 1</td><td colspan="3"></td><td colspan="3">SALLE 1</td><td colspan="3"></td><td colspan="3">SALLE 3</td></tr>
<tr><td>Chirurgien 2</td><td colspan="3">SALLE 3</td><td colspan="3"></td><td colspan="6">SALLE 1</td></tr>
<tr><td>Chirurgien 3</td><td colspan="3"></td><td colspan="3">SALLE 3</td><td colspan="3">SALLE 2</td><td colspan="3"></td></tr>
<tr><td>Chirurgien 4</td><td colspan="6"></td><td colspan="3">SALLE 3</td><td colspan="3"></td></tr>
<tr><td>Chirurgien 5</td><td colspan="6">SALLE 2</td><td colspan="3"></td><td colspan="3">SALLE 2</td></tr>
</table>

Figure 5 : **Un exemple de planning des lundis et mardis pour les 3 salles d'opération du bloc opératoire**[7]

Ce type de programmation opératoire devient de plus en plus courant dans les hôpitaux. Il a ses avantages et inconvénients. Dès que les plages horaires ont été établies, la planification des interventions est plus aisée que dans les autres types de programmation opératoire et de plus, il n'y a aucun risque pour qu'un chirurgien doive opérer deux patients en même temps dans deux salles d'opération différentes.

[7] Dans ce planning des plages horaires, la salle 1 est fermée le lundi matin pour la maintenance

Par ailleurs, les inconvénients sont :

1. La difficulté de construction du plan directeur d'allocation préalable de plages horaires. Le placement et la taille des plages horaires sont en effet les facteurs déterminants de la qualité d'allocation des plages horaires. Des plages surdimensionnées apporteront du confort pour les chirurgiens mais conduiront à une performance médiocre et des plages justes ou sous dimensionnées risquent, en cas d'aléas, de provoquer des tensions entre chirurgiens lors du changement de plages ;
2. La rigidité du programme impose aux chirurgiens une activité à volume constant. Au risque de voir s'écrouler les performances productives du bloc, il est de la « responsabilité » des chirurgiens de remplir de façon optimale les plages qui leur sont réservées et de respecter les durées d'interventions prévues ;
3. La perte de flexibilité de la planification car les plages horaires ne peuvent pas être modifiées. Donc il peut arriver que certains chirurgiens doivent refuser des interventions alors que d'autres ne remplissent pas leurs plages horaires.

3.2.2 Programmation ouverte – « open scheduling »

Contrairement à la stratégie du « block scheduling », la programmation ouverte est une programmation opératoire consistant à construire les plannings d'allocation des interventions pour une période de planification sans tenir compte des préférences des chirurgiens.

Deux approches [Kharraja, 2003] sont appliquées dans les hôpitaux pratiquant ce type de programmation pour gérer le planning d'allocation des interventions dans un hôpital : la règle du FCFS (« First Come First Served », premier arrivé premier servi) [Patterson, 1996] et la négociation entre les acteurs.

A. Premier arrivé premier servi - FCFS

Cette approche d'allocation des interventions se fait de manière chronologique durant toute la période de construction du planning. Il s'agit d'un « agenda collectif » [Kharraja, 2003] qui doit être connecté à un système d'information qui collecte les données sur les durées des interventions réalisées par type et par chirurgien et qui servira à estimer les durées d'opération des interventions à réaliser.

Cet agenda peut être géré par les secrétariats des différents services chirurgicaux. Dans le cas du placement d'une intervention occasionnant un dépassement d'horaire, le responsable du bloc doit être directement alerté, ce qui déclenchera un processus de régulation dont l'objectif serait l'obtention d'une solution de résolution négociée.

Cette technique présente l'avantage d'être extrêmement simple à mettre en œuvre. Cette simplification de la construction du planning apporte toutefois des inconvénients. L'inconvénient majeur est que cette méthode favorise les services chirurgicaux qui ont une activité planifiée sur du moyen et long terme, par exemple l'ophtalmologie, la chirurgie plastique ce qui va déranger les autres chirurgiens qui ne peuvent pas prédire leur agenda à moyen terme. De plus, comme le souligne Kontak-Forsyth et Grant [1995], cette pratique engendre un fort taux de déprogrammation, une sous-utilisation

des ressources, des dépassements horaires importants et engendre de fortes tensions entre les chirurgiens ou les services de chirurgie.

B. Négociation entre les acteurs

Compte tenu des inconvénients de la stratégie de FCFS, le programme peut s'élaborer normalement sous la direction du responsable du bloc opératoire suivant un processus de négociation entre les différents acteurs.

Kharraja [2003] présente un modèle de processus de construction du programme opératoire selon son étude effectuée au CHU (Centre Hospitalier Universitaire) de la Croix Rousse (Lyon). Les différentes étapes de ce processus sont schématisées dans la Figure 6.

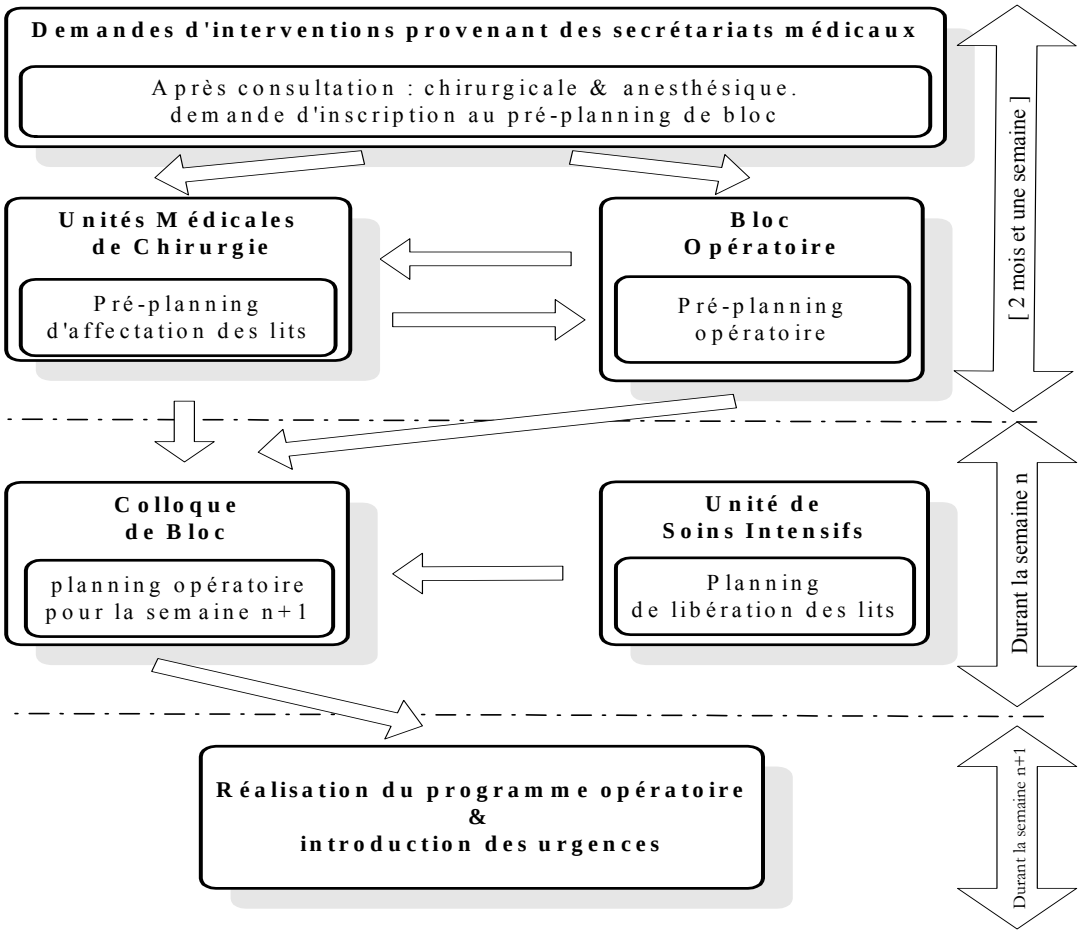

Figure 6 : Processus temporel de création du programme opératoire [Kharraja, 2003]

Dans ce processus, nous pouvons trouver que son élément clé se situe lors de la réunion « colloque de bloc » durant laquelle doivent être harmonisés tous les pré-plannings. À ce stade, la difficulté essentielle à surmonter est la recherche collective d'une solution « gagnant/gagnant » pour l'ensemble des acteurs, donc la construction du planning doit s'appuyer sur les contributions de bonnes méthodes théoriques et expériences réussies.

3.2.3 Programmation par allocation et ajustement de plages – « modified block scheduling »

Le modèle de programmation par allocation et ajustement de plages est une combinaison des stratégies de « block scheduling » et d'« open scheduling ». Cette stratégie s'appuie globalement sur les principes de « block scheduling » mais en offrant plus de flexibilité du fait de sa prise en compte de plusieurs facteurs :

1. Des plages horaires libres pour résorber d'éventuels dépassements horaires ou surcharge de travail. Généralement, ces plages sont affectées comme pour l'« open scheduling » par la négociation entre les chirurgiens et le responsable du bloc ;
2. La possibilité pour un chirurgien de céder une plage horaire (ou une partie) inutilisée à d'autres chirurgiens. Donc si les chirurgiens ou les unités chirurgicales n'utilisent pas suffisamment leurs plages horaires réservées dans une période convenue, par exemple 48 – 72 heures, ils n'ont plus de priorité sur ces plages horaires et le responsable du bloc peut alors les ajuster pour en faire bénéficier d'autres chirurgiens ou unités chirurgicales [Malhorta, 2001].

Comme nous le voyons, la stratégie du « modified block scheduling » offre plus de flexibilité que celle de « block scheduling » en terme de réutilisation des plages (ou parties) horaires inutilisées. Donc, elle est de plus en plus appliquée aux Etats-Unis notamment [Patterson, 1996]. Mais, cela entraîne aussi l'inconvénient de coordonner les demandes des chirurgiens avec la planification du responsable du bloc. Malgré le fait qu'elle soit la plus plébiscitée par les établissements hospitaliers américains, il y a très peu de littérature sur cette problématique excepté Kharraja et Marcon [2003] et Marcon *et al.* [2003] qui étudient l'intégrité du processus de fonctionnement du « modified block scheduling » pour l'amélioration et l'exploitation des plages non utilisées.

4. Notre problématique

Dans les parties précédentes, nous avons présenté le contexte des établissements hospitaliers, surtout celui des blocs opératoires. Etant donné que le bloc opératoire constitue le secteur le plus coûteux de l'hôpital, nous allons dans cette recherche tenter d'améliorer le fonctionnement de ce secteur. En fait, il y a plusieurs manières d'y arriver, comme la reconfiguration des blocs opératoires, l'amélioration des systèmes d'information, ou encore l'amélioration des programmes opératoires. Nous allons nous concentrer sur la gestion de la programmation opératoire. Comme nous avons dit, la typologie des problèmes concernant la programmation opératoire est variée et il est impossible de traiter tous les problèmes concernés. Nous allons donc nous préoccuper des deux politiques de la programmation opératoire les plus courantes : la programmation par allocation préalable de plages, « block scheduling », et la programmation ouverte, « open scheduling ».

Du fait de la difficulté de ces problèmes, nous allons les décomposer en deux étapes :

1. Etape de planification qui consiste à affecter une date d'opération aux patients (interventions) dans l'avenir, où nous ne considérons que les ressources les plus

critiques dans le bloc opératoire, à savoir les plages horaires dans les salles d'opération ;

2. Etape d'ordonnancement qui vise à déterminer l'ordre des interventions dans un bloc opératoire pour une journée. Dans cette étape, nous élaborons un planning plus détaillé en prenant en compte le fait qu'un chirurgien ne peut pas effectuer plusieurs opérations en même temps et intégrons les ressources moins critiques telles que le nombre des lits dans la salle de réveil.

En général, la structure de notre problématique peut être décrite comme ci-dessous :

A. Problème du « block scheduling »

Premièrement, un modèle mathématique va être construit pour le problème de planification des interventions sur une semaine. L'objectif de ce problème de planification est la minimisation du coût des heures de sous-utilisation et des heures supplémentaires sous la contrainte de la durée d'ouverture de chaque plage horaire réservée à un chirurgien. Selon la taille du problème, celui-ci sera résolu par une approche approximative ou exacte.

Dès que le problème de la planification est résolu, nous obtenons un ensemble d'interventions pour chaque plage horaire réservée dans la période de planification. Ensuite, nous allons résoudre le problème d'ordonnancement pour une journée où la contrainte relative à la disponibilité des lits dans la salle de réveil devra être prise en compte. C'est un problème à deux étages : des salles d'opération dans le premier étage et des lits de réveil dans le deuxième étage. L'objectif de ce problème d'ordonnancement est la minimisation du coût total des heures dans la salle d'opération et des heures en SSPI.

B. Problème de l'« open scheduling »

Nous allons aussi résoudre le problème d'« Open Scheduling » en deux étapes. Tout d'abord, un modèle mathématique est construit pour affecter sur une semaine les interventions aux salles d'opération. L'objectif est le même que celui du problème du « Block scheduling » : la minimisation des coûts des heures de sous-utilisation et des heures supplémentaires mais avec comme contrainte la capacité en nombre d'heures de travail du chirurgien ou en nombre de gestes.

Ensuite, nous allons résoudre le problème de l'ordonnancement. Ce problème est plus complexe que celui du « Block scheduling » car au lieu de ne traiter que le problème de l'ordonnancement des interventions pour un seul chirurgien, nous devons prendre en compte les interventions de tous les chirurgiens et toutes les salles d'opération disponibles pour ce jour-là. Donc, nous allons devoir traiter deux sous-problèmes d'ordonnancement :

1) Problème d'ordonnancement où nous ne prenons en compte que la contrainte relative à la durée d'ouverture des salles d'opération multifonctionnelles et la contrainte qu'un chirurgien ne peut opérer qu'une intervention en même temps. L'objectif est la minimisation du coût total des heures dans les salles d'opération ;

2) Problème d'ordonnancement où, en plus des contraintes citées en 1), nous prenons en compte la contrainte du nombre de lits disponibles en SSPI. C'est un problème à deux étages : les salles d'opération dans le premier étage et les lits en SSPI dans le deuxième étage. L'objectif est la minimisation du coût total des heures dans les salles d'opération et en SSPI.

5. Conclusion

La programmation opératoire est un problème important dans un établissement hospitalier mais aussi très complexe concernant les éléments qu'il implique et les contraintes qu'un programme opératoire réalisé doit respecter. Dans la pratique hospitalière, différentes stratégies de programmation opératoire sont utilisées et aucune n'est vraiment meilleure qu'une autre.

Partie II :

Revue de la littérature concernant la programmation opératoire et rapprochement avec la gestion de production

1. Introduction

Bien que la gestion du bloc opératoire, et plus particulièrement la planification et l'ordonnancement des interventions dans le bloc, soit un problème hautement important pour les gestionnaires hospitaliers, il n'y a que peu de chercheurs qui travaillent dans ce domaine. De plus, plupart d'entre eux visent à évaluer et améliorer les programmes opératoires existants alors que seulement certaines travaillent sur la construction de programmes opératoires réalisables et de bonne qualité.

Dans cette partie, nous allons montrer un état de l'art des études centrées sur la programmation opératoire à savoir l'évaluation et l'amélioration de la performance des programmes opératoires existants et l'élaboration de la programmation opératoire.

D'autre part, dans le milieu industriel, surtout dans les systèmes de production, les gestionnaires d'entreprises rencontrent aussi des problèmes similaires où de nombreuses contraintes de ressources humaines (ouvriers, techniciens, etc.) et matérielles (machines, outillages, convoyeurs, etc.) doivent être prises en compte. Dès les années 80, du fait d'une compétitivité économique accrue, la gestion de production vise l'optimisation des systèmes productifs en minimisant le coût de production ou en maximisant l'efficacité. De nombreux travaux de recherche portant sur cette optimisation ont été développés.

En analysant certains rapprochements existants entre le système hospitalier et le système de production, nous allons étudier dans cette partie si nous pouvons nous inspirer des méthodes et techniques de la gestion de production pour la gestion des blocs opératoires. Une analyse comparative entre ces deux systèmes sera donnée. Finalement, nous terminons cette partie en donnant quelques conclusions.

2. Etat de l'art

2.1 Evaluation de la performance des programmes opératoires

L'amélioration de l'exploitation des blocs opératoires est une problématique relevant du domaine d'optimisation combinatoire car elle implique la résolution de modèles de

planification, de modèles d'ordonnancement et aussi la combinaison de ces deux modèles.

En général, les évaluations sont effectuées par le biais de méthodes de simulation ou de méthodes d'analyse avec comme objectif de trouver où le système est embouteillé [Everett, 2002], la stratégie la plus efficace [Ramis *et al.*, 2001] [Dexter et Macario, 2002] [Mcleod *et al.*, 2003], ou d'analyser les impacts des modifications effectuées dans les plannings [Gallivan, 1998] [Blake *et al.*, 1995], [Lapierre *et al.*, 1999].

Pour améliorer la gestion d'un bloc opératoire, les gestionnaires ou les chercheurs doivent normalement tout d'abord évaluer la performance du système actuel, par simulation ou modèles statistiques, afin d'identifier les goulots d'étranglement. Ensuite, les améliorations possibles, comme le changement du processus, le déplacement des patients, le développement des systèmes d'information, la construction de modèles pour les problèmes importants [Lagergren, 1998], seront prises en compte pour rendre plus efficace la programmation opératoire, avec comme objectifs de maximiser l'utilisation des blocs opératoires, minimiser les coûts engendrés, etc.

L'étude de l'évaluation de la performance des programmes opératoires est l'une des plus développées dans le domaine de la gestion hospitalière. Les indicateurs utilisés pour évaluer cette performance sont variés. Citons les coûts variables périopératifs [Dexter *et al.*, 2002], la durée d'attente moyenne, la durée d'attente maximale, le nombre d'interventions effectuées [Gallivan, 1998], le taux d'utilisation des salles d'opération [Strum *et al.*, 1997], les coûts des programmes opératoires et l'allocation des budgets [Strum *et al.*, 1997][Murphy et Sigal, 1985], le nombre des équipes chirurgicales, les plages horaires, les coûts d'opération et les changements d'allocation des interventions [Dexter et Macario, 2002].

2.2 Amélioration des programmes opératoires

Comme nous l'avons dit, l'efficacité d'un programme opératoire se base sur de nombreux éléments car le fonctionnement d'un bloc opératoire implique presque tous les autres secteurs chirurgicaux. Dans un établissement hospitalier idéal, tous les secteurs se coordonnent, les durées d'opération des interventions sont connues, les équipements chirurgicaux sont toujours en bon état, les chirurgiens sont disponibles, arrivent à temps, etc. Il n'y a ni incident ni goulot d'étranglement. Bien que cette description soit utopique, tout doit être mis en œuvre pour se rapprocher de cette situation idéale.

Selon la littérature, les moyens d'amélioration des programmes opératoires peuvent être divisés en deux catégories :

A. Amélioration de la coordination des secteurs chirurgicaux

Pour ce faire, quelques chercheurs tentent d'améliorer la coordination des acteurs de soins travaillant dans les blocs opératoires, comme les chirurgiens, les anesthésistes, les infirmières de salle d'opération, les infirmières anesthésistes, les aides-soignants, les agents hospitaliers etc., pour réduire le gaspillage de leur temps de travail et favoriser

les conditions d'une performance optimale du système. Par exemple, Overdyk *et al.* [1998] mettent en relief une méthodologie de travail et une réflexion multidisciplinaire, précisant le rôle de chacun ainsi que sa responsabilité tout au long du processus opératoire, pour améliorer l'efficacité des salles d'opération. Dans le même contexte, Delesie [1998] propose trois étapes pour établir un rapprochement entre les cliniciens et les gestionnaires hospitaliers. Dexter *et al.* [2000a] nous présentent un système d'information EWPS (Enterprise-Wide Patient Scheduling Information System) pour coordonner le flux des patients entre le service clinique et les blocs opératoires et ainsi augmenter l'efficacité des blocs opératoires. De plus, Marty *et al.* [2001] donnent des propositions pour coordonner les activités des acteurs pour une meilleure prestation avec le maximum de sécurité et le moindre coût des blocs opératoires. D'autre part, quelques chercheurs [HFMA, 2003] se concentrent sur l'amélioration de la coordination des secteurs chirurgicaux dans les blocs opératoires pour rendre le processus des patients plus efficace et réduire le coût total d'opération des blocs opératoires.

B. Améliorer les estimations de la durée opératoire

Il est évident que nous avons besoin de connaître des informations concernant l'intervention et notamment sa durée pour obtenir un programme opératoire. Tout d'abord, citons un article de Fischer [1999] qui a relevé trois durées devant être prises en compte pour chaque intervention dans la construction d'un programme opératoire réalisable :

1. La durée de la prise en charge anesthésique : elle dépend du mode d'anesthésie et des moyens de surveillance nécessaires qui ont été définis lors de la consultation anesthésique ;
2. La durée de la prise en charge chirurgicale : les opérateurs sous-estiment fréquemment la durée de leurs interventions avec un pourcentage d'erreur compris entre 24% et 44% du temps opératoire [Wright, 1996] ;
3. La durée nécessaire à la remise en état de la salle d'opération et au changement de patient : elle dépend des produits mis en place et peut être, de ce fait, standardisée.

Généralement, l'hôpital détermine les durées opératoires, par pathologie, sur base d'un historique et les représente sous la forme de valeurs moyennes. Or une connaissance plus précise de ces durées opératoires est importante dans la réalisation d'un programme opératoire. En effet si la durée opératoire est mal estimée, le programme opératoire se heurte à un risque de non faisabilité et aussi à une augmentation du coût d'opération, y compris le coût d'ajustement des interventions, des groupes chirurgicaux, des équipements chirurgicaux, etc. Par conséquent, des chercheurs se sont donc attelés à spécifier plus précisément ces durées opératoires et nous pouvons distinguer deux courants d'idées.

Un premier groupe de chercheurs, Zhou et Dexter [1998], Strum *et al.* [1998, 2000a], Macario et Dexter [1999], Dexter *et al.* [1999a] sont d'avis de rattacher la durée opératoire à des lois statistiques. Dans un second courant d'idées, d'autres auteurs, Combes *et al.* [2004], Dexter *et al.*, [1999d], Shukla *et al.* [1990], Opit *et al.* [1991], Wright *et al.* [1996], Zhou *et al.* [1999], Strum *et al.* [2000b], Levecq *et al.* [2003]

essaient plutôt d'étudier l'impact de certains facteurs sur la variabilité de la durée opératoire. Citons les caractéristiques du patient (âge, sexe, statut physique ASA, historique médical) ; les facteurs chirurgicaux (type d'intervention et durée, type d'anesthésie, variables physiologiques, médicaments donnés) ; les événements intra-opératoires (problèmes cardiovasculaires, hypertension, ..) ; les facteurs propices aux complications (hypertension, diabète, angine de poitrine, …).

C. Améliorer la politique d'affectation des interventions

La manière la plus connue, pour améliorer un programme opératoire, consiste à trouver une affectation ou un ordonnancement des interventions satisfaisant l'objectif concerné, tel que de maximiser l'utilisation des salles d'opération, de minimiser les coûts d'opération, etc., en respectant les contraintes des ressources humaines et matérielles. En général, les chercheurs modifient ou échangent des morceaux de plannings opératoires existants et utilisent des modèles de simulation [Dexter et Macario, 1999][Lebowitz, 2003], les modèles d'analyse [Carter et Lapierre, 2001], ou même comparent la valeur d'objectif directement pour les cas simples [Dexter, 2000][Dexter *et al.*, 2001a, 2001b] pour évaluer les résultats de changement et ainsi trouver un meilleur programme opératoire.

2.3 Elaboration d'un programme opératoire

Bien que l'amélioration des programmes opératoires existants soit une idée pratique, il est aussi important de chercher la meilleure façon d'obtenir un programme opératoire le plus proche possible de l'optimum. Des recherches sont donc menées dans le domaine de l'optimisation de la programmation opératoire vers l'aide à la décision pour les gestionnaires hospitaliers.

Selon la littérature, comme synthétisé dans le Tableau 3, nous trouvons que de façon générale, les travaux d'optimisation de la programmation opératoire consistent à organiser au mieux l'activité des différentes salles d'opération de sorte à maximiser leur utilisation [Dexter *et al.*, 1999b][Dexter *et al.*, 1999c][Dexter et Traub, 2002], à minimiser leur coût d'utilisation [Dexter *et al.*, 2000b][Fei *et al.*, 2004, 2005a, 2005b][Kharraja *et al.*, 2002][Jebali *et al.*, 2003, 2006][Chaabane, 2004], à maximiser les bénéfices de l'hôpital [Gerchak *et al.*, 1996][Lovejoy et Li, 2002][Kuo *et al.*, 2003], à satisfaire les besoins des patients [Dexter *et al.*, 1999e][Guinet et Chaabane, 2003], à atteindre un faible niveau d'annulation [Marcon *et al.*, 2001a, 2001b] ou à satisfaire le plus possible d'objectifs en fonction de leur importance [Ogulata et Erol, 2003][Ozkarahan, 1995, 2000][Sier *et al.*, 1997].

Planification				
Articles	**Objectif**	**Contraintes**	**Méthode de résolution**	**Taille du problème expérimenté[8]**
[Fei *et al.*, 2005a]	- Minimiser le coût total des interventions	- Nombre des salles d'opération - Durée d'ouverture des salles d'opération - Date limite d'interventions	- Procédure heuristique basée sur la génération de colonnes	Une semaine/ 100 interventions/ 6 salles d'opération.
[Fei *et al.*, 2005b]	- Minimiser le coût total des interventions	- Nombre des salles d'opérations - Durées d'ouverture des salles d'opération - Date limite d'interventions	- Procédure de Branch-and-Price	Une semaine/ 50 interventions/ 4 salles d'opération.
[Guinet et Chaabane, 2003]	- Minimiser le coût du temps d'attente des patients	- Nombre limité des interventions opérées par un chirurgien - Date limite des interventions - Disponibilité de l'équipement chirurgical nécessaire pour chaque intervention	- Extension de la méthode Hongroise	Une semaine/ 85 interventions/ 3 salles d'opération.
[Kuo *et al.*, 2003]	- Maximiser les bénéfices	- Durée d'ouverture des salles d'opération - Disponibilité des chirurgiens	- Modèle de programmation linéaire résolu par EXCEL	Une semaine.

[8] La taille d'un problème se traduit notamment par le nombre de salles d'opération, le nombre d'interventions, le nombre de lits dans la salle de réveil, le nombre de chirurgiens pris en compte et l'horizon considéré.

[Lovejoy et Li, 2002]	- Minimiser le temps d'attente des patients - Maximiser la fiabilité de l'heure de départ des programmes opératoires - Maximiser le bénéfice de l'hôpital	- Contraintes concernées sont intégrées dans la fonction objectif	- Fixer un critère entre les trois dans le modèle mathématique et analyser les relations entre les deux critères restants afin de trouver le meilleur compromis entre les trois critères concernés	Une journée/ 25 salles d'opération.
[Marcon *et al.*, 2001a,2001b]	- Maximiser le bénéfice journalier en assurant un niveau d'annulation aussi faible que possible	- Durée d'ouverture des salles d'opération	- Modèles de Programmation linéaire en nombres entiers résolue par CPLEX	Une journée/ 45 interventions/ 8 salles d'opération.
[Ogulata et Erol, 2003]	- Minimiser la sous-utilisation des salles d'opération - Equilibrer la distribution des interventions entre les équipes chirurgicales - Minimiser le temps d'attente des patients	- Durée d'ouverture des salles d'opération - Heure d'arrivée des patients et leur priorité de sélection - Durée disponible des équipes chirurgicales	- Méthode de décomposition; - Modèles de programmation mathématique à buts multiples en nombres binaires (binary integer goal programming) résolus par « l'algorithme d'énumération implicite »	Une semaine/ 24 interventions/ 2 salles d'opération/ 3 équipes chirurgicales.
[Ozkarahan, 1995, 2000]	- Minimiser l'écart entre la durée opératoire totale des interventions assignées et la durée disponible des équipes chirurgicales - Minimiser le nombre des heures de sous-utilisation et de sur-utilisation des salles d'opération - Respecter les préférences des équipes chirurgicales -	- Durée disponible des équipes chirurgicales ; - Durée d'ouverture des salles d'opération - Préférences des équipes chirurgicales concernant les salles d'opération choisies et les dates d'opération	- Modèle de programmation mathématique à buts multiples en nombres binaires résolus par les heuristiques.	Une journée/ 40 interventions/ 7 salles d'opération.

Ordonnancement				
Articles	**Objectif**	**Contraintes spécifiques**	**Méthode de résolution**	**Taille du problème expérimenté**
[Dexter *et al.*, 1999b]	- Minimiser la sous-utilisation des salles d'opération	- Durée d'ouverture des salles d'opération	- Modèle de simulation comparant dix algorithmes d'ordonnancement existants	Une journée.
[Dexter et al., 1999c]	- Minimiser la sous-utilisation des salles d'opération	- Nombre des plages horaires assignées à un chirurgien par semaine - Durée disponible de chaque plage horaire ; - Durée moyenne d'attente des patients pour leur intervention	- Modèle de simulation comparant quatre algorithmes simples (*Next fit*, *First Fit*, *Best Fit*, *Worst Fit*)	Une semaine.
[Dexter *et al.*, 1999e]	- Minimiser le temps d'attente moyen des patients - Garder l'ordre des patients soumis à la salle d'opération - Minimiser la probabilité de mauvais résultats	- Toutes les interventions doivent être affectées aux salles d'opérations	- Méthode d'énumération pour le cas avec une salle d'opération - Heuristique simple pour le cas avec plusieurs salles d'opération	Une journée/ 3 interventions/ 1 salle d'opération.
[Dexter *et al.*, 2000b]	- Minimiser le coût des heures supplémentaires	- Durée supplémentaire maximale pour les interventions - Nombre des plages horaires assignées	- Une technique mathématique MCA (*Minimal Cost Analysis*) - Modèle de simulation	4 semaines.
[Dexter et Traub, 2002]	- Maximiser l'efficacité des salles d'opération (minimiser le coût des heures sous-utilisées et des heures supplémentaires)	- Toutes les interventions doivent être affectées aux salles d'opération.	- Comparer deux heuristiques simples (*Earliest Start Time* et *Latest Start Time*) afin de trouver la meilleure.	Une journée.

[Fei *et al.*, 2004]	- Minimiser le coût des heures supplémentaires des salles d'opération	- Durée d'ouverture des salles d'opération	- Algorithme génétique hybride	Une journée/ 60 interventions/ 6 salles d'opération.
[Kharraja *et al.*, 2002]	- Minimiser le *makespan*	- Durée d'ouverture des salles d'opération	- Modèle de programmation linéaire en nombres binaires résolu par CPLEX	Une journée/ 14 interventions/ 6 salles d'opération/ 6 lits de réveil.
[Sier *et al.*, 1997]	- Trouver une solution qui peut respecter le plus de contraintes	- Age des patients - Priorité des patients - Loi d'affectation préférée (l'intervention dont la durée opératoire la plus longue va être traitée en premier) - Disponibilité des équipements chirurgicaux - Durée d'ouverture des plages horaires.	- Modèle de programmation non linéaire en nombres entiers résolu par l'algorithme de Recuit simulé	Une journée/ 27 interventions/ 4 salles d'opération.
Planification & Ordonnancement				
Articles	**Objectif**	**Contraintes spécifiques**	**Méthode de résolution**	**Taille du problème expérimenté**
[Chaabane, 2004] (Chapitre 6)	- Minimiser le nombre de jours de retard et d'avance vis-à-vis des dates d'hospitalisation souhaitées ; - Minimiser le *makespan*.	- Durée d'ouverture des salles d'opération - Nombre d'heures disponibles des chirurgiens - Date limite des interventions - Nombre des lits d'hospitalisation disponibles	- Méthode de décomposition - Extension de la méthode Hongroise - Modèle de programmation linéaire résolu par LINGO - Heuristiques existantes pour les	Une semaine/ 90 interventions/ 3 salles d'opération/ 3 lits de réveil/ 60 lits

			problèmes de « flow shop » hybride sans temps d'attente entre les salles d'opération et la salle de réveil	d'hospitalisation.
[Jebali, 2004] (Chapitre 3 et 4)	- Minimiser le nombre des jours d'attente des patients - Minimiser le nombre des heures supplémentaires des salles d'opération	- Durée d'ouverture des salles d'opération - Disponibilité des chirurgiens - Disponibilité des équipements chirurgicaux - Disponibilité des lits d'hospitalisation et des lits de réanimation	- Modèle de programmation linéaire mixte résolu par CPLEX	Une semaine/ 60 interventions/ 3 salles d'opération/ 35 lits d'hospitalisation/ 8 lits de réanimation/ 4 lits de réveil/ 4 chirurgiens
[Jebali *et al.*, 2003, 2006]	- Minimiser le coût total des interventions	- Durée d'ouverture des salles d'opérations - Disponibilité des chirurgiens - Disponibilité des équipements chirurgicaux - Nombre de lits de réanimation - Nombre de lits de réveil - Date limite d'intervention	- Modèles de programmation linéaire mixte résolus par CPLEX	15 jours/ 15 interventions/ 3 salles d'opération/ 4 lits de réveil/ 4 chirurgiens.

Tableau 3 : Etat de l'art sur l'optimisation de programme opératoire

Par rapport aux études concernant le problème de planification des interventions [Fei *et al.*, 2005a,2005b][Gerchak *et al.*, 1996][Guinet et Chaabane, 2003][Kuo *et al.*, 2003][Lovejoy et Li, 2002][Marcon *et al.*, 2001a, 2001b][Ozkarahan, 1995, 2000][Ogulata et Erol, 2003], seulement une petite partie de travaux a été effectuée sur le problème de l'ordonnancement des interventions dans les salles opératoires ([Dexter *et al.*, 1999b], [Dexter *et al.*, 1999c], [Dexter *et al.*, 1999e], [Dexter *et al.*, 2000b], [Dexter et Traub, 2002], [Fei *et al.*, 2004]). De plus, un nombre encore plus restreint s'est intéressé à la planification suivie de l'ordonnancement dans le bloc opératoire (salles d'opérations + SSPI) [Chaabane, 2004(chapitre 6)][Jebali *et al.*, 2003, 2004 (chapitre 4), 2006].

Concernant les techniques utilisées pour la résolution du problème de planification, nous trouvons que presque tous ces problèmes sont décrits tout d'abord par un modèle mathématique et ensuite résolus par un logiciel existant (par exemple, EXCEL, CPLEX, LINGO etc.). Les méthodes les plus couramment utilisées sont les méthodes d'énumération, telles que la procédure de Branch-and-Bound, la procédure de Branch-and-Price, ou des procédures heuristiques comme par exemple, la procédure heuristique basée sur la technique de recherche du couplage maximal, sur la méthode Hongroise, sur la méthode de décomposition comme la méthode de génération de colonnes, etc..

Cependant, les problèmes d'ordonnancement sont résolus dans leur majorité par des règles de priorité, autrement dit des heuristiques simples (par exemple, « premier arrivé premier servi », « Best Fit», « Best Fit Descending » etc.), des modèles de simulation ou encore des procédures d'énumération. Seul un faible nombre de chercheurs utilisent des procédures sophistiquées.

2.4 Apport de notre recherche

Dans la section précédente, nous avons montré un état de l'art concernant la programmation opératoire et surtout détaillé la littérature récente sur l'optimisation de la programmation opératoire qui est la partie qui nous intéresse dans la gestion des blocs opératoires.

Dans le domaine de la planification des blocs opératoires, les modèles de programmation mathématique à buts multiples ("Goal Programming ») en nombres entiers, comme les travaux de Ozkarahan et Ogulata, combinent plusieurs objectifs entre eux et essaient de trouver un compromis parmi eux. La programmation mathématique à buts multiples est une bonne méthode car elle vise à trouver une solution respectant les préférences du décideur sur les différents objectifs. Cependant, tous les résultats existants dans ce domaine n'ont été obtenus que pour des problèmes de petite taille et de plus avec la seule prise en compte des salles d'opérations. Les modèles de programmation linéaire en nombres entiers ont été appliqués pour traiter des problèmes mono-objectifs de planification des blocs opératoires, comme les travaux de Kharraja, Marcon, Chaabane, Jebali, etc. La taille des problèmes modélisés par programmation linéaire en nombres entiers dépend de la méthode de résolution : la méthode exacte ne peut être appliquée qu'à des problèmes de petite taille pour trouver la solution optimale tandis que les heuristiques permettent de résoudre des problèmes de taille relativement grande pour obtenir des solutions approchées. Pour l'instant, les recherches utilisant la programmation linéaire en nombres entiers ne permettent de traiter que des problèmes de petite taille et plus il y a de contraintes à prendre en compte plus la taille du problème pouvant être traitée sera réduite. En conséquence, il faut trouver des techniques pour résoudre les problèmes de taille plus importante et pouvant tenir compte de nombreuses contraintes.

Concernant le problème d'ordonnancement, nous voyons que les recherches menées jusqu'à présent ne permettent de résoudre que des cas très simplifiés. Soit elles ne prennent pas en compte les principales contraintes comme la capacité de la salle de réveil ou la disponibilité des ressources (comme les chirurgiens, les équipements chirurgicaux, etc.). Soit elles émettent des hypothèses simplificatrices comme l'absence de temps d'attente entre les salles d'opération et la salle de réveil [Chaabane, 2004] ou absence de pénalité si les patients sont bloqués dans les salles d'opération après leur intervention [Kharraja, 2003][Kharraja et al., 2002] ou encore qu'une intervention peut être traitée par n'importe quel chirurgien disponible [Jebali et al., 2003, 2006]. Par conséquent, nous pouvons constater que le domaine d'ordonnancement des blocs opératoires, même pour les problèmes déterministes, est loin d'être exploité jusqu'à maintenant et qu'il y a donc une nécessité de trouver des méthodes efficaces pour construire un programme opératoire réalisable et de bonne qualité.

De plus, nous trouvons qu'il y a très peu de travaux concentrés sur les problèmes d'élaboration d'un programme opératoire pour plusieurs jours (normalement une semaine). Presque tous les travaux trouvés utilisent la méthode de décomposition (comme les travaux de Chaabane et Jebali) en décomposant le problème concerné en deux étapes : la planification et l'ordonnancement. Les méthodes de planification sont appliquées pour le problème de la première étape et celles d'ordonnancement sont appliquées pour le problème de la deuxième étape où les données sont obtenues par l'étape de planification.

Pour pallier non seulement à l'insuffisance des travaux portant sur l'élaboration d'un programme opératoire dans un bloc opératoire et aussi à l'absence de résultats remarquables sur des problèmes de taille réelle, nous visons dans cette thèse à construire un programme opératoire efficace. Nous prenons en compte des contraintes critiques, comme la durée d'ouverture des salles d'opération, la disponibilité des chirurgiens, la capacité limitée de la SSPI, tout en cherchant la satisfaction des patients (en respectant la date limite de leur intervention). Les objectifs sont la maximisation d'utilisation des salles d'opération et la minimisation du coût d'opération du bloc opératoire.

Comme c'est un problème extrêmement compliqué, nous utilisons aussi la méthode de décomposition. Notre problème de planification vise à affecter les interventions aux salles d'opération, en respectant la date limite des interventions, la durée d'ouverture des salles d'opération et la disponibilité des chirurgiens, avec comme objectifs la maximisation d'utilisation des salles d'opération (minimisation du coût des heures sous-utilisées) et de la minimisation du coût des heures supplémentaires. Notre problème d'ordonnancement vise à déterminer l'ordre des interventions aux salles d'opération, en prenant en compte la disponibilité des chirurgiens et la capacité limitée de la SSPI, avec comme objectif la minimisation du coût d'opération du bloc opératoire. De plus, nous allons résoudre des problèmes de taille relativement importante en nous basant sur le nombre moyen de salles d'opération et le nombre moyen d'interventions à traiter dans un hôpital européen de moyenne importance.

De plus, étant donné d'une part que le domaine de la gestion de production est quant à lui très bien développé et d'autre part qu'un rapprochement peut être effectué entre les problèmes rencontrés en gestion du bloc opératoire et ceux rencontrés (et résolus) en gestion industrielle, nous allons pouvoir emprunter quelques idées des techniques de planification et l'ordonnancement du système industriel.

Commençons tout d'abord par expliquer quelques notions de la gestion de production.

3. Gestion de production

La production est une transformation de ressources appartenant à un système productif et conduisant à la création de biens ou de services. La gestion de production a pour objet la recherche d'une organisation efficace de la production de biens et de services [Giard, 2003].

3.1 Description de la gestion de production

Un système de production est un processus d'addition de valeur à des biens ou à des services répondant à des impératifs de qualité, de prix, de quantité et de délai. Dans un système de production, la partie la plus importante est le processus de production, qui est formé par un ensemble d'actions élémentaires telles que la production proprement dite, le transport, le stockage, etc.

En général, les principaux objectifs d'une gestion de production efficace peuvent être considérés de la manière suivante :

1. Respecter les délais de commandes : il ne fait aucun doute que cet objectif est prioritaire vis à vis des pénalités de retard. Ainsi, il est nécessaire pour une entreprise de fournir un bien aux clients avec un retard le plus faible possible ;
2. Minimiser les stocks intermédiaires pour réduire les coûts de fonctionnement de l'entreprise, surtout le coût de stockage ;
3. Optimiser l'utilisation des moyens : il faut, en particulier, minimiser la charge des moyens les plus coûteux et détecter la présence de goulots d'étranglement limitant le débit de sortie des produits ;
4. Améliorer l'efficacité du système logistique et ainsi accélérer le processus de production.

Parmi les différents sous-systèmes de la gestion de production, deux nous intéressent plus particulièrement, à savoir la planification de production et la fonction d'ordonnancement.

3.2 Planification de production

Avec une définition généralisée, la planification de production se compose de trois niveaux de décision : le niveau stratégique, le niveau tactique et le niveau opérationnel, pour décider « Qui, Quoi et Comment produire ? » c'est-à-dire quel type de produit il faut produire, déterminer la combinaison optimale des ressources en entrée, comment affecter et ordonnancer les tâches aux ateliers.

Notons que la définition de la « planification de production » ne consiste qu'à trouver l'affectation optimale des tâches aux ateliers. Pour le problème de niveau opérationnel, nous le définissons comme « l'ordonnancement de production » et allons le décrire dans la partie suivante.

Comme nous l'avons dit, la planification de production vise à affecter un ensemble des tâches aux centres de production tout en respectant une ou plusieurs contraintes des ressources importantes. L'objectif de ce problème est normalement de minimiser le coût total de la fabrication sans se préoccuper comment et en particulier dans quel ordre ces tâches sont effectuées par ces centres de productivité. Cette dernière fonction est remplie par le niveau d'ordonnancement qui s'occupe d'un planning plus détaillé des produits [Hillier et Lieberman, 1980][Ross et Soland, 1975].

Comme le problème d'affectation des tâches est un problème classique NP-Difficile qui implique de nombreuses variables de décision, il est normalement résolu par les méthodes

d'énumération [Ross et Soland, 1975], la programmation dynamique [Hoesel *et al.*, 1994] ou les procédures heuristiques et méta-heuristiques [leBlanc *et al.*, 1999] etc..

3.3 Ordonnancement de production

Dès que nous affectons les tâches aux ateliers pour résoudre le problème de planification, un problème d'ordonnancement de production devra aussi être résolu pour définir où et à quel moment précis un certain nombre de tâches, correspondant aux fabrications des objets ou aux fournitures des prestations de service, doivent être réalisées dans les centres de production. Chaque tâche se décompose en un certain nombre d'opérations dans un centre de production, étant aussi bien une usine, un département de production, un atelier, un groupe de machines ou une machine.

L'organisation en ateliers spécialisés est le plus répandu des modes d'organisation des systèmes de production. C'est sans doute aussi celui qui pose les problèmes les plus compliqués en terme de résolution, en raison du nombre élevé de variables de décision, de la nature discrète de ces variables et de la prise en compte des contraintes de disponibilité de ressources qui conduit à la résolution de problèmes combinatoires de grande taille et implique l'utilisation d'heuristiques. Généralement, les problèmes d'ordonnancement de production dans les ateliers spécialisés peuvent être répartis en cinq grandes catégories (sans perte de généralité, nous remplaçons « centre de production » par « machine ») avec un ordre de complexité croissant :

1. Problèmes à une seule machine : pour ces problèmes, il n'y a qu'un centre de production. La littérature qui s'intéresse à ces problèmes est particulièrement riche en raison de leur simplicité relative, et surtout parce que les résultats obtenus par l'étude de ces problèmes servent de base pour résoudre des problèmes plus complexes [Gupta et Kyparisis, 1987] ;
2. Problèmes à machines parallèles : pour ces problèmes, toutes les tâches sont opérées sur n'importe quelle machine d'un ensemble de machines multifonctionnelles ;
3. Flow-shop : les tâches sont opérées par un sous-ensemble de machines. De plus, l'ordre de passage des tâches sur un même ensemble de machines utilisées est toujours le même pour toutes les tâches (Figure 7) ;
4. Job-shop : contrairement au système de flow-shop, les opérations élémentaires sont liées par un ordre total, non nécessairement identique pour toutes les tâches. Le cheminement des produits dans l'atelier est spécifique à chaque article et des produits différents en cours d'usinage devront vraisemblablement se croiser (un modèle de 5 machines et 2 produits est montré à la Figure 8) ;
5. Open-shop : dans ces problèmes, toutes les tâches sont opérées dans l'atelier avec des chemins libres, autrement dit, il n'y a pas de contraintes de précédence.

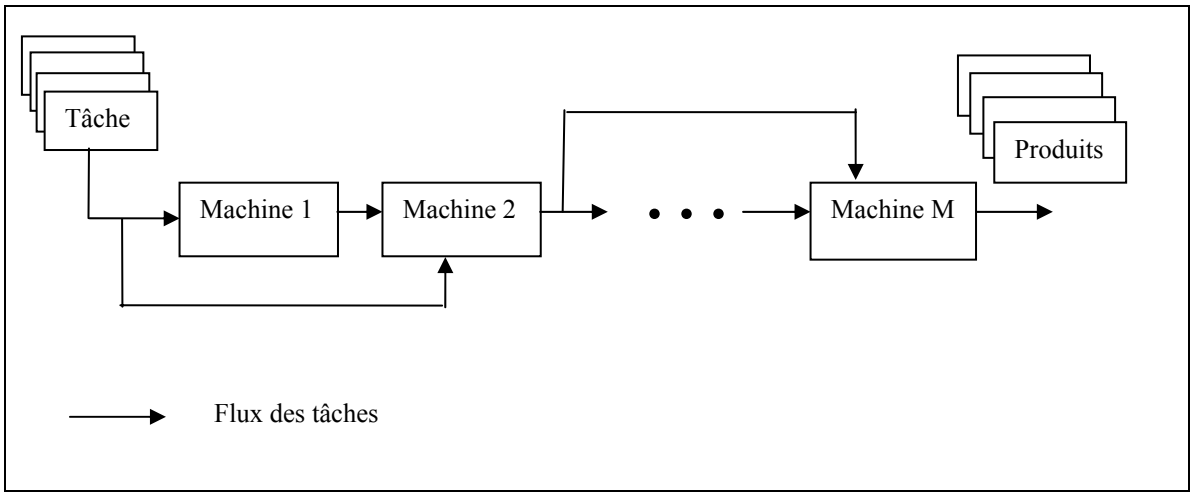

Figure 7 : Flux des tâches dans un modèle « flow Shop »

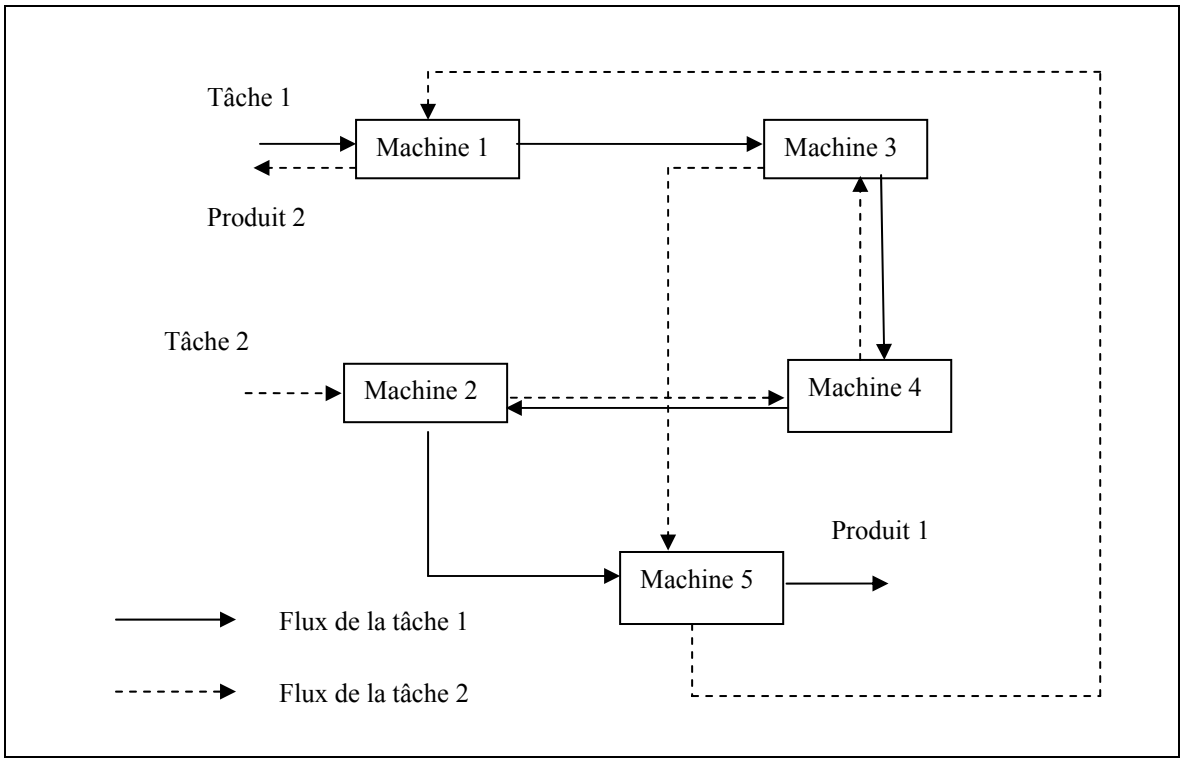

Figure 8 : **Flux des tâches dans un modèle « job Shop »**

Greene et Sadowski [1984], Cheng et Gupta [1989], Allahverdi *et al.* [1999], Jain et Meeran [1999] ainsi que Linn et Zhang [1999] présentent et détaillent les avancements de recherche pour ces problèmes d'ordonnancement.

4. Rapprochement entre la gestion des blocs opératoires et la gestion de production

Nous allons maintenant étudier les deux systèmes afin de pouvoir bénéficier des recherches déjà menées et éprouvées dans le cadre de la gestion de production et les adapter à la gestion des blocs opératoires. Les Tableau 4 à 8 nous montrent les analogies possibles et les différences entre ces deux types de systèmes du point de vue des

composants du système (Tableau 4), des objectifs (Tableau 5), des contraintes (Tableau 6), de la planification (Tableau 7) et de l'ordonnancement (Tableau 8).

Composants du système	Gestion des blocs opératoires	Gestion de production
Analogies	- Salles d'opération et les lits de réveil dans le bloc opératoire; - Equipes chirurgicales ; - Brancards ; - Patients devant être opérés ; - …	- Machines ou centres de production dans l'atelier ; - Techniciens ; - Convoyeurs ou véhicules ; - Tâches pour produire les produits ; - …
Différences	D'une part, la différence entre ces deux systèmes se situe sur leurs « produits ». Par rapport au système de production où les processus pour fabriquer les produits sont normalement déterministes, dans le système hospitalier, des services sont offerts aux patients dont l'état de santé varie et qui ont aussi leurs préférences. De plus, il y a plus fréquemment des cas d'urgences dans l'hôpital que dans les systèmes de production. D'autre part, à la différence du système de production où une machine (ou un centre de production) est normalement occupée par un technicien, les équipes chirurgicales se composent de membres de différentes spécialités (chirurgiens, infirmières, anesthésistes etc.) donc il faut coordonner les activités.	

Tableau 4 : Comparaison entre la gestion des blocs opératoires et la gestion de production du point de vue des composants

Principaux Objectifs	La gestion des blocs opératoires	La gestion de production
Analogies	- Minimiser le coût de stockage de médicaments - Minimiser le temps de passage des patients dans les blocs opératoires, en respectant la qualité du traitement d'intervention; - Minimiser la sous-utilisation des salles d'opération; - Minimiser le nombre d'heures supplémentaires de salle d'opération ; - …	- Minimiser le coût de stockage des ressources matérielles ; - Minimiser le temps de passages des produits dans l'atelier ; - Maximiser l'utilisation des moyens chers; - …

Différences	L'objectif d'un système hospitalier consiste principalement à minimiser ses coûts tandis que dans les systèmes de production c'est surtout une recherche d'une maximisation des bénéfices qui est réalisée.

Tableau 5 : Comparaison entre la gestion des blocs opératoires et la gestion de production du point de vue des objectifs

Principales Contraintes	La gestion des blocs opératoires	La gestion de production
Analogies	- Précédence des opérations dans les blocs opératoires ; - Capacité des salles d'opération et de la SSPI ; - Disponibilité des ressources humaines et matérielles nécessaires (les chirurgiens, les médecins anesthésistes ; les équipements chirurgicaux etc.); - Etat de santé des patients, comme la date limite (date au plus tard de la réalisation de l'opération) ; - Respecter les budgets fixés ; - …	- Précédence des opérations dans l'atelier; - Capacité des ateliers ; - Disponibilité des ressources humaines et matérielles nécessaires (les techniciens, les outils chers etc.); - Respect des délais des commandes, respect d'un niveau de qualité etc. ; - Respecter les budgets fixés ; - ...
Différences	Bien que les entreprises doivent respecter certaines contraintes comme les délais des commandes, la qualité du produit etc., il est possible de ne pas les respecter moyennant des pénalités. Cependant ce n'est pas possible dans les hôpitaux où certaines contraintes (comme la date limite de l'intervention) portant sur le patient doivent impérativement être respectées. Autrement dit, la satisfaction des clients dans les hôpitaux est une contrainte inviolable. D'autre part, le problème d'affectation des ressources du système hospitalier est plus complexe que celui du système de production car une équipe chirurgicale se compose de plusieurs personnes de spécialités différentes qu'il faut coordonner.	

Tableau 6 : Comparaison entre la gestion des blocs opératoires et la gestion de production du point de vue des contraintes

Planification	La gestion des blocs opératoires	La gestion de production
Analogies	- Approvisionnement, réception et stockage des produits médicaux ; - Affectation des patients aux lits d'hospitalisation ; - Affectation des équipes chirurgicales et ressources matérielles aux interventions ; - Affectation des interventions aux salles d'opération (ou plages horaires) pendant une période en respectant les contraintes critiques.	- Approvisionnement, réception et stockage des matières premières ; - Stockage des produits finaux aux stocks ; - Affectation des techniciens aux machines et des machines aux produits ; - Affectation des tâches aux ateliers pendant une période.
Différences	Les durées de production dans un système de production sont généralement connues avec précision. Par contre, c'est loin d'être le cas dans le système hospitalier où la durée opératoire dépend de nombreux paramètres (âge du patient, expérience du chirurgien, …)	

Tableau 7 : Comparaison entre la gestion des blocs opératoires et la gestion de production du point de vue de la planification

Ordonnancement	La gestion des blocs opératoires	La gestion de production
Analogies	- Flux des patients dans les blocs opératoires (salles d'opération + SSPI)	- Flux des tâches dans un modèle de « flow shop » à deux étages
Différences	Dans le système de production, normalement il y a du stockage intermédiaire entre deux étages quand il n'y a plus de machines disponibles dans le deuxième étage. Par contre, les patients dans un bloc opératoire doivent rester dans les salles d'opération où ils ont été opérés si aucun lit de réveil n'est disponible lorsque leur intervention est finie. Le patient commence à se réveiller dans la salle d'opération.	

Tableau 8 : Comparaison entre la gestion des blocs opératoires et la gestion de production du point de vue de l'ordonnancement

De ces comparaisons entre la gestion des blocs opératoires et la gestion de production, nous pouvons conclure que :

- L'optimisation de la programmation opératoire est apparentée à l'optimisation de la planification de production si nous considerons les interventions chirurgicales comme des tâches; les blocs opératoires comme des centres de service. La principale différence vient du fait que nous devons respecter la satisfaction des patients et quelques fois, nous devons aussi respecter les demandes des acteurs,

surtout des chirurgiens alors que ceci n'est pas un grand problème pour les entreprises où les opérateurs présent une certaine polyvalence ;

- L'ordonnancement de la programmation opératoire s'apparente à un modèle de « flow shop » à deux étages. Comme nous l'avons dit, un bloc opératoire se compose de deux parties : les salles d'opération et la SSPI où il y a plusieurs lits de réveil. Normalement, les patients vont d'abord être opérés dans les salles d'opération et ensuite être transférés en SSPI. Cet ordre est unique lorsque le patient est opéré sous anesthésie. Cette contrainte de précédence d'opération est la même que le modèle de « flow shop ». Ce qui est particulier à notre modèle est que le patient va commencer à se réveiller dans la salle d'opération s'il n'y a pas de lit disponible dans la SSPI et donc le temps dans la SSPI va être réduit. Par contre, dans le système industriel les tâches sont seulement bloquées.

Parce que les techniques et les méthodes ont été suffisamment développées dans le domaine de la planification et de l'ordonnancement de production, nous allons appliquer, en les adaptant, les méthodes révélées efficaces en gestion de production, comme les procédures heuristiques et méta-heuristiques, pour résoudre nos problèmes hospitaliers.

5. Conclusion

Dans la littérature, beaucoup de chercheurs se concentrent sur l'évaluation de la performance des programmes opératoires existants ainsi que sur leurs améliorations. Pour les études d'optimisation, presque tous visent à minimiser l'utilisation des salles d'opération ou le coût d'opération relatif aux temps supplémentaires. Pour résoudre ces modèles, quelques chercheurs utilisent les outils de simulation, d'autres proposent des méthodes heuristiques et méta-heuristiques ainsi que des procédures sophistiquées. De plus, ces modèles mathématiques ont été résolus par les outils déjà développés comme CPLEX, LINGO ou EXCEL.

Bien que le bloc opératoire se compose des salles d'opération et la SSPI, la plupart des chercheurs ne prennent en compte que les contraintes liées aux salles d'opération sans tenir compte des contraintes liées à la SSPI (exceptés [Kharraja *et al.*, 2002], [Jebali *et al.*, 2003, 2006] et [Chaabane , 2004]). De plus, aucun d'entre eux ne prend en considération la date limite de réalisation des interventions.

Nous allons donc nous concentrer sur l'optimisation de la programmation opératoire afin de construire les programmes opératoires réalisables et de bonne qualité, en respectant les contraintes nécessaires.

Partie III :

Planification des blocs opératoires

1. Introduction

Comme nous l'avons signalé auparavant, les programmes opératoires actuels dans les hôpitaux sont souvent les résultats d'une négociation entre les acteurs concernés et le responsable du bloc opératoire. Dans la littérature, la majorité des travaux traitent de l'évaluation et de l'amélioration des programmes opératoires existants, sans se préoccuper de concevoir de nouveaux programmes opératoires qui permettraient une meilleure gestion des blocs opératoires. De plus, même pour les travaux relatifs à l'optimisation des fonctionnements des blocs opératoires, la plupart d'entre eux visent à minimiser le coût relatif aux durées d'opération ou au taux d'utilisation des salles d'opération, en respectant la capacité des blocs opératoires mais peu prennent en compte les demandes des patients (par exemple la date limite des interventions).

D'autre part, du fait de la complexité de la construction des programmes opératoires, il est pratique de le décomposer en deux sous-problèmes séquentiels : la planification et l'ordonnancement des interventions.

Dans cette partie, nous nous concentrons sur la première étape : la planification des interventions dans les blocs opératoires, visant à créer un planning global des interventions pour la période de planification (une semaine). Nous traiterons deux modèles : un modèle de planification pour une stratégie du « block scheduling » et un modèle de planification pour une stratégie d' « open scheduling ». Les deux modèles sont résolus premièrement par les procédures heuristiques et ensuite par une procédure de « Branch and Price ». Ces deux procédures sont basées sur la procédure de génération de colonnes. Ces méthodes seront décrites dans cette partie. Les deux modèles ont été testés sur des données générées aléatoirement et les résultats expérimentaux sont présentés dans la partie V. Nous avons aussi testé la procédure heuristique sur un service d'endoscopie (partie VI).

2. Objectifs de la planification du bloc opératoire

Hamilton et Breslawki [1994] ont présenté dix éléments, les plus importants, devant être pris en compte dans la construction d'un programme opératoire. Neuf d'entre eux font partie des contraintes du problème générique de la planification du bloc opératoire :

1. Le nombre de salles d'opération disponibles chaque jour : il dépend de la taille de l'hôpital, de la disponibilité des infirmières et d'autres facteurs ;
2. Le nombre d'équipements chirurgicaux spécialisés : beaucoup d'opérations ont besoin d'équipement spécialisé et il est de la responsabilité du responsable de bloc d'assurer la disponibilité de l'équipement demandé ;
3. Les plages horaires disponibles réservées aux chirurgiens : pour une intervention acceptée, il doit y avoir assez de plages horaires libres pour l'opérer ;
4. Les politiques de la programmation opératoire dans un hôpital : Celles-ci peuvent différer d'un hôpital à l'autre. Par exemple, certains souhaiteront réserver les

plages horaires pour les chirurgiens à l'avance, alors que d'autres peuvent adopter l'approche d'organiser les plages horaires selon l'ordre des demandes ;

5. Les durées d'opération estimées : pour un chirurgien, les durées opératoires des interventions sont variées, même s'il opère la même sorte d'intervention. La durée dépend entre autres du profil du patient (âge, antécédents, etc.) et dépend des caractéristiques du chirurgien (expérience, etc.). Une des principales difficultés est donc d'arriver à estimer correctement les durées pour éviter la sous ou sur utilisation des blocs opératoires ;
6. Les temps d'ouverture et fermeture réguliers des blocs opératoires : les blocs opératoires ont normalement des programmes opératoires hebdomadaires ou mensuels avec des plages horaires disponibles. Les programmes opératoires devront respecter le temps d'ouverture et fermeture régulier des blocs opératoires car sinon il y a des coûts de pénalité ;
7. La disponibilité des anesthésistes : dans de nombreux hôpitaux commence à se faire sentir un manque d'anesthésistes. Donc les programmes opératoires doivent aussi respecter cette contrainte ;
8. La limitation des salles liées aux équipements chirurgicaux : les équipements sont habituellement assignés aux salles d'opération qui les réservent. Mais, quelques équipements chirurgicaux font partie intégrante de la salle d'opération et ne peuvent donc pas être déplacés ;
9. L'heure de départ assignée aux chirurgiens : les chirurgiens sont habituellement assignés à une heure d'arrivée pour commencer leur opération dans une salle d'opération. Cependant, s'il y a un changement de chirurgien dans une salle d'opération, l'heure d'arrivée du chirurgien suivant dépend du chirurgien précédent, ayant accompli toutes ses interventions.

Quant aux objectifs, ils dépendent des intérêts des gestionnaires des hôpitaux. Normalement, les hôpitaux publics sont financés par l'Etat et donc leurs gestionnaires doivent organiser le fonctionnement en respectant un budget fixé, mais les gestionnaires des hôpitaux privés s'intéressent peut-être plus à maximiser les bénéfices de l'établissement de soins. De toute façon, tous les hôpitaux devront prendre en compte la satisfaction des patients.

Ici, nous présentons deux objectifs, les plus souvent pris en compte dans les modèles de la programmation opératoire pour les hôpitaux publics : la maximisation de l'utilisation des salles d'opération et la minimisation des coûts opératoires.

2.1 Maximisation de l'utilisation des salles d'opération

Le taux d'utilisation des salles d'opération est un des indicateurs les plus utilisés pour évaluer notamment l'efficacité des salles d'opération [Weinbroum *et al.*, 2003]. Ce taux peut se calculer comme suit [Donham *et al.*, 1996] [Dexter *et al.*, 1999b, 1999c] :

$$\%\ \text{utilisation} = \frac{\text{Les heures ordinaires utilisées + les heures supplémentaires}}{\text{La durée d'ouverture ordinaire de cette salle d'opération}} \times 100$$

D'autre part, Strum et ses collègues [Strum *et al.*, 1997, 1999] définissent quatre modes d'utilisation :

- Sous-utilisation lorsque la durée d'opération totale du programme opératoire affecté est moindre que la plage horaire définie ;
- Sur-utilisation si la durée d'opération totale d'un programme opératoire dépasse la durée de la plage horaire ordinaire réservée à l'avance. Le coût d'une heure supplémentaire est normalement plus élevé que celui d'une heure ordinaire ;
- Utilisation idéale : un phénomène rare dans l'expérience hospitalière, où un programme opératoire utilise la même durée d'une plage horaire que celle réservée ;
- Sur- et sous-utilisation : il y a des créneaux libres entre les interventions assignées à un programme opératoire pendant la durée ordinaire réservée et il y a aussi des interventions réalisées durant les heures supplémentaires.

2.2 Minimisation du coût total des interventions

La minimisation du coût total des interventions est un objectif souvent pris en compte par les gestionnaires et les chercheurs [Ogulata et Erol, 2003][Ozkarahan, 1995,2000] [Jebali *et al.*, 2003,2006][Guinet et Chaabane, 2003].

Le coût total des interventions est déterminé par de nombreux facteurs comme par exemple le coût pré-opératoire, les salaires de l'équipe chirurgicale et les assistants (les brancardiers, les secrétaires etc.), la dépense des équipements chirurgicaux et d'autres matériels relatifs [Viapiano, 1999]. En effet, la minimisation des coûts totaux des interventions des blocs opératoires est un critère important pour réduire les dépenses d'un établissement hospitalier. Les gestionnaires des établissements hospitaliers tentent d'améliorer l'efficacité des blocs opératoires afin de réduire les coûts et d'accroître la productivité.

Une manière importante de réduire les coûts opératoires est de minimiser la sous-utilisation ainsi que la sur-utilisation des blocs opératoires pour une meilleure efficacité. Pour nous, l'objectif de nos modèles de programmation opératoire sera de minimiser les coûts relatifs aux heures de sous-utilisation et aux heures supplémentaires.

3. Modèles de planification

3.1 Modèle de planification pour la stratégie du « block scheduling »

Nous rappelons que la stratégie du « block scheduling » se compose de deux sous-problèmes : la confection de plages horaires par chirurgien et l'allocation des interventions aux plages horaires réservées à un chirurgien durant une période de planification.

Dans cette recherche, nous présumons que les plages horaires ont été définies préalablement (généralement annuellement). Nous nous concentrons donc sur le dernier problème – l'allocation des interventions aux plages horaires réservées à un chirurgien durant une semaine de la planification, en prenant en compte la date limite des interventions, avec comme objectif de minimiser le coût de sous-utilisation et des heures supplémentaires.

Du fait de la difficulté du problème, nous le simplifions en posant les hypothèses suivantes :

1. Les plages horaires affectées à un chirurgien sont fixées par la direction hospitalière et ne peuvent être modifiées jusqu'au redéploiement de ces dernières ;

2. Les salles d'opération sont considérées comme étant multifonctionnelles. Cette hypothèse n'a été introduite que dans le but de simplifier les notations. Cependant, notre modèle peut aussi être appliqué au cas où les salles d'opération sont spécialisées, moyennant de légères modifications. Il suffit en fait d'ajouter un indicateur pour chaque patient qui indique la salle d'opération où il peut être opéré ;
3. Les lits dans la SSPI sont toujours disponibles pour les patients qui viennent d'être opérés dans les salles d'opération donc nous ne prenons pas en considération la contrainte concernant le nombre de lits dans ce modèle ;
4. Les autres ressources humaines et matérielles nécessaires sont toujours disponibles pour le bon déroulement des interventions réalisées par le chirurgien ;
5. Le chirurgien travaille cinq jours par semaine ;
6. Les cas urgents ne sont pas pris en compte et nous supposons que dès qu'une intervention a débuté, cette salle ne peut pas être affectée aux autres interventions, c'est-à-dire, il n'y a ni urgence et ni préemption. Cette hypothèse n'est pas restrictive dans la mesure où nous cherchons à établir un planning global prévisionnel dans lequel nous ne pouvons pas anticiper tous les événements aléatoires possibles du fait de l'horizon de planification. Le planning global ainsi obtenu ne sera qu'une indication pour la réalisation. Il doit être remis à jour lorsque de nouvelles données telles que les urgences sont disponibles.

De par les hypothèses émises, nous pouvons décrire le problème concerné sous forme d'une programmation en nombres entiers.

Notation :

N_{interv} : le nombre d'interventions à réaliser par le chirurgien considéré ;
Ω : l'ensemble des interventions en attente, $\Omega = \{1, ..., N_{interv}\}$;
t_i : la durée opératoire estimée de l'intervention i ;
D_i : la date limite de l'intervention i ;
N_{jour} : la période de la planification exprimée en nombre de jours ;
$\overline{\Omega}$: l'ensemble des interventions en attente dont la date limite est inférieure ou égale à N_{jour},
$\overline{\Omega} = \{i | D_i \le N_{jour}, i \in \Omega\}$, donc $\Omega \setminus \overline{\Omega}$ représente l'ensemble des interventions en attente qui pourraient être réalisées au-delà de la période de la planification ;
NB_d : le nombre de plages horaires disponibles le jour d pour le chirurgien en question ;
TOR_k^d : le nombre d'heures normales disponibles de la $k^{ème}$ plage horaire réservée du jour d ;
TS_k^d : le nombre maximal d'heures supplémentaires possibles de la $k^{ème}$ plage horaire réservée du jour d ;
β : le rapport entre le coût d'une heure supplémentaire et le coût d'une heure de sous-utilisation ;
C_k^d : le coût d'une plage horaire réservée k le jour d (= coût des heures supplémentaires ou coût de la sous-utilisation de cette plage horaire).

Variable de décision:

z_{ik}^d = 1 si l'intervention i est planifiée dans la plage horaire k le jour d ; 0 sinon.

Modèle Mathématique du problème de « Block Scheduling » (MMBS):

$$\text{Min} \sum_{d=1}^{N_{jour}} \sum_{k=1}^{NB_d} C_k^d = \sum_{d=1}^{N_{jour}} \sum_{k=1}^{NB_d} \max\left\{ (TOR_k^d - \sum_{i\in\Omega} t_i z_{ik}^d), \beta(\sum_{i\in\Omega} t_i z_{ik}^d - TOR_k^d) \right\} \quad (3.1.1)$$

sous les contraintes

$$\sum_{d=1}^{D_i} \sum_{k=1}^{NB_d} z_{ik}^d = 1 \qquad i \in \overline{\Omega} \quad (3.1.2)$$

$$\sum_{d=1}^{N_{jour}} \sum_{k=1}^{NB_d} z_{ik}^d \leq 1 \qquad i \in \Omega \quad (3.1.3)$$

$$\sum_{i\in\Omega} t_i z_{ik}^d \leq TOR_k^d + TS_k^d \qquad k \in \{1,..., NB_d\} \text{ et } d \in \{1,..., N_{jour}\} \quad (3.1.4)$$

$$z_{ik}^d \in \{0,1\} \quad i \in \Omega, d \in \{1,...,N_{jour}\}, k \in \{1,...,NB_d\} \quad (3.1.5)$$

La fonction objectif vise la minimisation du coût total d'utilisation des plages horaires sur la période de planification. Le coût d'une plage horaire, représenté par la formule (3.1.1), représente soit le coût du nombre d'heures sous-utilisées soit le coût du nombre d'heures supplémentaires.

La contrainte (3.1.2) impose que toute intervention dont la date limite se situe dans la semaine, soit réalisée une et une seule fois avant sa date limite.

La contrainte (3.1.3) impose que toute intervention ne peut être prise en compte qu'au plus une fois.

La contrainte (3.1.4) implique que la durée opératoire totale des interventions affectées à une plage horaire ne peut pas excéder la durée maximale possible de la plage horaire concernée (le nombre d'heures normales + le nombre d'heures supplémentaires possibles).

3.2 Modèle de planification pour la stratégie de l'« open scheduling »

À l'heure actuelle, dans beaucoup d'établissements, l'élaboration du programme opératoire avec une stratégie « open scheduling » se fait de manière chronologique suivant la règle du « premier arrivé premier servi » ou se construit par la négociation entre les acteurs [Patterson, 1996][Fogg, 2001][Kharraja, 2003]. Cependant, la plupart des travaux ayant été réalisés pour étudier la construction du programme opératoire avec une idée d'optimisation du fonctionnement des blocs opératoires ne prennent pas en compte la contrainte de disponibilité des chirurgiens sauf Jebali *et al.* [2006] qui supposent que les interventions peuvent être réalisées par n'importe quel chirurgien mais leur capacité d'opération est limitée.

Dans cette section, nous allons nous concentrer sur un problème d'« open scheduling » pour construire un programme opératoire avec toujours comme objectif la minimisation du coût des heures de sous-utilisation et des heures supplémentaires des salles d'opération, pour une semaine.

Comme le problème d'« open scheduling » est plus compliqué que celui de « block scheduling », nous prenons les mêmes hypothèses que pour ce dernier à l'exception de la première hypothèse qui ne nous concerne pas ici. Notons que les plages horaires dans le modèle de « block scheduling » doivent être remplacées ici par les salles d'opération ouvertes.

De plus, nous ajoutons d'autres notations :

Notation :

NS_d : le nombre des salles d'opération disponibles le jour d ;
N_{chir} : le nombre de chirurgiens pendant cette période de planification ;
TA_l^d : le nombre d'heures maximum disponibles du chirurgien l le jour d ;
ch_i : l'indice d'un chirurgien qui opère l'intervention i ;
Ω_l : l'ensemble des interventions à réaliser par le chirurgien l, $\Omega_l = \{i | ch_i \equiv l, i \in \Omega\}$. Bien entendu, nous avons $\bigcup_{l=1}^{N_{chir}} \Omega_l = \Omega$.

Variable de décision :

z_{ik}^d = 1 si l'intervention i est planifiée dans la salle d'opération k le jour d ; 0 sinon.

Modèle mathématique du problème de « Open Scheduling » (MMOS):

$$\text{Min} \sum_{d=1}^{N_{jour}} \sum_{k=1}^{NS_d} C_k^d = \sum_{d=1}^{N_{jour}} \sum_{k=1}^{NS_d} \max\left\{(TOR_k^d - \sum_{i\in\Omega} t_i z_{ik}^d), \beta(\sum_{i\in\Omega} t_i z_{ik}^d - TOR_k^d)\right\} \quad (3.2.1)$$

sous les contraintes

$$\sum_{d=1}^{D_i} \sum_{k=1}^{NS_d} z_{ik}^d = 1, \quad i \in \overline{\Omega} \quad (3.2.2)$$

$$\sum_{d=1}^{N_{jour}} \sum_{k=1}^{NS_d} z_{ik}^d \le 1, \quad i \in \Omega \quad (3.2.3)$$

$$\sum_{i\in\Omega} t_i z_{ik}^d \le TOR_k^d + TS_k^d, \quad k \in \{1,\cdots,NS_d\}, d \in \{1,\cdots,N_{jour}\} \quad (3.2.4)$$

$$\sum_{k=1}^{NS_d} \sum_{i\in\Omega_l} t_i z_{ik}^d \le TA_l^d, \quad d \in \{1,\cdots,N_{jour}\}, l \in \{1,\cdots,N_{chir}\} \quad (3.2.5)$$

$$z_{ik}^d \in \{0,1\} \quad i \in \Omega, k \in \{1,\cdots,NS_d\}, \ d \in \{1,\cdots,N_{jour}\} \quad (3.2.6)$$

Comme pour le modèle du « block scheduling », l'objectif de ce modèle vise la minimisation du coût total des heures de sous-utilisation et supplémentaires pour toutes les salles d'opération concernées pendant la période de planification.

Les contraintes (3.2.2) – (3.2.4) représentent les mêmes contraintes que (3.1.2) – (3.1.4), sauf que nous prenons en compte des salles d'opération plutôt que des plages horaires.

La contrainte (3.2.5) signifie que la durée opératoire totale des interventions affectées à un chirurgien pendant une journée ne peut pas excéder la capacité du chirurgien concerné.

Bien qu'un programme opératoire réalisable doive aussi prendre en compte les contraintes de capacité et s'assurer qu'un chirurgien n'opère pas deux interventions en même temps, nous ne les prenons pas en considération dans ce modèle de planification parce que toutes ces contraintes vont être prises en compte au niveau de l'ordonnancement pour une journée afin de construire finalement un programme opératoire réalisable et efficace.

4. Méthodes et techniques de résolution

Dans la section précédente, nous avons décrit les modèles mathématiques d'un problème de « block scheduling » et d'un problème d'« open scheduling ».

Dans cette section, nous allons présenter une méthode efficace pour les résoudre.

Tout d'abord, nous allons décomposer les modèles mathématiques MMBS et MMOS en deux parties : un problème principal, qui sera reformulé comme un problème de partitionnement, et un ensemble de problèmes dits « auxiliaires ». Ensuite des procédures heuristiques seront utilisées pour obtenir une solution réalisable et enfin une procédure de Branch-and-Price sera utilisée pour obtenir une solution exacte pour le cas de « block scheduling ».

4.1 Décomposition des modèles mathématiques

En appliquant l'idée de la décomposition de Dantzig-Wolfe (décrit en annexe 1), qui est prouvée comme étant une méthode efficace pour une grande catégorie de programmes linéaires en nombres entiers, nous décomposons les modèles mathématiques MMBS et MMOS en deux parties : un problème principal MP et un ensemble de problèmes auxiliaires *SPs*. De plus, le problème principal sera reformulé en tant que variante du problème de partitionnement avec des variables binaires afin de résoudre leur relaxation linéaire par la procédure de génération de colonnes. Les détails du problème principal (Master Problem, MP) sont décrits ci-dessous.

4.1.1 Problème principal MP du MMBS

En effet, notre problème peut être ramené à un problème de partitionnement puisqu'il consiste à décomposer l'ensemble des interventions en différents sous-ensembles, chaque sous-ensemble, sauf éventuellement un qui correspond aux interventions à ne pas réaliser dans la semaine considérée, étant affecté à une plage horaire réservée.

Puisque tout problème de partitionnement peut être modélisé en un programme linéaire en nombres entiers, notre problème peut l'être également. Cette modélisation suppose que nous puissions identifier tous les sous-ensembles possibles de l'ensemble des interventions.

Le MP du modèle mathématique MMBS reprend le problème MMBS mais reformulé à l'exception des contraintes (3.1.4). Les problèmes auxiliaires SP_k^d ($k = 1, \ldots, NB_d$, $d = 1, \ldots, N_{jour}$) visent à intégrer ces contraintes. Pour reformuler le MP en un problème de partitionnement, nous introduisons la notion de planning partiel (appelé simplement dans la suite « planning ») qui signifie une plage horaire réservée avec un ensemble d'interventions qui y sont affectées. Notons que la date limite des interventions ne doit pas dépasser le jour de la plage horaire. Nous reprenons les notations du modèle MMBS et ajoutons les autres notations suivantes :

Notations :

C_j : le coût du planning j (= coût des heures supplémentaires ou coût de la sous-utilisation de la plage horaire concernée) ;
N_{plan} : le nombre de plannings possibles pendant la durée de planification ;
Ξ : l'ensemble des plannings possibles pendant cette période de planification;
a_{ij} : = 1 si l'intervention i figure dans le planning j ; 0 sinon ;
b_j^d : = 1 si le planning j concerne une plage horaire du jour d ; 0 sinon ;
e_{kj} : = 1 si le planning j est réalisé dans la $k^{\text{ème}}$ plage horaire d'une journée; 0 sinon.

Variable de décision:

$$x_j = \begin{cases} 1 & \text{si le planning } j \text{ est retenu;} \\ 0 & \text{sinon;} \end{cases}$$

Le MP du MMBS peut être reformulé comme ci-dessous :

$$\text{Min} \quad \sum_{j\in\Xi} C_j x_j$$

sous les contraintes

$$\sum_{j\in\Xi} a_{ij} x_j = 1, \quad i \in \overline{\Omega} \tag{4.1.1}$$

$$\sum_{j\in\Xi} a_{ij} x_j \le 1, \quad i \in \Omega \setminus \overline{\Omega} \tag{4.1.2}$$

$$\sum_{j\in\Xi} b_j^d e_{kj} x_j \le 1 \quad k \in \{1,\cdots,NB_d\}, d \in \{1,\cdots,N_{jour}\} \tag{4.1.3}$$

$$x_j \in \{0,1\}, \quad j \in \Xi \tag{4.1.4}$$

La fonction objectif, comme celle du modèle MMBS, vise la minimisation du coût total des plannings retenus sur la période de planification.

Les contraintes (4.1.1) et (4.1.2) représentent les mêmes contraintes que (3.1.2) et (3.1.3). Notons que la contrainte (4.1.1) est une version compacte de la $\sum_{j\in\Xi}\left(\sum_{d=1}^{D_i}\sum_{k=1}^{NB_d} b_j^d e_{kj}\right) a_{ij} x_j = 1$ du fait que les manières de générer des plannings initiaux et de résoudre les problèmes auxiliaires (détaillé en section 4.2) peuvent garantir le fait que chaque intervention i dont la date limite est inférieure ou égale au dernier jour de l'horizon de planification doit être affectée à un planning j retenu avant sa date limite, autrement dit, $\sum_{d=1}^{D_i}\sum_{k=1}^{NB_d} b_j^d e_{kj} = 1$.

La contrainte (4.1.3) exige que chaque plage horaire réservée soit affectée à au plus un planning retenu.

4.1.2 Problème principal MP du MMOS

Semblable à la décomposition du MMBS, le MP du modèle mathématique MMOS reprend le problème MMOS à l'exception des contraintes (3.2.4). Les problèmes auxiliaires SP_k^d ($k = 1, \ldots, NS_d$, $d = 1, \ldots, N_{jour}$) visent à intégrer ces contraintes. Pour reformuler le MP sous forme de « set partition », nous reprenons les notations du modèle MMOS et celles du MP du MMBS en remplaçant une plage horaire par une salle d'opération:

Variable de décision:

$$x_j = \begin{cases} 1 & \text{si le planning } j \text{ est retenu;} \\ 0 & \text{sinon;} \end{cases}$$

Le MP du MMOS peut être reformulé comme ci-dessous :

$$\min \sum_{j \in \Xi} C_j x_j$$

Sous les contraintes

$$\sum_{j \in \Xi} a_{ij} x_j = 1, \quad i \in \overline{\Omega} \tag{4.1.5}$$

$$\sum_{j \in \Xi} a_{ij} x_j \le 1, \quad i \in \Omega \setminus \overline{\Omega} \tag{4.1.6}$$

$$\sum_{j \in \Xi} b_j^d e_{kj} x_j \le 1 \quad d \in \{1, \cdots, N_{jour}\}, k \in \{1, \cdots, NS_d\} \tag{4.1.7}$$

$$\sum_{j \in \Xi} b_j^d \left(\sum_{i \in \Omega_l} a_{ij} t_i\right) x_j \le TA_l^d, \quad d \in \{1, \cdots, N_{jour}\}, l \in \{1, \cdots, N_{chir}\} \tag{4.1.8}$$

$$x_j \in \{0,1\} \quad j \in \Xi \tag{4.1.9}$$

La fonction objectif, comme celle du modèle MMOS, vise à la minimisation du coût total des plannings retenus sur la période de planification.

Les contraintes (4.1.5), (4.1.6) et (4.1.8) représentent les mêmes contraintes que (3.2.2), (3.2.3) et (3.2.5) respectivement.

La contrainte (4.1.7) exige que chaque salle d'opération soit affectée à au plus un planning retenu.

4.2 Génération de colonnes (GC)

Examinant les problèmes principaux en formulation de « set partition » pour MMBS et MMOS, nous peuvons trouver que la relation entre les variables de décision, z_{ik}^d, des modèles mathématiques décrits dans la section 3 et les x_j, des modèles de la section 4.1, peut être représentée par la formulation suivante :

$$z_{ik}^d = \sum_{j \in \Xi} a_{ij} b_j^d e_{kj} x_j \tag{4.1.10}$$

De plus, chaque colonne du problème principal en formulation de « set partition », MP, correspond à un planning réalisable qui est constitué d'un sous-ensemble d'interventions à réaliser dans une salle d'opération pour une plage horaire (si « block scheduling ») ou une journée (si « open scheduling »). La méthode de génération de colonnes est appliquée pour résoudre la relaxation linéaire de nos problèmes. Dans la relaxation linéaire du MP, appelée LMP (Linear Master Problem, Problème Principal Linéaire), la contrainte d'intégrité est relâchée. La contrainte d'intégrité du MP ((4.1.4) du MMBS ou (4.1.9) du MMOS) est remplacée par $x_j \geq 0, j = 1, \cdots, N_{plan}$. En général, le nombre de plannings possibles est extrêmement grand mais il n'est pas nécessaire de générer tous les plannings à l'avance. En fait, les plannings ne sont générés qu'au cas où ils vont entrer dans la base du problème. Donc, au début de la procédure de résolution, nous ne générons qu'un sous-ensemble de plannings du problème concerné (MMBS ou MMOS) pour construire le RMP (Restricted Master Problem, Problème Principal Restreint). À partir d'ici, une procédure de génération de colonnes, dont les problèmes auxiliaires sont résolus par la programmation dynamique, est utilisée afin de résoudre le LMP pour obtenir une borne inférieure du problème concerné (MMBS ou MMOS).

Dans la dernière décennie, la méthode de génération de colonnes, pour les programmations linéaires en nombres entiers, est aussi appliquée avec succès pour les problèmes pouvant être formulés comme des variantes du « set partitioning problem » avec des variables binaires [Vanderbeck et Wolsey, 1996][Barnhart *et al.*, 1998], [Valvério de Carvalho,1998][Chen et Powell, 1999a, 1999b][Lübbecke et Desrosiers, 2004]. Nous utilisons aussi la méthode de génération de colonnes qui permettra de servir de base, d'une part, pour la construction d'heuristiques et d'autre part, pour la construction d'une procédure exacte. Ces deux méthodes de résolution sont détaillées respectivement dans les sections suivantes pour résoudre le problème du « block scheduling ».[9]

4.2.1 Description générale de la génération de colonnes

La Figure 9 montre succinctement la description de la procédure de la génération de colonnes.

[9] Etant donné que la technique pour la résolution du modèle « open scheduling » est pratiquement similaire à celle du « block scheduling », à l'exception d'une contrainte supplémentaire (la contrainte de la disponibilité des chirurgiens) pour son MP (Master Problème, Problème Principale) par rapport à celui du « block scheduling », nous ne la détaillons pas ici.

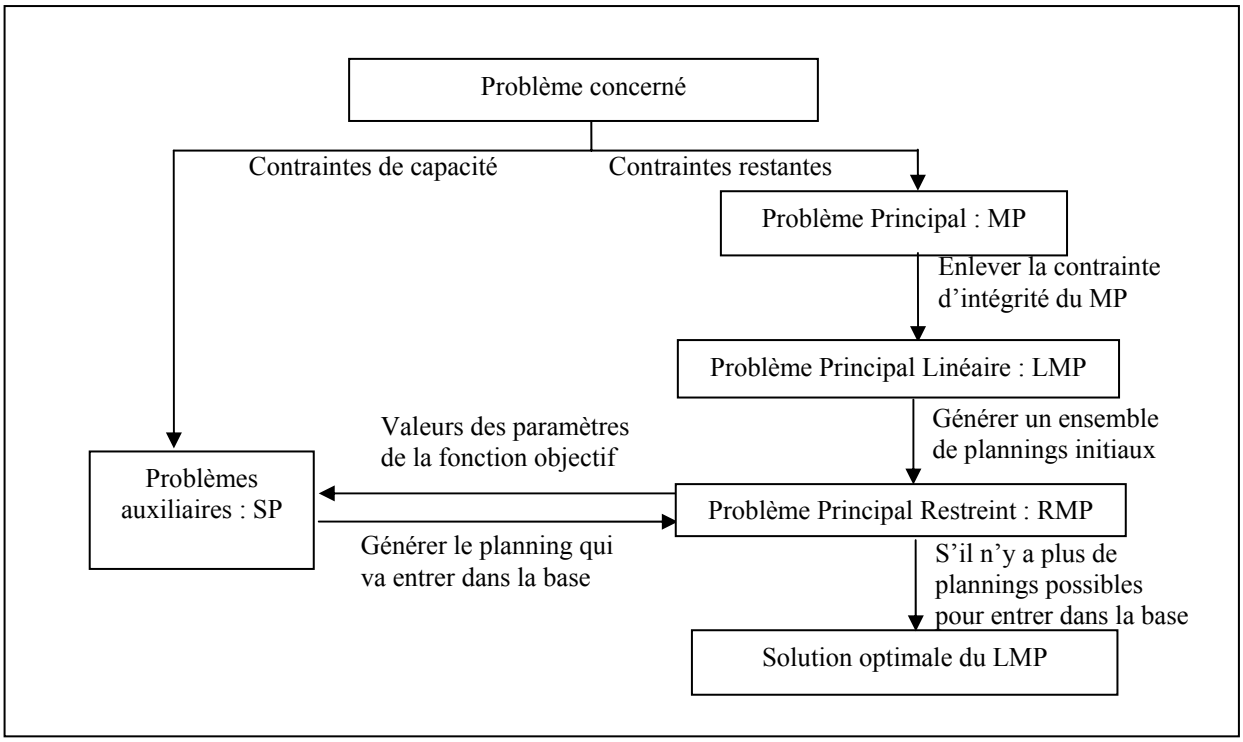

Figure 9 : Description de la génération de colonnes

Cette procédure peut être détaillée comme suit :

Etape 1 : Générer un ensemble de plannings initiaux par une variante de la procédure « Best Fit Descending » qui est présentée dans [Dexter *et al.*, 1999b], pour construire le RMP (Restricted Master Problem, Problème Principal Restreint) initial du problème actuel sur la base d'un LMP (Linear Master Problem, Problème Principal Linéaire). Cette petite variante de « Best Fit Descending » se déroule comme suit : tout d'abord, un tri des interventions par ordre croissant des dates limites est d'abord effectué. Ensuite les interventions avec la même date limite sont triées par ordre décroissant des durées opératoires. Puis, l'intervention à la tête de la liste est affectée à une plage horaire, dont la date est inférieure ou égale à la date limite de cette intervention, ayant assez mais le moins de temps disponible. Et ainsi de suite jusqu'à ce que toutes les interventions aient été affectées ou qu'il n'y ait plus de plages horaires disponibles. Dans le tableau simplexe de la programmation linéaire RMP, chaque colonne correspondra à un planning réalisable ;

Etape 2 : Résoudre le RMP actuel. Les valeurs duales obtenues, πI_i ($i \in \Omega$), πII_k^d ,($d = 1, \cdots, N_{jour}$, $k = 1, \cdots, NB_d$) correspondant d'ailleurs aux contraintes (4.1.1), (4.1.2) et (4.1.3) respectivement, vont constituer les coefficients de la fonction objectif des problèmes auxiliaires. Nous avons un problème auxiliaire par jour par plage horaire ;

Etape 3 : Résoudre tous les problèmes auxiliaires par la programmation dynamique (décrite en section 4.2.3). Nous obtenons donc pour chaque jour le «meilleur» planning à entrer dans la base du RMP. La valeur de leur fonction objectif représente le coût réduit d'une colonne ;

Etape 4 : En comparant les valeurs des fonctions objectifs de toutes les plages horaires, nous allons choisir une colonne dont le coût réduit est le plus petit.

Si cette valeur minimale est négative, nous acceptons cette colonne pour qu'elle rentre dans la base du RMP (nous ajoutons une colonne) à la place d'un planning existant (selon les règles habituelles de la méthode de simplexe) et nous retournons à l'étape 2 ;

Etape 5 : Si toutes les valeurs de la fonction objectif des problèmes auxiliaires sont positives ou nulles alors la procédure de génération de colonnes (GC) s'arrête et dans ce cas, la base actuelle du LMP est optimale.

Après avoir présenté le problème principal MP du MMBS et MMOS, nous allons détailler ci-dessous les problèmes auxiliaires.

4.2.2 Description des problèmes auxiliaires

En programmation linéaire, les coefficients de la fonction objectif finale représente le coût réduit des plannings. Chaque colonne *j* du RMP, qui n'est pas dans la base, a un coût réduit dont la valeur peut être calculée par la formule suivante :

$$\sigma_j = C_j - \sum_{i \in \overline{\Omega}} a_{ij} \left(\sum_{d=1}^{D_i} \sum_{k=1}^{NB_d} b_j^d e_{kj}\right) \pi I_i - \sum_{i \in \Omega \setminus \overline{\Omega}} a_{ij} \pi I_i - \sum_{d=1}^{N_{jour}} \sum_{k=1}^{NB_d} b_j^d e_{kj} \pi II_k^d \qquad (4.2.1)$$

Où πI_i est la valeur de la variable duale correspondant aux contraintes (4.1.1) et (4.1.2), πII_k^d est la valeur de la variable duale correspondant aux contraintes (4.1.3).

Du fait que chaque colonne du RMP correspond à un planning réalisable, si nous voulons trouver une colonne qui va entrer dans la base du RMP et dont le coût réduit est le plus petit, nous devons résoudre un problème auxiliaire pour chaque plage horaire *k* dans le jour *d* ($k = 1,\ldots, NB_d$, $d = 1, \ldots, N_{jour}$) qui vise à trouver un planning réalisable dont le coût réduit est le plus petit, tout en respectant les contraintes de nombre d'heures possibles pour la plage horaire concernée (la contrainte (3.1.4)). Cela consiste à résoudre un problème appelé problème auxiliaire en décidant des valeurs de a_{ij}, b_j^d et e_{kj} de telle sorte que σ_j soit le plus petit possible.

Un problème auxiliaire SP_k^d, dont l'objectif est de trouver un planning de coût réduit le plus petit pour une plage horaire *k* le jour *d* ($k = 1, \ldots, NB_d$, $d = 1, \ldots, N_{jour}$), peut être décrit comme suit :

$$\underset{y_i}{Min} \quad \max\left\{ TOR_k^d - \sum_{i \in \Omega^d} y_i t_i, \beta\left(\sum_{i \in \Omega^d} y_i t_i - TOR_k^d \right) \right\} - \sum_{i \in \Omega^d} y_i \pi I_i - \pi II_k^d$$

sous les contraintes

$$\sum_{i \in \Omega^d} y_i t_i \le TOR_k^d + TS_k^d \qquad (4.2.2)$$

$$y_i \in \{0,1\}, i \in \Omega^d \qquad (4.2.3)$$

Où $\Omega^d = \{i \mid D_i \ge d, i \in \Omega\}$, pour garantir que dans un planning, il n'y ait que des interventions de date limite supérieure ou égale au jour *d* considéré. Autrement dit, aucune intervention ne peut être planifiée après sa date limite.

De plus, en prenant en compte la formule de la fonction objectif de SP_k^d, nous pouvons encore décomposer chaque sous-problème SP_k^d en deux cas suivants. Dans un cas, ou $\sum_{i\in\Omega^d} y_i t_i \le TOR_k^d$ et dans l'autre, $\sum_{i\in\Omega^d} y_i t_i > TOR_k^d$.

A. Cas_SP_1

$$\underset{y_i}{Min}\left[TOR_k^d - \pi I_k^d - \sum_{i\in\Omega^d} y_i (t_i + \pi I_i) \right]$$

sous les contraintes

$$\sum_{i\in\Omega^d} y_i t_i \le TOR_k^d \tag{4.2.4}$$

$$y_i \in \{0,1\}, i \in \Omega^d \tag{4.2.5}$$

B. Cas_SP_2

$$\underset{y_i}{Min}\left[-\beta TOR_k^d - \pi I_k^d + \sum_{i\in\Omega^d} y_i (\beta t_i - \pi I_i) \right]$$

sous les contraintes

$$\sum_{i\in\Omega^d} y_i t_i > TOR_k^d \tag{4.2.6}$$

$$\sum_{i\in\Omega^d} y_i t_i \le TOR_k^d + TS_k^d \tag{4.2.7}$$

$$y_i \in \{0,1\}, i \in \Omega^d \tag{4.2.8}$$

Etant donné que le premier sous-problème Cas_SP_1 est un problème de type « sac à dos » [Martello et Toth, 1990][Hifi et Zissimopoulos, 1996] et que le deuxième Cas_SP_2 s'y apparente, nous les résolvons au moyen de la programmation dynamique que nous décrivons ci-après.

4.2.3 Procédures de la programmation dynamique pour résoudre les problèmes auxiliaires

Nous notons

N^d : le nombre d'interventions dans l'ensemble $\Omega^d = \{i \mid D_i \ge d, i \in \Omega\}$;

$F(j,tf)$: le coût réduit minimal d'un planning actuel, où les interventions sont choisies parmi un sous-ensemble d'interventions $B_j = \{1,\cdots,j\} \subseteq \Omega^d$ dont le temps d'opération total est égal à une valeur *tf*.

A. Programmation dynamique pour le Cas_SP_1

Etape 1 : (Etape initiale)

$F(0,tf) = +\infty$ si $0 < tf \le TOR_k^d$ ou $t < 0$;

$F(0,0) = TOR_k^d - \pi II_k^d$;

Etape 2 : (Formules de récurrence)

Pour tout $tf = 0$ à TOR_k^d ,

$F(j,tf) = Min\{F(j-1,tf), F(j-1,tf - t_j) - (t_j + \pi I_j)\}$;

Etape 3 : (Solution optimale)

La solution optimale est obtenue en calculant

$F(N^d, tf^*) = \underset{0 \le tf \le TOR_k^d}{Min} \{F(N^d, tf)\}$.

B. Programmation dynamique pour le Cas_SP_2

Etant donné que la programmation dynamique pour le Cas_SP_2 est pratiquement similaire à celle du Cas_SP_1, nous ne le détaillons pas ici.

Si la valeur obtenue pour F(N^d ; *tf**) est infinie alors aucun planning n'est réalisé dans le Cas_SP_2 ; autrement, nous comparons sa solution avec celle du Cas_SP_1 pour trouver un planning dont le coût réduit est le plus petit. Ce planning entrera dans la base du RMP.

4.3 Procédures heuristiques basées sur la génération de colonnes

Dans cette section, nous allons résoudre le problème MMBS par la procédure PHBGC (Procédure Heuristique Basée sur la Génération de Colonnes) afin de trouver les solutions approchées. La Figure 10 montre succinctement la méthodologie suivie :

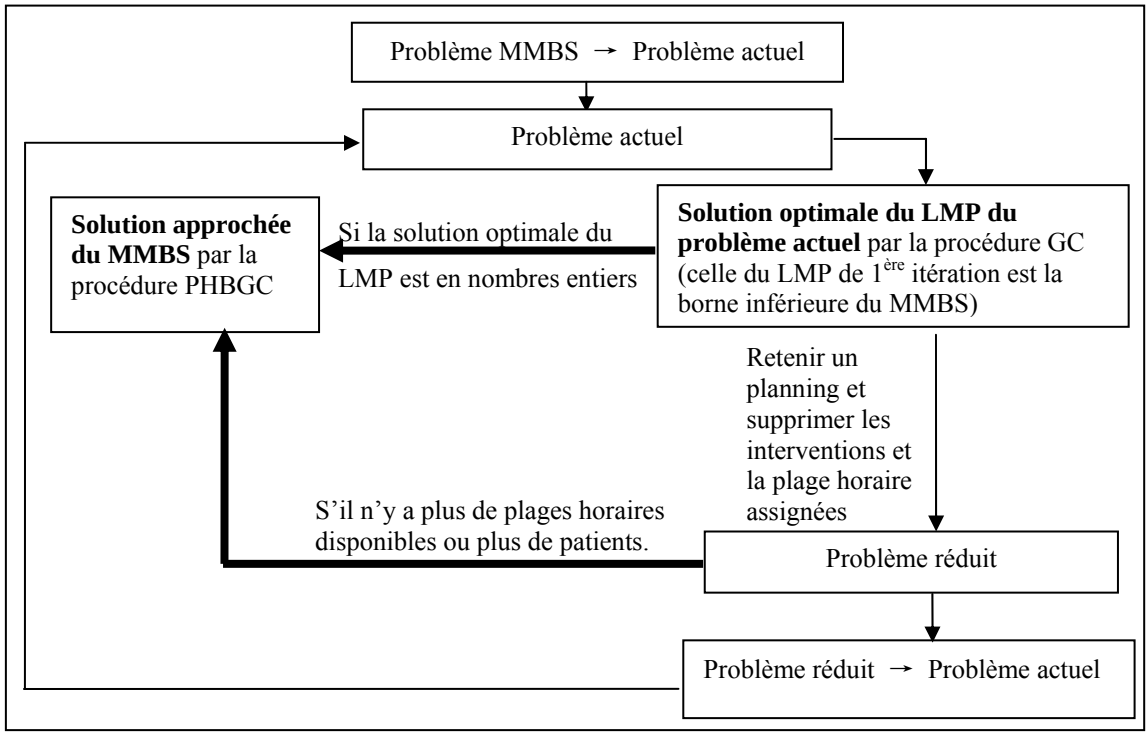

Figure 10 : Description générale de la méthodologie

Toutes les heuristiques utilisées dans ce cadre fonctionnement selon le même schéma. A chaque itération, nous résolvons un problème avec un ensemble d'interventions à assigner et un ensemble de plages horaires disponibles. Ce problème est résolu à l'aide de la génération de colonnes en relâchant la contrainte d'intégrité. Nous retenons ensuite un planning optimal parmi les colonnes de base. Nous obtenons alors un problème réduit où les interventions planifiées et la plage horaire utilisée dans le planning retenu sont retirées. Ce processus continue jusqu'à ce qu'il n'y ait plus d'interventions ou de plage horaire. Suivant la manière dont un planning est retenu, nous obtenons quatre versions de la procédure PHBGC.

Description générale de la procédure PHBGC:

Etape 1 : Le problème actuel représente le problème MMBS ;

Etape 2 : Résoudre le LMP du problème actuel par la génération de colonnes (GC) ;

Etape 3 : Si la solution trouvée par l'étape 2 respecte la contrainte d'intégrité sur les variables de décision $x_j, j \in \Xi$ (contrainte (4.1.4)), nous avons terminé. Si c'est la première itération, nous avons trouvé une solution optimale réalisable du MMBS, sinon, nous avons trouvé une solution approchée réalisable du MMBS ;

Etape 4 : Si la solution obtenue à l'étape 2 ne respecte pas la contrainte d'intégrité sur les variables de décision x_j (vu que cette contrainte a été enlevée du LMP), cela signifie que nous avons juste obtenu une borne inférieure (LB) de la fonction objectif du problème actuel. Dans ce cas-ci, s'il y a des plannings correspondant aux x_j =1, nous cherchons parmi eux un planning j dont le coût du planning est le plus petit et s'il n'y a aucun planning tel que x_j =1 pour le moment, nous choisirons un planning par une des manières suivantes :

1. Nous cherchons un planning j dont x_j est le plus proche de 1. S'il y en a plusieurs, nous choisirons celui dont le coût du planning est le plus petit (version 1 de la procédure PHBGC) ;
2. Nous cherchons un planning j dont le coût C_j est le plus petit. S'il y en a plusieurs, nous choisirons celui dont x_j est le plus grand (version 2 de la procédure PHBGC) ;
3. Nous cherchons un planning dont la valeur du C_j / x_j est la plus petite. S'il y en a plusieurs, nous choisirons celui dont x_j est le plus grand (version 3 de la procédure PHBGC) ;
4. Nous cherchons un planning dont la valeur du C_j / x_j est la plus petite. S'il y en a plusieurs, nous choisirons celui dont C_j est le plus petit (version 4 de la procédure PHBGC).

Etape 5 : Supprimer toutes les interventions déjà affectées ainsi que la plage horaire déjà utilisée dans le planning accepté, nous obtenons un problème réduit ;

Etape 6 : S'il n'y a plus de plages horaires disponibles ou plus d'interventions à assigner, cette procédure PHBGC se termine. Sinon, ce problème réduit devient le problème actuel. Ensuite, aller à l'étape 2.

Notons que dans les expérimentations, les quatre versions de la procédure PHBGC sont testées et comparées. Puis nous choisissons la meilleure solution approchée d'entre elles comme la solution approchée finale.

4.4 Méthode exacte basée sur la génération de colonnes

Comme nous le savons, le problème de type « sac-à-dos » est un problème NP-complet [Martello et Toth, 1990][Hifi et Zissimopoulos, 1996]. Notre problème, étant plus général qu'un problème de type « sac à dos », l'est aussi (voir partie IV).

Pour résoudre les problèmes NP-complets de manière exacte, nous ne pouvons qu'utiliser des méthodes d'énumération pour les problèmes bien structurés. Dans cette section, nous allons présenter une procédure de Branch-and-Price, qui est une procédure de Branch-and-Bound basée sur la génération de colonnes.

Comme une application très vaste de la procédure de Branch-and-Bound existe, nous allons exposer son principe général et ensuite une des variantes, la procédure de Branch-and-Price, pour la résolution du problème concerné, qui est modélisé comme un programme linéaire en nombres entiers.

4.4.1 Principe général de Branch-and-Bound

La procédure de Branch-and-Bound parcourt implicitement toutes les solutions. Elle procède par décomposition d'un ensemble de solutions (ramification) en sous-ensembles. Au lieu de chercher directement la meilleure solution parmi l'ensemble, nous cherchons la meilleure solution de chacun des sous-ensembles. La meilleure solution de l'ensemble est nécessairement la meilleure solution parmi les meilleures solutions des sous-ensembles. Ce processus peut encore être appliqué sur la recherche de la meilleure solution parmi chacun des sous-ensembles et ainsi de suite jusqu'à ce que les sous-ensembles deviennent tellement petits que le calcul d'une solution optimale est immédiat (en particulier lorsque les sous-ensembles deviennent des singletons). La décomposition de l'ensemble des solutions en sous-ensembles de plus en plus petits est le principe de ramification.

Mais avant de rechercher une solution dans un sous-ensemble (nous disons aussi ce sous-ensemble est exploré), celui-ci est évalué. Le but de l'évaluation est d'éliminer des sous-ensembles sans les explorer et tout en garantissant qu'au moins une solution optimale n'appartienne pas aux sous-ensembles éliminés.

L'évaluation se fait de deux manières différentes : la dominance et le calcul de borne inférieure (pour la minimisation) ou de borne supérieure (pour la maximisation).

Si quelle que soit la solution σ d'un ensemble *S*, il existe au moins une autre solution qui n'est pas dans l'ensemble *S* et qui est au moins aussi bonne que σ, alors l'ensemble *S* est dit dominé. Nous pouvons alors abandonner la recherche d'une solution optimale dans *S*.

Une borne inférieure (LB) ou une borne supérieure (UB) d'un ensemble donné de solutions est une estimation de la valeur de la fonction objectif de la meilleure solution parmi l'ensemble considéré, dans une situation plus favorable que la réalité. S'agissant d'une estimation, elle doit être aussi précise que possible. Par conséquent, une borne inférieure est d'autant meilleure qu'elle est grande et une borne supérieure est d'autant meilleure qu'elle est petite.

Le calcul de borne inférieure (pour la minimisation) ou de borne supérieure (pour la maximisation) s'effectue en se mettant dans une situation plus favorable que la réalité c'est-à-dire en relâchant des contraintes et/ou en modifiant la fonction objectif. Pour un problème de minimisation, si la borne inférieure d'un sous-ensemble de solutions est plus grande que la valeur de la fonction objectif d'une solution déjà connue, nous pouvons être

sûr que ce sous-ensemble ne contient pas une solution strictement meilleure que la solution déjà connue. Par conséquent, nous n'explorons plus ce sous-ensemble. Le calcul de borne inférieure et/ou borne supérieure doit être plus facile que la résolution du problème initial, sinon il n'y aurait qu'à résoudre directement le problème initial. D'habitude, une borne inférieure est obtenue en résolvant un problème relaxé où soit une partie des contraintes, comme les contraintes d'intégrité, sont relâchés, soit la fonction objectif est modifiée (par exemple, nous remplaçons la maximisation de sin *x* par la maximisation de x pour les $x \in [0,1]$).

Pour un problème donné, il peut y avoir plusieurs manières de construire une procédure par ramification et évaluation. Elles se différent par la manière de décomposer une ensemble (un nœud père dans l'arbre d'énumération) en sous-ensembles (les nœuds fils dans l'arbre d'énumération), de calculer les bornes inférieures ou les bornes supérieures, et par l'ordre d'exploration des sous-ensembles en attente d'exploration.

Par rapport à une procédure générale de Branch-and-Bound, les particularités de la procédure de Branch-and-Price se situent au niveau :

- de l'obtention de la première borne supérieure (UB) ;
- du calcul de la borne inférieure (LB) d'un nœud ;
- de la sélection de la variable de ramification ;
- de l'ordre les nœuds d'exploration des nœuds.

De plus, contrairement à une procédure générale de Branch-and-Bound où les propriétés de dominance servent à éliminer des nœuds dominés, ce n'est pas le cas ici.

Dans cette section, nous allons présenter premièrement la description générale de cette procédure et ensuite, les points clefs : les stratégies de la sélection du nœud et de la variable de ramification.

4.4.2 Procédure de Branch-and-Price

Etape 1 : (Initialisation)
Liste des nœuds candidats = vide ; le premier nœud actuel représente le problème général GP du MMBS ;

Etape 2 : (Résoudre le nœud actuel)

a. Résoudre le LMP du nœud actuel par la procédure GC (la procédure de génération de colonnes) ;

b. Si le LMP actuel est irréalisable ou si sa solution optimale est supérieure ou égale à la borne supérieure actuelle (UB), aller à l'étape 3 ;

c. Mettre la solution de LMP du nœud actuel comme sa borne inférieure (LB). Si ce nœud actuel est le premier nœud exploré, une procédure heuristique PHBGC, présenté dans la section 4.2, est appliquée pour obtenir une solution réalisable qui constituera la première borne supérieure (UB) de cette procédure Branch-and-Price.

Etape 3 : (Evaluation)

a. Si le nœud actuel est le premier nœud exploré et si son LMP est irréalisable, cette procédure de Branch-and-Price se termine. nous concluons que le problème n'admet pas de solution réalisable ;

b. Si la borne inférieure (LB) du nœud actuel est supérieure ou égale à la

borne supérieure actuelle (UB) et si la liste actuelle des nœuds candidats est vide, la procédure s'arrête. Dans ce cas-ci, la UB actuelle est la solution optimale. Si la liste actuelle des nœuds candidats n'est pas vide, aller à l'étape 3(d) ;

c. Si la LB du nœud actuel est une solution réalisable (en nombres entiers), nous remplaçons UB actuelle par cette LB; sinon, aller à l'étape 4 ;

d. Remplacer le nœud actuel par la tête de la liste des noeuds candidats dont la LB est la plus petite et enlever le nœud choisi de la liste en même temps. Ensuite, aller à l'étape 2.

Etape 4 : (Ramification)
Calculer et choisir une variable de ramification selon les critères présentés dans la section 4.4.3, puis ramifier le nœud actuel en deux nœuds fils ;

Etape 5 : (Etendre la liste actuelle de nœuds)
Mettre le nœud fils gauche comme le nœud actuel à explorer et, en même temps, ajouter le nœud fils droit dans la liste actuelle des nœuds candidats où tous les nœuds sont triés par ordre croissant de leurs bornes inférieures, ensuite, aller à l'étape 2.

Nous allons présenter les points les plus importants de la procédure de Branch-and-Price : les stratégies de la sélection du nœud et de la variable de ramification.

4.4.3 Stratégies de la sélection du nœud et de la variable de Ramification

La stratégie de la sélection du nœud vise à choisir un nœud actuel dans la liste de nœuds candidats et la stratégie de la sélection de la variable de ramification vise à choisir une variable non entière pour effectuer la ramification, autrement dit comment créer les nœuds fils.

A. Stratégie de la sélection du nœud

La stratégie de la sélection du nœud appliquée ici est la combinaison de la règle de « Depth-first-search » et celle de « Best-lower-bound », c'est-à-dire que si le nœud actuel ne peut pas être élagué, la règle de Depth-first-search est appliquée et donc un de ses nœuds fils (dans ce travail, nous prenons son nœud fils gauche) est choisi comme prochain nœud à être traité. En outre, si le nœud actuel est élagué, la règle Best-lower-bound est appliquée et un nœud dans la liste de noeuds candidats, dont la borne inférieure est le minimum, est choisi pour être exploré.

B. Préparation pour la stratégie de la sélection de la variable de ramification

Dans la procédure générale de Branch-and-Bound, les variables de décision sont normalement utilisées directement pour les actions de ramification. Mais ce n'est plus le cas pour la procédure de Branch-and-Price, qui est basée sur la génération de colonnes. La fixation et la ramification d'une variable du MP provoque des modifications des problèmes auxiliaires utilisés pour générer la colonne entrante dans la base du RMP. En général, ces modifications vont détruire la structure du problème exploité or cette structure est exigée pour résoudre les problèmes auxiliaires efficacement. De plus, il entraîne probablement un arbre d'énumération non équilibré ([Barnhart *et al.*, 1998], [Chen et Powell, 1999b]). Donc, les stratégies de ramification devraient être

particulièrement développées pour la procédure de Branch-and-Price. Pour ce faire, nous reprenons la première description du modèle mathématique MMBS où les variables de décision sont représentées par z_{ik}^{d}.

Puisque le MMBS est équivalent au problème principal de partitionnement MP où chaque colonne correspond à un planning réalisable qui satisfait les contraintes des problèmes auxiliaires, et que ces deux modèles mathématiques représentent le même problème concerné, leurs solutions optimales seront elles aussi équivalentes. Car il est plus simple d'utiliser les variables de décision du modèle MMBS, z_{ik}^{d}, pour les actions de ramification mais la relaxation linéaire du modèle de partitionnement peut être résolue efficacement par la génération de colonnes, donc, nous combinons les deux modèles pour construire une procédure de Branch-and-Price permettant de trouver la solution optimale du problème de « Block scheduling » concerné, où les variables de décision du MMBS, z_{ik}^{d}, sont utilisées pour l'action de ramification.

C. Stratégie de la sélection de la variable de ramification

Pour choisir une variable pour l'action de ramification, tout d'abord, nous définissons les *q*-variables comme suit :

$$q_i^d = \sum_{k=1}^{NB_d} z_{ik}^d \quad i \in \Omega, d \in \{1, \cdots, N_{jour}\} \tag{4.4.1}$$

Chaque variable q_i^d indique si une intervention i est planifiée au jour d.

La stratégie de la sélection de la variable de ramification se compose de deux étapes :

Premier étape :

Ramifier le noeud actuel selon la valeur de q-variable. S'il y a plusieurs q-variables fractionnaires, nous choisissons une paire de (i_0, d_0) selon le critère 1[4]. Puis, deux nœuds fils sont construits. Dans le nœud fils gauche nous forçons $q_{i_0}^{d_0}$=0 et dans celui de droite nous forçons $q_{i_0}^{d_0}$=1. Pour le cas où $q_{i_0}^{d_0}$ est à 0, nous mettons $z_{i_0 k}^{d_0} = 0$ pour tous les k$\in \{1,\ldots, NB_{d_0}\}$, et le RMP initial de ce nœud se compose de toutes les colonnes de son nœud père sauf celles où l'intervention i_0 est planifiée le jour d_0. En même temps, nous mettons $\Omega_k^{d_0} = \Omega_k^{d_0} \setminus \{i_0\}, k \in \{1,..., NB_{d_0}\}$, autrement dit, nous interdisons la planification de i_0 au jour d_0. Pour les nœuds où $q_{i_0}^{d_0}$ est mise à 1, toutes les variables $z_{i_0 k}^{d}\Big|_{d \neq d_0}$ sont mises à 0 pour tous les *k* possibles et dans ce cas-là, son RMP initial se compose de toutes les colonnes de son nœud père sauf celles où l'intervention i_0 est planifiée les jours autres que le jour d_0. De plus, nous enlevons i_0 de Ω_k^d pour tout *d* tel que $d \neq d_0$;

Deuxième étape:

---

[4] Critère 1: $q_{i_0}^{d_0}$ est fractionnaire et le plus proche de 0.5, i.e.

$$q_{i_0}^{d_0} = \arg\min_{q_i^d} \{\left| q_i^d - 0.5 \right|, 0 < q_i^d < 1, i \in \Omega, d \in \{1, \cdots, N_{jour}\}\}$$

Si toutes les q-variables sont entières, nous vérifions encore s'il y a des valeurs de z-variables fractionnaires. S'il y en a, nous allons ramifier selon les valeurs de z-variables comme suit : choisir (i_0, d_0, k_0) selon le critère 2[5]. Comme lors de la première étape, deux nœuds fils sont créés. Pour le nœud fils gauche, nous mettons $z_{i_0k_0}^{d_0}$ =0 et $\Omega_{k_0}^{d_0} = \Omega_{k_0}^{d_0} \setminus \{i_0\}$. Son RMP initial se compose de toutes les colonnes de son nœud père sauf celles où l'intervention i_0 est planifiée à la plage horaire k_0 le jour d_0. Le nœud fils droit correspond au $z_{i_0k_0}^{d_0}$ =1, son RMP initial reprend toutes les colonnes de son noeud père pour lesquelles soit l'intervention i_0 est planifiée à la plage horaire k_0 le jour d_0 soit l'intervention i_0 ne figure pas. Nous mettons $\Omega_k^d = \Omega_k^d \setminus \{i_0\}$ pour tous les $\{k \in \{1, \cdots, NB_d\}, d \in \{1,..., N_{jour}\} \setminus \{d_0\}\}$ et $\{d = d_0, k \in \{1,..., NB_d\} \setminus \{k_0\}\}$.

5. Conclusion

Dans cette partie, nous avons tout d'abord présenté le problème générique de la planification opératoire. Deux modèles mathématiques en nombres entiers, un pour la stratégie du « block scheduling » et l'autre pour la stratégie de l'« open scheduling », ont été construits. La procédure de génération de colonnes (GC) est ensuite proposée pour résoudre leurs relaxations linéaires afin de trouver les bornes inférieures de chacun. Basée sur cette procédure, deux méthodes sont proposées pour résoudre les modèles concernés : la heuristique PHBGC et la méthode exacte Branch-and-Price.

Jusqu'à maintenant, nous ne traitons que les problèmes de planification hebdomadaire ; cependant un programme opératoire réalisable doit indiquer l'ordre de chaque intervention et donc les problèmes d'ordonnancement des interventions doivent aussi être pris en compte. Dans la partie suivante, nous allons nous concentrer sur la deuxième étape de la planification opératoire, les problèmes d'ordonnancement journaliers, afin de construire finalement un programme opératoire réalisable. Comme pour cette partie, deux problèmes d'ordonnancement, un pour la stratégie du « block scheduling » et l'autre pour la stratégie de l'« open scheduling » seront traités dans la partie suivante.

[5] Critère 2: $z_{i_0k_0}^{d_0}$ est fractionnaire et le plus proche de 0.5, i.e.

$$z_{i_0k_0}^{d_0} = \arg\min_{z_{ik}^d} \{\left| z_{ik}^d - 0.5 \right|, \quad 0 < z_{i_0k_0}^{d_0} < 1, i \in \Omega, d \in \{1, \cdots, N_{jour}\}\}$$

Partie IV :

Ordonnancement des blocs opératoires

1. Introduction

Dans la partie précédente, nous avons traité la planification des blocs opératoires, en prenant en considération les stratégies de « block scheduling » et d'« open scheduling ». Nous avons vu que la programmation opératoire est non seulement constituée de la planification mais aussi de l'ordonnancement des interventions dans un bloc opératoire. C'est donc sur cette dernière partie que nous allons maintenant nous concentrer afin de construire un programme opératoire réalisable et efficace en terme de coût des heures opératoires du bloc. Dans cette partie, nous intégrons la contrainte relative à la disponibilité des lits dans la SSPI et pour l'« open scheduling » des contraintes concernant les chirurgiens.

Deux modèles généraux d'ordonnancement des interventions dans un bloc opératoire seront présentés : un pour la stratégie du « block scheduling », un autre pour la stratégie de l'« open scheduling ». Ils prendront chacun en ligne de compte plusieurs salles d'opération et plusieurs lits en SSPI.

Pour construire un modèle d'ordonnancement, nous examinons le flux de patients dans le service chirurgical. En Figure 11 nous reprenons un schéma présenté dans [Jebali, 2004].

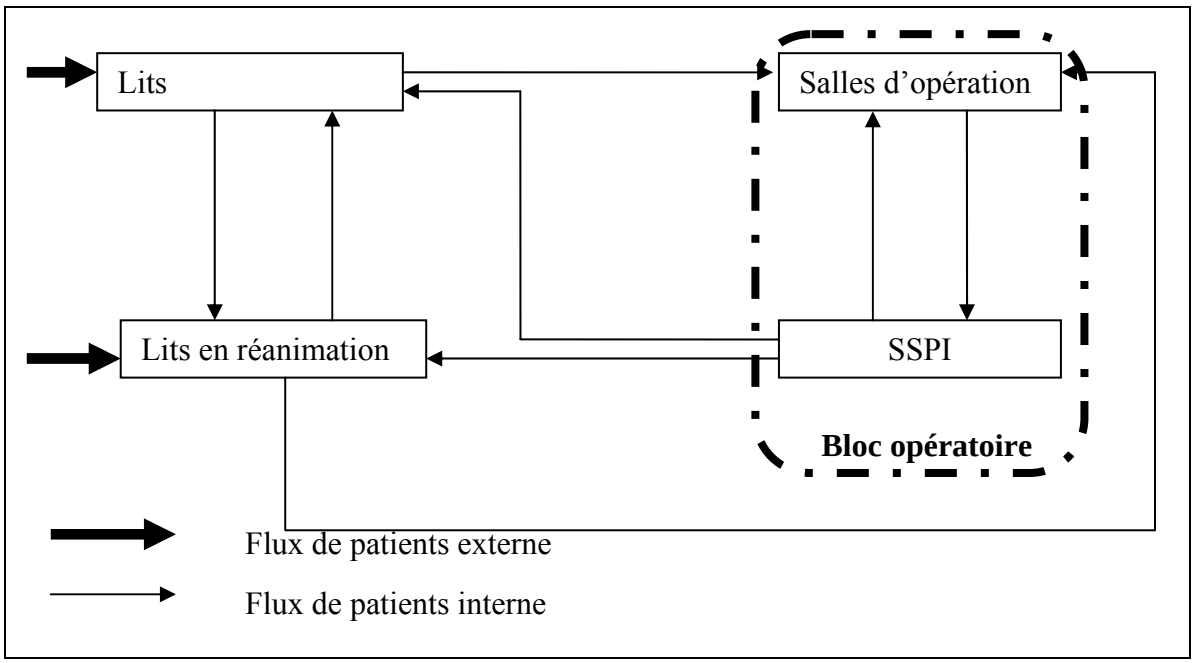

Figure 11 : Flux de patients dans le service chirurgical

Nous voyons que le flux de patients passant par le bloc opératoire montre une diversité de routages possibles. Les deux routages les plus fréquents pour un patient programmé sont :

- Lit d'hospitalisation → salle d'opération → SSPI → lit d'hospitalisation ;

- Lit d'hospitalisation → salle d'opération → SSPI → lit en réanimation → lit d'hospitalisation.

Nos modèles d'ordonnancement vont se concentrer sur le bloc opératoire, c'est-à-dire l'ensemble « salles d'opération − SSPI ».

Notre objectif est de minimiser le coût des heures opératoires du bloc. Pour se faire, il suffit de minimiser la durée d'accomplissement de toutes les interventions affectées dans un planning pour les salles d'opération ainsi que pour la SSPI. Comme les ressources matérielles et humaines sont beaucoup plus coûteuses pour les salles que pour la SSPI, notre objectif sera non seulement de minimiser le *makespan* du bloc opératoire, c'est-à-dire la durée qui s'écoule entre l'heure de début de la première opération dans la salle d'opération et l'heure de fin de la dernière intervention dans la SSPI, mais aussi la durée qui s'écoule entre l'heure de début de la première intervention dans une salle d'opération et l'heure de fin de la dernière intervention dans les salles d'opération (le *makespan* des salles d'opération) en accordant à ce dernier objectif un poids plus important.

L'objectif de minimisation du *makespan* est un objectif largement étudié par la communauté de chercheurs en gestion industrielle. Etant donné que le bloc opératoire est vide en début et en fin de journée d'ordonnancement, nous nous baserons sur les travaux effectués en gestion industrielle en tenant compte aussi de contraintes supplémentaires.

Comme le montre la Figure 12, le bloc opératoire est constitué de deux ensembles : d'une part, un ensemble de salles d'opération, spécialisées dans le cas d'une stratégie du « block scheduling » et multifonctionnelles dans le cas d'une stratégie d' « open scheduling » et d'autre part, une salle de réveil (la SSPI) qui contient un certain nombre de lits de réveil identiques. Le patient, après son opération, est transféré dans la SSPI s'il y a un lit disponible sinon il reste dans la salle d'opération jusqu'à ce qu'un lit se libère ou que le patient soit tout-à-fait réveillé.

Le modèle d'ordonnancement des interventions dans un bloc opératoire normalement constitué de plusieurs salles d'opération et de lits en SSPI, vise à ordonnancer un ensemble d'interventions affectées lors de la phase de planification pour une journée dans le cas où les lits en SSPI ne sont pas toujours disponibles pour les demandes. Notons que l'affectation des interventions dans les salles d'opération décidée au niveau planification pour la stratégie de l'« open scheduling » peut être remise en cause au niveau d'ordonnancement.

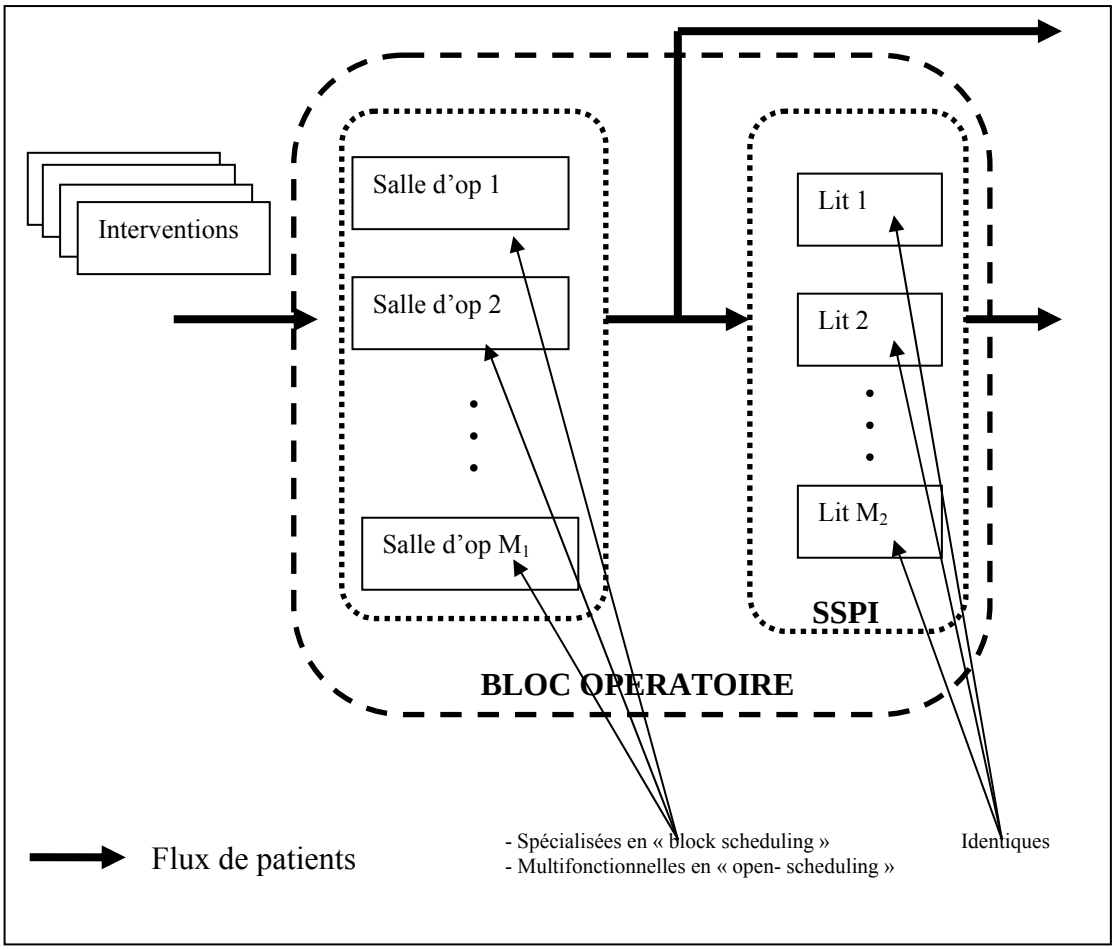

Figure 12 : Description du modèle d'ordonnancement

Pour résoudre notre problème d'ordonnancement, des hypothèses communes aux deux stratégies ont été émises :

- Chaque intervention est déjà affectée à une équipe chirurgicale ;
- Les ressources humaines (à l'exception du chirurgien) et matérielles sont toujours disponibles pour les demandes pendant cette journée ;
- Toutes les salles d'opération ouvrent simultanément pour ce jour d'ordonnancement ;
- Tous les patients sont prêts pour leur intervention, c'est-à-dire, nous ne prenons pas en compte l'heure d'arrivée des patients ;
- Les cas urgents ne sont pas pris en compte et nous supposons que dès qu'une intervention commence, cette salle ne peut plus être affectée à d'autres interventions, c'est-à-dire, il n'y a ni urgence ni préemption ;
- Si aucun lit en SSPI n'est disponible à la fin d'une intervention alors le patient reste dans la salle d'opération jusqu'à ce qu'un lit se libère ou est transféré vers son lit d'hospitalisation s'il se réveille avant qu'un lit de réveil ne se libère. Les lits de réveil dans la SSPI sont identiques, c'est-à-dire que le patient peut être transféré vers n'importe quel lit de réveil disponible ;
- Comme pour l'hypothèse concernant les salles d'opération, il n'y ni urgence ni préemption dans la SSPI, c'est-à-dire que dès qu'un patient est installé dans un lit de réveil, il n'en bouge plus jusqu'à ce qu'il se réveille ;
- Le coût d'une heure d'ouverture des salles d'opération est beaucoup plus important qu'une heure d'ouverture de la SSPI ;

- Les durées de reconfection des lits de réveil et les durées de nettoyage des salles d'opération ne sont pas prises en compte.

Peu de travaux de recherche ont été réalisés à ce jour pour résoudre les problèmes d'ordonnancement des blocs opératoires et aucun, à notre connaissance, n'a traité le problème que nous considérons. Nous constatons certaines analogies entre le problème d'ordonnancement de blocs opératoires et celui d'ateliers de production pour lequel de nombreuses recherches ont été réalisées. Nous pourrons ainsi nous baser sur ces travaux pour résoudre le problème d'ordonnancement des blocs opératoires.

Tous les problèmes d'ordonnancement décrits dans cette thèse peuvent être prouvés NP-complets. Le cas le plus simple apparenté à un modèle de « flow shop » à deux machines est prouvé NP-complet en annexe 2.

2. Résolution de problèmes d'ordonnancement d'ateliers de production et rapprochement avec l'ordonnancement des blocs opératoires

Dans cette section, nous présentons la résolution de problèmes d'ordonnancement d'ateliers de production car parmi eux, les problèmes du « flow shop » hybride à deux étages à contraintes supplémentaires offrent de grandes similitudes avec nos problèmes d'ordonnancement des blocs opératoires.

En conséquence, nous allons donc comparer les problèmes d'ordonnancement et ceux des blocs opératoires. Ensuite, nous présentons les principales méthodes permettant de les résoudre et plus particulièrement les problèmes de type « flow shop » hybride.

Enfin, nous décrivons la structure générale d'un algorithme génétique hybride qui servira à l'ordonnancement des blocs opératoires.

2.1 Principaux problèmes d'ordonnancement d'ateliers de production

2.1.1 Définitions des termes usuels principaux

Avant d'entrer dans les détails, il est nécessaire de préciser quelques définitions des termes usuels que nous utiliserons dans ce travail de recherche pour la description des problèmes d'ordonnancement d'ateliers de production. Nous reprenons les définitions données par Duvivier [2000].

Définition 2.1.1_1 (Opération)
Une opération (ou une tâche) est un travail élémentaire à être localisé dans le temps par une date de début ou de fin et dont la réalisation nécessite un certain intervalle de temps appelé durée.

Définition 2.1.1_2 (Produit)
Un produit est le résultat de plusieurs opérations élémentaires appliquées à une matière première ou à un produit intermédiaire. Dans le cas général, ces opérations sont partiellement ordonnées dans le temps selon un graphe de précédence appelé gamme de fabrication.

Définition 2.1.1_3 (Ressource)

Une ressource est un moyen technique ou humain destiné à être utilisé pour la réalisation d'une opération et disponible en quantité (capacité) limitée.

Définition 2.1.1_4 (Machine)
Dans le cadre des problèmes d'ordonnancement d'atelier, le terme machine désigne les ressources utilisées pour mener à bien le projet global.

Définition 2.1.1_5 (*Makespan*)
Le *makespan* est la durée nécessaire à la fabrication de tous les produits de l'instance considérée selon l'ordre de fabrication imposé par un ordonnancement fixé.

Définition 2.1.1_6 (Etage)
Un étage représente un ensemble de machines parallèles, chacune d'entre elles peut réaliser, éventuellement avec des performances différentes, une même opération sur un produit.

Définition 2.2.1_7 (Ressource supplémentaire)
Le terme ressource supplémentaire représente un moyen technique ou humain critique, à l'exception des machines, qui est demandé par les opérations des différents produits à un même étage dans un système de fabrication.

En général, les problèmes d'ordonnancement peuvent être caractérisés par trois ensembles d'éléments :

- T : un ensemble des produits qui se composent d'une ou plusieurs opérations ;
- E : un ensemble d'étages qui se composent d'une ou plusieurs machines parallèles ;
- R : un ensemble d'une ou de plusieurs sortes de ressources supplémentaires.

Les ressources supplémentaires peuvent être renouvelables ou non renouvelables. Une ressource est renouvelable si après avoir été allouée à une ou plusieurs tâches, elle redevient disponible en même quantité après la fin de ces tâches. Dans notre cas hospitalier, les chirurgiens peuvent être considérés comme des ressources supplémentaires renouvelables.

Définition 2.1.1_8 (Heure d'arrivée)
L'heure d'arrivée d'une opération est l'heure où cette opération est prête à être opérée dans un étage de fabrication. Une opération est disponible à l'heure t si son heure d'arrivée est inférieure ou égale à l'heure t, toutes ses opérations précédentes, prenant en compte les contraintes de précédence, étant finies avant ou juste à l'heure et les ressources supplémentaires (s'il y en a) étant aussi prêtes à l'heure t.

Définition 2.1.1_8 (Ordonnancement)
Un ordonnancement consiste en une programmation prévisionnelle détaillée des ressources mobilisées dans l'exécution des opérations nécessaires à la production élémentaire de biens sur un horizon très court. [Giard, 2003]

2.1.2 Notations générales pour uniformiser la description des problèmes d'ordonnancement

De nombreuses variantes de problèmes d'ordonnancement existent. [Duvivier, 2000] Celles-ci sont liées à la structure de l'atelier (les machines parallèles, séquentielles, etc.), au type de ressources supplémentaires (renouvelables, non renouvelables) et au type de production (unitaire, par lots, etc.).

Pour uniformiser la description des problèmes d'ordonnancement, nous utilisons ici les notations proposées par Graham *et al.* [1979] et Blazewicz *et al.*, [1983,1986,1996]. Chaque notation se compose de trois éléments : $\alpha|\beta|\gamma$. Le premier élément α représente, au moyen de deux types de paramètres α_1 et α_2, respectivement la structure et le nombre de machines dans le système d'ordonnancement.

$\alpha_1 = \varnothing$: Machine unique, c'est-à-dire qu'il n'y a qu'une machine dans l'atelier et que toutes les opérations y sont réalisées ;

$\alpha_1 = P$: Machines identiques, c'est-à-dire qu'il y a un ensemble de machines identiques parallèles dans l'atelier et que toutes les opérations sont opérées sur n'importe laquelle de ces machines avec une même vitesse ;

$\alpha_1 = Q$: Machines uniformes, c'est-à-dire qu'il y a un ensemble de machines parallèles où chaque machine a une vitesse constante différente pour effectuer les opérations ;

$\alpha_1 = R$: Machines non-reliées, c'est-à-dire qu'il y a un ensemble de machines parallèles dont les vitesses d'opération varient en fonction des produits qu'elles opèrent ;

$\alpha_1 = O$: Problème de type « open shop », c'est-à-dire qu'il y a un ensemble de machines ayant différentes fonctionnalités et que les opérations de tout produit suivent des chemins libres, autrement dit, dans un ordre quelconque sans chevauchement dans le temps ;

$\alpha_1 = F$: Problème de type « flow shop », c'est-à-dire qu'il y a un ensemble de machines séquentielles. Tous les produits, ayant un ensemble d'opérations, passent successivement sur toutes les machines avec une même gamme de fabrication fixée. Il existe une variante : le problème de type « flow shop » hybride lorsque les machines existent en plusieurs exemplaires. Ces exemplaires forment un étage. Un étage peut se composer de machines identiques, uniformes, non reliées ou même spécialisées ;

$\alpha_1 = J$: Problème de type « job shop », c'est-à-dire qu'il y a un ensemble de machines séquentielles mais où les opérations élémentaires sont liées par un ordre total, non nécessairement identique pour tous les produits. Le cheminement des produits dans l'atelier est spécifique à chaque article et des produits différents en cours d'usinage devront très vraisemblablement se croiser.

Le paramètre $\alpha_2 \in \{\varnothing, k\}$ représente le nombre de machines utilisées dans le problème d'ordonnancement :

$\alpha_2 = \varnothing$: Le nombre de machines peut varier ;

$\alpha_2 = k$: Le nombre de machines est égal à $k \geq 1$. Pour les problèmes de type hybride, il représente le nombre d'étages.

L'élément β représente notamment les caractéristiques des opérations et des ressources pour décrire un problème d'ordonnancement. Blazewicz *et al.* [1996] ont présenté huit

paramètres pour décrire cet élément β, nous présentons ici deux d'entre eux, ceux qui sont importants pour décrire nos modèles d'ordonnancement.

Le paramètre $\beta_1 \in \{\varnothing, pmtn\}$ indique la possibilité de préemption des opérations.

$\beta_1 = \varnothing$: Aucune préemption n'est permise ;

$\beta_1 = pmtn$: Il y a de la préemption, donc nous pouvons interrompre une opération en cours de réalisation sur une machine, avant son achèvement, pour en démarrer une autre.

Le paramètre $\beta_2 = \{\varnothing, res\}$ caractérise les ressources supplémentaires.

$\beta_2 = \varnothing$: Il n'y a pas de contrainte de ressources supplémentaires ;

$\beta_2 = res$: Il y a des contraintes de ressources supplémentaires à prendre en compte.

De plus, les paramètres présentés par Blazewicz *et al.* [1996] ne nous permettant pas de décrire de manière complète nos problèmes à contraintes de ressources supplémentaires, nous avons dû en rajouter trois autres :
$\beta_3 = \{\varnothing, \lambda\delta\rho\}$, $\beta_4 = \{\varnothing, \{n_1, \cdots, n_m\}\}$ et $\beta_5 = \{\varnothing, \{F(t^{(1)}, t^{(2)}), \cdots, F(t^{(m-1)}, t^{(m)})\}\}$.
Comme précédemment, $\varnothing$ indique que ce cas n'existe pas.

$\beta_3 = \lambda\delta\rho$: Il y a λ types de ressources supplémentaires dont les limites de capacité sont définies par δ et les unités des ressources demandées par les opérations sont représentées par ρ ;

$\beta_4 = \{n_1, \cdots, n_m\}$: Il y a m étages dans un problème de type « flow shop » hybride et n_i machines identiques à chaque étage $i (1 \le i \le m)$. Si le nombre ou/et le type de machines n'est pas identique pour tous les étages, nous ajoutons des paramètres supplémentaires destinés à apporter les précisions requises. Par exemple, Pn_s représente n_s machines identiques à l'étage s ; Rn_s représente n_s machines non reliées à l'étage s ; Dn_s représente n_s machines spécialisées à l'étage s ; Qn_s représente n_s machines uniformes à l'étage s, etc. ;

$\beta_5 = \{(F(t^{(1)}, t^{(2)}), \cdots, F(t^{(m-1)}, t^{(m)})\}$: Pour chaque produit, les relations entre les durées opératoires de deux opérations successives sont fournies par la formule $F(t^{(i)}, t^{(i+1)}), 1 \le i \le m-1$.

Le troisième élément γ représente le critère d'optimisation. Les critères les plus couramment utilisés sont le « *makespan* », le retard moyen, l'avance moyenne, le nombre de commandes en retard, etc.. Dans cette thèse, nous utilisons le *makespan* $C_{\max}$ comme le premier critère d'optimisation et puis le temps moyen de passages des patients dans le bloc opératoire comme le deuxième critère.

2.1.3 Problème d'ordonnancement de type « flow shop » hybride à deux étages

Grâce aux notations présentées ci-dessus, nous pouvons décrire les principaux problèmes d'ordonnancement.

Par exemple, la notation $F2\left|\{\phi, \phi, \phi, P5, R3\}, \{t^{(2)} = t'^{(2)} + \omega(t^{(1)} - t'^{(1)})\ (\omega > 1)\}\right| C_{\max}$ signifie que nous sommes face à un problème d'ordonnancement de type « flow shop » hybride à deux étages (montré à la Figure 13). Le premier étage est composé de cinq machines identiques et le deuxième étage de trois machines non reliées. La gamme de

fabrication est identique puisque chaque produit passe dans un premier temps sur une machine du premier étage et puis sur une du deuxième. De plus, le paramètre $\beta_5 = \{t^{(2)} = t'^{(2)} + \omega(t^{(1)} - t'^{(1)})\ (\omega > 1)\}$, où $t^{(2)}$ représente la durée opératoire normale au deuxième étage dans le cas ce que le produit est transféré sans attente de premier étage au deuxième étage, $t'^{(2)}$ la durée opératoire passée au deuxième étage, $t'^{(1)}$ la durée opératoire de l'opération au premier étage et $t^{(1)}$ la durée passée actuelle au premier étage, indique que la durée opératoire de l'opération au deuxième étage d'un produit dépend de sa durée d'attente entre les deux étages. Plus précisément, si dans une séquence, un produit devait attendre une durée $t^{(1)} - t'^{(1)}$ avant d'entrer dans le deuxième étage, sa durée opératoire au deuxième étage deviendrait $t'^{(2)} + \omega(t^{(1)} - t'^{(1)})$ où $\omega > 1$ est un paramètre donné.

Comme dans un problème de « flow shop » classique (une machine à chaque étage), la résolution d'un tel problème consiste à déterminer une séquence de passage de tous les produits sur chacun des étages. Cela soulève bien évidemment une ambiguïté puisqu'il faut aussi déterminer pour chaque produit, la machine qui réalise chaque opération. Il existe aussi un problème de séquencement pour connaître l'ordre de passage des produits sur chaque machine de chaque étage. Le problème de « flow shop » à deux machines est un modèle « simple » par rapport au problème de « flow-shop » hybride à deux étages.

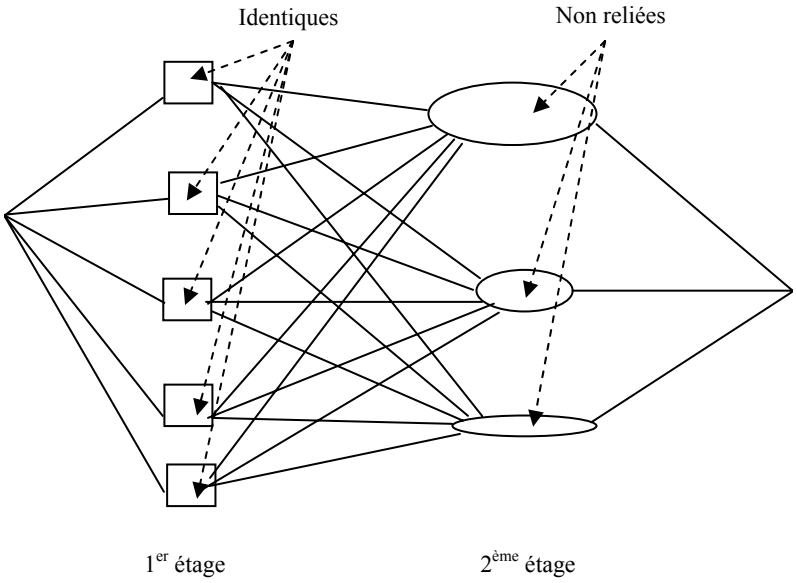

Figure 13 : Exemple de « flow shop » hybride à deux étages (cinq machines identiques au premier étage et trois machines non reliées au deuxième étage).

2.2 Rapprochement avec les modèles d'ordonnancement de blocs opératoires

Quelle que soit la stratégie de la programmation opératoire adoptée (« block scheduling » ou « open scheduling »), le problème d'ordonnancement des interventions au sein d'un bloc opératoire peut être assimilé à un problème de type « flow shop » hybride à deux étages avec des durées opératoires différentes entre les étages avec ou sans contraintes de ressources supplémentaires. Le premier étage est composé des salles d'opération, identiques ou spécialisées selon les stratégies de programmation opératoire, et le deuxième étage consiste en la SSPI contenant un ou plusieurs lits identiques.

Il faut souligner que notre modèle diffère du problème classique de type « flow shop » hybride sur deux points :

1. Les salles d'opération au premier étage sont spécialisées dans le cas du « block scheduling » et identiques dans le cas de l' « open scheduling ». Donc, dans le premier cas, les interventions affectées à une salle d'opération ne pourront pas être déplacées vers d'autres salles d'opération. En fait, dans ce cas-ci, il s'agit d'un cas particulier d'un problème de type « job shop » car le choix de la salle d'opération d'un patient dépend de son chirurgien et aussi du fait que la relation successive entre les deux étages (salles d'opération et la SSPI) est identique pour tous les patients ;
2. S'il n'y a aucun lit disponible en SSPI à la fin d'une intervention alors le patient va devoir rester dans la salle d'opération jusqu'à ce qu'un lit se libère en SSPI ou sera directement transféré dans son lit d'hospitalisation sans passer par la SSPI si aucun lit de la SSPI n'est libéré avant son réveil. Donc, les durées de réveil des patients au sein de la SSPI dépendent du temps de réveil déjà effectué dans la salle d'opération.

Le Tableau 9 établit le rapport entre la terminologie utilisée en gestion industrielle et celle utilisée pour la gestion d'un bloc opératoire.

Gestion d'un atelier de production	Gestion du bloc opératoire
Opération	Intervention dans la salle d'opération Réveil dans la SSPI
Produit	Intervention entière (dans la salle d'opération et la salle de réveil), patient
Ressource	Ressources matérielles : brancards, lits, … Ressources humaines : infirmières, chirurgiens, brancardiers, anesthésistes,…
Machine	Salles d'opération Lits de réveil
Makespan	Le *makespan* du bloc opératoire : la durée qui s'écoule entre l'heure de début de la première intervention dans la salle d'opération et l'heure de fin de la dernière intervention dans la SSPI ; Le *makespan* des salles d'opération : la durée qui s'écoule entre l'heure de début de la première intervention dans la salle d'opération et l'heure de fin de la dernière intervention dans les salles d'opération.
Etage	Salles d'opération au premier étage Salle de réveil au deuxième étage
Ressource supplémentaire	Aucune dans la stratégie du « block scheduling » Chirurgiens dans la stratégie de l'« open scheduling »

Tableau 9 : Terminologie utilisée en gestion industrielle et en gestion hospitalière

2.2.1 Problème d'ordonnancement pour la stratégie du « block scheduling »

Pour la stratégie du « block scheduling », toutes les plages horaires sont réservées préalablement aux chirurgiens et les chirurgiens ne traitent leurs interventions que dans

leurs plages horaires réservées. Dès que les interventions sont affectées pour chaque jour aux plages horaires par l'étape de planification, elles ne peuvent plus être déplacées vers d'autres plages horaires de ce jour. En conséquence, nous sommes dans la même situation que si nous avions des salles opératoires spécialisées. Si la SSPI n'est pas prise en compte, le problème serait semblable à un problème d'ordonnancement sur une seule machine ce qui est un problème simple qui peut être facilement résolu en minimisant le temps de passage moyen des interventions avec un tri de type « SPT» (*Shortest Processing Time*, la durée opératoire la plus petite) [Baker, 1974]. Des modèles d'ordonnancement à machine unique avec d'autres objectifs comme la minimisation du retard total ont aussi été étudiés [Abdul-Razaq *et al.*, 1990][Sen *et al.*, 2003].

En réalité, le nombre de lits de réveil ne peut pas être considéré comme illimité même s'ils sont beaucoup moins critiques que les salles d'opération. Nous allons nous concentrer sur le problème d'ordonnancement à deux étages où il y a plusieurs salles d'opération spécialisées et plusieurs lits de réveil dans la SSPI en prenant en compte le fait que la durée de post-anesthésie d'un patient peut être partagée entre les salles d'opération et la SSPI. Cette condition n'est pas prise compte par Kharraja [2002] ni par Guinet et Chaabane [2003] qui ont traité le problème d'ordonnancement pour une salle d'opération et un lit de réveil.

Dans le cas simpliste où le bloc opératoire n'est constitué que d'une salle d'opération et d'un lit de réveil en SSPI et où le problème d'ordonnancement a pour objectif la minimisation de la somme pondérée du *makespan* du bloc opératoire et du *makespan* de la salle d'opération, nous constatons que ce modèle est NP_Complet (cf. annexe 2). Pour le démontrer, nous nous sommes basés sur les travaux de Sriskandarajah et Wagneur [1991], qui ont montré qu'un modèle classique de « flow shop » à deux machines (avec un temps d'opération sur un étage dépendant de celui de l'étage précédent) $F2\left|\phi,\phi,\phi,\{R1,P1\},\{t^{(2)}=t'^{(2)}+\omega(t^{(1)}-t'^{(1)})\}\right|f(C_{\max})$ est NP-complet. En conséquence, il est évident que l'extension de ce modèle au cas plus général d'un problème d'ordonnancement de type « block scheduling » $F2\left|\phi,\phi,\phi,\{Rn_1,Pn_2\},\{t^{(2)}=t'^{(2)}+\omega(t^{(1)}-t'^{(1)})\}\right|f(C_{\max})$ est lui aussi NP-complet.

2.2.2 Problème d'ordonnancement pour la stratégie de l'« open scheduling »

Dans le cas d'une stratégie d' « open scheduling », contrairement au « block scheduling », aucune plage horaire n'est attribuée préalablement aux chirurgiens. Les salles d'opération disponibles sont donc polyvalentes.

Dans ce cas, nous devons prendre en compte la disponibilité des chirurgiens pour la construction d'un programme opératoire réalisable et efficace en terme de coût de la durée d'ouverture du bloc opératoire : un chirurgien ne peut opérer qu'une intervention dans une salle d'opération dans un intervalle de temps. De plus, dans ce problème d'ordonnancement, les durées de post-anesthésie peuvent aussi se partager entre les salles d'opération et la SSPI.

Nous sommes face ici à un des problèmes d'ordonnancement les plus complexes dans un bloc opératoire. Dans le cas plus simpliste où tous les lits dans la SSPI seraient disponibles, le problème s'apparenterait alors à un problème de « machines parallèles » avec des contraintes de ressources supplémentaires. Selon Ullman [1976] (présenté dans [Lenstra et Rinnooy Kan, 1980]**)**, Kellerer et Strusevich [2003, 2004], c'est un problème NP-complet. De plus, Sriskandarajah et Wagneur [1991] ont montré que le

problème $F2|\phi,\phi,\phi,\{P1,P1\},\{t^{(2)} = t'^{(2)} + \omega(t^{(1)} - t'^{(1)})\}|f(C_{\max})$ est NP-complet et par conséquent, il est évident que le problème d'ordonnancement pour la stratégie de l'« open scheduling » pour $F2|\phi,res,N_{chir}\ 1\ 1,\{Pn_1,Pn_2\},\{t^{(2)} = t'^{(2)} + \omega(t^{(1)} - t'^{(1)})\}|f(C_{\max})$ est NP-complet ; où N_{chir} représente le nombre des chirurgiens présents ce jour d'ordonnancement, $f(C_{\max})$ la somme pondérée du *makespan* du bloc opératoire et du temps d'achèvement de la dernière intervention des salles d'opération, $\beta_3 = \{N_{chir}\ 1\ 1\}$ signifie que chaque chirurgien ne peut opérer qu'une seule intervention à la fois et que chaque intervention ne nécessite qu'un seul chirurgien.

A notre connaissance, seuls Jebali *et al.* [2003, 2004, 2006] ont traité le problème d'ordonnancement du bloc opératoire en prenant en compte la contrainte de la disponibilité des chirurgiens, le nombre des lits dans la SSPI et le fait qu'un patient doive rester dans sa salle d'opération si aucun lit n'est disponible dans la SSPI (contrainte de blocage). Cependant, le dernier modèle développé par Jebali *et al.* [2006] est un modèle mathématique en nombres entiers limité à quinze interventions journalières, 3 salles d'opération, 4 chirurgiens et 4 lits de réveil. Or dans la réalité hospitalière, nous rencontrons fréquemment des problèmes de taille plus importante. Citons, par exemple, le bloc opératoire du Centre Hospitalier Universitaire de Tivoli, un hôpital belge de taille moyenne, qui se compose de huit salles d'opération et dix lits de réveil et où environ 8500 interventions annuelles y sont réalisées (environ 30 interventions par jour). De plus, Jebali *et al.* [2006] présument que les chirurgiens peuvent opérer n'importe quel patient. Or dans la réalité, le patient qui consulte un chirurgien, désire aussi être opéré par celui-ci.

2.3 Etat de l'art sur l'ordonnancement de « flow shop » hybride

A notre connaissance, aucun travail de recherche n'a été réalisé pour résoudre un problème d'ordonnancement des interventions, de taille réelle, dans un bloc opératoire constitué de plusieurs salles d'opération et de plusieurs lits dans la salle de réveil. De ce fait, nous sommes amenés à établir des analogies avec le modèle de « flow shop » hybride à deux étages avec contrainte des ressources supplémentaires renouvelables. Malheureusement, les problèmes de « flow shop » hybride avec contrainte des ressources renouvelables sont des problèmes parmi les plus complexes dans le domaine industriel et donc très peu de travaux y sont consacrés. Nous nous basons sur les résultats obtenus sur des modèles variants, c'est-à-dire, les modèles de « flow shop » hybride avec contraintes supplémentaires pour lesquels aucun temps de changement n'est pris en compte puisque c'est notre cas dans le domaine hospitalier.

Le problème de « flow shop » hybride est l'un des problèmes les plus difficiles et également les plus rencontrés dans le domaine industriel. Beaucoup de chercheurs l'ont étudié. Gupta, en 1988, a ouvert le champ d'études des « flow shop » hybrides et prouvé que le cas le plus simple de ce problème (deux étages avec une machine à un étage et deux à l'autre) est NP-complet. Depuis, de nombreuses recherches ont été menées et les méthodes et techniques utilisées s'étendent des heuristiques (comme le premier arrivé, le premier servi (FCFS), le « Best Fit Descending » etc.), aux méta-heuristiques (comme le recuit simulé, la recherche Tabou, les algorithmes génétiques, etc.) et aux méthodes hybrides en passant par des méthodes exactes avec comme objectif la minimisation du *makespan* ou le temps de séjour total (ou moyen) des travaux ou du nombre des travaux en retard ou encore des temps de retard totaux, etc. Par exemple, Allaoui et Artiba [2006] ont étudié un modèle de « flow shop » hybride, une machine au premier étage et plusieurs

machines au deuxième étage, avec des contraintes de disponibilité des machines. Ils l'ont résolu par une procédure de Branch-and-Bound. Pirlot et Teghem [2003] ont présenté un problème de type « hoist scheduling », proche de notre problème, où les machines sont des cuves dans lesquelles les produits trempent, la durée des opérations est comprise entre une durée minimale et une durée maximale, il n'y a pas de tampon intermédiaire entre deux étages successifs, et les produits sont déplacés au moyen de crochets manipulés par un ou plusieurs robots dont chacun ne peut déplacer qu'un seul produit à la fois. Les robots peuvent être considérés comme des ressources supplémentaires. Pirlot et Teghem ont utilisé deux approches génétiques proposées par Lim [1997] et Matile *et al.* [1999] pour résoudre le problème. Malheureusement, celui-ci ne prenait en compte qu'une cuve à chaque étage et un seul robot ; donc, c'est un problème de « flow shop » avec une seule contrainte de ressource renouvelable. D'autre part, Gupta *et al.* [2002] ont proposé des heuristiques pour résoudre un problème de « flow shop » hybride où les durées opératoires des tâches varient en fonction de la ressource supplémentaire.

Dans le Tableau 10, nous avons classifié les études récentes, dans le domaine des problèmes d'ordonnancement de type « flow shop » hybride, selon la fonction objectif. Pour plus de détails, Chen [1995], Kis et Pesch [2005], Linn et Zhang [1999], Negenman [2001], Vignier *et al.* [1999] et Wang [2005] ont réalisé un état de l'art pour les études centrées sur les problèmes d'ordonnancement de « flow shop » hybride.

Objectif / Méthodes	*Makespan*	Temps de séjour	Retard	Autre objectif
Heuristiques	[Brah et Loo, 1999] [Ding et Kittichartphayak, 1994] [Guinet *et al.*, 1996] [Guinet et Solomon, 1996] [Gupta, 1988] [Gupta *et al.*, 1997] [Gupta et Tunc, 1991, 1994] [Havill et Mao, 1998] [Hunsucker et Shah, 1994] [Kyparisis et Koulamas, 2006] [Lee et Vairaktarakis 1994] [Logendran *et al.*, 2005] [Nawaz *et al.*, 1983] [Oguz *et al.*, 2003] [Riane *et al.*, 1998, 2002] [Sriskandarajah et Sethi, 1989] [Su, 2003] [Wang et Hunsucker, 2003]	[Azizglu *et al.*, 2001] [Brah et Loo, 1999] [Guirchoun et Martineau, 2002] [Nawaz *et al.*, 1983]	[Botta-Genoulaz, 2000] [Guinet et Solomon, 1996] [Gupta *et al.*, 2002]	[Aghezzaf et Artiba, 1998] [Bertel et Billaut, 2004]
Méta-Heuristiques	[Amor *et al.*, 2005] (algo. génétiques) [Engin et Döyen, 2004] (système immunitaire artificiel) [NowicKi et Smutnicki, 1998] (recherche Tabou) [Pirlot et Teghem, 2003] (algo.génétiques) [Ulusoy et Sivrikarya-Serifoglu, 1998] (algo. génétique) [Wardono et Fathi] (recherche Tabou)			[Bertel et Billaut, 2004] (algo génétique)
Méthodes hybrides	[Brockmann et Dangelmaier, 1998] (Branch and Bound + Heuristiques) [Haourai et M'Hallah, 1997] (recuit simulé + recherche Tabou)	[Low, 2005] (heuristique + recuit simulé)	[Kreutz *et al.*, 2000] (algorithme génétique +	[Kreutz *et al.*, 2000] (algorithme génétique +

	[Jin *et al.*, 2005] (Heuristiques + recuit simulé) [Kreutz *et al.*, 2000] (algo. génétique + heuristique) [Portmann *et al.*, 1998] (Branch-and-Bound + algo.génétique) [Sevaux *et al.*, 2005] (algo. génétique + programmation de contraint) [Suresh, 1997] (deux heuristiques)		heuristique)	heuristique)
Méthodes exactes (Branch-and-Bound)	[Allaoui et Artiba, 2006] [Arthanari et Ramamurthy, 1971] [Brah et Hunsucker, 1991] [Moursli et Pochet, 2000] [Néron *et al.*, 2001] [Vignier et Venturini., 1996]	[Azizglu *et al.*, 2001] [Yokoyama, 2001]		
Autres	[Santos *et al.*, 1995] (proposent une borne inférieure globale)			[Riane *et al.*, 2001](simul.)

Tableau 10 : Etat de l'art pour la résolution des problèmes de « flow shop » hybride

En général, nous trouvons que la plupart des études est basée sur les heuristiques, méta-heuristiques ainsi que les méthodes hybrides pour obtenir des solutions acceptables vu que ces méthodes permettent de trouver des solutions acceptables en temps raisonnable pour des problèmes NP-difficiles et pour le cas le plus simple du problème de type « flow shop » qui est déjà NP-complet.

Dans la section 3, nous allons présenter les principales méthodes de résolution de problèmes d'ordonnancement d'un « flow shop » hybride, qui nous seront utiles pour résoudre notre problème d'ordonnancement hospitalier.

3. Méthodes développées pour résoudre les problèmes d'ordonnancement de « flow shop » hybride

3.1 Méthodes principales de résolution

Nous allons présenter brièvement les méthodes principales de résolution en suivant une classification en quatre catégories décrite par Duvivier [2000] :

- **Méthodes exactes** : elles fournissent systématiquement une solution optimale (avec parfois la garantie de l'optimalité) si une telle solution existe ou affirment qu'il n'existe pas de solution au problème traité ;
- **Méthodes de relaxation** : elles fournissent une solution approchée, en relâchant quelques contraintes difficiles (par exemple, les contraintes d'intégrité) afin d'obtenir un modèle plus facile à résoudre par rapport au problème d'origine. Normalement, les solutions des problèmes relâchés ne constituent pas des solutions réalisables du problème d'origine mais les solutions obtenues peuvent servir de bornes inférieures ou supérieures de la valeur optimale du problème d'origine ;
- **Méthodes heuristiques** : elles permettent d'obtenir une solution acceptable au problème en un temps raisonnable, c'est-à-dire une solution optimale ou une solution réalisable, dont la valeur de la fonction objectif est suffisamment proche de la valeur optimale. En général, les méthodes heuristiques ne peuvent pas garantir la qualité de la solution obtenue ;

- **Méthodes hybrides** : elles sont des méthodes de recherche combinant au moins deux méthodes de recherche distinctes afin de renforcer les avantages des méthodes les composant et d'affaiblir leurs inconvénients.

Pour les problèmes NP-complets, il est peu probable que nous puissions systématiquement trouver une solution optimale au moyen d'algorithmes polynomiaux ([Garey et Johnson, 1979]). Donc, pour résoudre ceux-ci, il faut utiliser

- soit des méthodes énumératives, comme la méthode de « *Branch-and-Bound* », qui cherchent l'optimum parmi l'espace de toutes les solutions possibles et où des méthodes de relaxation sont appliquées pour obtenir de bonnes bornes inférieures ou supérieures afin de réduire l'espace de recherche ;
- soit des méthodes heuristiques ou hybrides afin d'obtenir des solutions acceptables en un temps raisonnable.

Vu que le temps de calcul et la mémoire du calculateur consommés par une méthode énumérative dépendent de la structure de l'espace des solutions et augmentent de manière exponentielle avec la taille du problème, nous nous intéressons particulièrement aux méthodes hybrides.

3.2 Méthodes méta-heuristiques

Les méthodes heuristiques peuvent encore être subdivisées en deux catégories : les heuristiques dédiées et les méta-heuristiques. Les heuristiques dédiées sont proposées pour résoudre des problèmes bien spécifiques. Ces heuristiques donnent donc généralement de piètres résultats lorsqu'elles sont appliquées à d'autres types de problèmes. Par ailleurs, les méta-heuristiques, recherchant une solution acceptable dans un espace de solutions réalisables, peuvent être utilisées. Il s'agit de méthodes génériques optimisant une large gamme de problèmes différents, sans nécessiter de changements importants dans l'algorithme employé. Ces méthodes ont en commun, en outre, les caractéristiques suivantes [Dréo et al., 2003] :

- elles sont, au moins pour partie, stochastiques : cette approche permet de faire face à l'explosion combinatoire des possibilités ;
- elles sont souvent inspirées par des analogies : avec la physique (recuit simulé…), avec la biologie (algorithmes évolutionnaires, colonies de fourmis, essaims particulaires, recherche tabou…) ;
- elles utilisent la mémoire d'itérations précédentes sous différentes formes, par exemple la meilleure solution actuelle, la population courante etc. ;
- elles partagent aussi les mêmes inconvénients : les difficultés de réglage des paramètres de la méthode et le temps de calcul élevé.

Dans l'état actuel de la recherche, il est le plus souvent impossible de prévoir avec certitude l'efficacité ou la rapidité d'une méthode donnée, quand elle est appliquée à un problème donné. La tendance actuelle est donc l'émergence de méthodes hybrides, qui s'efforcent de tirer parti des avantages spécifiques d'approches différentes en les combinant.

Les méta-heuristiques généralement comprennent trois mécanismes : l'intensification, l'apprentissage et la diversification. L'intensification vise à améliorer une solution de bonne qualité en explorant les solutions voisines. L'apprentissage permet à l'algorithme de tirer parti des informations contenues dans les solutions qui ont déjà été visitées. La diversification permet à la méthode d'explorer des zones de l'espace des solutions qui ont été peu ou pas visitées. Les méta-heuristiques les plus courantes sont basées sur un algorithme itératif de recherche locale nécessitant l'introduction d'un opérateur de

voisinage permettant de définir la notion de solution voisine et de voisinage d'une solution. Dans cette classe de méta-heuristiques, nous trouvons notamment le recuit simulé [Kirkpatrick *et al.*, 1983] et la recherche Tabou [Glover, 1986] [Hansen, 1986]. D'autres méta-heuristiques utilisent la notion de population et manipulent donc un ensemble de solutions en parallèle, à chaque itération. Dans cette catégorie, se trouvent les algorithmes génétiques [Holland, 1962][Goldberg, 1994], les essaims particulaires [Eberhart et Kennedy, 1995], les colonies de fourmis [Dorigo *et al.*, 1991] etc..
Après un bref récapitulatif du principe de fonctionnement des algorithmes itératifs de recherche locale, la méthode Tabou et les algorithmes génétiques sont expliquées et ensuite, nous les combinerons afin d'obtenir un algorithme hybride permettant à la fois la diversification et l'intensification pour améliorer significativement les solutions obtenues et traiter des problèmes de plus grande taille.

3.2.1 Principe général de fonctionnement de l'algorithme itératif de recherche locale

Le principe de fonctionnement d'un algorithme itératif de recherche locale est le suivant :

Etape 1 : Tout d'abord, créer une solution initiale s_0, qui est normalement une solution réalisable du problème traité et mettre cette solution initiale obtenue comme la solution courante s ;
Etape 2 : A partir de la solution courante s, générer un ensemble de voisins de s afin de construire son voisinage (ou une partie) $V(s)$;
Etape 3 : Chercher parmi le voisinage généré $V(s)$, s'il y a un voisin s' dont la valeur de la fonction objectif est meilleure que celle de la solution courante s, mettre $s = s'$;
Etape 4 : Si la condition d'arrêt est satisfaite, l'algorithme se termine et la solution courante s est la solution finale obtenue ; sinon, retourner à l'étape 2.

Cette méthode de recherche est une heuristique très simple qui requiert peu de temps de calcul et de mémoire. Cependant, elle aboutit facilement à un optimum local et n'en sort pas. En fait, en intégrant des mécanismes complémentaires dans l'algorithme itératif de recherche locale, nous obtenons un ensemble de méta-heuristiques telles que le recuit simulé, la recherche Tabou, ou les algorithmes génétiques. Ces méthodes permettent de trouver des solutions acceptables en un temps raisonnable.

Avant tout, présentons quelques définitions qui nous seront utiles :

Définition 3.2.1_1 (mouvement)

Un mouvement est une modification apportée à une solution. Elle consiste à modifier une partie de cette solution courante. Il permet de définir un voisinage pour une solution.

Il existe une grande variété de voisinage plus ou moins efficace selons les problèmes. Naturellement, selon le problème à résoudre, des voisinages plus élaborés tenant mieux compte de la structure des bonnes solutions peuvent être envisagés. La Figure 14 nous montre trois types de mouvement. Le voisinage obtenu du premier type de mouvement est le plus restreint puisque de taille n-1. Le deuxième type de mouvement définit un voisinage comprenant $n(n-1)/2$ voisins et le troisième voisinage est de taille $n(n-2)$ +1. Les capacités de ces divers types de voisinages à se diriger en peu d'itérations vers de bonnes solutions sont très différentes et variables selon le type d'application. Le premier type est le plus approprié dans le cas du problème du voyageur de commerce (TSP). Le second peut être utilisé dans le cas du problème d'affectation quadratique (QAP). Alors

que pour des applications en ordonnancement, c'est le troisième type qui est le meilleur [Taillard 1990]. En conséquence, dans cette thèse, nous utilisons le troisième type de mouvement pour obtenir les solutions voisines.

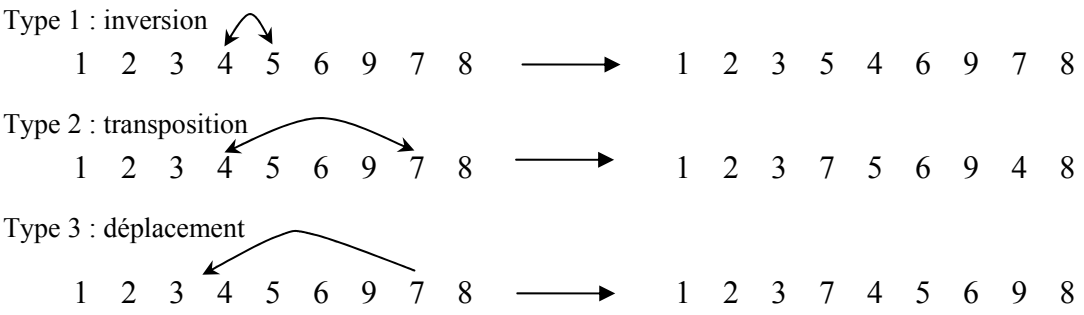

Figure 14 : Trois types de mouvement possible pour obtenir un voisin dans le cas d'une permutation

Définition 3.2.1_2 (Voisin, voisinage)
Un voisin de la solution courante *s*, ou une solution voisine, est une solution accessible en un seul mouvement élémentaire à partir de *s*. Le voisinage de *s* est l'ensemble de ces voisins obtenus, noté *V(s)*.

Notons ici que si le voisinage concerné est assez grand, nous peuvons se contenter de choisir la meilleure solution parmi une partie du voisinage de la solution, cet ensemble pouvant être sélectionné aléatoirement. Ce mécanisme permet d'aboutir à un compromis entre les coûts d'exécution et la qualité des mouvements locaux.

Définition 3.2.1_3 (condition d'arrêt)
La condition d'arrêt est un critère sur lequel se base la méthode pour dire quand une procédure se termine. La condition d'arrêt peut être, par exemple, un nombre d'itérations ou la convergence des solutions.

3.2.2 Recherche Tabou

Initialement élaborée par Glover [1986] et indépendamment par Hansen [1986] la même année, la recherche Tabou utilise un mécanisme afin d'échapper aux optima locaux. Le principe de ce mécanisme est de permettre de choisir un voisin dont la valeur de la fonction objectif est la meilleure dans le voisinage mais pas obligatoirement meilleure que la solution courante. Tel quel, le mécanisme précédent crée un risque important de retourner à une configuration déjà retenue lors d'une itération précédente, ce qui engendre un cycle. Pour éviter ce phénomène, la recherche Tabou tient à jour et exploite, à chaque itération. Une liste des derniers mouvements effectués est utilisée pour empêcher le « cyclage » (refaire à l'infini les mêmes mouvements). En même temps, un mécanisme d'aspiration est utilisé pour lever l'interdiction d'utilisation d'un mouvement si ce dernier conduit à une meilleure solution.

Présentons quelques définitions importantes :

Définition 3.2.2_1 (Mouvement Tabou)
Un mouvement Tabou est un mouvement interdit.

Définitions 3.2.2_2 (Liste Tabou)

La liste Tabou est une structure de données, dans laquelle les mouvements Tabous sont mémorisés, servant à interdire de revenir sur les mouvements précédents pendant un certain nombre d'itérations. La liste Tabou est mise à jour à chaque itération selon une règle de premier arrivé premier sorti (First In First Out, FIFO).

La taille de la liste Tabou constitue un paramètre ajustable de la méthode et elle peut varier selon les caractéristiques du problème considéré. Il est clair qu'il est important de choisir une taille appropriée : une liste Tabou contenant trop peu d'éléments peut s'avérer inutile et mener à des mouvements cycliques, mais une liste Tabou avec trop d'éléments peut devenir très restrictive. Il a été observé que trop de contraintes (tabous) forcent le programme à visiter des solutions voisines peu intéressantes à la prochaine itération, autrement dit, qu'il peut interdire des mouvements intéressants qui nous auraient conduits vers de nouvelles solutions de bonne qualité. Les règles déterminant la taille de la liste Tabou peuvent être statiques ou dynamiques selon les auteurs [Gadenne, 2005][Glover *et al.*, 1992][Glover et Laguna, 1992].

Les règles statiques choisissent une valeur fixe pour la taille, qui va normalement de 7 à 20. Les règles dynamiques font varier la valeur de la taille de la liste au fur et à mesure des itérations. Les expériences pratiques indiquent que généralement les règles dynamiques sont plus robustes que les règles statiques mais beaucoup plus complexes.

Au début de la procédure de recherche Tabou, la liste Tabou est initialisée à vide et ensuite se remplit pendant le déroulement de la procédure. Lorsque la liste est pleine, alors son élément le plus ancien sera remplacé par le nouveau.

Etant donné que ce mécanisme de liste Tabou, dans certains cas, pourrait interdire un mouvement qui aurait pu conduire à un voisin meilleur que les solutions rencontrées dans les itérations précédentes, la recherche Tabou est dotée d'un critère d'aspiration qui lève le statut Tabou d'un mouvement si ce dernier conduit à une solution intéressante.

Définition 3.2.2_3 (Critère d'aspiration)

Le critère d'aspiration est la condition nécessaire pour qu'un mouvement Tabou soit accepté comme prochain mouvement, malgré son statut Tabou. Par exemple, un mouvement qui mène à une solution meilleure que toutes celles visitées par la recherche dans les itérations précédentes n'a aucune raison d'être interdit, l'éventuel statut tabou est donc modifié pour ces mouvements, ces mouvements sont dit aspirés. Il est naturellement possible d'imaginer d'autres critères d'aspiration, basés moins directement sur la valeur de l'objectif à optimiser.

Définition 3.2.2_4 (Meilleure solution courante)

La meilleure solution courante représente la meilleure solution découverte depuis le début de la recherche en cours d'exécution.

Nous pouvons maintenant donner un exemple de critère d'aspiration : si un mouvement Tabou de la solution courante s peut générer un voisin s' de la solution courante s dont la valeur de fonction objectif est meilleure que celle de la meilleure solution courante s^*, nous acceptons le mouvement Tabou et le voisin s' comme la solution courante s et nous mettons $s^* = s'$.

Définition 3.2.2_5 (solution admissible)
Une solution admissible est une solution obtenue par un mouvement qui n'est pas interdit par la liste Tabou ou par un mouvement qui satisfait le critère d'aspiration.

Le principe de fonctionnement de l'algorithme de la recherche Tabou, comme montré par la Figure 15, est :

Etape 1 : Tout d'abord, créer une solution initiale s_0, qui est normalement une solution réalisable du problème traité et mettre la solution courante $s = s_0$;

Etape 2 : A partir de la solution courante *s*, générer une liste des mouvements candidats, $V(s)$, dont chacun peut conduire à une solution admissible voisine de *s* en prenant en compte les contraintes de la liste Tabou et le critère d'aspiration ;

Etape 3 : Choisir la meilleure solution admissible voisine $s' \in V(s)$ et ensuite mettre $s = s'$;

Etape 4 : Si aucune condition d'arrêt n'est satisfaite, mettre à jour la liste Tabou et le critère d'aspiration; ensuite, retour à l'étape 2 ;

Etape 5 : Si une des conditions d'arrêt est atteinte, la recherche Tabou se termine et la meilleure solution courante est choisie comme la solution finale obtenue.

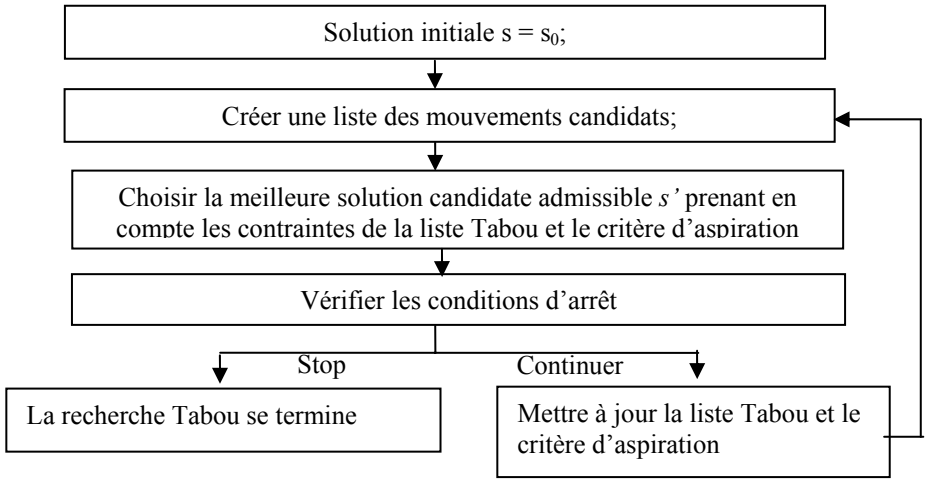

Figure 15 : Schéma de base de la recherche Tabou [Glover, 1990]

Dans cette procédure, une étape critique est la sélection de la meilleure solution admissible candidate (l'étape 3). Comme montré par la Figure 16, la sélection de la solution admissible candidate se déroule comme ci-dessous :

Etape A : Tout d'abord, évaluer un des mouvements candidats ;

Etape B : Ensuite, vérifier le statut tabou de la solution voisine obtenue par le mouvement candidat courant selon les contraintes de la liste Tabou courante ;

Etape C : Si la solution voisine courante n'est pas interdite par la liste Tabou, elle est acceptée comme une solution admissible ;
Si la solution voisine courante est interdite par la liste Tabou, vérifier son statut d'aspiration ;

Etape D : Si la solution voisine courante satisfait le critère d'aspiration, elle est acceptée comme une solution admissible ;

Etape E : Tester la liste candidate. S'il y a encore des mouvements candidats à évaluer, aller à l'étape A ;

Etape F : Choisir, parmi tous les mouvements candidats, le mouvement qui peut générer la meilleure solution admissible.

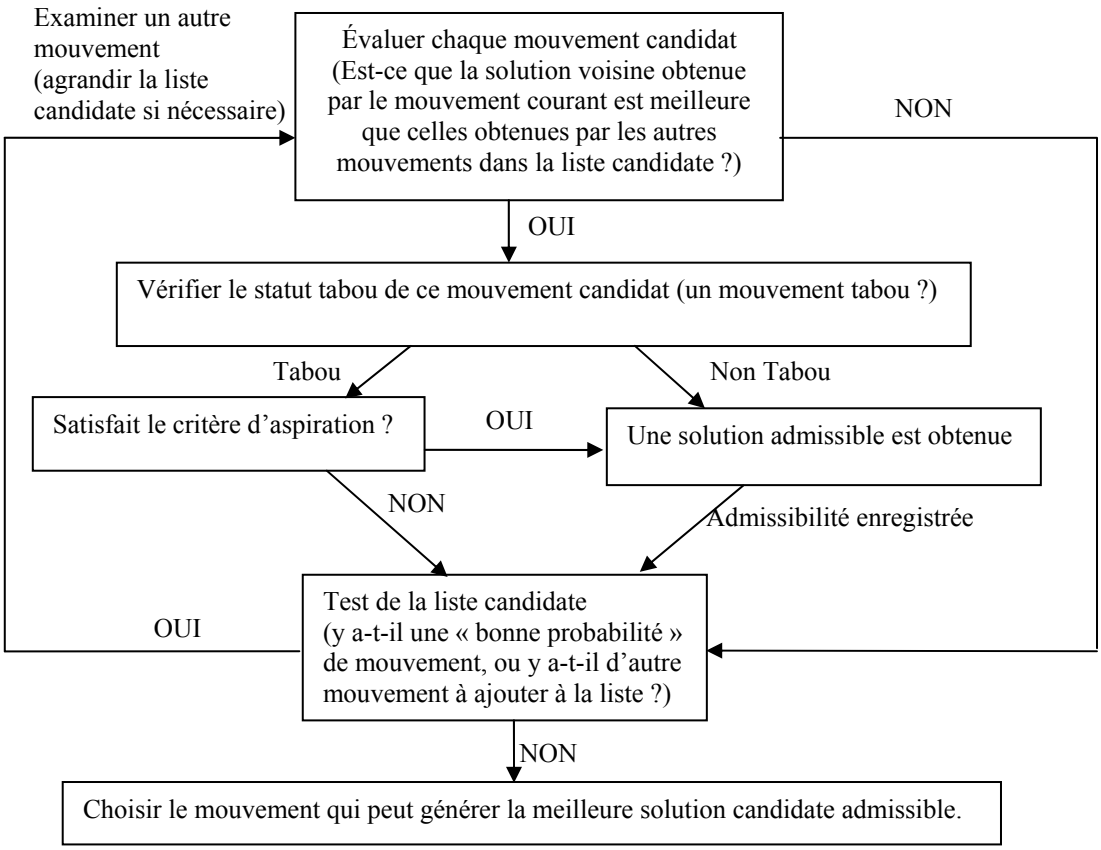

Figure 16 : Processus pour sélectionner la meilleure solution candidate admissible

La recherche Tabou est reconnue comme étant une heuristique donnant de très bons résultats pour bon nombre de problèmes d'optimisation complexe tels que le voyageur de commerce (TSP) [Hasegawa *et al.*, 2002], le coloriage de graphes [Galinier et Hertz, 2005], les tournées de véhicules (VRP) [Ho et Haugland, 2004], les problèmes d'ordonnancement [Gupta *et al.*, 2002][Janiak et al, 2005][Negenman, 2001] [Nowicki et Smutnicki, 1998] [Oguez *et al.*, 2004].

3.2.3 Algorithmes génétiques

Développés par Holland [1962], les algorithmes génétiques (Genetic Algorithm, GA) tentent de simuler le processus d'évolution naturelle en suivant le modèle darwinien [Darwin, 1859] dans un environnement donné. Ils utilisent un vocabulaire similaire à celui de la génétique naturelle. Cependant, les processus naturels, auxquels ils font référence, sont beaucoup plus complexes.

Définition 3.2.3_1 (Codage)

Etant donné l'ensemble des configurations possibles par la structure des données manipulée par les algorithmes génétiques, un codage est l'établissement d'une fonction injective de l'ensemble des solutions à l'ensemble des configurations, c'est-à-dire qu'un codage doit confirmer que deux solutions différentes du problème concerné correspondent nécessairement à deux configurations différentes.

Les codages peuvent être répartis en deux catégories principales :

- Le codage indirect : La structure de données manipulée par l'algorithme génétique ne représente pas directement une solution du problème traité : soit ne contient pas toutes les informations nécessaires à la reconstitution d'une solution réalisable ; soit les données sont exprimées sous une forme non directement exploitable pour évaluer la solution ;
- Le codage direct : La structure de données manipulée par l'algorithme génétique représente exactement une solution au problème. L'algorithme génétique dispose de toutes les informations sur les solutions, puisqu'elles sont directement intégrées dans la structure de données manipulée.

Les codages peuvent être classifiés selon le type de code qu'ils utilisent [Duvivier, 2000] : le codage binaire, le codage de permutation, le codage de valeur, le codage d'arbre, etc. Pour les problèmes d'ordonnancement, le codage de permutation, qui représente la permutation des tâches sur les machines, est généralement utilisé [Portmann *et al.*, 1988][Pirlot et Teghem, 2003] etc.

Définition 3.2.3_2 (Individu)
Un individu représente une solution potentielle dans l'espace de recherche du problème concerné. Par analogie avec la génétique, un individu est codé par un ou plusieurs chromosomes constitués de gènes qui contiennent les caractères héréditaires de l'individu. Pour l'algorithme génétique standard, un individu est composé d'un chromosome codé par une chaîne de bits.

Définition 3.2.3_3 (Population)
Une population est un groupe d'individus sur lequel les algorithmes génétiques effectuent un certain nombre d'opérations.

Définition 3.2.3_4 (Génération)
Une génération est une itération de la boucle de base d'un algorithme génétique.

Définition 3.2.3_5 (Parent)
Un parent est un individu de la génération courante choisi par un opérateur de sélection pour générer des nouveaux individus.

Définition 3.2.3_6 (Enfant)
Dès que deux parents sont choisis dans une génération courante, les opérateurs de croisement ou de mutation vont opérer afin d'obtenir un ou plusieurs nouveaux individus, appelés « enfants ».

Définition 3.2.3_7 (Croisement)
Un opérateur de croisement est une combinaison de deux parents pour former un ou deux enfants.

Définition 3.2.3_8 (Mutation)
L'opérateur de mutation est la modification élémentaire d'un individu.

Définition 3.2.3_9 (Indicateur d'adaptation ou fonction de performance)
L'indicateur d'adaptation est calculé sur la base de la valeur de la fonction objectif des parents (et les enfants générés) de la génération courante et va permettre aux individus ayant les meilleurs valeurs de fonction objectif d'avoir une plus grande probabilité d'être choisis par le processus de sélection. Cet indicateur est, par exemple, l'écart entre la valeur de la fonction objectif de l'individu en question et la pire valeur de fonction

objectif de la génération courante. L'indicateur d'adaptation est un cas particulier dans le domaine pour l'algorithme génétique.

Définition 3.2.3_10 (Convergence)
Nous disons qu'une population converge lorsque 95% des individus de la population ont la même valeur de fonction objectif.

Contrairement à la recherche Tabou, les algorithmes génétiques n'agissent pas sur un individu isolé mais sur une population. En général, les algorithmes génétiques cherchent, à partir d'une population initiale (génération 0), évoluant durant une succession d'itérations jusqu'à ce qu'une condition d'arrêt, à trouver une solution proche de l'optimum ou même optimale. Durant chaque génération, une succession d'opérateurs (l'opérateur de sélection, de croisement et de mutation) est appliquée aux individus d'une population pour engendrer la nouvelle population à la génération suivante. En conséquence, pour utiliser un algorithme génétique sur un problème d'optimisation, nous devons disposer d'un principe de codage afin de générer la population initiale; une technique acceptable de sélection pour choisir les parents ou reproduire la prochaine génération, et d'opérateurs de croisement et de mutation permettant de diversifier la population au cours des générations et d'explorer l'espace de recherche. Bien sûr, les individus obtenus avec une bonne qualité (c'est-à-dire avec une bonne valeur de fonction objectif) peuvent renforcer la performance de l'algorithme génétique.

Jusqu'à présent, beaucoup de variantes de l'algorithme génétique ont été proposées pour résoudre les problèmes d'optimisation, mais tous les algorithmes génétiques sont basés sur trois composantes qui constituent un algorithme génétique simple : opérateurs de sélection, de croisement et de mutation.

Opérateurs de sélection

A chaque génération, des individus se reproduisent, survivent ou disparaissent de la population sous l'action de deux opérateurs de sélection :

- La sélection pour la reproduction qui détermine combien de fois un individu sera reproduit en une génération ;
- La sélection pour le remplacement qui détermine quels individus devront disparaître de la population à chaque génération de façon que, de génération en génération, la taille de la population reste constante, ou plus rarement, soit contrôlée selon une politique définie.

En toute généralité, la capacité d'un individu à être sélectionné, que ce soit pour la reproduction ou le remplacement, dépend de sa performance. L'opérateur de sélection est ainsi chargé de déterminer un nombre de sélections pour chaque individu en fonction de sa performance.

Il existe plusieurs techniques de sélection [Goldberg, 1994][Dréo *et al.*, 2003] dont les principales sont :

- Sélection selon le rang des individus : cette technique de sélection choisit toujours les individus possédant les meilleurs scores d'adaptation, qui ne dépendent que des rangs de l'individu par valeur décroissante, le hasard n'entre donc pas dans ce mode de sélection ;
- Sélection proportionnelle : ce type de sélection a été conçu à l'origine par J. Holland pour les algorithmes génétiques. Elle n'est utilisée que pour la reproduction. Pour chaque individu, la probabilité d'être sélectionné est proportionnelle à sa performance. Afin de sélectionner un individu de cette

manière, deux techniques principales sont utilisées : la méthode de roulette (Roulette Wheel Selection, RWS), proposée à l'origine des algorithmes génétiques, qui exploite la métaphore d'une roulette de casino, qui comporterait autant de cases que d'individus dans la population et où la taille de ces cases serait proportionnelle à la performance de chaque individu; et la méthode d'échantillonnage stochastique universel (Stochastic Universal Sampling, SUS) [Baker, 1987], qui considère toujours un segment de droite partitionné en autant de zones qu'il y a d'individus dans la population, chaque zone étant de taille proportionnelle à la performance. En fait, la variance du processus avec la méthode SUS est plus faible qu'avec la méthode RWS ;

- Sélection par tournois : le tournoi le plus simple consiste à choisir aléatoirement un nombre k d'individus dans la population et à sélectionner, pour la reproduction, celui qui a la meilleure performance. Les individus qui participent à un tournoi sont remis dans la population ou en sont retirés, selon le choix de l'utilisateur. Avec le tournoi binaire stochastique, sur deux individus en compétition, le meilleur gagne avec une probabilité p comprise entre 0,5 et 1 ;
- Sélection uniforme : la sélection se fait aléatoirement, uniformément et sans intervention de la valeur d'adaptation. Chaque individu a donc une probabilité $1/P$ d'être sélectionné, où P est le nombre total d'individus dans la population.
- Sélection déterministe : cette sélection ne fait que choisir les n meilleurs individus d'une population, n étant un paramètre que l'utilisateur doit fixer. Si l'opérateur est utilisé dans le cadre de la sélection pour la reproduction, pour engendrer λ enfants à partir des n parents choisis, chaque parent aura λ/n enfants. Si l'opérateur est utilisé pour le remplacement et engendre ainsi la population de μ individus à la génération suivante, alors $n = \mu$.

A l'exception des opérateurs de sélection décrits ci-dessus, il existe aussi des opérateurs qui ne sont utilisés que pour le remplacement :

- Remplacement générationnel : ce type de remplacement est le plus simple, puisque la population des parents à la génération suivante sera constituée par tous les enfants, et seulement eux, engendrés à la génération courante. Les algorithmes génétiques d'origine utilisent un remplacement générationnel ;
- Remplacement des stratégies d'évolution « (μ,λ) -ES» : une sélection déterministe des meilleurs μ individus parmi λ enfants forme la population à la génération suivante ;
- Remplacement stationnaire : à chaque génération, un petit nombre de descendants (un ou deux) sont engendrés et remplacent un nombre inférieur ou égal de parents, pour former la population à la génération suivante. Cette stratégie est spécialement utile lorsque la représentation d'une solution se répartit sur plusieurs individus, éventuellement la totalité de la population. Le choix des parents remplacés obéit à des critères variés. Avec le remplacement uniforme, les parents remplacés sont désignés au hasard. Le choix peut aussi dépendre de la performance : le parent le moins performant est remplacé, ou alors il est choisi stochastiquement, selon une distribution de probabilité dépendant de la performance ou d'autres critères ;
- Elitisme : une stratégie élitiste consiste à conserver dans la population, d'une génération à l'autre, au moins l'individu ayant la meilleure performance.

Opérateurs de croisement et de mutation

Les opérateurs de croisement et de mutation sont deux composants essentiels de l'algorithme génétique. Ils permettent de créer de la nouveauté dans une population en

construisant des individus « enfants », qui héritent en partie des caractéristiques d'individus « parents ».

A. Croisement

L'opérateur de croisement utilise deux parents pour former un ou deux enfants. L'opérateur est généralement stochastique, dans la mesure où le croisement répété d'un même couple de parents distincts donnera des enfants différents.

Selon la littérature, plusieurs opérateurs de croisement sont proposés comme croisement en un point (*one-point crossover*), croisement en deux points (*two-point crossover*), PMX (*partial-mapped crossover*), OX(*order crossover*), CX (*cycle crossover*), LOX (*linear order crossover*), SXX (*subsequence exchange crossover*), JOX (*job-based order crossover*), ER(*Edge recombination crossover*), *uniforme crossover* etc. Pour les problèmes d'ordonnancement, normalement les opérateurs PMX, OX, et ER sont souvent appliqués. De plus, il est possible de les utiliser dans des proportions différentes à l'intérieur d'une même population dans le but d'introduire de la diversité [Goldberg, 1989].

B. Mutation

Classiquement, l'opérateur de mutation modifie aléatoirement un individu pour en former un autre qui le remplacera. Comme pour les croisements, de nombreuses méthodes de mutation ont été développées dans la littérature du domaine. Par exemple, la mutation la plus simple consiste à inverser les deux gènes choisis aléatoirement ; il est aussi possible de réarranger l'ordre des gènes entre deux positions choisies afin d'obtenir un nouvel individu, etc. De toute façon, la mutation évite la dégénérescence de la population. Cependant, pour les algorithmes génétiques, la mutation est vue comme un opérateur mineur, chargé de maintenir un minimum de diversité dans la population, ce que ne peut pas assurer le croisement. En conséquence, la probabilité de mutation est typiquement faible, de l'ordre de 0,01 à 0,05. En fait, ce pourcentage peut aussi varier au cours du processus.

La plupart des mutations modifient un individu de telle façon que le résultat de la transformation lui soit proche. De cette façon, l'opérateur assure une recherche locale aléatoire autour de chaque individu et donc l'utilisation de la mutation, en tant qu'opérateur de recherche local, suggère de le combiner avec d'autres techniques locales plus efficaces, bien que davantage dépendantes de la nature du problème. Ce genre d'approche conduit à la conception d'algorithmes génétiques hybrides.

Nous avons présenté ci-dessus en détail les mécanismes des algorithmes génétiques de base avec trois opérateurs. Pour mieux comprendre le fonctionnement de l'algorithme génétique, nous présentons ci-dessus le principe de fonctionnement général d'un algorithme génétique simple, comme montré par [Glodberg, 1994].

Le principe de fonctionnement général de l'algorithme génétique :

Etape 1 : Génération aléatoirement d'une population. Celle-ci contient un pool génétique qui représente un ensemble d'individus (solutions possibles) ;

Etape 2 : Calcul d'une valeur d'adaptation pour chaque individu. Elle sera fonction directe de la proximité des différents individus avec l'objectif afin de confirmer que les meilleurs individus ont les plus grandes possibilités d'être choisis ;

Etape 3 : Sélection d'un couple de parents en utilisant une technique de sélection

selon leurs valeurs d'adaptation ;

Etape 4 : Croisement des génomes des parents avec une probabilité P_c pour générer deux enfants ;

Etape 5 : Mutation des deux enfants avec une probabilité P_m ;

Etape 6 : Répéter les étapes (3), (4), et (5) jusqu'à ce que la nouvelle population se produise ;

Etape 7 : Itérer à partir de l'étape (2) jusqu'à ce qu'une des conditions d'arrêt soit satisfaite.

3.3 Méthodes hybrides

Les méta-heuristiques sont applicables à un grand nombre de problèmes sans nécessiter de connaissances approfondies sur le problème traité. Elles permettent, en général, d'obtenir de bonnes performances, mais sur certains problèmes, ou certaines instances d'un problème, leurs performances ne sont parfois pas assez bonnes. Au contraire, les méthodes dédiées permettent d'obtenir d'excellents résultats pour certains problèmes, ou certaines instances particulières mais ces méthodes sont bien souvent difficilement généralisables à d'autres types de problèmes car elles exploitent toutes les spécificités du problème pour lequel elles se sont concentrées.

En fait, une « bonne » méthode de recherche ne doit pas uniquement être applicable à un grand nombre de problèmes mais elle doit également être performante, dans le sens de trouver une solution acceptable et robuste. L'idéal serait donc de combiner, en une méthode hybride, plusieurs méthodes de recherche, exactes, heuristiques dédiées ou méta-heuristiques. Pour cela, il faut savoir caractériser les points forts et également les points faibles des méthodes de recherche que nous désirons choisir pour construire une méthode hybride. Par exemple, les algorithmes génétiques s'avèrent très performants lorsqu'il s'agit d'explorer l'espace de recherche, mais ils sont incapables d'exploiter efficacement la zone vers laquelle la population converge, et donc il serait intéressant d'utiliser une autre méthode pour pallier cet inconvénient.

Etant donné que nos problèmes d'ordonnancement des interventions sont prouvés NP-complets et que les méthodes exactes seront inefficaces ou même incapables de résoudre les problèmes vu leur taille, nous allons, en conséquence, nous concentrer sur deux méthodes méta-heuristiques : la recherche Tabou et l'algorithme génétique. L'algorithme génétique va intervenir comme mécanisme de diversification, tandis que la recherche Tabou va agir comme intensificateur de la recherche en explorant intensément le voisinage des solutions. L'hybridation des deux méthodes permet donc d'améliorer les capacités d'exploration de l'espace de recherche en vue de maintenir un compromis acceptable entre intensificateur et diversificateur, gage de performance de toute méthode de recherche qui se respecte.

3.4. Algorithme génétique hybride

Nous basant sur les principes de fonctionnement de la recherche Tabou et de l'algorithme génétique, nous allons maintenant les combiner pour construire un algorithme génétique hybride. Celui-ci va donc disposer d'un principe de codage des individus, d'un mécanisme de génération de la population initiale et d'opérateurs permettant de diversifier la population au cours des générations et d'explorer l'espace de recherche. De plus, à l'instar notamment de Kreutz *et al.* [2000], Wang *et al.*, [2005] Liaw [2000] et surtout de Nowicki et Smutnicki, [1998], nous allons utiliser la recherche Tabou pour améliorer un nouvel individu généré à chaque itération. Negennman [2001] a montré que

l'algorithme de la recherche Tabou proposé par Nowicki et Smutnicki [1996a, 1996b] donne les meilleurs résultats par rapport à d'autres méthodes de recherche locale pour résoudre des problèmes d'ordonnancement dans le cadre d'un « flow shop ». Par ailleurs, Nowicki et Smunicki [1998] ont étendu leur méthode au « flow shop » hybride où les machines à chaque étage sont identiques. Comme montré dans la Figure 17, un algorithme génétique hybride peut être décrit comme ci-dessous. Cet algorithme hybride est obtenu en remplaçant les étapes 5 et 6 de l'algorithme génétique classique comme suit :

Etape 5 : (Mutation)
Choisir aléatoirement un enfant généré par l'opérateur de croisement dans l'étape 4 et puis lui appliquer l'opérateur de mutation avec une probabilité de mutation pour obtenir un nouvel individu ;

Etape 6 : (Amélioration locale)
Appliquer une variante de la procédure de recherche Tabou, comme un opérateur d'amélioration locale, à l'autre enfant généré par l'opérateur de croisement non choisi par l'étape 5 pour la mutation. Un nouvel individu est obtenu.

Notons que nous appliquons une technique d'élitisme dans notre algorithme afin d'assurer que le meilleur individu actuel est toujours gardé dans la mémoire de l'algorithme. Autrement dit, un individu de la population courante, dont la valeur d'adaptation est la meilleure, va sûrement être choisi comme un individu de la génération prochaine.

La condition d'arrêt de cet algorithme génétique hybride : le nombre d'itération est égal à 500 ou 95% des individus ont la même valeur de la fonction objectif.

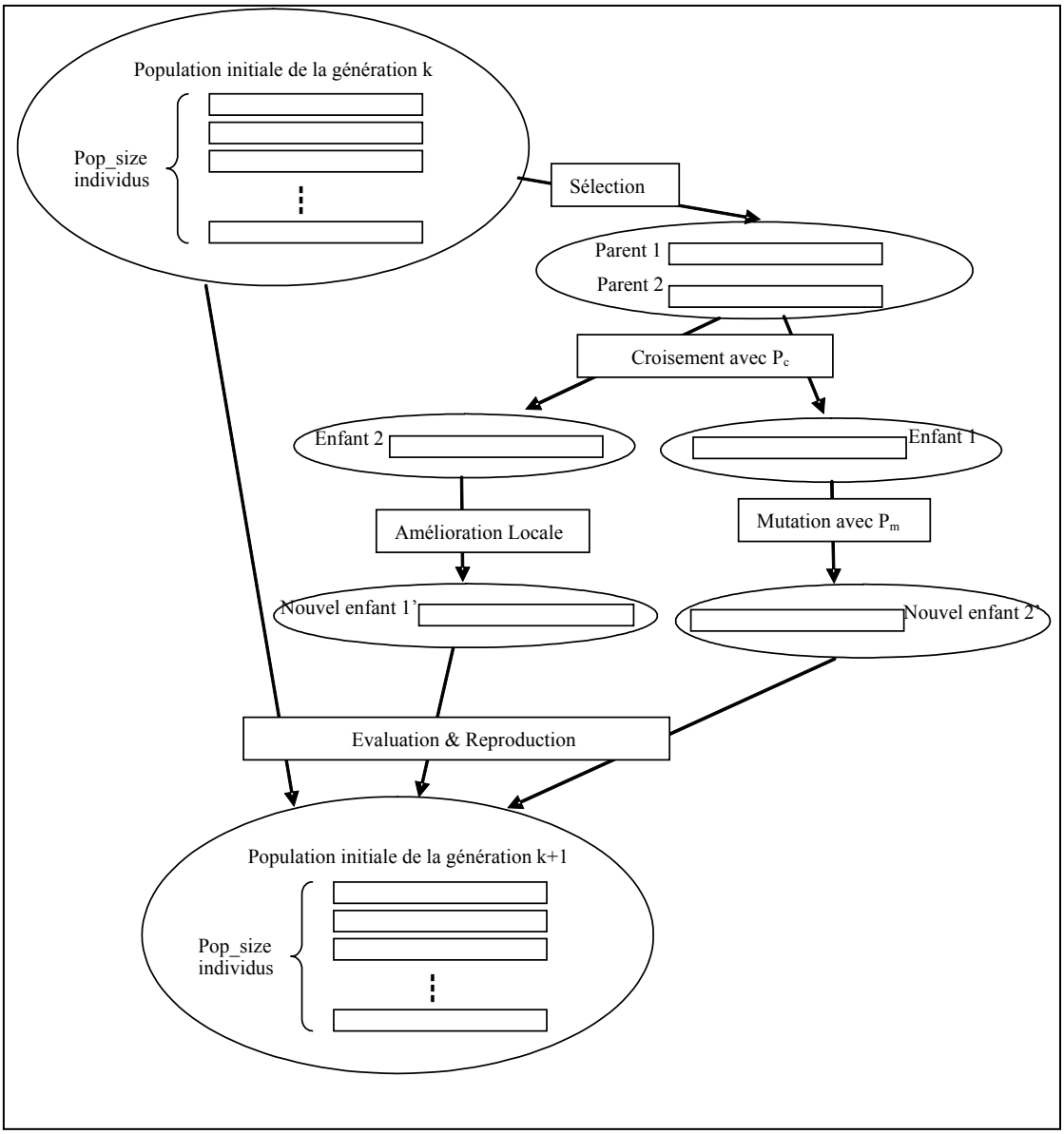

Figure 17 : Description générale de l'algorithme génétique hybride

4. Résolution des problèmes d'ordonnancement de blocs opératoires au moyen d'un algorithme génétique hybride

Nous avons décrit un modèle d'ordonnancement des interventions chirurgicales pour chacune des stratégies de programmation opératoire : le «block scheduling » et l'«open scheduling ». Dans cette section, un algorithme génétique hybride sera utilisé pour résoudre ces deux modèles d'ordonnancement.

4.1 Cas d'une stratégie du « block scheduling »

4.1.1 Description du modèle et des notations

Comme nous l'avons déjà signalé précédemment, le modèle d'ordonnancement centré sur le bloc opératoire pour la stratégie du « block scheduling » peut être considéré comme un modèle d'ordonnancement d'un « flow shop » hybride à deux étages où les plages

horaires avec l'heure de départ (normalement, la durée d'ouverture d'une salle d'opération se compose d'une ou plusieurs plages horaires spécialisées, les heures de départ des plages horaires sont données d'avance selon le plan directeur d'allocation préalable de plages horaires) constituent des « machines spécialisées parallèles » en premier étage et où les lits de la salle de réveil sont des « machines identiques parallèles » en deuxième étage. Dans cette phase d'ordonnancement, les interventions, affectées lors de l'étape de planification dans les plages horaires, seront ordonnancées avec comme objectif la minimisation du coût total du bloc opératoire c'est-à-dire le coût des salles d'opération et de la SSPI.

Traditionnellement, les problèmes d'ordonnancement classique d'un « flow shop » hybride visent à minimiser le *makespan*. Cependant, pour notre problème d'ordonnancement cette seule minimisation n'est pas suffisante car nous devons tenir compte du fait que le coût d'une heure d'ouverture de la salle d'opération est toujours beaucoup plus élevé que celui de la SSPI à cause des différentes ressources humaines et matérielles qui y sont affectées. Le rapport entre le coût d'une heure d'ouverture de la salle d'opération et celui de la SSPI peut varier entre 4.8 à 15[10] selon le type d'interventions. Donc, notre fonction objectif consistera en une minimisation de la somme pondérée des *makespan*s : celui relatif à la salle d'opération (les plages horaires) et celui relatif à la salle de réveil. Un poids plus important est attribué au *makespan* des plages horaires.

Plus précisément, notre modèle peut être décrit comme suit :

Notre problème d'ordonnancement est à deux étages. Le premier étage se compose de M_1 plages horaires spécialisées, qui sont affectées préalablement aux R salles d'opération et le deuxième étage se compose de M_2 lits de réveil identiques dans la SSPI. Par jour, N patients sont à assigner dans le bloc opératoire. Etant donné que chaque patient ne peut être opéré que dans sa plage horaire spécialisée, l'ensemble des patients à assigner est divisé en M_1 sous-ensembles dans les plages horaires (le premier étage). Et donc pour chaque plage horaire spécialisée k, $N_k^{(1)}$ patients (interventions) y sont affectés ($\sum_{k=1}^{M_1} N_k^{(1)} = N$).

Chaque patient i (i =1,…, N) va d'abord être traité dans sa plage horaire spécialisée où sa durée opératoire $t_i^{(1)}$ est estimée à l'avance, et ensuite va être transféré vers un lit de réveil dans la SSPI. Il va y rester jusqu'à ce qu'il se réveille sauf, cas particulier, s'il n'y a pas de lit disponible dans la SSPI, dans quel cas, il reste « bloqué » dans la salle d'opération où il est opéré. La durée de réveil $t_i^{(2)}$ peut être partagée entre la salle d'opération et la salle de réveil.

Si Δ_i ($0 \le \Delta_i \le t_i^{(2)}$) représente le temps d'attente du patient i dans sa salle d'opération avant d'être transféré soit à son lit de réveil dans la SSPI soit à son lit d'hospitalisation, sa durée réelle dans sa salle d'opération est $t'^{(1)}_i = t_i^{(1)} + \Delta_i$ ($t_i^{(1)} \le t'^{(1)}_i \le t_i^{(1)} + t_i^{(2)}$) et sa durée réelle dans la SSPI est $t'^{(2)}_i = t_i^{(2)} - \Delta_i$ ($0 \le t'^{(2)}_i \le t_i^{(2)}$). Il est évident que $t'^{(1)}_i + t'^{(2)}_i = t_i^{(1)} + t_i^{(2)}$. Ce problème d'ordonnancement vise à minimiser le coût total du bloc opératoire : $\omega C_{\max}^{(1)} + C_{\max}^{(2)}$ où ω représente le rapport entre le coût d'une heure d'ouverture des salles d'opération et le coût d'une heure d'ouverture de la SSPI ;

[10] http://www.asge.org/nspages/practice/management/186.cfm

$C_{\max}^{(1)}$ représente le temps d'achèvement de l'intervention du dernier patient sortant du premier étage (les plages horaires) ; $C_{\max}^{(2)}$ représente le temps de sortie du dernier patient du deuxième étage (de la SSPI).

Pour mieux comprendre l'algorithme génétique hybride proposé pour résoudre ce problème, nous détaillons les notations utilisées dans la description de l'algorithme.

Ω : l'ensemble de toutes les interventions (patients) à assigner pour ce jour d'ordonnancement, $\Omega = \{1, \cdots, N\}$;

N_r : le nombre des plages horaires composant de la durée d'ouverture de la salle d'opération r (r = 1, …, R), $\sum_{r=1}^{R} N_r = M_1$;

TD_k : l'heure d'ouverture de la plage horaire k ;

TF_k : l'heure de fermeture de la plage horaire k ;

PS_k : l'indice de salle d'opération où la plage horaire k se situe ;

$\Omega_k^{(1)}$: l'ensemble des interventions (patients) affectées à la plage horaire k, $\Omega_k^{(1)} = \left\{1, \cdots, N_k^{(1)}\right\}$, $k = 1, \cdots, M_1$;

π : un programme opératoire réalisable, c'est-à-dire, une séquence de tous les patients passant par une plage horaire de la salle d'opération et la salle de réveil. Nous notons $\pi = \left(\pi^{(1)}, \pi^{(2)}\right)$ où $\pi^{(1)}$ représente la séquence des patients (interventions) du premier étage (les plages horaires) et $\pi^{(2)}$ la séquence des patients du deuxième étage (la SSPI). Étant donné qu'il y a M_1 plages horaires au premier étage et M_2 lits de réveil au deuxième étage, nous notons la séquence des patients (interventions) pour une plage horaire k comme étant $\pi_k^{(1)} = \left(\pi_k^{(1)}[1], \cdots \pi_k^{(1)}[i], \cdots, \pi_k^{(1)}[N_k^{(1)}]\right)$ ($k = 1, \cdots, M_1$) et celle pour un lit de réveil k comme étant $\pi_k^{(2)} = \left(\pi_k^{(2)}[1], \cdots \pi_k^{(2)}[i], \cdots, \pi_k^{(2)}[N_k^{(2)}]\right)$ ($k = 1, \cdots, M_2$) où $\pi_k^{(s)}$[i] représente l'indice du patient qui est assigné à la $i^{\text{ième}}$ position de la séquence $\pi_k^{(s)}, i \in \{1, \cdots N_k^{(s)}\}, s \in \{1,2\}, k \in \{1, \cdots, M_s\}$. En conséquence, un programme opératoire peut être formalisé comme ci-dessous :

$$\begin{aligned}\pi &= \left(\pi^{(1)}, \pi^{(2)}\right) \\ &= \left(\pi_1^{(1)}, \cdots, \pi_{M_1}^{(1)}, \pi_1^{(2)}, \cdots, \pi_{M_2}^{(2)}\right) \quad ; \\ &= \begin{pmatrix} \pi_1^{(1)}[1], \cdots, \pi_1^{(1)}[N_1^{(1)}], \cdots, \pi_{M_1}^{(1)}[1], \cdots, \pi_{M_1}^{(1)}[N_{M_1}^{(1)}], \\ \pi_1^{(2)}[1], \cdots, \pi_1^{(2)}[N_1^{(2)}], \cdots, \pi_{M_2}^{(2)}[1], \cdots, \pi_{M_2}^{(2)}[N_{M_2}^{(2)}] \end{pmatrix}\end{aligned}$$

$CI_i^{(1)}$: le temps d'achèvement idéal de l'intervention i (i =1,…, N) du premier étage (les salles d'opération). Un cas idéal se présente lorsque le patient i peut être transféré à la SSPI sans attente c'est-à-dire dès que son intervention est terminée dans la salle d'opération ;

$s_i^{(1)}$: le temps d'arrivée du patient (intervention) i au premier étage (les plages horaires) dans le programme opératoire, c'est le temps de début de l'intervention i si les autres ressources nécessaires au premier étage (les plages horaires) sont disponibles ;

$s_i^{(2)}$: le temps d'arrivée du patient i au deuxième étage (la SSPI) si les ressources nécessaires sont disponibles ;

$C_i^{(s)}$: le temps d'achèvement d'une intervention $i (i \in \Omega)$ à l'étage s . Ce temps

d'achèvement représente le moment où ce patient sort de sa salle d'opération (plage horaire) spécialisée (pour $s = 1$) ou de la salle de réveil (pour $s = 2$).

Le temps d'achèvement d'une intervention à un étage peut se calculer au moyen de trois éléments :

1. le temps d'arrivée de cette intervention ;
2. la durée opératoire de l'intervention concernée ;
3. le temps d'attente de cette intervention entre l'étage s et l'étage suivant.

Supposons que le patient affecté à la $i^{ème}$ position dans la plage horaire k (la « machine » k au premier étage) serait le $i'^{ème}$ patient transféré au lit de réveil k' dans la SSPI (la « machine » k' au deuxième étage) dans le programme opératoire π, donc les interventions $\pi_k^{(1)}[i]$ et $\pi_{k'}^{(2)}[i']$ correspondent au même patient.

La Figure 18 illustre les notations utilisées et montre le cas où le $(i\text{-}1)^{ème}$ patient rentre dans la salle de réveil dès que son intervention dans la plage horaire k est achevée. Il y a un temps d'attente entre le $i^{ème}$ patient et le $(i\text{-}1)^{ème}$ patient dans la plage horaire k.[11] Le $(i\text{-}2)^{ème}$ patient s'étant réveillé avant la fin de l'intervention du $(i\text{-}1)^{ème}$ patient dans la plage horaire k, le lit de réveil k' est disponible. Le patient i, quant à lui, devra attendre, après son opération, dans la salle d'opération où il a été opéré jusqu'à ce que le lit de réveil k' soit disponible.

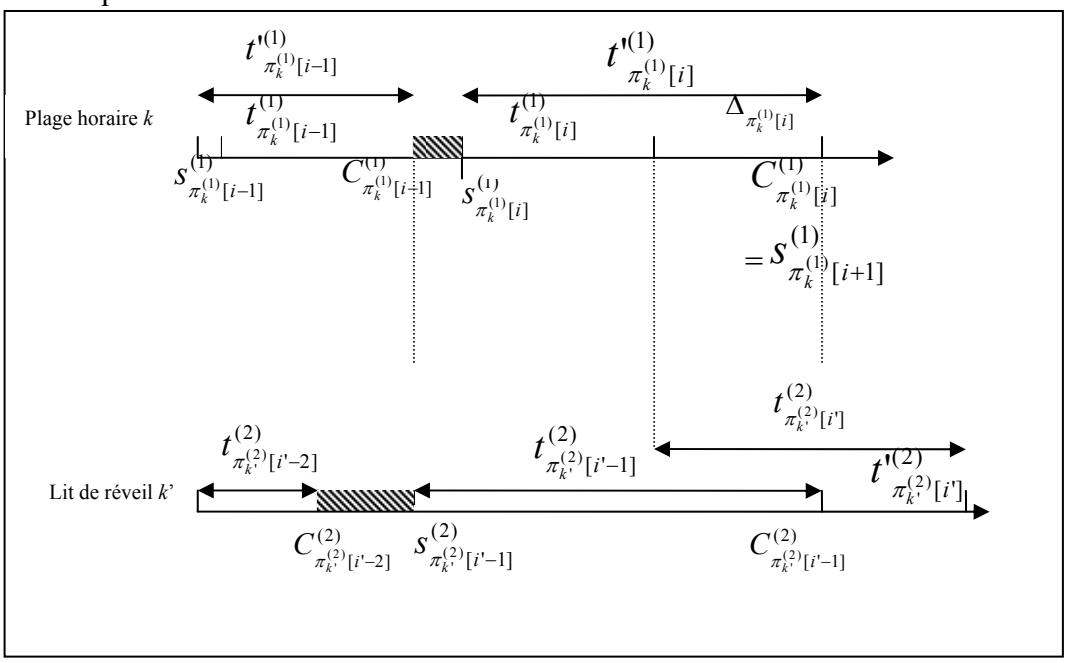

Figure 18 : Illustration des notations utilisées

Donc pour calculer les temps d'achèvement des interventions relatives au $i^{ème}$ patient en salle d'opération, nous détaillons ci-dessous les trois éléments composant le temps d'achèvement $C^{(s)}_{\pi_k^{(1)}[i]}$.

1. Le temps d'arrivée du patient $\pi_k^{(s)}[i]$ à l'étage s : $s^{(s)}_{\pi_k^{(s)}[i]}$.

[11] Il n'existe normalement pas de temps d'attente entre deux patients successifs dans les plages horaires pour la stratégie de « block scheduling ». Cependant, nous illustrons le cas général dans la figure 18 afin d'illustrer valablement le problème d'ordonnancement pour les deux stratégies.

Si s=2, $s^{(2)}_{\pi^{(1)}_k[i]} = s^{(2)}_{\pi^{(2)}_{k'}[i']} = \max\left\{C^{(2)}_{\pi^{(2)}_{k'}[i'-1]}, C^{(1)}_{\pi^{(1)}_k[i]}\right\}$ ce qui signifie que le temps d'arrivée du patient en salle de réveil est égal au moment où ce patient quitte sa salle d'opération spécialisée.

Ce temps de début dépend notamment du temps d'achèvement de l'intervention précédente dans la même plage horaire k (le temps d'arrivée de la première intervention dans la chaque plage horaire k est égale à l'heure de départ de cette plage horaire) pour l'étage s =1 et sur le même lit de réveil k' pour l'étage 2.

2. La durée opératoire.

   Si s = 1, c'est la durée opératoire dans la salle d'opération (plus précisément, sa plage horaire spécialisée), c'est-à-dire $t^{(1)}_{\pi^{(1)}_k[i]}$.

   Si s = 2, c'est la durée de post-anesthésie (de réveil) qui se déroule, dans le cas idéal, dans la SSPI, c'est-à-dire $t^{(2)}_{\pi^{(2)}_{k'}[i']}$.

3. Le temps d'attente dans la salle d'opération, où la plage horaire k se situe, entre le moment où l'intervention du $i^{ème}$ patient est finie dans cette salle d'opération et le moment où il peut sortir de la salle d'opération: $\Delta_{\pi^{(1)}_k[i]}$. Ce temps d'attente peut être calculé par la formule

$$\Delta_{\pi^{(1)}_k[i]} = \min\left\{\max\left\{0, C^{(2)}_{\pi^{(2)}_{k'}[i'-1]} - (s^{(1)}_{\pi^{(1)}_k[i]} + t^{(1)}_{\pi^{(1)}_k[i]})\right\}, t^{(2)}_{\pi^{(2)}_k[i']}\right\}$$

Par conséquent, la formule pour calculer le temps d'achèvement de l'intervention $\pi^{(1)}_k[i]$ à l'étage s est comme ci-dessous :

1. s=1 (les plages horaires):

$$\begin{aligned} C^{(1)}_{\pi^{(1)}_k[i]} &= s^{(1)}_{\pi^{(1)}_k[i]} + t^{(1)}_{\pi^{(1)}_k[i]} + \Delta_{\pi^{(1)}_k[i]} \\ &= s^{(1)}_{\pi^{(1)}_k[i]} + t^{(1)}_{\pi^{(1)}_k[i]} + \min\left\{\max\left\{0, C^{(2)}_{\pi^{(2)}_{k'}[i'-1]} - (s^{(1)}_{\pi^{(1)}_k[i]} + t^{(1)}_{\pi^{(1)}_k[i]})\right\}, t^{(2)}_{\pi^{(1)}_k[i]}\right\} \end{aligned}$$

2. s=2 (la SSPI)

$$\begin{aligned} C^{(2)}_{\pi^{(2)}_k[i]} &= s^{(2)}_{\pi^{(2)}_{k'}[i']} + t^{(2)}_{\pi^{(2)}_{k'}[i']} - \Delta_{\pi^{(1)}_k[i]} \\ &= s^{(2)}_{\pi^{(2)}_{k'}[i']} + t^{(2)}_{\pi^{(2)}_{k'}[i']} \\ &\quad - \min\left\{\max\left\{0, C^{(2)}_{\pi^{(2)}_{k'}[i'-1]} - (s^{(1)}_{\pi^{(1)}_k[i-1]} + t^{(1)}_{\pi^{(1)}_k[i]})\right\}, t^{(2)}_{\pi^{(1)}_k[i]}\right\} \end{aligned}$$

   Pour le cas particulier où nous aurions $\Delta_{\pi^{(1)}_k[i]} = t^{(2)}_{\pi^{(1)}_k[i]}$, nous obtenons alors $C^{(2)}_{\pi^{(2)}_k[i]} = s^{(2)}_{\pi^{(2)}_{k'}[i']} = C^{(1)}_{\pi^{(1)}_k[i]}$

$C^{(1)}_{\max}$: Le temps de sortie du dernier patient des plages horaires du programme opératoire π, $C^{(1)}_{\max} = \max\{C^{(1)}_i \,|\, i \in \Omega\}$.

$C^{(2)}_{\max}$: Le temps de sortie du dernier patient de la SSPI du programme opératoire π, $C^{(2)}_{\max} = \max\{C^{(2)}_i \,|\, i \in \Omega\}$.

f : La valeur de la fonction objectif du programme opératoire π, $f = \omega C^{(1)}_{\max} + C^{(2)}_{\max}$.

Nous allons maintenant détailler les composants importants de notre algorithme génétique hybride (le codage, la génération des individus pour la population initiale, l'opérateur de croisement, l'opérateur de mutation, l'évaluation et l'opérateur de reproduction et finalement l'opérateur d'amélioration locale). Ensuite, nous proposons une borne inférieure pour l'évaluation de la performance de l'algorithme proposé.

4.1.2 Codage utilisé dans les algorithmes génétiques

Au vu des spécificités de notre problème d'ordonnancement, le chromosome proposé pour représenter un individu (une solution réalisable) se constitue de cinq parties:

1. V_1 : le vecteur séquentiel, à N dimensions, des patients au premier étage représentant l'ordre des patients (interventions) dans les plages horaires ;
2. V_2 : le vecteur d'affectation à N dimensions représentant les indices des lits de réveil au deuxième étage (la SSPI) correspondant à l'ordre des patients représenté par V_1 ;
3. V_3 : le vecteur de séparation des plages horaires, qui est de $(M_1 - 1)$ dimensions, représentant le nombre de positions dans le vecteur V_1 affectées à la première plage horaire spécialisée, puis à la deuxième, … jusqu'à la $(M_1 - 1)$ plage horaire ;
4. V_4 : le vecteur séquentiel des patients au deuxième étage, qui est de N dimensions, représentant l'ordre des patients sur chaque lit de réveil dans la SSPI ;
5. V_5 : le vecteur de séparation des lits de réveil, qui est de $(M_2 - 1)$ dimensions, indiquent combien de positions dans le vecteur V_4 sont affectées au premier lit de la SSPI.

Pour mieux comprendre cette stratégie de codage, nous visualisons un exemple ci-dessous :

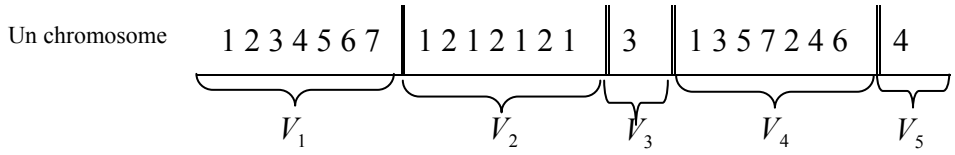

Figure 19 : Un exemple de la stratégie du codage

La Figure 19 nous permet de visualiser un exemple de la stratégie de codage. Ce chromosome représente le codage d'un programme opératoire réalisable pour un bloc opératoire constitué de deux plages horaires spécialisées et de deux lits de réveil identiques dans la SSPI.

Ce chromosome nous montre un programme opératoire où trois interventions (1, 2, et 3) seront opérées dans l'ordre {1-2-3} dans la première plage horaire et quatre interventions (4, 5, 6 et 7) dans l'ordre {4-5-6-7} dans la deuxième plage horaire. Après leurs opérations, les patients 1, 3, 5 et 7 seront transférés au lit de réveil 1 et les autres au lit de réveil 2 dans la SSPI. L'ordre des patients passant sur le lit de réveil 1 est 1-3-5-7 et l'ordre pour le lit de réveil 2 est de 2-4-6. Dès qu'un chromosome est défini, nous obtenons un programme opératoire pour ce bloc opératoire.

4.1.3 Génération de la population initiale

Dans un algorithme génétique, une population initiale doit être générée où chaque individu représente une solution réalisable du problème d'optimisation (dans notre cas, c'est un programme opératoire réalisable pour le problème concerné). Les individus de la population initiale sont normalement générés aléatoirement ou par des procédures heuristiques qui ont la possibilité d'obtenir des individus de bonne qualité.

Dans notre étude, la procédure que nous avons utilisée pour générer la population initiale des individus, adaptée à notre problème est une procédure de décomposition, où les patients (interventions) sont tout d'abord assignés au premier étage (les plages horaires) et ensuite au deuxième étage (la SSPI).

Algorithme heuristique par décomposition (AHD) va générer aléatoirement un programme opératoire réalisable π, représenté par le chromosome $(V_1, V_2, \cdots, V_5)$.

Etape 1 : (Génération des vecteurs relatifs aux plages horaires)

1. Le vecteur V_3 est connu et fixé par les données des problèmes car tous les patients devront être opérés dans leur plage horaire spécialisée ;
2. Générer aléatoirement le vecteur séquentiel V_1 représentant l'ordre des interventions dans toutes les plages horaires spécialisées ;

Etape 2 : (Génération des vecteurs relatifs à la SSPI)

1. Générer aléatoirement le vecteur d'affectation des lits V_2 pour tous les patients dont l'ordre au premier étage est déjà fixé par V_1 ;
2. A partir du vecteur V_2, le vecteur V_5 est créé ;
3. Générer le vecteur séquentiel des patients au deuxième étage, V_4, en respectant l'ordre des patients sortant d'une même plage horaire et l'heure de départ des plages horaires. Cette étape se déroule comme suit :
   a) Affecter les patients sortant de la première plage horaire aux lits de la SSPI dans l'ordre ;
   b) Insérer les patients sortant des autres salles d'opération, un par un dans l'ordre, dans le vecteur V_4 en prenant en compte leurs durées opératoires et en les comparant aux temps d'achèvement des interventions déjà mises dans le vecteur. La Figure 20 montre un exemple. Les patients de la première plage horaire sont mis dans le vecteur V_4 d'après leur lit (voir V_2) et dans l'ordre de passage dans la plage horaire. Ainsi, les patients 1 et 3 sont affectés au lit 1 et dans cet ordre tandis que le lit 2 est attribué au patient 2. Ensuite, nous regardons le premier patient de la plage horaire 2. Il est affecté au lit 1. Nous comparons donc son temps d'achèvement par rapport au temps d'achèvement des autres patients déjà affectés à ce lit. Si nous constatons que le patient 4 sortira de la plage horaire 2 avant que le patient 1 n'ait terminé dans la salle 1. Donc, nous pourrons venir insérer le patient 4, dans le vecteur V_4, avant le patient 1. Nous effectuons le même raisonnement pour le patient 5 qui pourra donc être inséré avant le patient 2 dans le lit 2, et le patient 6 qui pourra être inséré entre le patient 1 et 3 dans le lit 1. Nous obtenons donc au final comme vecteur V_4 : {4 1 6 3 5 2}.

Etape 3 : (Calculer les valeurs de la fonction objectif et du critère auxiliaire)

1. Calculer $C_i^{(1)}$ et $C_i^{(2)}$ pour chaque patient *i*. Dans le cas où un patient *i* est transféré à son lit d'hospitalisation sans passer par la SSPI, $C_i^{(1)} = C_i^{(2)}$;
2. La valeur de la fonction objectif de ce programme opératoire généré est $f = \omega * \max\{C_i^{(1)} \mid i = 1, \cdots, N\} + \max\{C_i^{(2)} \mid i = 1, \cdots, N\}$;
3. La valeur du critère auxiliaire de ce programme opératoire généré est

$$f' = \omega * \frac{1}{N} \sum_{i=1}^{N} C_i^{(1)} + \max\{C_i^{(2)} \mid i = 1, \cdots, N\}.$$

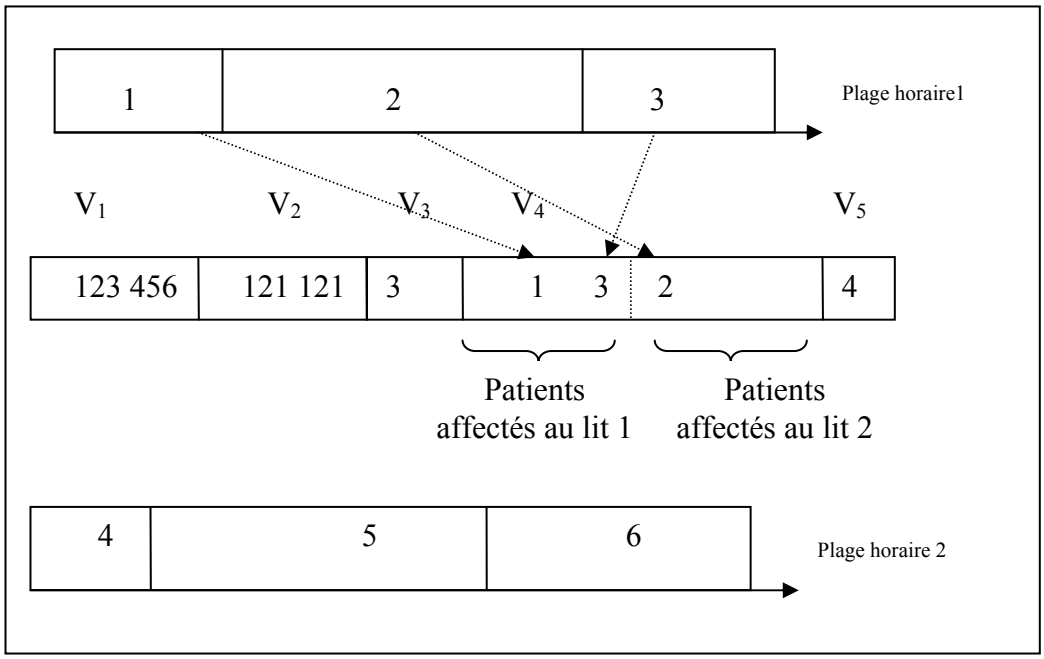

Figure 20 : Illustration de l'algorithme AHD

4.1.4 Opérateur de croisement

Pour obtenir de nouveaux individus (enfants) à partir d'une population initiale d'une itération, nous utiliserons deux opérateurs de croisement, basés sur ceux de Goldberg [1989]. Le croisement s'effectuera soit sur le vecteur V_1 représentant l'ordre des interventions dans les salles d'opération, soit sur le vecteur V_2 représentant l'affectation des lits aux patients dans la SSPI. Il faudra que les nouveaux individus obtenus par ces opérations de croisement soient aussi des solutions réalisables.

A. Opérateur de croisement traitant le premier vecteur V_1 du chromosome

Etape 1 : Choisir deux individus de la population actuelle comme étant les parents P_1 et P_2 pour générer deux enfants O_1 et O_2 ;

Etape 2 : Copier tous les éléments des parents choisis, à l'exception de V_1 et V_4, aux deux enfants respectivement, ici, nous copions ceux du parent P_1 à l'enfant O_1, ceux du parent P_2 à l'enfant O_2 ;

Etape 3 : Copier une partie aléatoire du vecteur V_1 du patient P_1 à l'enfant O_2 et en même temps, copier la partie correspondante du vecteur V_1 du patient P_2 à l'enfant O_1 ;

Etape 4 : Compléter la partie restante du vecteur V_1 des enfants par les éléments de leurs parents respectifs (O_1 par P_1, O_2 par P_2) en balayant de gauche à droite et en ne reprenant que les éléments non encore transmis lors de l'étape 3 ;

Etape 5 : Remplacer le vecteur V_4 de chaque enfant en prenant en compte les modifications opérées dans le vecteur V_1 ;

La Figure 21 nous montre un exemple du fonctionnement de cet opérateur de croisement où les éléments de même couleur proviennent du même parent. Les éléments soulignés

sont les éléments choisis pour l'opération de croisement. Les éléments non coloriés sont modifiés selon les modifications de leur vecteur V_1.

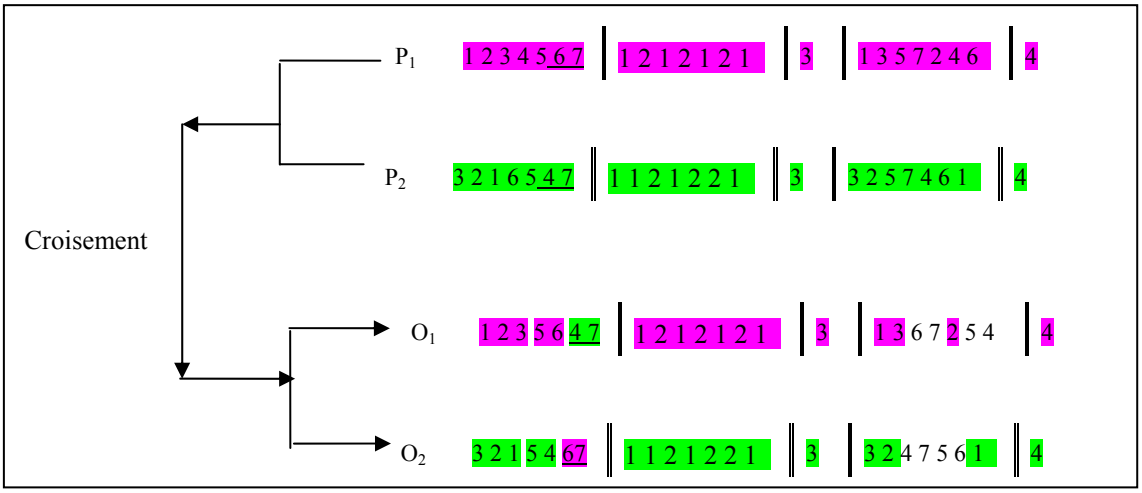

Figure 21 : Opérateur de croisement traitant le vecteur V_1

B. Opérateur de croisement traitant le vecteur V_2 du chromosome

Cet opérateur de croisement est similaire à celui utilisé pour le vecteur V_1 du chromosome sauf qu'ici à l'étape 3, il n'y a pas de balayage, le recopiage s'effectuant directement et à l'étape 4, l'opérateur d'exploration va chercher à trouver les positions disponibles pour les patients déplacés vers d'autres lits de réveil à cause des opérations de croisement effectués sur le vecteur V_2. Cet opérateur de croisement est visualisé à la Figure 22.

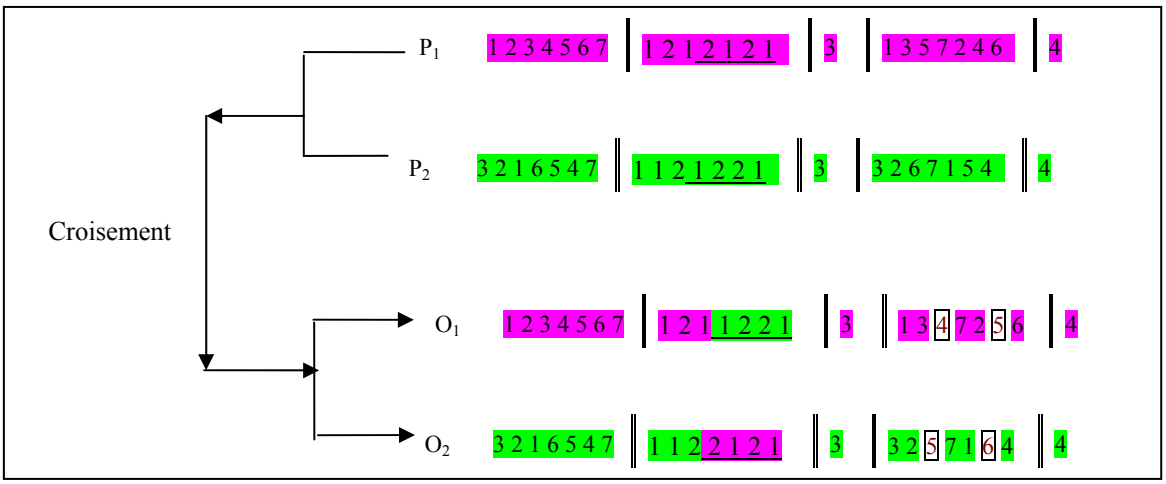

Figure 22 : Opérateur de croisement traitant le vecteur V_2

4.1.5 Opérateur de mutation

Bien que chaque individu se compose de cinq parties, nous trouvons que le vecteur V_3 est fixé par les données initiales des interventions et les deux derniers vecteurs, V_4 et V_5 sont définis par les trois premiers vecteurs V_1, V_2 et V_3. Donc, les opérations de mutation ne seront réalisées que sur les deux premiers vecteurs V_1 et V_2.

Dans cet algorithme, la probabilité de mutation P_m est fixée à 1%. L'opérateur de mutation sera choisi aléatoirement entre les trois opérateurs de mutation proposés ci-dessous :

A. Opérateur de mutation traitant le vecteur V_1

- Choisir deux éléments du vecteur V_1 tout en veillant à ce que ces deux interventions appartiennent à la même plage horaire ;
- Echanger les deux éléments choisis ;
- Modifier en conséquence les éléments du vecteur V_4.

B. Premier opérateur de mutation pour le vecteur V_2

- Choisir aléatoirement deux éléments du vecteur V_2 ;
- Si les éléments choisis sont identiques, refaire le choix jusqu'à ce que ces deux éléments soient différents. Echanger les deux éléments choisis ;
- Modifier le vecteur V_4 en conséquence.

C. Deuxième opérateur de mutation pour le vecteur V_2

- Générer aléatoirement un chiffre j entre [1,..., N] et un chiffre k entre [1,…, M_2] ;
- Si le $j^{ème}$ élément du vecteur V_2 est égal à k, nous régénèrons aléatoirement un chiffre k parmi [1, …, M_2] jusqu'à ce que le chiffre généré soit différent du $j^{ème}$ élément du vecteur V_2 ;
- Remplacer le $j^{ème}$ élément du vecteur V_2 par le chiffre k généré ;
- Modifier les vecteurs V_4 et V_5 en conséquence.

4.1.6 Evaluation et l'opérateur de reproduction

Afin de régénérer la population initiale pour la prochaine génération, nous utilisons une variante de la règle de la méthode de roulette (Roulette Wheel, RW) comme opérateur de sélection assisté par les évaluations des indicateurs d'adaptation (fitness) des candidats.

Avant de passer au détail du calcul des indicateurs d'évaluation, nous présentons tout d'abord les notations.

Pop_Size : le nombre d'individus de la population initiale de chaque itération ;

POP : l'ensemble courant des candidats pour l'opérateur de sélection. Dans le cas où l'opérateur de sélection est appliqué pour choisir deux parents pour l'opération de croisement, le *POP* initial est constitué de la population initiale de la génération courante. Dans le cas où l'opérateur de sélection est appliqué pour générer la population initiale pour la prochaine génération, le *POP* initial se compose de tous les individus de la génération courante, y compris les enfants générés par l'opérateur de croisement, de mutation et d'amélioration locale. Ensuite, il diminue au fur et à mesure par élimination des individus choisis ;

PSize : le nombre d'individus dans *POP*, $PSize = Pop\_Size + 2$ au début et puis diminue suite à l'élimination des individus choisis ;

π_s : un individu (chromosome) dans *POP* qui représente un programme opératoire réalisable du problème d'ordonnancement concerné ($1 \le s \le PSize$) ;

f_s : la valeur de la fonction objectif de l'individu s, $f_s = \omega C_{\max}^{(1)} + C_{\max}^{(2)}$;

f_w : la valeur la plus grande de fonction objectif parmi les individus dans l'ensemble actuel des individus, $f_w = \max\{f_s \,|\, s \in POP\}$;

p_s : la probabilité de la sélection de l'individu s ;

nb_c : le nombre d'individus choisis comme faisant partie de la population initiale de la prochaine itération.

Supposons que nous allons choisir *Nb* individus dans *POP* par l'opérateur de sélection :
Nb est égal à 2 ; *Psize* de *POP* initial est égal à *Pop_Size* dans le cas où l'opérateur de sélection choisit deux parents pour l'opérateur de croisement ;
Nb est égal à *Pop_size* et *Psize* de *POP* initiale est égal à *Pop_Size* +2 dans le cas où l'opérateur de sélection est appliqué pour générer la population initiale de la prochaine génération.

Opérateur de sélection :

Etape 1 : POP représente l'ensemble initial des candidats. $nb_c = 0$;

Etape 2 : Calculer $C(s) = \sqrt{f_w - f_s}$, $s \in POP$;

Etape 3 : Pour chaque individu *s*, calcul de sa probabilité de distribution :
$p_s = \frac{C(s)}{\sum_{k=1}^{PSize} C(k)}, s \in POP$;

Etape 4 : Calcul de $RW = \left\{ rw_s \middle| rw_0 = 0, rw_s = \sum_{k=1}^{s} p_s, s \in POP \right\}$;

Etape 5 : Générer aléatoirement un chiffre λ avec une loi uniforme entre 0 et 1. Si nous trouvons que $rw_{s-1} < \lambda \le rw_s$, l'individu π_s sera choisi.
$POP = POP \setminus \{\pi_s\}$; $PSize = PSize$ -1 ; $nb_c = nb_c + 1$;

Etape 6 : Si $nb_c \ge Nb$, cette procédure se termine et *Nb* individus sont choisis; sinon retourner à l'étape 2.

4.1.7 Opérateur d'amélioration locale

Dans cette thèse, une procédure de la recherche Tabou est utilisée afin d'améliorer localement un individu choisi.
Nous reprenons les mêmes notations que celles décrites en section 4.1.1 et décrivons celles qui seront utilisées dans cette procédure de Recherche Tabou.

$P(\Omega_k^{(1)})$: l'ensemble de toutes les permutations (séquences) des interventions pour la salle d'opération *k*. $\pi_k^{(1)} \in P(\Omega_k^{(1)})$;

$P(\Omega_k^{(2)})$: l'ensemble de toutes les permutations (séquences) des patients pour le lit *k* dans la SSPI. $\pi_k^{(2)} \in P(\Omega_k^{(2)})$.

Décrivons dès à présent les composantes importantes de cet opérateur d'amélioration locale.

A. Solution initiale
La solution initiale de cette procédure de la recherche Tabou est un enfant, provenant d'une itération de l'algorithme génétique, obtenu par une opération de croisement.

B. Stratégies de mouvement pour générer les voisins
Etant donné que la recherche complète du voisinage de la solution initiale n'est pas toujours une bonne idée du fait du temps de calcul, nous allons seulement chercher une partie du voisinage de la solution courante.
Nous basant sur les résultats de Nowicki et Smutnicki [1996a, 1996b, 1998], nous utilisons la stratégie de mouvement d'insertion pour générer le voisinage de la solution courante. Étant donné que les salles d'opération sont spécialisées et que les lits de réveil sont identiques, nous détaillons les stratégies de mouvement d'insertion pour ces deux étages respectivement ci-dessous.

B_1. Stratégie de mouvement d'insertion pour les plages horaires spécialisées :

Étant donné que $\pi_k^{(1)}$ représente une permutation des interventions dans la plage horaire k et x, y représentent deux positions différentes ($x \neq y$) dans $\pi_k^{(1)}$. Un mouvement d'insertion indique que le patient qui était affecté à la $x^{ème}$ position de cette séquence est déplacé pour être mis juste après (cas 1 où $x < y$) ou juste avant (cas 2 où $x > y$) la $y^{ème}$ position de la Figure 23. Ce mouvement est noté par un vecteur $mv = (1, x, y)$.

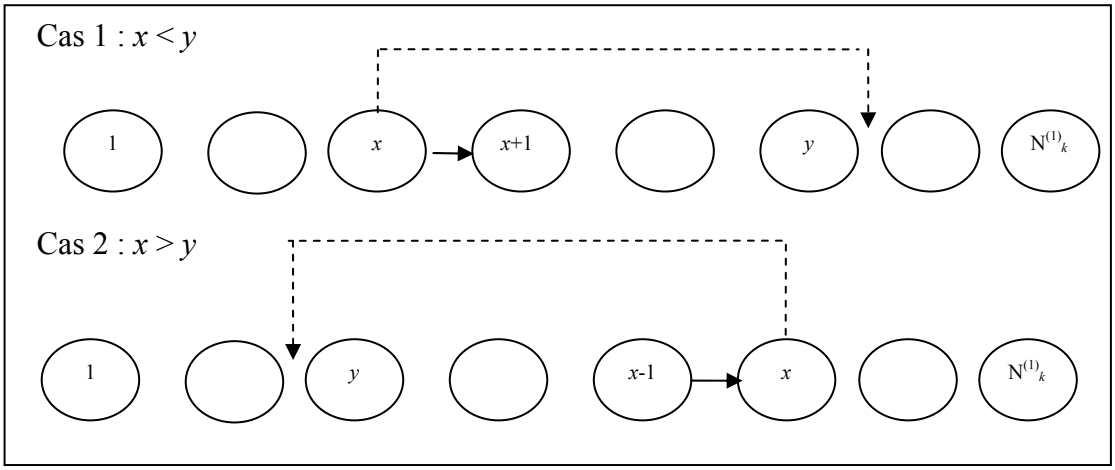

Figure 23 : Permutation des interventions dans une plage horaire spécialisée

B_2. Stratégie de mouvement d'insertion pour les lits dans la SSPI :

Étant donné que les lits dans la SSPI sont identiques, les patients sortants des salles d'opération peuvent donc être placés dans n'importe quel lit de réveil. Donc, la stratégie de mouvement d'insertion est plus flexible dans ce cas que dans celle du premier étage : les mouvements peuvent concerner le même lit ou deux lits différents.

Pour généraliser la stratégie de mouvement d'insertion afin de permettre les mouvements entre deux lits de réveil, nous utilisons dans cette stratégie un vecteur à cinq dimensions $mv = (2, a, x, b, y)$. Ce vecteur représente un mouvement d'insertion entre une position x dans la séquence $\pi_a^{(2)}$ et une autre position y dans la séquence $\pi_b^{(2)}$ [Nowicki et Smutnki, 1998]. Si $a = b$ cela signifie qu'un mouvement d'insertion est fait dans le même lit de réveil. C'est donc similaire à la stratégie de mouvement d'insertion au premier étage. Sinon, dans le cas où un mouvement d'insertion est effectué entre deux lits de réveil différents, le patient de la $x^{ème}$ position de la séquence $\pi_a^{(2)}$ est déplacé et inséré juste avant la $y^{ème}$ position de la séquence $\pi_b^{(2)}$. Donc, pour ce deuxième étage, il y a trois mouvements d'insertion possibles comme montré par la Figure 24.

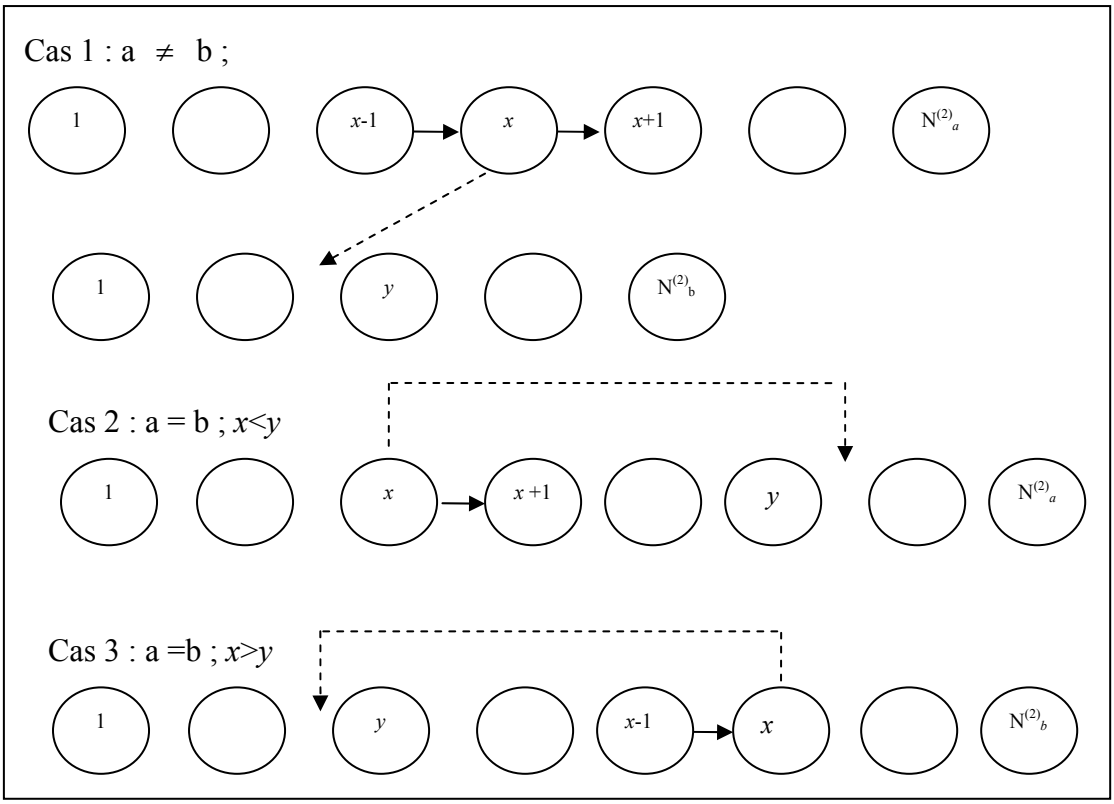

Figure 24 : Stratégie de mouvement d'insertion pour les lits dans la SSPI

En général, après avoir effectué un mouvement mv $(=(1,x,y)$ pour les plages horaires ; $=(2,a,x,b,y)$ pour la SSPI) dans un programme opératoire π, nous obtenons un nouveau programme opératoire π^{mv} dans l'espace du voisinage de π. A partir de tous les mouvements possibles, nous construisons alors une partie importante du voisinage de π parmi lequel le nouveau programme opératoire (programme opératoire initial pour l'itération suivante) sera trouvé.

En fait, pour une position aléatoirement choisie x dans la séquence $\pi_a^{(1)}$ $(1 \le a \le M_1$, $1 \le x \le N_a^{(1)})$ dans un programme opératoire π, il y aura au plus $(N_a^{(1)} - 1)$ mouvements d'insertion à faire et au total $\sum_{a=1}^{M_1} N_a^{(1)}(N_a^{(1)} - 1)$ mouvements d'insertion pourront être faits pour générer les voisins. De manière similaire, pour une position aléatoirement choisie x dans la séquence $\pi_a^{(2)}$ ($1 \le a \le M_2, 1 \le x \le N_a^{(2)}$) dans un programme opératoire π, il y aura au plus $(N_a^{(2)} - 1)$ mouvements d'insertion à faire pour le cas où la position y est dans la même séquence que celle de x et $(N_b^{(2)} + 1)(1 \le b \le M_2, b \ne a)$ mouvements d'insertion à faire pour le cas où la position y est dans une autre séquence que celle de x.

Alors, au total $\sum_{a=1}^{M_2} N_a^{(2)} \{(N_a^{(2)} - 1) + \sum_{b \in \{1,\cdots,M_2\} \setminus \{a\}} (N_b^{(2)} + 1)\} = \sum_{a=1}^{M_2} \sum_{b=1}^{M_2} N_a^{(2)}(N_b^{(2)} + 1)$ mouvements d'insertion seront effectués pour générer les voisins.

Par conséquence, nous obtenons un voisinage d'un programme opératoire π composé de $\sum_{a=1}^{M_1} N_a^{(1)}(N_a^{(1)}-1)+\sum_{a=1}^{M_2}\sum_{b=1}^{M_2} N_a^{(2)}(N_b^{(2)}+1)$ nouveaux programmes opératoires, chacun de ceux-ci ayant été obtenu par un mouvement d'insertion du programme opératoire π.

C. Liste Tabou

Dans la recherche Tabou, une liste Tabou est toujours utilisée pour éviter le piège des optima locaux. Dans notre procédure de recherche Tabou, la taille de la liste Tabou est fixée à *MaxL*, c'est-à-dire, qu'il y a au plus *MaxL* restrictions des mouvements d'insertion. Chaque restriction est associée à un vecteur à trois dimensions $T=(s,i_1,i_2)$ qui indique que le patient i_1 est ordonnancé avant le patient i_2 à l'étage *s* (Si *s* = 1(les plages horaires), $i_1,i_2\in\Omega_k^{(1)}, k\in\{1,\cdots,M_1\}$; si *s* = 2 (la SSPI), $i_1,i_2\in\Omega$). Pour chaque itération, dès qu'un nouveau mouvement d'insertion est effectué, un ou deux vecteurs seront ajoutés à la liste Tabou. Par exemple, si un mouvement $(2,a,x,b,y)$ est effectué au deuxième étage, deux vecteurs $(2,\pi_a^{(2)}[x-1],\pi_a^{(2)}[x])$ et $(2,\pi_a^{(2)}[x],\pi_a^{(2)}[x+1])$ seront ajoutés à la liste Tabou dans le cas où $a\neq b$. Si $x=1$, nous n'ajoutons que le dernier élément et si $x=N_a^{(2)}$, nous n'ajoutons que le premier élément. Dans le cas *a=b*, nous ajoutons le vecteur $(2,\pi_a^{(2)}[x-1],\pi_a^{(2)}[x])$ pour $x<y$ et le vecteur $(2,\pi_a^{(2)}[x],\pi_a^{(2)}[x+1])$ pour $x>y$. Pour le premier étage, c'est exactement le même cas que celui du $a=b$ au deuxième étage[12].
Au début de la procédure de la recherche Tabou, la liste Tabou est initialisée à vide, puis à chaque itération, un à un les éléments sont ajoutés à cette liste. Dans le cas où la liste est pleine, le mouvement le plus ancien sera remplacé par le nouveau mouvement venant d'être effectué.
Avec cette liste Tabou, nous pouvons savoir si un mouvement d'insertion est interdit dans une itération de la recherche Tabou. Comme montré dans la Figure 25, un mouvement d'insertion $mv=(2,a,x,b,y)$ du programme opératoire π sera interdit si au moins une relation de précédence $(2,i_1,i_2)$ entre deux patients apparaissant dans le nouveau programme opératoire généré π_{mv} est interdit par la liste Tabou. Les relations de précédence possibles concernant la SSPI sont :

cas où $a\neq b$

- $(2,\pi_a^{(2)}(x),\pi_b^{(2)}(i)), y\le i\le N_b^{(2)}$ ou $(2,\pi_a^{(2)}(i),\pi_a^{(2)}(x))$, $1\le i<y$;

cas $a=b$

- $(2,\pi_a^{(2)}(i),\pi_a^{(2)}(x)), x<i\le y$, si $x<y$;
- $(2,\pi_a^{(2)}(x),\pi_a^{(2)}(i)), y\le i<x$ si $x>y$;

et concernant les plages horaires :

- $(1,\pi_a^{(1)}(i),\pi_a^{(1)}(x)), x<i\le y$, si $x<y$;
- $(1,\pi_a^{(1)}(x),\pi_a^{(1)}(i)), y\le i<x$ si $x>y$.

[12] Tous les éléments qui seront ajoutés à la liste Tabou sont montrés par les lignes normales dans les figures 19 et 20.

Dans la Figure 25, les lignes normales représentent les relations de précédence (les restrictions Tabou) et les lignes discontinues représentent les mouvements d'insertion.

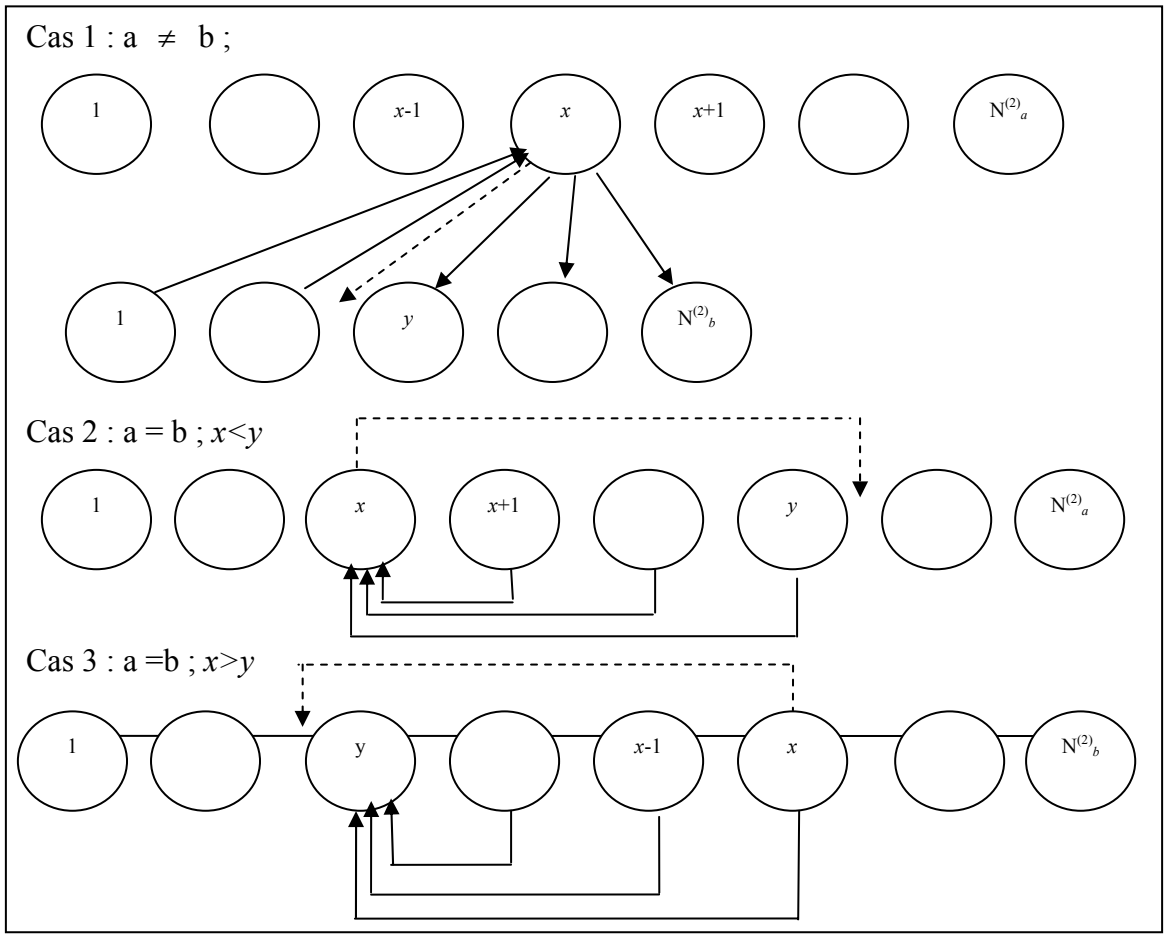

Figure 25 : Restrictions Tabou d'un mouvement d'insertion par la liste Tabou.[13]

D. Stratégie de recherche

Avec la construction du voisinage et en tenant compte de la liste Tabou, à chaque itération, il faut choisir un programme opératoire (un voisin) afin de poursuivre la recherche jusqu'à ce que la condition d'arrêt soit satisfaite.

Normalement, tous les mouvements d'insertion « voisins » possibles sont évalués afin de trouver un mouvement acceptable pour construire un nouveau programme opératoire. Mais il ne faut pas perdre de vue qu'un compromis entre le temps consommé et sa contribution à l'algorithme génétique hybride, c'est-à-dire l'amélioration de la valeur de la fonction objectif pour l'individu choisi, doit être trouvé. Donc, nous proposons une procédure pour éviter d'évaluer tous les mouvements d'insertion et trouver un mouvement d'insertion acceptable.

Etape 1 : Evaluer tous les mouvements d'insertion voisins dans les plages horaires $(1, a, x, y)$ ($1 \le x \le N_a^{(1)}, 1 \le a \le M_1$) afin de trouver un meilleur mouvement d'insertion $mv^{(1)*} = (1, a^*, x^{(1)*}, y^{(1)*})$ à cet étage. Cela

[13] Dans cette figure, nous ne listons que les restrictions Tabou pour les mouvements d'insertion dans la SSPI car les restrictions Tabou pour les salles d'opération sont similaires à celles des cas 2 et 3 dans cette figure.

signifie qu'en effectuant ce mouvement d'insertion $mv^{(1)*}$, le nouveau programme opératoire $\pi_{mv}^{(1)*}$ obtenu a la plus petite valeur de fonction objectif parmi tous les candidats voisins. Si ce mouvement choisi permet d'améliorer la valeur de la fonction objectif du π, nous acceptons ce mouvement $mv^{(1)*}$ pour construire un nouveau programme opératoire (voisin) ;

Etape 2 : Si le mouvement d'insertion choisi $mv^{(1)*}$ ne peut pas améliorer la valeur de la fonction objectif du π, nous évaluons tous les mouvements d'insertion voisins dans la SSPI afin de trouver un meilleur mouvement d'insertion $mv^{(2)*} = (2, a*, x^{(2)*}, b*, y^{(2)*})$ à cet étage. Si le mouvement d'insertion choisi peut améliorer la valeur de fonction objectif du π, nous acceptons ce mouvement $mv^{(2)*}$ pour construire un nouveau programme opératoire (voisin) ;

Etape 3 : Si aucun des deux mouvements d'insertion $mv^{(1)*}$ et $mv^{(2)*}$ ne peut améliorer le programme opératoire π, celui parmi eux qui peut générer un nouveau programme opératoire (voisin) dont la valeur de la fonction objectif est la plus petite et n'est pas interdit par la liste Tabou, sera choisi pour construire un voisin π_{mv} et un ou deux vecteurs correspondants[14] seront ajoutés en liste Tabou. Si la liste Tabou est déjà remplie par les itérations précédentes, les éléments (vecteurs) les plus anciens sont remplacés par les nouveaux. Si les deux mouvements sont interdits par la liste Tabou, cette procédure se termine ainsi que la recherche Tabou et nous acceptons le π^{TS} comme étant la solution final.

E. Description générale de l'opérateur d'amélioration locale

Notre opérateur d'amélioration locale, à partir d'un enfant π^0 généré par l'opérateur de croisement à une itération de l'algorithme génétique, se déroule comme suit :

Etape 1 : Mettre $\pi = \pi^0$; $\pi^{TS} = \pi^0$; $f^{TS} = f(\pi^0) = \omega C_{\max}^{(1)}(\pi^0) + C_{\max}^{(2)}(\pi^0)$; la liste Tabou à vide c'est-à-dire $TL = 0$; $Itr = 0$; $Mov = 0$; $MaxItr = 50$; $MaxMov = 20$; $MaxL = 7$;

Etape 2 : Pour le programme opératoire actuel π, chercher un mouvement d'insertion mv^* selon les étapes décrites au point D de la section 4.1.7 afin de construire un nouveau programme opératoire π^{mv*} dont la valeur de la fonction objectif est $f(\pi^{mv*})$;

Etape 3 : Mettre $\pi = \pi^{mv*}; f(\pi) = f(\pi^{mv*})$;
Si $f(\pi) \geq f^{TS}$

---

[14] Si le mouvement $mv^{(1)*}$ est choisi, un vecteur $(1, \pi_{a*}^{(1)}(x^{(1)*}), \pi_{a*}^{(1)}(x^{(1)*}+1))$ (si $x^{(1)*} < y^{(1)*}$) ou un vecteur $(1, \pi_{a*}^{(1)}(x^{(1)*}-1), \pi_{a*}^{(1)}(x^{(1)*}))$ (si $x^{(1)*} > y^{(1)*}$) sera ajouté dans la liste Tabou.
Si le mouvement $mv^{(2)*}$ est choisi, deux vecteurs $(2, \pi_{a*}^{(2)}(x^{(2)*}-1), \pi_{a*}^{(2)}(x^{(2)*}))$ et $(2, \pi_{a*}^{(2)}(x^{(2)*}), \pi_{a*}^{(2)}(x^{(2)*}+1)$ seront ajoutés à la liste Tabou dans le cas où $a* \neq b*$ et $1 < x < N_{a*}^{(2)}$ (si $x = 1$, on ajoute le dernier vecteur ; si $x = N_{a*}^{(2)}$, on ajoute le premier). Dans le cas ou $a* = b*$, un seul vecteur sera ajouté à la liste : $(1, \pi_{a*}^{(2)}(x^{(2)*}), \pi_{a*}^{(2)}(x^{(2)*}+1))$ si $x^{(2)*} < y^{(2)*}$ ou $(1, \pi_{a*}^{(2)}(x^{(2)*}-1), \pi_{a*}^{(2)}(x^{(2)*}))$ si $x^{(2)*} > y^{(2)*}$.

mettre à jour la liste Tabou en y ajoutant un ou deux vecteurs décrits au point D de la section 4.1.7 et $TL = TL+1$ ou $TL+2$ selon le cas. Si $TL \geq MaxL$, les éléments plus anciens de la liste Tabou sont remplacés par les nouveaux vecteurs ajoutés. $Mov = Mov + 1$.

Si $f(\pi) < f^{TS}$

mettre $\pi^{TS} = \pi^{mv*}$; $f^{TS} = f(\pi^{mv*})$; $Mov = 0$; $Itr = Itr + 1$.

Etape 4 : Si $Itr < MaxItr$ et $Mov < MaxMov$, aller à l'étape 2 ; sinon, la procédure se termine et nous acceptons le π^{TS} comme la solution finale de cette procédure d'amélioration locale.

4.1.8 Borne inférieure pour l'évaluation de la performance de l'algorithme proposé

Afin d'avoir un bon indicateur pour l'évaluation de la performance de nos algorithmes, une borne inférieure est proposée. Gupta, en 1988, a été le premier à avoir proposé une borne inférieure pour un modèle de « flow shop » hybride à deux étages. Puis, Lee et Vairaktarakis [1994] ont proposé une autre borne inférieure meilleure que celle de Gupta [1988] ; Haouari et M'Hallah [1997] l'ont encore améliorée. Pour le cas du « flow shop » hybride à plusieurs étages, de nombreux auteurs s'y sont aussi intéressés. Citons Santos *et al.* [1995], Brockman et Dangelmaier [1998], Jin *et al.* [2005] qui ont développé des bornes inférieures pour le cas où les machines à chaque étage sont identiques et où les durées opératoires à chaque étage sont connues. Ce n'est pas le cas pour notre problème d'ordonnancement où les salles d'opération sont spécialisées et où les durées de réveil peuvent être partagées entre les salles d'opération et la SSPI. Nous allons donc proposer une nouvelle borne inférieure globale NLB, adaptée à notre problème, en nous basant sur les idées développées par Haouari et M'Hallah [1997].

Lemme 1 :

$$\textbf{NLB} = \omega TST_{\max} + \max \left\{ \frac{SPT(M_2) + T^{(2)} - \sum_{k=1}^{M_1} TFT_k}{M_2}, \; TST_{\max} + \min\{ t_i^{(2)} \,|\, i \in \Omega_k \text{ dont } TST_k \equiv TST_{\max} \} \right\}$$

est une borne inférieure de la valeur optimale de la fonction objectif du problème concerné.

Ce Lemme est prouvé en annexe 3.

4.2 Cas d'une stratégie d'«Open Scheduling »

Ayant développé dans le détail le cas du « block scheduling » dans la section précédente, nous allons maintenant nous intéresser au cas de l' «open scheduling » et surtout nous attarder sur les différences entre les deux stratégies au sein de l'algorithme génétique hybride. Ce problème consiste à décider de l'ordre de passage des interventions affectées à une journée par le niveau planification dans les salles d'opération. Notons que l'affectation des interventions dans les salles indiquée par le niveau planification peut être remise en cause.

4.2.1 Description du modèle et les notations

Rappelons que la seule différence entre la stratégie du « block scheduling » et celle de l'«open scheduling » réside dans la manière d'affecter les plages horaires : la stratégie du «block scheduling » permet aux chirurgiens de réserver préalablement leurs plages horaires tandis que la stratégie de l'«open scheduling » ne le permet pas. Cette différence rend le problème d'ordonnancement de cette dernière bien plus complexe du fait que la disponibilité des chirurgiens, dans le cas où il y a plusieurs salles d'opération identiques parallèles, doit être prise en compte. En effet, il pourrait arriver que deux patients devant être opérés dans deux salles d'opération différentes, requièrent au même moment le même chirurgien. Dans ce cas-là, le chirurgien ne peut être « disponible » que pour l'un d'entre eux et donc les autres interventions (patients) devront attendre dans leur salle d'opération jusqu'à ce que le chirurgien en question se libère. Cela peut entraîner d'importants temps d'attente dans les salles d'opération et donc l'augmentation des dépenses totales du bloc opératoire.

Nous avions déjà signalé dans la section 2.2, que dans le cas simpliste où tous les lits de réveil dans la SSPI sont toujours disponibles pour les patients venant d'être opérés dans les salles d'opération, le problème s'apparente alors à un problème de « machines parallèles » avec une contrainte de ressources (la contrainte de la disponibilité des chirurgiens). Ce problème a été prouvé comme étant NP-complet par Ullman [1976] et Kellerer et Strusevich [2004]. Nous avons traité le cas particulier d'un problème d'ordonnancement réel d'un centre d'endoscopie composé de deux salles d'opération spécialisées parallèles et l'avons résolu (voir détails en annexe 7) par une combinaison des technologies de groupe [Kellerer et Strusevich, 2004] avec l'algorithme de Gonzalez et Sahni [1976] (présenté par Breit *et al.* [2001]).

Lorsque le nombre des salles d'opération est supérieur à deux, nous sommes devant un cas particulier du modèle de « flow shop » hybride où le fonctionnement de la SSPI peut venir perturber le fonctionnement des salles d'opération. Dans ce contexte, le problème d'ordonnancement des interventions peut être résolu par un algorithme génétique hybride. De manière similaire à la stratégie du « block scheduling », le problème d'ordonnancement concerné ici est aussi à deux étages. Le premier étage se compose de M_1 salles d'opération identiques et le deuxième étage se compose de M_2 lits de réveil identiques dans la SSPI.

Durant une journée, N patients sont à assigner au bloc opératoire. Pour chaque patient *i*, son chirurgien (ch_i, $i \in \Omega$) ainsi que sa durée opératoire $t_i^{(1)}$ sont connus à l'avance. Comme dans le cas de la stratégie du « block scheduling », à la fin de son intervention, le patient i va devoir rester dans sa salle d'opération si aucun lit dans la salle de réveil n'est disponible. La durée de réveil $t_i^{(2)}$ peut donc s'écouler entre la salle d'opération et la salle de réveil.

Rappelons qu'un chirurgien ne peut opérer qu'une intervention en même temps. Donc si à un moment t, un patient *i* est transféré à sa salle d'opération assignée mais que son chirurgien ch_i est en train d'opérer une autre intervention, nous disons alors que ce chirurgien ch_i est indisponible pour le moment t. Comme nous avons supposé qu'il n'y a pas de préemption pour ce problème d'ordonnancement, ce patient *i* devra attendre dans sa salle d'opération jusqu'à ce que son chirurgien ch_i soit à nouveau disponible pour l'opérer. La notation r_{lt} va représenter le temps disponible au plus tôt du chirurgien l à partir du moment t. Nous supposons que tous les chirurgiens sont prêts pour leurs interventions au début de l'ouverture du bloc opératoire (le moment 0) et donc $r_{l0} = 0$.

L'objectif du problème reste inchangé et consiste toujours en la minimisation du coût total du bloc opératoire. Par conséquent, dans l'étape d'ordonnancement, nous visons tout d'abord à minimiser la durée maximale d'ouverture du bloc opératoire et surtout celle des salles d'opération et utilisons $\min\{\omega C_{\max}^{(1)} + C_{\max}^{(2)}\}$ comme fonction objectif, où ω est un paramètre représentant le rapport entre le coût d'une heure d'ouverture de la salle d'opération et celui d'une heure d'ouverture de SSPI. S'il existe un ensemble de solutions dont la valeur de fonction objectif est la même, nous utilisons un critère auxiliaire pour trouver la meilleure solution parmi elles. Ce critère auxiliaire vise à la minimisation du coût moyen des heures d'ouverture des salles d'opération ainsi que la durée d'ouverture du bloc opératoire (le *makespan* du système, autrement dit, $C_{\max}^{(2)}$), c'est-à-dire $\min\{\omega * moyenne(C_i^{(1)}) + C_{\max}^{(2)}\}$. Ainsi, nous pouvons trouver un programme opératoire dont le coût d'opération est le plus petit que possible.

Constituant un composant important de la fonction objectif, nous avions vu que dans le cas d'une stratégie du « block scheduling », le temps d'achèvement $C_{\pi_k^{(1)}[i]}^{(s)}$ $(i \in \Omega)$ dépendait de 3 éléments :

- Le temps d'arrivée du patient $\pi_k^{(s)}[i]$ à l'étage s : $s_{\pi_k^{(s)}[i]}^{(s)}$;
- La durée opératoire ;
- Le temps d'attente entre le moment où l'intervention *i* est finie dans la salle d'opération *k* et le moment où le patient peut sortir de la salle d'opération k : $\Delta_{\pi_k^{(1)}[i]}$.

Dans le cas d'une stratégie « open scheduling », le temps d'achèvement d'une intervention à l'étage 1 dépend aussi d'un quatrième élément :

- le temps disponible au plus tôt du chirurgien qui va effectuer cette intervention.

Supposons que le patient affecté en $i^{ème}$ position à la salle d'opération k sera le $i'^{ème}$ patient transféré au lit de réveil k' dans la SSPI, donc l'intervention $\pi_k^{(1)}[i]$ et l'intervention $\pi_{k'}^{(2)}[i'-1]$ correspondent au même patient, c'est-à-dire $\pi_k^{(1)}[i] \equiv \pi_{k'}^{(2)}[i']$.

Étant donné que chaque intervention *i* doit être effectuée par son chirurgien ch_i, l'intervention $\pi_k^{(1)}[i]$ sera donc effectuée par le chirurgien $ch_{\pi_k^{(1)}[i]}$ dont le temps disponible au plus tôt à partir d'un moment t est r_{lt}. Ce temps dépend de l'ordre des interventions dans le programme opératoire π. En conséquence de quoi il est possible que bien qu'un patient $\pi_k^{(1)}[i]$ soit prêt à être opéré au moment *t* dans la salle d'opération *k*, son chirurgien ne soit pas disponible avant r_{lt}. Une intervention $\pi_k^{(1)}[i]$ ne peut donc commencer qu'au temps : $\max\left\{t, r_{lt} \mid t = C_{\pi_k^{(1)}[i-1]}, l = ch_{\pi_k^{(1)}[i]}\right\}$.

Le temps d'achèvement de la « tâche » $\pi_k^{(1)}[i]$ à l'étage *s* peut donc être formalisé comme ci-dessous :

1. s = 1 (les salles d'opération):
$$C_{\pi_k^{(1)}[i]} = \max\{t, r_{lt} \mid t = s^{(1)}{}_{\pi_k^{(1)}[i]}, l = ch_{\pi_k^{(1)}[i]}\} + t_{\pi_k^{(1)}[i]}^{(1)} + \Delta_{\pi_k^{(1)}[i]} \qquad (4.2.1)$$
2. s = 2 (la SSPI):
$$C_{\pi_{k'}^{(2)}[i']} = s_{\pi_{k'}^{(2)}[i']}^{(2)} + t_{\pi_{k'}^{(2)}[i']}^{(2)} - \Delta_{\pi_{k'}^{(2)}[i']} \qquad (4.2.2)$$

Où $\Delta_{\pi_k^{(1)}[i]} = \Delta_{\pi_{k'}^{(2)}[i']} = \min\left\{\max\left\{0, C_{\pi_{k'}^{(2)}[i'-1]} - \left(\max\left\{s^{(1)}_{\pi_k^{(1)}[i]}, r_{lt}\right\} + t^{(1)}_{\pi_k^{(1)}[i]}\right)\right\}, t^{(2)}_{\pi_k^{(1)}[i]}\right\}$ (4.2.3)

Ayant déjà auparavant détaillé l'algorithme génétique hybride, nous n'allons plus qu'expliquer ici les différences existantes par rapport au cas du « block scheduling » déjà développé.

4.2.2 Codage

Le chromosome proposé pour représenter le programme opératoire de la stratégie de l'« open scheduling » se compose aussi de cinq parties (V_1,V_2,V_3,V_4,V_5) à la seule différence près que V_3 n'est pas fixé au moyen des données initiales vu que toutes les salles d'opération sont identiques et que les interventions peuvent être affectées à n'importe laquelle des salles d'opération. Dans ce codage, nous n'explicitons pas l'ordre dans lequel chaque chirurgien effectue ses interventions. Cet ordre est défini implicitement par l'ordre de passage des patients dans les salles d'opération, par une règle de priorité. En effet, un chirurgien opère le patient disponible le plus tôt et en cas d'égalité, celui nécessitant moins de temps.

4.2.3 Générateur des individus

L'algorithme proposé dans la section précédente pour générer la population initiale est aussi utilisé pour cette stratégie. Certaines modifications y ont été apportées.

Algorithme Heuristique de Décomposition Modifié (AHDM) pour générer aléatoirement un programme opératoire réalisable π pour le problème d' « open scheduling »:

Etape 1 : (Pour les salles d'opération)

1. Générer aléatoirement le vecteur séquentiel V_1 représentant l'ordre de toutes les interventions (patients) i (i = 1,…, N) au premier étage ;
2. Générer aléatoirement le vecteur de séparation des salles d'opération V_3 ainsi le vecteur séquentiel V_1, divisé en M_1 parties dont chacune correspond à l'ordre des interventions (patients) dans une salle d'opération (il y a au total M_1 salles d'opération). Ce vecteur V_3 est généré comme ci-dessous :
   a) Générer M_1 - 1 nombres entiers dont les valeurs sont comprises entre 1 et N+1 ;
   b) Trier les M_1 -1 nombres entiers par ordre croissant de leur valeur ;
   c) Le vecteur V_3 se compose de ces M_1 -1 nombres entiers triés.

Etape 2 : (Pour la SSPI)

1. Générer aléatoirement le vecteur d'affectation des lits V_2, dont la valeur de chaque élément est un nombre entier entre 1 et M_2 (il y a totalement M_2 lits de réveil dans la SSPI) pour tous les patients dont l'ordre aux salles d'opération est déjà fixé par le vecteur V_1 dans l'étape 1 ;
2. Prendre les patients affectés à leur lit par le vecteur V_2, qui est généré dans l'étape 2.1, et nous obtenons les valeurs du vecteur de séparation des lits de réveil V_5 ;
3. Générer aléatoirement le vecteur séquentiel des patients aux lits de réveil, V_4, pour ordonnancer aléatoirement les patients sur leur lit de réveil.

Etape 3 : (Pour calculer la valeur de la fonction objectif)

1. Calculer $C_i^{(1)}$ et $C_i^{(2)}$ pour chaque patient $i, i \in \Omega$ selon la formule

(4.2.1)-(4.2.3) ;

2. Calculer $C_{\max}^{(1)} = \max\left\{C_i^{(1)} \middle| i \in \Omega\right\}$; $C_{\max}^{(2)} = \left\{C_i^{(2)} \middle| i \in \Omega\right\}$;
3. Calculer la valeur de la fonction objectif de ce programme opératoire généré : $f = \omega * C_{\max}^{(1)} + C_{\max}^{(2)}$ et la valeur du critère auxiliaire : $f' = \omega * moyenne(C_i^{(1)}) + C_{\max}^{(2)}$ où ω est un paramètre donné.

4.2.4 Opérateur de croisement

Nous reprenons les deux opérateurs de croisement décrits dans le cadre du « block scheduling » et proposons un troisième opérateur de croisement traitant le vecteur de séparation des salles d'opération V_3 comme ci-dessous.

Étant donné deux parents P_1 et P_2 choisis pour les opérations de croisement,

Etape 1 : Copier tous les éléments à l'exception de V_3 et V_4 du parent P_1 à l'enfant O_1 et du parent P_2 à l'enfant O_2 ;

Etape 2 : Copier une partie aléatoire du vecteur V_3 du patient P_1 à l'enfant O_2 ; en même temps, copier la partie correspondante du vecteur V_3 du parent P_2 à l'enfant O_1 ;

Etape 3 : Insérer les éléments de la partie restante du vecteur V_3 de chaque parent à l'enfant correspondant en veillant à ce que tous les éléments du vecteur V_3 des enfants soient triés par ordre croisant de valeurs ;

Etape 4 : Remplir le vecteur V_4 des enfants en se basant sur le vecteur V_4 du parent correspondant et en prenant en compte les changements intervenus dans le vecteur V_3.

4.2.5 Opérateur de mutation

Nous reprenons les trois opérateurs de mutation décrits précédemment et en ajoutons un de plus pour le vecteur V_3.

Etape 1 : Générer aléatoirement un nombre entier $k \in \{1, \cdots, M_1 - 1\}$

Etape 2 : Supprimer le $k^{ème}$ élément du vecteur V_3, ainsi V_3 devient un vecteur à $M_1 - 1$ dimensions.

Etape 3 : Générer aléatoirement un nombre entier $j \in \{1, \cdots, N + 1\}$ et l'insérer dans V_3 en tenant compte que tous les éléments de V_3 sont triés par ordre croissant de leur valeur ;

Etape 4 : Remplir le vecteur V_4 des enfants en se basant sur le vecteur V_4 du parent correspondant et en prenant en compte les changements intervenus dans le vecteur V_3.

4.2.6 Evaluation et l'opérateur de reproduction

Comme il n'y a pas de différence à ce niveau entre les deux stratégies, nous ne détaillons plus ce point.

4.2.7 Opérateur d'amélioration locale

La méthode de recherche Tabou, basée sur celle du Nowicki et Smunicki [1998], peut toujours s'appliquer pour notre problème dans cette section. La seule différence se situe dans la stratégie de mouvement d'insertion au premier étage (les salles d'opération).

Étant donné que toutes les salles d'opération sont identiques dans le problème concerné dans cette section, nous pouvons uniformiser la stratégie de mouvement d'insertion pour les salles d'opération et les lits de réveil dans la SSPI par la stratégie appliquée à cette dernière.

Donc, un vecteur de cinq dimensions $mv = (s, a, x, b, y)$ ($s \in \{1,2\}$, $a, b \in \{1, \cdots, M_s\}$, $x \in \{1, \cdots, N_a^{(s)}\}, y \in \{1, \cdots, N_b^{(1)}\}$) représente un mouvement d'insertion entre la position x dans la séquence $\pi_a^{(s)}$ et l'autre position y dans la séquence $\pi_b^{(s)}$, comme celui présenté dans la section 4.1.7.

4.2.8 Borne inférieure pour l'évaluation de la performance de l'algorithme proposé

La borne inférieure que nous avons proposée pour le problème d'ordonnancement de « block scheduling » n'est plus valable pour le cas d'un « open scheduling » et donc une nouvelle borne inférieure doit être proposée

Lemme 2 : $OLB = \omega * \max\left\{ t_{i^0}^1, \frac{\sum_{i \in \Omega} t_i^{(1)}}{M_1} \right\} + \max\left\{ t_{i^0}^1 + t_{i^0}^{(2)}, \frac{SPT(M_2) + \sum_{i \in \Omega} (t_i^{(2)} + t_i^{(2)})}{M_1 + M_2} \right\}$

est une borne inférieure de la valeur optimale de la fonction objectif du problème d'ordonnancement d'une stratégie d' « open scheduling ».

Ce Lemme est prouvé en annexe 4.

Partie V :

Résultats expérimentaux

1. Introduction

Dans les parties précédentes de ce travail, nous avons présenté nos problèmes de planification et d'ordonnancement des interventions au sein d'un bloc opératoire ainsi que les méthodes et techniques envisagées pour les résoudre. Dans cette partie, les résultats expérimentaux seront montrés.

Etant donné que nous traitons les problèmes de la gestion du bloc opératoire en deux étapes : planification d'une part et ordonnancement d'autre part, nous allons présenter tout d'abord les résultats expérimentaux des méthodes pour les problèmes de planification hebdomadaire et ensuite pour les problèmes d'ordonnancement journaliers. A chaque étape, les deux stratégies, « block scheduling » et « open scheduling » sont traitées respectivement.

Pour réaliser nos expérimentations, nous avons utilisé un IBM ThinkPad T42 (CPU : PM 1,6GHz, Mémoire : 256 Mo) avec comme logiciel Microsoft VC++ 6.0 et comme solveur de programmation linéaire : COIN.[15]

2. Résultats expérimentaux pour les problèmes de planification

Dans cette section, nous montrons les résultats expérimentaux pour les problèmes de planification des deux stratégies : « block scheduling » et « open scheduling ».

Rappelons que deux stratégies, le « block scheduling » et l'« open scheduling », ont été étudiées pour affecter un ensemble d'interventions aux salles d'opération pour une semaine et deux types de méthodes, une heuristique et une méthode exacte, ont été proposés pour résoudre le problème de planification. En effet, nous comparons tout d'abord les résultats obtenus par la manière heuristique et ceux obtenus par la méthode exacte, en prenant en compte le cas du chirurgien 1 comme exemple afin de comparer l'efficacité et la rapidité de ces deux méthodes.

Puis, une méthode de résolution va être choisie (dans cette thèse, nous choisissons les procédures heuristiques) pour résoudre le problème de même taille pour les deux stratégies concernées, le « block scheduling » et l'« open scheduling », afin de comparer les efficacités de ces deux stratégies pour le problème de planification hebdomadaire où plusieurs chirurgiens sont pris en compte. Notons que la stratégie du « block scheduling » traite le problème de planification pour chaque chirurgien (le nombre de problèmes de planification traités est égal au nombre de chirurgien) et puis les résultats de ces problèmes sont agrégés comme données pour l'étape d'ordonnancement et ce du fait de la contrainte de disponibilité des lits de la SSPI. Par contre pour la stratégie de l'« open scheduling », le problème de planification est résolu pour l'ensemble des chirurgiens et donc les résultats de ce problème constituent les données pour l'étape d'ordonnancement.

[15] http://www.coin-or.org/Clp/index.html

2.1 Stratégie du « block scheduling »

2.1.1 Données

La plupart des études menées à ce jour utilisent une loi normale [Dexter, 2000] ou Lognormale [Chaabane, 2004][Dexter *et al.*, 1999c, 2001a, 2003, etc.][Kharraja, 2003][Jabali, 2004][Strum *et al*., 1998][Zhou et Dexter, 1998] pour générer les données expérimentales concernant des durées opératoires et de réveil des interventions [Dexter et Tinker, 1995].

En se basant sur les analyses de données provenant du service d'endoscopie de l'hôpital de la Croix-Rousse de Lyon (France), Combes *et al.* [2004] ont mené une étude statistique qui conduit à considérer la loi Log-Normale comme un modèle non approprié, contrairement à ce qui est préconisé par May *et al.* [2000]. Ils ont montré qu'une loi prenant en compte plus de deux dimensions n'est pas nécessaire pour la modélisation du phénomène observé et que la durée opératoire peut être approchée simplement par une loi de Pearson III (loi gamma avec décalage). En conséquence, nous générons les durées opératoires et de réveil des interventions par la loi de Pearson III.

Etant donné un plan directeur d'affectation des plages horaires aux chirurgiens dans un hôpital pour une semaine (montré par le Tableau 11), les problèmes de planification pour la stratégie du « block scheduling » visant à affecter les interventions de chaque chirurgien à leurs plages horaires réservées seront résolus par la procédure heuristique PHBGC et la méthode exacte de Branch-and-Price.

	Lundi (13 plages horaires)						Mardi (11 plages horaires)					
	Salle 1	Salle 2	Salle 3	Salle 4	Salle 5	Salle 6	Salle 1	Salle 2	Salle 3	Salle 4	Salle 5	Salle 6
9:00	Chir 1	Chir 3	Chir 2	Chir 5	Chir 4	Chir 6	Chir 7	Chir 3	Chir 5	Chir 8	Chir 4	
10:00												
11:00												
12:00								Chir 2				
13:00												
14:00										Chir 3		
15:00	Chir 2	Chir 1					Chir 8					
16:00			Chir 5						Chir 7		Chir 2	
17:00				Chir 6		Chir 3						
18:00		Chir 4			Chir 1			Chir 5				
19:00												
20:00												
21:00												

	Mercredi (12 plages horaires)						Jeudi (12 plages horaires)					
	Salle 1	Salle 2	Salle 3	Salle 4	Salle 5	Salle 6	Salle 1	Salle 2	Salle 3	Salle 4	Salle 5	Salle 6
9:00	Chir 1	Chir 4	Chir 6	Chir 2	Chir 7	Chir 8	Chir 2	Chir 4	Chir 5	Chir 6	Chir 8	Chir 3
10:00												
11:00												
12:00												
13:00												
14:00			Chir 1					Chir 6	Chir 4			
15:00	Chir 2	Chir 5		Chir 4			Chir 3			Chir 5		
16:00												
17:00					Chir 6						Chir 2	Chir 7
18:00												
19:00		Chir 6										
20:00												
21:00												

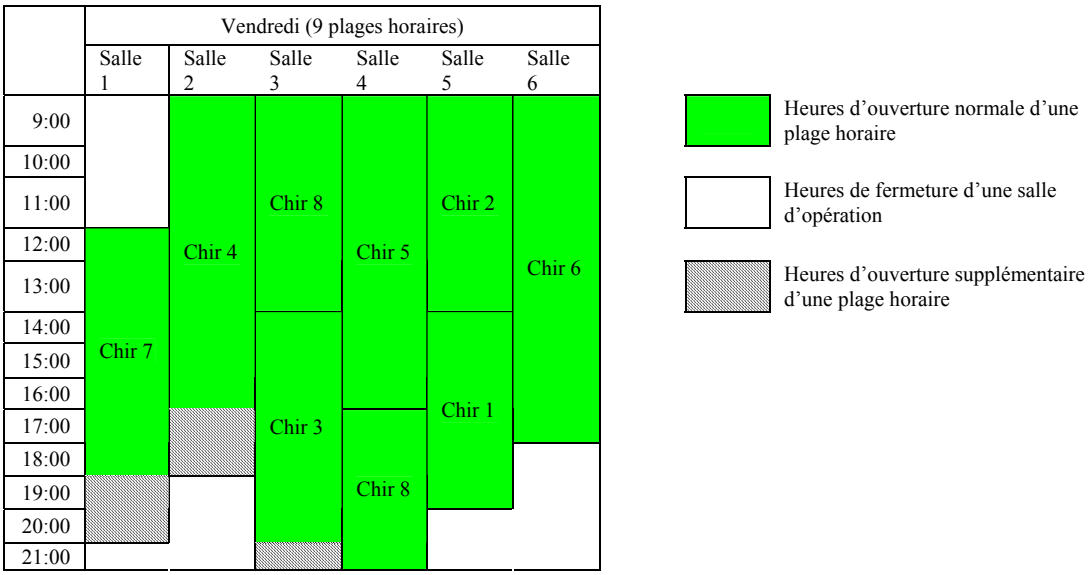

Tableau 11 : Plan directeur d'affectation des plages horaires aux chirurgiens pour une semaine dans un hôpital

Comme montré par le Tableau 11, nous sommes déjà informés sur toutes les plages horaires réservées dans un bloc opératoire: leurs heures d'ouverture et fermeture, les salles d'opération affectées et les chirurgiens correspondants etc. Par exemple, pour le chirurgien 1, il y a trois plages horaires réservées le lundi, la première est de 9h à 13h dans la salle d'opération 1, y compris une durée supplémentaire (de 13 à 15). Sa deuxième plage horaire est de 15h à 18h dans la salle d'opération 2 et sa dernière plage horaire de ce jour est de 18h à 22h dans la salle d'opération 5 sans durée supplémentaire.

La stratégie du « block scheduling » vise à affecter les interventions d'un chirurgien en question à ses plages horaires réservées. Donc, pour montrer la qualité des solutions obtenues lors de l'étape de planification par nos méthodes proposées, nous ne testons nos procédures de PHBGC et de Branch-and-Price pour le chirurgien 1 uniquement. Puis dans les sections suivantes, nous allons traiter toutes les affectations des interventions pour chaque chirurgien concerné et comparer les résultats agrégés avec ceux de la stratégie de l'« open scheduling ». Tout d'abord, nous précisons les données utilisées pour les expérimentations (le problème de planification pour le chirurgien 1) dans cette section ci-dessous :

- La durée de planification : $N_{jour} = 5$ (jours), c'est donc une planification pour une semaine ;
- Le nombre des plages horaires réservées préalablement au chirurgien 1 : $NB_1 = 3$, $NB_2 = 0$, $NB_3 = 2$, $NB_4 = 0$, $NB_5 = 1$;
- Le nombre d'heures normales disponibles d'une plage horaire : $TOR_1^1 = 4$, $TOR_2^1 = 3$, $TOR_3^1 = 4$, $TOR_1^3 = 4$, $TOR_2^3 = 6$, $TOR_1^5 = 6$, les autres sont égales à zéro, unité : une heure ;
- Le nombre maximal d'heures supplémentaires possibles pour les plages horaires : $TS_1^1 = 1$, $TS_1^3 = 1$, $TS_2^3 = 2$, les autres sont égales à zéro, unité : une heure ;
- Le rapport entre le coût d'une heure normale et d'une heure supplémentaire $\beta = 1{,}5$;
- Le nombre d'interventions à assigner aux plages horaires réservées du chirurgien 1 sur la durée de planification (une semaine) : $N_{interv} = 20, 30, 40, 50, 60$;
- La date limite d'intervention, D_i ($i = 1, \ldots, N_{interv}$), est générée entre [1, 14] par la loi uniforme dont l'unité est d'un jour ;

- La durée opératoire de l'intervention i, t_i, est générée entre [30, 150] par la loi de Pearson III où la moyenne est 80, l'écart type est 15 dont l'unité est d'une minute.

2.1.2 Résultats numériques par la procédure PHBGC pour le chirurgien 1

Nous avons exécuté 20 exemples pour chaque cas de N_{interv} et fait une comparaison entre la solution approchée du problème de planification trouvée au moyen de la procédure PHBGC (prenant en compte les contraintes d'intégrité) et la borne inférieure (avec relaxation des contraintes d'intégrité), *LP*, trouvée par la procédure GC. Vu que nous avons élaboré quatre versions de la procédure PHBGC (selon le type de sélection effectué), nous comparons tout d'abord les résultats de ces quatre règles et puis acceptons la meilleure solution comme la solution approchée finale. Ainsi, nous combinons les mérites des différentes règles de sélection.

Les indicateurs de comparaison déployés sont les suivants :

N_{interv}	:	le nombre d'interventions à assigner sur la durée de planification (une semaine) ;
LP	:	la valeur de la fonction objectif de la solution obtenue par la procédure GC, c'est une solution « optimale » sans respect des contraintes d'intégrité ;
H1 à H4	:	la valeur de la fonction objectif de la solution approchée (respectant toutes les contraintes d'intégrité) obtenue par une règle de sélection correspondante dans la procédure PHBGC. H1 correspond à la version 1 de la procédure PHBGC , H2 correspond à la version 2 de la procédure PHBGC, H3 correspond à la version 3 de la procédure PHBGC et H4 correspond à la version 4 de la procédure PHBGC ;
H	:	la valeur de la fonction objectif de la solution approchée finale (la meilleure entre les solutions approchées de H1 à H4) ;
(H-LP)/H	:	l'écart relatif entre H et LP ;
MLP	:	La valeur moyenne des LP ;
MH	:	La valeur moyenne des H pour les problèmes de même taille ;
Min((H-LP)/H)	:	l'écart relatif minimal entre H et LP ;
NColGenLP	:	Le nombre total de colonnes générées dans la procédure GC ;
NColGenH	:	le nombre total de colonnes générées dans la procédure PHBGC pour obtenir la solution approchée H ;
CPULP	:	le temps moyen de calcul de LP (en secondes) ;
CPUH	:	le temps moyen de calcul de H (en secondes) ;
CPULP/CPUH	:	le rapport moyen de CPULP sur CUPH ;
pNbS	:	le rapport entre le nombre des interventions dont la date limite est supérieure à une semaine et le nombre total des interventions à assigner ;
pNbSAsgnH	:	le rapport entre le nombre d'interventions, dont la date limite est supérieure à une semaine, assignées par la procédure PHBGC et le nombre total des interventions dont la date limite est supérieure à une semaine ;
pNbAsgnH	:	le rapport entre le nombre des interventions assignées par la procédure PHBGC et le nombre total des interventions ;
PTAsgnH	:	le rapport de la durée opératoire totale des interventions assignées par la procédure PHBGC à la durée totale des interventions à

assigner ;

PORUH : l'utilisation moyenne des salles d'opération, c'est le rapport de la durée opératoire totale des interventions assignées par la procédure PHBGC aux heures normales d'ouverture totale des salles d'opération.

Tous les résultats sont les résultats moyens obtenus sur les 20 exemples, pour chaque cas de N_{interv}.

Ninterv	LP	H1	H2	H3	H4	H
20	779,50	826,58	781,58	781,58	781,58	779,50
30	386,00	422,00	386,00	386,00	386,00	386,00
40	20,38	35,58	20,38	20,77	32,88	20,38
50	0,00	0,36	0,36	4,29	49,46	0,36
60	0,00	3,38	3,38	5,96	15,35	2,88

Tableau 12 : Comparaison entre les valeurs de fonction objectif des solutions approchées et la borne inférieure LP

Le Tableau 12 nous montre que H2 est systématiquement meilleure que les autres versions de l'heuristique selon les valeurs moyennes (pour les cas où tous les heuristiques peuvent trouver la solution réalisable) de la fonction objectif. Cependant, il faut noter que la version 2 ne peut résoudre que 98% des exemples testés mais les versions 1 et 3 peuvent trouver la solution réalisable pour 100% des cas. Il est évident que la version 4 est pire car pour 13% des exemples, elle ne peut pas trouver de solution réalisable et ses valeurs de fonction objectif sont aussi les pires dans la plupart des cas. Par conséquent, nous constatons que les solutions obtenues par les différentes versions de l'heuristique varient parmi les exemples testés, mais que la meilleure solution est toujours très proche de la borne inférieure.

(H-LP)/H%	0	]0 5]	]5 10]	]10 15]	]15 20]	]20 100]
Proportion	90,00%	0,00%	0,00%	0,00%	1,00%	9,00%
Proportion cumulée	90,00%	90,00%	90,00%	90,00%	91,00%	100,00%

Tableau 13 : Proportion de problèmes pour chaque fourchette d'écarts relatifs de la solution approchée H par rapport à la borne inférieure LP

Le Tableau 13 nous montre que dans 90% des exemples l'écart entre la borne inférieure et la solution approchée est zéro, cela signifie que nous avons trouvé la solution optimale par la procédure PHBGC.

N_{interv}	LP	H	Min{(H-LP)/H} en %	Moyenne des écarts relatifs (H-LP)/H en %	Ecart relatif des moyennes (MH-MLP)/MH en %
20	779,5	779,5	0	0	0
30	386	386	0	0	0
40	13,25	16,35	0	18,88	22,63
50	0	0,25	0	5	- (car LP = 0)
60	0	2,75	0	20	- (car LP = 0)

Tableau 14 : Ecart relatif de la solution approchée H par rapport à la borne inférieure LP

Dans le Tableau 14, nous pouvons trouver que l'écart entre les bornes inférieures et les solutions approchées est toujours très faible. Notons que pour les cas de taille 50 et 60, nous ne montrons pas les valeurs de la dernière colonne car LP est égale à zéro dans ces cas. De plus, comme nous pouvons trouver les solutions réalisables dans 100% des

exemples en choisissant la meilleure solution parmi celles obtenues par les quatre versions de l'heuristique, les valeurs montrées dans le Tableau 14 sont des valeurs moyennes sur 20 exemples. Cela explique la légère différence obtenue entre les valeurs des Tableaux 12 et 14.

N_{interv}	NcolGenLP	NcolGenH	CPULP	CPUH	CPULP/CPUH
20	30,5	207,75	0,16	1,03	0,16
30	47,3	348,65	0,30	2,12	0,14
40	62,38	308,69	0,44	3,41	0,13
50	65,25	297,2	0,49	4,21	0,12
60	67,15	320,75	0,52	3,98	0,14

Tableau 15 : Comparaison du nombre de colonnes générées et du temps de calcul

Dans le Tableau 15, nous comparons la performance de calcul des algorithmes. Le temps de calcul augmente avec la taille du problème mais le temps de calcul de la procédure PHBGC est toujours plus élevé que celui de la procédure GC. Cependant, tous les problèmes peuvent être résolus dans la minute. Plus il y a de colonnes générées dans une procédure, plus le temps de calcul augmente.

N_{interv}	PNbs	PNbsAsgnH	PNbAsgnH	PTAsgnH	PORUH
20	0,60	0,91	0,95	0,95	0,92
30	0,63	0,95	0,97	0,97	0,96
40	0,63	0,90	0,94	0,94	0,99
50	0,61	0,66	0,80	0,78	1,00
60	0,61	0,43	0,66	0,64	1,00

Tableau 16 : Taux d'affectation des interventions et taux d'utilisation des plages horaires réservées au chirurgien 1

Le Tableau 16 nous montre d'autres résultats numériques comme le taux d'affectation du nombre des interventions dont la date limite est supérieure à une semaine, le taux d'utilisation des salles d'opération pour les interventions dont la date limite est supérieure à une semaine. Dans notre algorithme, toutes les interventions dont la date limite est inférieure ou égale à une semaine doivent être affectées dans cette semaine, sinon, la solution est considérée comme non réalisable.

2.1.3 Résultats numériques par la procédure Branch-and-Price

Nous avons effectué une comparaison entre la solution exacte trouvée par la procédure de Branch-and-Price, la borne inférieure trouvée par la procédure GC et la solution approchée obtenue par la meilleure solution trouvée par la procédure PHBGC (dans les parties suivantes, nous l'appellerons la solution approchée trouvée par la procédure PHBGC). Le temps de calcul et le nombre de noeuds explorés sont montrés ici. Il faut noter que nous n'avons pu exécuter que trois cas (N_{interv} = 40, 50, 60) du fait que la procédure de Branch-and-Price prend toujours beaucoup trop de temps de calcul pour trouver la solution optimale si la procédure de PHBGC ne peut pas trouver la solution optimale. Par exemple, pour le cas où N_{interv} = 60, il a fallu plus de 7 heures pour trouver la solution exacte.

Pour analyser les résultats numériques obtenus, nous reprenons les notations des paramètres présentés dans la section précédente et ajoutons de nouveaux indicateurs de comparaison :

OPT : la solution exacte de la procédure de Branch-and-Price ;

(OPT-LP)/OPT : L'écart relatif entre OPT et LP ;
(H-OPT)/H : l'écart relatif de H par rapport à OPT ;
Nb résolus : le nombre de problèmes résolus à leur nœud initial (nœud de racine), c'est-à-dire, sans l'action de ramification, parmi les 20 exemples de chaque cas ;
MinCPUOPT : le temps minimal de calcul mis, par la procédure de Branch-and-Price, pour résoudre les problèmes non résolus par la procédure de PHBGC. De plus, MaxCPUOPT et CPUOPT représentent le temps maximal et moyen de calcul des 20 exemples pour chaque cas de N_{interv} (unité : secondes) ;
MinCPUH : le temps minimal de calcul par la procédure PHBGC parmi l'ensemble des 20 exemples de chaque cas de N_{interv} ; de plus, MaxCPUH et CPUH représentent les temps maximal et moyen de calcul des 20 exemples pour chaque cas de N_{interv} (unité : secondes) ;
CPU par nœud : le temps moyen de calcul pour évaluer un nœud dans la procédure Branch-and-Price, y compris les exemples résolus par la procédure PHBGC ;
Nb nœuds explorés : le nombre total des nœuds explorés dans la procédure de Branch-and-Price (au moins un nœud) ;
Nb colonnes générées : le nombre total de colonnes générées dans la procédure de Branch-and-Price.

Tous les résultats présentés ici sont des résultats moyens sur les 20 exemples pour chaque cas de N_{interv}.

(OPT-LP)/OPT%	0	]0 5]	]5 10]	]10 15]	]15 20]	]20 100]
Proportion	100,00%	0,00%	0,00%	0,00%	0,00%	0,00%
Proportion cumulée	100,00%	0,00%	0,00%	0,00%	0,00%	0,00%

Tableau 17 : Proportion de distribution d'écarts relatifs de OPT par rapport à LP

Le Tableau 17 nous montre que, dans 100% des exemples, la borne inférieure est égale à la solution optimale, cela implique que la procédure de génération de colonnes est capable de trouver une borne inférieure de très bonne qualité.

(H-OPT)/H%	0	]0 5]	]5 10]	]10 15]	]15 20]	]20 100]
Proportion	90,00%	0,00%	0,00%	0,00%	1,00%	9,00%
Proportion cumulée	90,00%	90,00%	90,00%	90,00%	91,00%	100,00%

Tableau 18 : Proportion de distribution d'écarts relatifs de H par rapport à OPT

Le Tableau 18 nous montre que dans 90% des exemples, l'écart entre la solution exacte et la solution approchée est égal à 0%, cela signifie que la procédure PHBGC a trouvé pour ces exemples la solution optimale.

N_{interv}	Min CPUOPT	Min CPUH	Max CPUOPT	Max CPUH	Moyenne CPUOPT	Moyenne CPUH
20	0,75	0,75	1,27	1,27	1,03	1,03
30	1,77	1,77	2,52	2,52	2,12	2,12
40	3,10	3,04	1768,06	4,19	140,83	3,41
50	3,28	3,28	786,97	5,06	43,37	4,21
60	2,63	2,63	27319,60	7,54	1384,06	3,98

Tableau 19 : Comparaison entre les temps de calcul par la procédure PHBGC et la procédure Branch-and-Price (en secondes)

Le Tableau 19 montre que le temps de calcul de la procédure Branch-and-Price est beaucoup plus long que celui de la procédure PHBGC, surtout pour calculer les problèmes de grande taille. En général, plus la taille d'un problème est grande, plus le temps de calcul sera long, surtout pour la procédure de Branch-and-Price dont le temps de calcul varie de 1,27 secondes à 27319,60 secondes (plus que 7 heures).

N_{interv}	LP	OPT	H	(H-OPT)/H en %	Nb résolus	Nb nœuds explorés	CPU par nœud
20	779,5	779,5	779,5	0	20	1	1,03
30	386	386	386	0	20	1	2,10
40	13,25	13,25	16,35	18,88	15	214,46	2,82
50	0	0	0,25	5	19	70,1	4,05
60	0	0	2,75	20	16	947,45	3,13

Tableau 20 : Autres résultats numériques

Le Tableau 20 nous montre que les bornes inférieures obtenues par la procédure GC et les solutions approchées obtenues par la procédure PHBGC sont très proches des solutions optimales obtenues par la procédure Branch-and-Price. De plus, la plupart des problèmes sont résolus sans l'action de ramification, c'est-à-dire, résolus directement par la procédure PHBGC. Grâce à des solutions initiales de bonne qualité, les temps de calcul pour les solutions optimales sont diminués. Nous pouvons aussi voir qu'en général plus il y a de nœuds explorés, plus long est le temps de calcul. De plus, nous trouvons aussi que moins l'écart est grand entre OPT et H, moins de nœuds sont explorés pour la solution optimale.

2.1.4 Conclusion

Considérant une stratégie du « block scheduling », les résultats expérimentaux des procédures PHBGC et des procédures de Branch-and-Price, effectués sur les données générées par la loi de Pearson III pour les interventions d'un chirurgien (le chirurgien 1), nous montrent que la procédure PHBGC peut trouver des solutions de bonne qualité pour la plupart des problèmes testés. En général, la procédure de Branch-and-Price peut résoudre des problèmes de planification de petite taille (moins de 60 interventions) mais sa performance s'avère moins efficace quand la taille de problème devient de plus en plus grande. Par exemple, le traitement de 60 interventions consomme déjà beaucoup de temps de calcul.

En général, nous avons montré grâce à nos expérimentations que la procédure PHBGC trouve de bonnes solutions pour la majorité des exemples, et ce quel que soit le jeu de données utilisé, avec un temps de calcul raisonnable.

Prenant en compte, d'une part, la grande différence entre les temps de calcul de la procédure PHBGC et de la procédure de Branch-and-Price et d'autre part, les qualités des

solutions obtenues par la procédure PHBGC, nous nous intéresserons donc dans la suite à la procédure PHBGC et l'utiliserons pour résoudre le problème de planification pour la stratégie de l'« open scheduling », plus complexe que celui pour la stratégie du « block scheduling ». De plus, afin de comparer les performances de deux stratégies concernées, nous comparons ensuite les résultats expérimentaux de ces deux stratégies sur un même ensemble d'interventions qui seront opérées par un même ensemble de chirurgiens dans un même bloc opératoire. La seule différence entre ces deux cas testés se situe sur l'affectation des durées d'ouverture des salles d'opération aux chirurgiens.

2.2 Résultats expérimentaux pour la stratégie de l'« open scheduling »

2.2.1 Données

Pour faciliter la comparaison entre la performance de la stratégie du « block scheduling » et celle d'«open scheduling », nous reprenons les informations données par le plan directeur (le Tableau 11) sauf que nous traitons les salles d'opération au lieu des plages horaires dans cette section car dans la stratégie de l'« open scheduling », les durées d'ouverture des salles d'opération peuvent être affectées à n'importe quel chirurgien et aucun chirurgien ne peut réserver préalablement des plages horaires. Selon les informations données par le Tableau 11, il y a au total 29 salles d'opération où nous pouvons assigner les interventions (six salles d'opération pour chaque jour sauf 5 salles pour mardi). D'une part, nous accumulons les durées disponibles d'une même journée de chaque chirurgien selon leurs plages horaires et nous obtenons les données TA_l^d (d = 1, …, N_{jour}, l = 1,…, 8), comme montré dans le Tableau 21 où les temps sont exprimés en heure et les zéros signifient que le chirurgien en question ne peut pas réaliser d'interventions ce jour-là.

jour / Chirurgien	Lundi	Mardi	Mercredi	Jeudi	Vendredi
Chir 1	13	0	13	0	6
Chir 2	13	13	13	12	5
Chir 3	12	11	13	13	8
Chir 4	11	6	9	13	10
Chir 5	12	11	0	12	8
Chir 6	12	0	5	13	9
Chir 7	0	12	11	6	9
Chir 8	0	12	13	8	10

Tableau 21 : Durées disponibles maximales des chirurgiens (unité : une heure)

D'autre part, nous accumulons les durées normales des plages horaires dans chaque salle d'opération pour obtenir leur durée d'ouverture normale, TOR_k^d (k =1,…, NS_d , d =1 ,…, N_{jour}) et aussi les durées supplémentaires des plages horaires dans la même salle d'opération pour obtenir leur durée supplémentaire, TS_k^d (k =1,…,NS_d, d =1,…, N_{jour}). Nous le montrons dans le Tableau 22.

Jour / Salle	Lundi		Mardi		Mercredi		Jeudi		Vendredi	
	TOR	TS	TOR	TS	TOR	TS	TOR	TS	TOR	TS
1	8	5	10	3	9	3	13	0	7	2
2	13	0	13	0	12	1	11	2	8	2
3	9	2	11	2	9	4	11	2	12	1
4	9	3	10	3	12	1	10	2	13	0
5	9	2	13	0	11	2	11	2	11	0
6	13	0			11	2	11	2	9	0

Tableau 22 : Durée d'ouverture des salles d'opération pendant une semaine (unité : une heure)

De plus, nous traitons un ensemble d'interventions dont le nombre est de 40, 80, 120 et 160 pour l'ensemble des chirurgiens. Les proportions de distribution des interventions parmi les différents chirurgiens sont générées aléatoirement selon la loi uniforme. Autrement dit, nous générons préalablement pour chaque intervention un chiffre entre [1, 8] par la loi uniforme, s'il est égal à *k*, cela signifie que l'intervention correspondante va être réalisée par le chirurgien *k*. La durée opératoire et la date limite de chaque intervention sont générées de la même manière que dans la section précédente et donc nous ne les détaillons plus.

Avant de faire la comparaison entre les deux stratégies, nous montrons tout d'abord les résultats expérimentaux de la stratégie de l'« open scheduling » par la procédure PHBGC et les comparons avec les bornes inférieures obtenues par la procédure GC. Nous avons exécuté pour chaque cas de N_{interv} 10 exemples, donc au total 40 exemples ont été testés.

2.2.2 Résultats numériques obtenus par la procédure PHBGC

Pour analyser les résultats numériques, nous reprenons les notations des paramètres présentées dans la section 2.1 sauf que nous remplaçons les valeurs H par les valeurs HO, qui représentent la valeur de la fonction objectif de la solution obtenue par la procédure GC pour la stratégie de l'« open scheduling », et LP par LPO, qui représente la valeur de la fonction objectif de la solution obtenue par la procédure GC pour la stratégie de l'« open scheduling ». Les résultats sont montrés dans les tableaux suivants.

(HO-LP)/HO (en %)	0	]0 5]	]5 10]	]10 15]	]15 20]	]20 100]	non résolus
Proportion	70,00%	7,50%	2,50%	5,00%	2,50%	0,00%	12,50%
Proportion cumulée	70,00%	77,50%	80,00%	85,00%	87,50%	87,50%	100,00%

Tableau 23 : Proportion de distributions des résultats

Le Tableau 23 nous montre que pour 80% des exemples, l'écart entre la solution approchée et la borne inférieure du problème est inférieur ou égal à 10%. De plus, pour 70% des cas, l'écart est égal à zéro. Cela implique que la solution obtenue par la procédure de génération de colonnes est très proche de la solution optimale. Notons qu'il y a 12,5% des exemples non résolus par la procédure PHBGC bien que la procédure GC ait résolu leurs relaxations linéaires.

N_{interv}	LPO	HO	Min {(HO-LPO)/HO} En%	(HO-LPO)/HO En%	(MHO-MLPO)/MHO En %
40	2860,1	2860,1	0	0	0
80	1976,64	1976,70	0,00	0,00	0,00
120	1119,30	1120,20	0,00	0,07	0,08
160	442,18	495,60	3,33	10,35	10,78

Tableau 24 : Ecart relatif de HO par rapport à LPO

Le Tableau 24 nous montre pour différentes valeurs de N_{interv}, la moyenne des LPOs et des HOs ainsi que l'écart relatif minimum et moyen. Dans ce tableau, nous voyons que dans tous les cas, nous avons obtenu un écart relatif minimum très faible et proche de 0. Nous trouvons aussi que presque tous les écarts relatifs moyens sont très faibles, c'est-à-dire que tous les problèmes testés sont bien résolus par la procédure PHBGC. Nous constatons donc que la solution obtenue par la procédure PHBGC est toujours très proche de la borne inférieure, ce qui signifie que la solution obtenue par la procédure PHBGC est toujours très proche de la solution optimale du problème.

N_{interv}	NColGenLPO	NcolGenHO	CPULPO	CPUHO	CPULPO/CPUHO
40	44,6	44,6	0,83	0,83	1
80	98,4	138,7	3,23	12,14	0,9
120	152,5	787,1	7,65	59,03	0,81
160	238	3574,4	18,59	169,62	0,11

Tableau 25 : Nombre de colonnes générées et temps de calcul

Le Tableau 25 compare la vitesse des algorithmes (la procédure GC et la procédure PHBGC). Le temps de calcul des deux algorithmes augmente avec la taille du problème. Le temps de calcul de la procédure PHBGC est toujours plus important que celui de la procédure GC. Cependant, tous les problèmes peuvent être résolus dans quelques minutes. Comme nous pouvons nous y attendre, nous constatons aussi que plus il y a de colonnes générées dans une procédure, plus le temps de calcul augmente.

N_{interv}	PNbS	PNAsgnSHO	PTAsgnHO	PTORUHO
40	63,25%	100,00%	100,00%	87,39%
80	62,38%	100,00%	100,00%	87,74%
120	60,83%	98,83%	99,16%	90,84%
160	63,25%	83,36%	88,84%	94,63%

Tableau 26 : Taux d'affectation des interventions et utilisation des salles d'opération

Le Tableau 26 montre d'autres résultats numériques, comme le taux d'affectation des durées opératoires, le taux d'utilisation des salles d'opération, le taux d'affectation des interventions dont la date limite est supérieure à une semaine (dans notre algorithme, toutes les interventions dont la date limite est inférieure ou égale à une semaine doivent être affectées dans cette semaine, sinon, nous ne trouvons pas la solution réalisable du problème).

2.3 Comparaison entre les deux stratégies

Dans les sections précédentes, nous avons présenté les résultats expérimentaux des deux stratégies respectivement où les problèmes de « block scheduling » pour un chirurgien (le chirurgien 1) sont résolus par la procédure heuristique PHBGC ainsi que par la procédure

de Branch-and-Price. Les exemples d'« open scheduling » ne sont résolus que par la procédure heuristique PHBGC. En effet vu que le temps de calcul consommé par la procédure de Branch-and-Price est déjà très long pour les problèmes avec seulement 60 interventions et que les solutions approchées obtenues par la procédure PHBGC sont toujours très bonne qualité pour le problème de « block scheduling », nous abandonnons la méthode exacte pour n'utiliser que la procédure PHBGC pour résoudre le problème d'« open scheduling ». De plus, ce problème est plus complexe que le problème de « block scheduling » et donc la procédure de Branch-and-Price pour le problème d'« open scheduling » va consommer encore plus temps de calcul ainsi que plus de mémoire d'ordinateur que pour le problème de « block scheduling ». En fait, les résultats obtenus montrent que la procédure PHBGC peut aussi trouver des solutions de bonne qualité pour la stratégie de l'« open scheduling ».

Dans cette section, nous faisons la comparaison entre les deux stratégies sur les données montrées dans la section 2.2 (les données des plages horaires de stratégie du « block scheduling » sont montrées dans le Tableau 11). Du fait qu'un modèle de l'« open scheduling » traite les interventions opérées par un ensemble des chirurgiens mais un modèle de la stratégie du « block scheduling » ne traite que les interventions opérées par un chirurgien, nous utilisons tout d'abord les modèles du « block scheduling » pour traiter respectivement un sous-problème d'affectation d'interventions pour chaque chirurgien. Selon les données montrées dans la section 2.2, il y a au total 8 chirurgiens dans notre exemple et donc 8 sous-problèmes de planification du « block scheduling » ont été résolus par la procédure heuristique PHBGC. Puis nous accumulons leurs résultats et les comparons avec les résultats du modèle de l'«open scheduling ».

Nous reprenons les notations de la section 2.1 et ajoutons les indicateurs de comparaison suivants :

LPB	:	la somme des valeurs de fonction objectif des bornes inférieures (les solutions obtenues par la procédure GC) des sous-problèmes de planification (un sous-problème de planification traite l'affectation d'interventions d'un chirurgien et donc LPB est la somme des bornes inférieures de 8 sous-problèmes de planification) pour la stratégie du « block scheduling » ;
HB	:	la somme des valeurs de fonction objectif des solutions approchées (les solutions btenues par la procédure PHBGC) des sous-problèmes de planification pour la stratégie du « block scheduling » ;
(HB-HO)/HB	:	l'écart relatif entre HB et HO ;
NColGenHB	:	la somme des nombres totaux de colonnes générées dans la procédure PHBGC pour tous les sous-problèmes de planification pour la stratégie du « block scheduling » ;
CPUHB	:	le temps moyen de calcul de HB qui est la somme moyenne de temps de calcul de tous les sous-problèmes de planification pour la stratégie du « block scheduling » (en seconde) ;
CPUParColHO	:	le temps de calcul pour générer une colonne pour la stratégie de l'« open scheduling » ;
CPUParColHB	:	le temps de calcul pour générer une colonne pour la stratégie du « block scheduling ».

N_{interv}	LPO	HO	LPB	HB	(HB-HO)/HB En %
40	2860,1	2860,1	2860,1	2860,1	0,00
80	1976,64	1976,7	1980,6	1980,6	0,19
120	1119,3	1120,2	1160,07	1162,3	3,54
160	442,18	495,6	587,10	591,2	16,60

Tableau 27 : Ecart relatif entre les résultats de la stratégie du « block scheduling » et ceux d'« open scheduling »

Selon le Tableau 27, nous trouvons que plus grande est la taille du problème, meilleur est le résultat de la stratégie de l'« open scheduling ». Pour les problèmes de taille petite, il n'y a pas de grande différence entre les deux stratégies. Il faut noter que la stratégie du « block scheduling » peut trouver une solution réalisable pour 100% des exemples mais la stratégie de l'« open scheduling » ne peut trouver une solution réalisable que pour 87.5% des exemples car 5 exemples de N_{interv} = 160 ne peuvent pas être résolus par la procédure heuristique PHBGC de la stratégie de l'« open scheduling ».

N_{interv}	NColGenHO	CPUHO (en secondes)	NColGenHB	CPUHB (en secondes)	CPUParColHO (en secondes)	CPUParColHB (en secondes)
40	44,600	0,830	381,200	1,541	0,019	0,004
80	138,700	12,135	869,600	4,076	0,047	0,005
120	787,100	59,027	1407,000	7,610	0,056	0,005
160	3574,400	169,621	1951,600	11,202	0,047	0,006

Tableau 28 : Nombre de colonnes générées et temps de calcul

Le Tableau 28 compare la performance de calcul des algorithmes (la procédure PHBGC pour la stratégie du « block scheduling » et celle pour la stratégie de l'« open scheduling »). Le temps de calcul de la stratégie du « block scheduling » est plus élevé que celui de la stratégie de l'« open scheduling » pour les problèmes de petite taille (N_{interv} = 40) mais le temps de calcul de la stratégie de l'« open scheduling » devient toujours plus important que celui de l'autre stratégie. Ceci est dû au fait que le temps de calcul par colonne de la stratégie de l'« open scheduling » est toujours beaucoup plus important et que le nombre de colonnes générées augmente plus vite que celui de la stratégie du « block scheduling ». En général, tous les exemples testés sont résolus dans un temps de calcul raisonnable (moins de 5 minutes).

2.4 Conclusion

Dans cette section, nous avons présenté les résultats des expérimentations numériques de la procédure heuristique PHBGC et de la procédure exacte de Branch-and-Price pour résoudre des problèmes de planification la stratégie du « block scheduling », et ceux de la procédure heuristique PHBGC pour la stratégie de l'« open scheduling ». Nous avons aussi comparé les résultats de la procédure PHBGC entre ces deux stratégies sur un même ensemble de données.

Au vu des résultats obtenus, nous trouvons que la procédure PHBGC peut trouver des solutions approchées de très bonne qualité pour la plupart des problèmes testés par les deux stratégies et que les bornes inférieures trouvées par la procédure GC sont toujours très proches des solutions optimales obtenues par la procédure de Branch-and-Price pour la stratégie du « block scheduling ». Pour la procédure de Branch-and-Price, elle peut résoudre des problèmes de petite taille (moins de 60 interventions) en un temps raisonnable mais le temps de calcul augmente brutalement avec la croissance de la taille du problème.

Selon la comparaison faite entre les deux stratégies, la stratégie de l'« open scheduling » peut trouver une affectation meilleure que celle de « block scheduling » bien qu'il existe quelques exemples non résolus. Concernant le temps de calcul, celui de la stratégie d'« open scheduling » est plus important pour les problèmes de grande taille mais les deux stratégies peuvent résoudre le problème de planification dans un temps raisonnable.

3. Résultats expérimentaux pour les problèmes d'ordonnancement

A partir de la planification hebdomadaire obtenue lors des sections 2.2 et 2.3, nous ordonnançons maintenant les interventions par l'algorithme génétique hybride présenté dans la partie IV et présentons les résultats obtenus.

Comme dans la section précédente, nous montrons les résultats des deux stratégies respectivement et puis effectuons la comparaison entre les deux.

3.1 Résultats numériques pour la stratégie du « block scheduling »

3.1.1 Données

Selon le plan directeur d'allocation des plages horaires (Tableau 11), les interventions de chaque chirurgien ont été affectées à leurs plages horaires réservées dans l'étape de planification. Donc nous allons ordonnancer les interventions dans les plages horaires pour chaque jour de planification (par exemple, pour le lundi, nous accumulons les interventions affectées aux 13 plages horaires réservées) afin de minimiser le coût total d'opération du bloc opératoire (les salles d'opération et la SSPI). Notons que dès qu'un ensemble d'interventions est assigné à une plage horaire, elles ne peuvent pas être affectées à d'autres plages horaires, autrement dit, toutes les plages horaires sont spécialisées (par exemple, il y a 13 plages horaires spécialisées le lundi), mais les lits de réveil sont communs pour tout patient sortant de la salle d'opération. De plus, les résultats que nous montrons dans cette section sont les programmes opératoires construits pour une semaine.

Les données complémentaires utilisées dans cette étape d'ordonnancement sont les suivantes :

- la durée de post-anesthésie de l'intervention i, $t_i^{(2)}$, est générée entre [15, 60] par une loi de Pearson III où la moyenne est égale à la durée opératoire correspondante en salle d'opération $t_i^{(1)}$ moins 10. Cette manière de générer la durée de post-anesthésie est utilisée par Dexter et Tinker [1995], Kharraja *et al.* [2002] et Jebali *et al.*[2006] pour générer les données par une loi Lognormale. Nous remplaçons ici la loi de Lognormale par la loi de Pearson III. L'écart type est égal à 15 et l'unité utilisée est d'une minute ;
- le rapport entre le coût d'une heure d'ouverture de la salle d'opération et d'une heure d'ouverture de la SSPI : $\omega = 5{,}8$;
- Il y a 10 lits de réveil dans la SSPI.

Nous avons exécuté pour chaque cas 10 exemples, donc au total 40 exemples ont été testés.

3.1.2 Résultats numériques par l'algorithme génétique hybride

Nous avons fait une comparaison entre la valeur de la fonction objectif des solutions obtenues par l'algorithme génétique hybride et la borne inférieure NLB que nous avons proposé dans la partie IV. Notre algorithme hybride se compose de deux parties : l'algorithme génétique et la recherche Tabou. Donc, nous avons aussi fait les comparaisons entre les résultats de l'algorithme génétique sans la recherche Tabou et ceux de la pure recherche Tabou. Nous avons également fait une comparaison entre la valeur de critère auxiliaire des solutions obtenues par les trois algorithmes.

Tout d'abord, nous présentons les nouveaux indicateurs de comparaison utilisés :

GATS : la valeur de la fonction objectif, $f = \omega * C_{max}^{(1)} + C_{max}^{(2)}$, de la solution obtenue par l'algorithme génétique hybride ;

GATS_A : la valeur du critère auxiliaire, $f' = \omega * moyenne(C_i^{(1)}) + C_{max}^{(2)}$, de la solution obtenue par l'algorithme génétique hybride ;

GA : la valeur de la fonction objectif, $f = \omega * C_{max}^{(1)} + C_{max}^{(2)}$, de la solution obtenue par l'algorithme génétique sans le traitement de l'amélioration locale ;

GA_A : la valeur du critère auxiliaire, $f' = \omega * moyenne(C_i^{(1)}) + C_{max}^{(2)}$, de la solution obtenue par l'algorithme génétique sans le traitement de l'amélioration locale ;

TS : la valeur de la fonction objectif de la solution obtenue par la seule recherche Tabou ;

TS_A : la valeur du critère auxiliaire obtenue par la seule recherche Tabou ;

NLB : la borne inférieure proposée dans la partie IV du problème d'ordonnancement pour la stratégie du « block scheduling » ;

CPUGATS : le temps moyen de calcul de l'algorithme génétique hybride proposé (en secondes) ;

CPUGA : le temps moyen de calcul de l'algorithme génétique simple (sans l'amélioration locale) (en secondes) ;

CPUTS : le temps moyen de calcul de la recherche Tabou (en secondes).

N_{interv}	GATS	GA	TS	NLB
40	4817,24	4845,8	4845,9	4817,04
80	4760,16	4760,16	4771,88	4759,36
120	5178,3	5178,6	5181,14	5177,2
160	5320,14	5320,34	5334,24	5317,54

Tableau 29 : Comparaison des valeurs moyennes de la fonction objectif des solutions obtenues par les différentes méthodes

N_{interv}	GATS_A	GA_A	TS_A
40	4282,247	4295,341	4293,187
80	4136,997	4151,923	4155,6
120	4404,359	4447,124	4448,601
160	4659,867	4734,231	4721,121

Tableau 30 : Comparaison des valeurs moyennes du critère auxiliaire des solutions obtenues par les différentes méthodes

Les Tableau 29 et 30 montrent que l'algorithme génétique hybride peut toujours trouver la meilleure valeur de la fonction objectif ainsi que la valeur de critère auxiliaire par rapport à d'autres algorithmes bien qu'il n'y ait pas de grande différence entre les

résultats des trois algorithmes. De plus, la solution obtenue par l'algorithme hybride est très proche de la borne inférieure. Cela implique que notre borne inférieure est de très bonne qualité et proche de la solution optimale.

N_{interv}	CPUGATS	CPUGA	CPUTS
40	4,32	0,87	0,27
80	24,75	1,38	0,67
120	38,31	1,92	1,00
160	68,53	2,71	1,45

Tableau 31 : Comparaison entre les temps de calcul (en secondes)

Le Tableau 31 montre la comparaison sur le temps de calcul. Nous trouvons que tous les problèmes peuvent être résolus dans un temps raisonnable (dans la minute).

3.2 Résultats numériques pour la stratégie de l'« open scheduling »

Nous prenons les mêmes données et notations qu'à la section 3.1 sauf que les informations concernant des plages horaires sont remplacées par les informations des salles d'opération.

Comme à la section 3.1, nous avons aussi fait des comparaisons entre les solutions obtenues par l'algorithme hybride, l'algorithme génétique et la recherche Tabou. De plus, nous présentons un nouvel indicateur de comparaison :

OLB : la borne inférieure proposée dans la partie IV du problème d'ordonnancement pour la stratégie de l'« open scheduling ».

Les résultats sont montrés dans les tableaux suivants.

N_{interv}	GATS	GA	TS	OLB
40	2523,12	2605,9	2991,28	2413,69
80	3937,7	4182,04	4955,76	3726,8
120	4677,38	5518,14	6861,4	4444,73
160	5617,15	6222,28	7504,24	4986,4

Tableau 32 : Comparaison entres les valeurs de la fonction objectif des solutions obtenues par différentes méthodes

N_{interv}	GATS_A	GA_A	TS_GA
40	2405,96	2510,64	2552,51
80	3667,29	3949.95	3935,11
120	4383,51	5067,85	5030,33
160	5620,96	5635,37	5638,57

Tableau 33 : Comparaison entre les valeurs du critère auxiliaire des solutions obtenues par différentes méthodes

Par les résultats montrés dans les Tableau 32 et 33, nous trouvons que les solutions obtenues par l'algorithme génétique hybride sont nettement meilleures que celles obtenues par les deux autres algorithmes.

Selon les comparaisons avec les bornes inférieures, les écarts sont plus élevés que ceux de la stratégie du « block scheduling ». Cela signifie que la borne inférieure proposée pour la stratégie de l'« open schduling » n'apparaît pas être très proche de la solution

optimale. Cependant, dans le cas de petite taille (N_{interv} = 40) cette borne inférieure est de bonne qualité. Cela signifie que les solutions de notre algorithme génétique hybride sont encore très satisfaisants mais une meilleure borne inférieure ou d'autres benchmarks devraient être trouvés pour une meilleure analyse des résultats.

N_{interv}	CPUGATS	CPUGA	CPUTS
40	99,92	2,09	0,54
80	329,8	3,93	1,62
120	974,76	5,29	4,71
160	1635,21	9,98	9,58

Tableau 34 : Comparaison entre les temps de calcul (en secondes)

Concernant les temps de calcul, le Tableau 34 montre que l'algorithme génétique hybride consomme beaucoup plus de temps de calcul que les autres mais reste néanmoins raisonnable. Par conséquent, il s'agit d'un bon compromis entre la rapidité et la qualité de la solution.

3.3 Comparaison entre les deux stratégies

Comme à la section 2, nous comparons maintenant les performances des deux stratégies concernées. Les notations utilisées sont les suivantes :

GTO : la valeur de la fonction objectif, $f=\omega*C_{max}^{(1)} + C_{max}^{(2)}$, de la solution obtenue par l'algorithme génétique hybride pour la stratégie de l'« open scheduling » ;

GTO_A : la valeur du critère auxiliaire, $f' =\omega*moyenne(C_i^{(1)})+C_{max}^{(2)}$, de la solution obtenue par l'algorithme génétique hybride pour la stratégie de l'« open scheduling » ;

CPUGTO : le temps moyen de calcul de l'algorithme génétique hybride proposé (en secondes) pour la stratégie de l'« open scheduling » ;

GTB : la valeur de la fonction objectif, $f=\omega*C_{max}^{(1)} + C_{max}^{(2)}$, de la solution obtenue par l'algorithme génétique hybride pour la stratégie du « block scheduling » ;

GTB_A : la valeur du critère auxiliaire, $f' =\omega*moyenne(C_i^{(1)})+C_{max}^{(2)}$, de la solution obtenue par l'algorithme génétique hybride pour la stratégie du « block scheduling » ;

CPUGTB : le temps moyen de calcul de l'algorithme génétique hybride proposé (en secondes) pour la stratégie du « block scheduling ».

N_{interv}	GTO	CPUGTO (en secondes)	GTB	CPUGTB (en secondes)	GTO_A	GTB_A
40	2523,12	99,92	4817,24	4,32	2552,51	4282,25
80	3937,70	329,80	4760,16	24,75	3935,11	4137,00
120	4677,38	974,76	5178,30	38,32	5030,33	4404,36
160	5617,15	1635,21	5318,85	68,96	5638,57	4610,04

Tableau 35 : Comparaison entre les deux stratégies

Le Tableau 35 nous montre que les solutions obtenues par la stratégie de l'« open scheduling » sont meilleures que celles obtenues par l'autre stratégie pour les problèmes dont la taille varie de 40 à 80. Mais à partir de 120, les solutions trouvées par la stratégie du « block scheduling » commencent à être meilleures que celles obtenues par la stratégie de l'« open scheduling ». C'est probablement à cause de la disponibilité des chirurgiens. De plus, pour les exemples de taille 160, il y a 5 exemples non résolus pour la stratégie de l'« open scheduling » parce que la première étape de panification n'a pas trouvé de

solution réalisable. Concernant le temps de calcul, la stratégie du « block scheduling » consomme moins temps de calcul que la stratégie de l'« open scheduling ».

3.4 Conclusion

Dans cette section, nous avons présenté des résultats numériques de l'algorithme génétique hybride proposé pour résoudre nos problèmes d'ordonnancement. Nous avons effectué une comparaison entre les solutions obtenues par notre algorithme hybride, l'algorithme génétique simple, la recherche Tabou et les bornes inférieures. Nous concluons que les solutions obtenues par l'algorithme hybride proposé sont de bonne qualité. De plus, les bornes inférieures, surtout celles pour le problème d'ordonnancement de « block scheduling » sont très proches de la solution optimale.

Selon la comparaison entre les deux stratégies, nous trouvons que la stratégie de l'« open scheduling » a de meilleures performances pour les problèmes de petite taille mais que la stratégie du « block scheduling » devient de plus en plus performante quand la taille du problème augmente.

Partie VI :

Traitement d'un cas réel : planification et ordonnancement de l'activité d'Endoscopie

1. Contexte de l'étude et problématique

Dans la partie V, nous avons présenté l'application et la comparaison de nos méthodes de résolution sur des données générées aléatoirement selon la loi de Pearson III. Dans cette partie, nous allons présenter le traitement d'un cas réel : la construction d'un programme opératoire pour le centre d'endoscopie de l'hôpital de la Croix-Rousse (Lyon).

1.1 Organisation du service d'endoscopie de l'hôpital de la Croix-Rousse

L'endoscopie est un examen d'une cavité interne du corps humain au moyen d'un endoscope (instrument muni d'un tube optique et d'un système d'éclairage est introduit dans les cavités naturelles du corps afin de les examiner).

La Figure 26 présente l'organisation très schématisée de l'organisation du service d'endoscopie étudié.

En général, tout patient admis dans le service d'endoscopie suit l'une des deux trajectoires suivantes :

- ✓ Si le geste doit être réalisé sous anesthésie alors le patient est transféré dans la salle d'endoscopie. Le médecin anesthésie-réanimateur (MAR) et l'infirmier anesthésiste diplômé d'état (IADE) anesthésient le patient. Le médecin gastro-entérologie (GE) réalise ensuite le ou les gestes assistés de l'infirmier(ère) diplômé(e) d'état (IDE) pendant que le MAR et l'IADE surveillent l'état du patient. Une fois le geste réalisé, le patient est transféré dans la SSPI. Un(e) IDE surveille le réveil du patient. Une fois la phase de réveil terminée, le patient est transféré dans son service d'origine d'hospitalisation. Les gestes sous anesthésie générale sont obligatoirement réalisés dans une salle nommée « salle d'endoscopie ».
- ✓ Si le geste est réalisé sans anesthésie générale, la seule différence réside dans le fait qu'il n'est pas nécessaire durant le geste, d'avoir un MAR et un(e) IADE. L'acte est réalisé dans la salle appelée « salle de gastroscopie » et le patient n'effectue aucun séjour en SSPI.

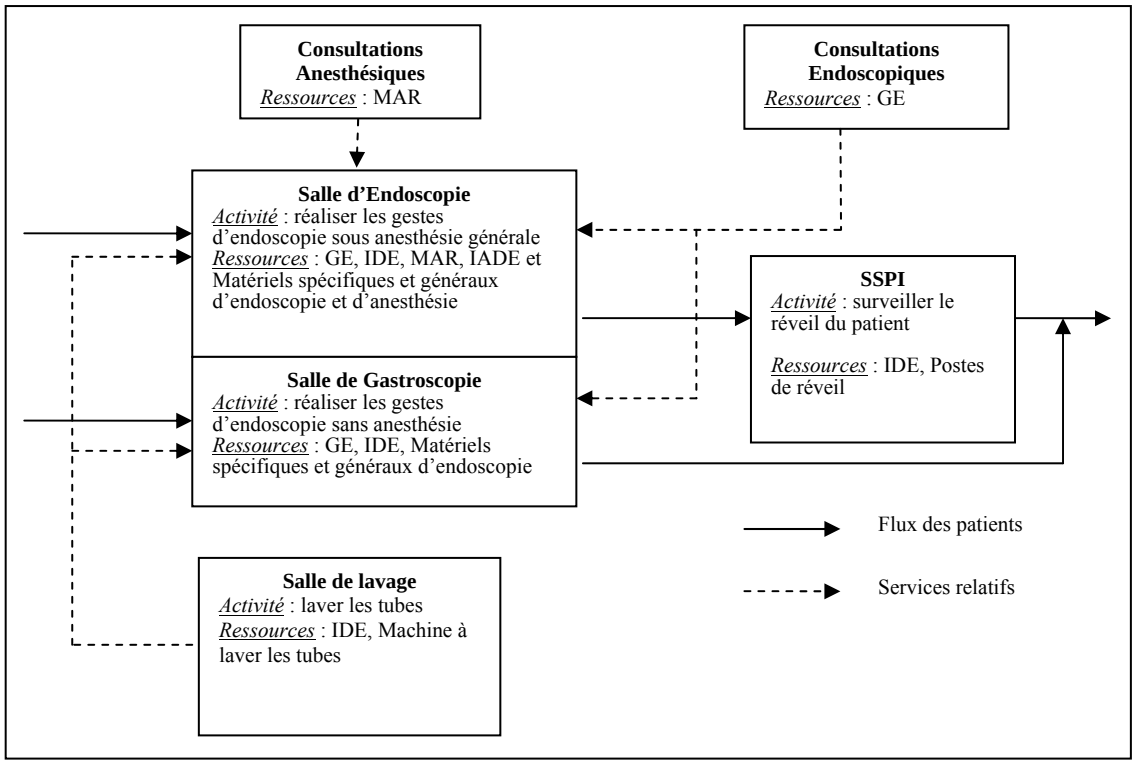

Figure 26 : Topologie et circulation des flux au sein du service d'endoscopie de l'hôpital de la Croix-Rousse – Hospices Civils de Lyon (France)

Tous les gestes réalisés par un médecin gastro-entérologue sur les patients, dans le service d'endoscopie, peuvent être considérés comme des interventions chirurgicales.

Dans la salle d'endoscopie, trois types d'interventions peuvent avoir lieu : la coloscopie ou la gastroscopie (intervention consistant en un seul geste) et la coloscopie avec une gastroscopie (intervention constituée de deux gestes sur un même patient).

Par contre, dans la salle de gastroscopie n'est pratiquée qu'une seule sorte d'intervention : la gastroscopie qui compte pour un seul geste.

Il faut préciser que le médecin gastro-entérologue est limité à un nombre maximal de gestes chirurgicaux par jour.

La durée opératoire d'une intervention correspond à la durée totale des gestes la composant (soit 1 ou 2 gestes par patient).

1.2 Problématique

Peu de travaux de recherche traitent de la planification et l'ordonnancement des blocs opératoire comme nous l'avons présenté dans la partie II, et il y en a évidemment encore moins qui se préoccupent spécifiquement de la gestion de la programmation opératoire d'un service d'endoscopie [Sonnenberg, 2000a, 2000b] [Patrick et Carl, 2005] du fait que l'endoscopie n'est qu'une des spécialités réalisées au sein d'un bloc opératoire (centralisé ou décentralisé).

L'endoscopie a la particularité de devoir faire face à des actes courts avec un « turnover » très important.

L'objectif réside dans la construction d'un programme opératoire efficace pour une période d'une semaine pour un service d'endoscopie comportant deux salles d'interventions.

Les hypothèses suivantes sont émises:

1. Chaque patient à assigner est préalablement affecté à une des salles spécialisées et à un chirurgien.
2. Les ressources humaines et matérielles sont en nombre suffisant pour un bon fonctionnement du service durant la semaine de planification, excepté les chirurgiens qui sont à prendre en compte vu qu'ils ne peuvent pas opérer plusieurs patients en parallèle.
3. Il existe toujours au moins un lit disponible pour les patients venant d'être opérés dans la salle d'endoscopie.
4. Il n'existe pas de préemption de salle d'opération : une intervention en-cours n'est pas interruptible.
5. Les urgences ne sont pas prises en compte.
6. Aucune plage horaire n'est affectée à l'avance à certains chirurgiens. Nous nous trouvons donc dans le cas le plus complexe, celui de l' «open scheduling ».
7. Tous les patients sont prêts pour leur intervention. Les salles d'opération sont opérationnelles dès leur ouverture.

Nous résolvons le problème de planification hebdomadaire par la procédure heuristique que nous avons présentée dans la partie III. Pour le problème d'ordonnancement journalier avec une contrainte de la disponibilité de chirurgiens, nous n'utilisons pas la méthode présenté dans la partie IV pour le résoudre car le modèle d'ordonnancement avec deux salles d'opération est un système particulier et donc un algorithme polynomial peut être utilisé pour obtenir une solution approchée de bonne qualité. Nous obtenons ainsi un programme opératoire réalisable pour ce service d'endoscopie sur une semaine.

2. Planification hebdomadaire du service d'endoscopie

Dans cette partie, nous allons présenter le modèle de planification permettant d'affecter les patients à une date d'intervention sur une période hebdomadaire de planification avec pour objectifs la minimisation du coût des heures supplémentaires et la maximisation d'utilisation des salles d'opération.

Nous proposons un modèle basé sur la programmation linéaire en nombres entiers (plus précisément binaires) pour résoudre le problème de planification concernée où une intervention correspond à un ensemble des gestes exigés pour un patient.

En plus des notations définies précédemment, nous introduisons quelques notations complémentaires spécifiques à ce cas particulier :

Notations :

g_i : le nombre de gestes composant l'intervention i ;

s_i : l'indice de la salle d'opération spécialisée pour opérer l'intervention i ;

N_{salle} : le nombre de salles d'opération. Dans ce cas précis, il y a une salle d'endoscopie et une salle de gastroscopie, $N_{salle} = 2$;

NG_l : le nombre maximum de gestes que le chirurgien l peut opérer durant une journée.

Modèle Mathématique du Problème d'Endoscopie (MMPE)

$$\text{Min} \quad \sum_{d=1}^{N_{jour}} \sum_{k=1}^{N_{ssalle}} C_k^d = \sum_{d=1}^{N_{jour}} \sum_{k=1}^{N_{salle}} \max\left\{ (TOR_k^d - \sum_{i\in\Omega} t_i z_{ik}^d), \beta(\sum_{i\in\Omega} t_i z_{ik}^d - TOR_k^d) \right\} \tag{2.1.1}$$

sous les contraintes

$$\sum_{d=1}^{D_i} \sum_{k=1}^{N_{salle}} z_{ik}^d = 1, \quad i \in \overline{\Omega} \tag{2.1.2}$$

$$\sum_{d=1}^{N_{jour}} \sum_{k=1}^{N_{salle}} z_{ik}^d \le 1, \quad i \in \Omega \tag{2.1.3}$$

$$\sum_{i\in\Omega} t_i z_{ik}^d \le TOR_k^d + TS_k^d, \quad k \in \{1,\cdots,N_{salle}\}, d \in \{1,\cdots,N_{jour}\} \tag{2.1.4}$$

$$\sum_{k=1}^{N_{salle}} \sum_{i\in\Omega_l} g_i z_{ik}^d \le NG_l, \ \ l \in \{1,\cdots,N_{chir}\}, \quad d \in \{1,\cdots,N_{jour}\} \tag{2.1.5}$$

$$\sum_{k=1}^{N_{salle}} \sum_{i\in\Omega_l} t_i z_{ik}^d \le \max\{TOR_k^d + TS_k^d, k = 1,\cdots,N_{salle}\}, l \in \{1,\cdots,N_{chir}\}, d \in \{1,\cdots,N_{jou} \tag{2.1.6}$$

$$z_{ik}^d \in \{0,1\} \quad i \in \Omega, k \in \{1,\cdots,NS_d\}, \ d \in \{1,\cdots,N_{jour}\} \tag{2.1.7}$$

La fonction objectif et les contraintes de ce modèle sont les mêmes que celles du modèle MMOS que nous avons présenté dans la partie III sauf que la seule contrainte de la disponibilité de chirurgiens (3.2.5) est remplacée par deux contraintes (2.1.5) et (2.1.6).

La contrainte (2.1.5) implique que le nombre total des gestes opérés par un chirurgien pendant une journée ne peut pas excéder le nombre autorisé.

La contrainte (2.1.6) implique que la durée opératoire totale d'un chirurgien pendant une journée ne peut pas excéder la durée d'ouverture maximale des salles d'opération.

Nous ne détaillons pas la procédure heuristique PHBGC (nous utilisons la première version de l'heuristique PHBGC dans cette partie) et allons directement à la deuxième étape : l'ordonnancement.

3. Problème d'ordonnancement pour le service d'endoscopie pour une journée

3.1 Description du problème d'ordonnancement pour le service d'endoscopie

La programmation opératoire étant non seulement constituée de la planification mais aussi de l'ordonnancement des patients dans le service d'endoscopie, nous allons maintenant nous concentrer sur l'ordonnancement afin de construire un programme opératoire réalisable et efficace.

Pour chaque jour d'ordonnancement, deux salles d'opération spécialisées sont disponibles : une salle d'endoscopie pour les interventions avec anesthésie et une salle de gastroscopie pour les interventions sans anesthésie. Pour chaque salle d'opération, un ensemble d'interventions a déjà été affecté par l'étape de planification et chaque intervention doit être réalisée par un chirurgien connu au préalable. Pour chaque chirurgien en question, ses interventions peuvent être réparties dans les deux salles, mais chaque chirurgien ne peut opérer deux interventions en même temps. Nous allons résoudre ce problème d'ordonnancement avec comme objectif la minimisation de la durée totale de l'ensemble des interventions (*makespan* , C_{max}).

Comme nous ne prenons pas en compte la contrainte concernant la SSPI, notre problème d'ordonnancement peut être transposé dans le domaine industriel à un modèle d'ordonnancement sur machines parallèles où toutes les machines sont spécialisées avec une contrainte de ressources renouvelables (les chirurgiens). Reprenant les notations proposées par Blazewicz *et al.* [1986] et Lawler *et al.* [1993] pour décrire ce problème d'ordonnancement, nous représentons notre problème par la notation $PDm|res\ \lambda\ \sigma\ \rho|C_{\max}$, où PDm représente m ($m = N_{salle}$) salles d'opération parallèles spécialisées, $res\ \lambda\ \sigma\ \rho$ implique qu'il y a λ ($\lambda = N_{chir}$) chirurgiens disponibles ; chaque chirurgien ne pouvant opérer qu'au plus $\sigma(\sigma = 1)$ interventions en même temps et chaque intervention ne pouvant être opérée que par au plus $\rho(\rho = 1)$ chirurgiens.

Comme prouvé par Kellerer et Strusevich [2004], un problème $PD2|res\ 2\ 1\ 1|C_{\max}$ peut être résolu par un algorithme polynomial mais dès que λ dépasse la valeur de deux, les autres paramètres restant identiques, cela devient un problème NP-difficile. De plus, la différence avec les problèmes classiques de gestion industrielle est qu'ici la variable de décision est binaire : une intervention a lieu ou n'a pas lieu. Cela rend, selon Ullman [1976], notre problème NP-complet même pour le cas avec seulement deux salles et un chirurgien ($m = 2, \lambda = 1$) et où les patients peuvent être opérés dans n'importe quel ordre. Nous pouvons donc conclure que notre problème d'ordonnancement $PDN_{salle}|res\ N_{chir}\ 1\ 1|C_{\max}$ où $N_{salle} = 2$, $N_{chir} > 2$ est aussi un problème NP-complet.

3.2 Une méthode polynomiale pour résoudre ce problème d'ordonnancement

Comme notre problème d'ordonnancement est un problème NP-Complet, nous allons adopter une méthode de technologie de groupe qui va regrouper les interventions en blocs, comme présenté par Kellerer et Strusevich [2004].

Nous appelons « tâche composée » (ou « bloc ») l'ensemble des interventions opérées par un même chirurgien le même jour dans la même salle. L'ensemble de ces interventions est traité comme un bloc sans temps d'attente entre les interventions d'un même bloc. Les interventions au sein d'un bloc peuvent être opérées dans n'importe quel ordre. Nous supposons aussi qu'il n'y a pas de temps de changement entre les chirurgiens et donc entre les blocs. Le programme opératoire débute dès l'ouverture de la salle d'opération.

Un problème d'ordonnancement de tâches composées d'un modèle d' « open shop » à deux machines spécialisées ($O_2\|C_{\max}$) a déjà été détaillé par Lawer *et al.* [1993] avec comme objectif la minimisation du « makespan ». Nous allons donc utiliser les idées de résolution développées pour le modèle d' « open shop » à notre problème. Grâce à l'algorithme de Gonzalez et Sahni présenté par Breit *et al.* [2001], nous pouvons résoudre

un modèle simplifié qui prend en compte deux salles d'opération. Le temps de calcul de cet algorithme est une fonction polynomiale du nombre de tâches composées, c'est-à-dire le nombre des chirurgiens. La complexité d'exécution de l'algorithme Gonzalez-Sahni est $O(N_{chir})$.

Vu les contraintes déjà respectées lors de l'étape de planification, l'algorithme de Gonzalez-Sahni va nous fournir d'office un ordonnancement réalisable c'est-à-dire respectant la contrainte portant sur les chirurgiens (un chirurgien n'opérera bien qu'un bloc à la fois).

Avant de détailler cette méthode, nous présentons tout d'abord les notations utilisées.

Notation :

O_{kl} : la tâche composée opérée le jour considéré par le chirurgien l en salle d'opération k, $l \in \{1,\cdots,N_{chir}\}, k \in \{1,2\}$;

t_{kl} : la durée opératoire de la tâche composée O_{kl}, elle est le somme des durées opératoires des interventions composantes de cette tâche composée ;

Π_k : l'ensemble des tâches composées le jour considéré en salle d'opération k, $\Pi_k = \{O_{k1}, O_{k2}, \cdots, O_{kN_{chir}}\}, k \in \{1,2\}$;

Π_{1A} : l'ensemble des tâches composées en salle d'endoscopie (salle 1) dont la durée opératoire est inférieure ou égale à celle de la tâche composée correspondante (c'est-à-dire la tâche composée opérée par le même chirurgien) en salle de gastroscopie (salle 2), $\Pi_{1A} = \{O_{1i} \in \Pi_1 \mid t_{1i} \le t_{2i}\}$;

Π_{2A} : l'ensemble des tâches composées en salle de gastroscopie (salle 2) dont la durée opératoire est supérieure ou égale à celle de la tâche composée correspondante (c'est-à-dire la tâche composée opérée par le même chirurgien) en salle d'endoscopie (salle 1), $\Pi_{2A} = \{O_{2i} \in \Pi_2 \mid t_{1i} \le t_{2i}\}$;

Π_{1B} : l'ensemble des tâches composées en salle d'endoscopie dont la durée opératoire est supérieure à celle correspondante en salle de gastroscopie, $\Pi_{1B} = \Pi_1 \setminus \Pi_{1A}$;

Π_{2B} : l'ensemble des tâches composées en salle de gastroscopie dont la durée opératoire est inférieure à celle correspondante en salle d'endoscopie, $\Pi_{2B} = \Pi_2 \setminus \Pi_{2A}$;

$\xi(Q)$: un programme opératoire composé aléatoirement d'un ensemble de tâches composées Q, $Q \subseteq \Pi_1$ ou Π_2.

L'algorithme « Gonzalez-Sahni »

Etape 0 : Nous regroupons toutes les interventions opérées par le chirurgien *l* dans une salle d'opération *k* en un bloc (tâche composée). Étant donné que N_{chir} chirurgiens sont disponibles et qu'ils peuvent opérer les interventions dans deux salles d'opération, il y a donc au maximum $2N_{chir}$ tâches composées. Au cas où un chirurgien n'effectue aucune opération dans une salle pour le jour considéré, une tâche composée sera malgré tout créée pour lui avec une durée opératoire de zéro. Donc, nous avons exactement $2N_{chir}$ tâches composées à ordonnancer.

L'ordonnancement à établir devra respecter plusieurs contraintes, à savoir :

- chaque salle ne peut traiter qu'un bloc à la fois ;

- un chirurgien ne peut opérer qu'une tâche composée à la fois et donc ne peut pas être présent dans plusieurs salles en même temps.

Etape 1 : génération d'un programme opératoire provisoire S_φ

a) Choisir deux tâches composées $O_{2l} \in \Pi_{2A}$ et $O_{1r} \in \Pi_{1B}$ vérifiant que $t_{2l} \geq \max\{t_{1j} \mid O_{1j} \in \Pi_{1A}\}$; $t_{1r} \geq \max\{t_{2j} \mid O_{2j} \in \Pi_{2B}\}$. Si un des ensembles de tâches composées Π_{2A} ou Π_{1B} est vide, nous mettons $\{O_{2l}\}$ vide ou $\{O_{1r}\}$ vide respectivement.

b) Ensuite, nous construisons un programme provisoire S_φ où les tâches composées dans chaque salle d'opération k sont opérées dans l'ordre $\varphi(\Pi_k)$: $\{O_{kl}, \xi(\Pi_{kA} \setminus \{O_{kl}\}), \xi(\Pi_{kB} \setminus \{O_{kr}\}), O_{kr}\}, k \in \{1,2\}$.

Etape 2 : construction du programme opératoire final

Si le programme opératoire obtenu S_φ satisfait la condition suivante : $\sum_{O_{1i} \in \Pi_1} t_{1i} - t_{1l} \leq \sum_{O_{2i} \in \Pi_2} t_{2i} - t_{2r}$

1. Alors, nous construisons un programme opératoire S_{GS} dont l'ordre des tâches composées en salle d'endoscopie devient $(\xi(\Pi_{1A} \setminus \{O_{1l}\}), \xi(\Pi_{1B} \setminus \{O_{1r}\}), O_{1r}, O_{1l})$ et l'ordre des tâches composées en salle de gastroscopie reprend l'ordre $\varphi(\Pi_2) = \{O_{2l}, \xi(\Pi_{2A} \setminus \{O_{2l}\}), \xi(\Pi_{2B} \setminus \{O_{2r}\}), O_{2r}\}$
   Dans ce cas-ci, la tâche composée O_{1l} commencera à être opérée en salle d'endoscopie au moment $\max\{\sum_{O_{1i} \in \Pi_1 \setminus \{O_{1l}\}} t_{1i}, t_{2l}\}$.

2. Sinon, nous construisons un programme opératoire S_{GS} dont l'ordre des tâches composées en salle d'endoscopie reprend l'ordre $\varphi(\Pi_1) = \{O_{1l}, \xi(\Pi_{1A} \setminus \{O_{1l}\}), \xi(\Pi_{1B} \setminus \{O_{1r}\}), O_{1r}\}$ et l'ordre des tâches composées en salle de gastroscopie devient $(O_{2r}, O_{2l}, \xi(\Pi_{2A} \setminus \{O_{2l}\}), \xi(\Pi_{2B} \setminus \{O_{2r}\}))$.

Dans ce cas-ci, la tâche composée O_{1r} commencera à être opérée en salle d'endoscopie au moment $\max\{\sum_{O_{1i} \in \Pi_1 \setminus \{O_{1r}\}} t_{1i}, t_{2r}\}$.

4. Résultats expérimentaux

4.1 Données

Pour réaliser nos expérimentations, nous nous sommes basé sur le fonctionnement du service d'endoscopie de l'hôpital de la Croix-Rousse qui contient deux salles d'opération : une salle d'endoscopie et une salle de gastroscopie ($N_{salle} = 2$).

Nous avons effectué une planification sur une semaine ($N_{jour} = 5$).

La durée d'ouverture normale de la salle d'endoscopie est de 4 heures quelque soit le jour ($TOR_1^d = 240$ minutes) tandis que celle de la salle de gastroscopie est de 7 heures ($TOR_2^d = 420$ minutes).

Les durées d'ouverture supplémentaire maximale autorisée pour les deux salles d'opération sont de deux heures ($TS_k^d = 120$ minutes), $d = 1,\cdots,N_{jour}, k = 1,\cdots,N_{salle}$.

Pour la salle d'endoscopie, trois sortes d'interventions vont y être opérées : la coloscopie, la gastroscopie et la coloscopie + gastroscopie.

Pour la salle de gastroscopie, il n'y a qu'une sorte d'intervention qui y est opérée : la gastroscopie.

Il y a au total 8 chirurgiens disponibles pour chaque jour (N_{chir} = 8). Chacun d'entre eux peut effectuer au maximum 11 gestes par jour, $NG_l = 11, l = 1,\cdots,N_{chir}$.

Les dates limites pour les interventions ont été générées aléatoirement en utilisant une loi uniforme U[1,10].

Étant donné que l'efficacité varie suivant le chirurgien, nous devons générer des durées opératoires qui dépendent non seulement du type d'intervention mais aussi du chirurgien qui opère. Le Tableau 36 indique, pour la salle d'endoscopie, la répartition des interventions entre les différents chirurgiens et le type d'intervention. Etant donné que Combes [2004] a montré que c'est une loi de type Pearson III qui s'ajuste le mieux aux données opératoires, nous génèrerons nos données en suivant cette loi. Les paramètres utilisés se trouvent dans le Tableau 37.

Chirurgien / Sorte d'intervention	1	2	3	4	5	6	7	8	Total
Coloscopie	5,73%	18,14%	7,64%	5,09%	5,41%	6,05%	6,05%	4,77%	58,86%
Gastroscopie	1,27%	0,63%	0,95%	1,27%	1,90%	1,58%	0,32%	1,58%	9,49%
Coloscopie + Gastroscopie	3,48%	5,06%	1,58%	4,43%	5,38%	6,33%	4,11%	1,27%	31,65%
total	10,47%	23,83%	10,17%	10,79%	12,69%	13,96%	10,48%	7,62%	100,00%

Tableau 36 : Distribution globale des interventions en salle d'endoscopie par type d'intervention et par chirurgien

Sorte Interv. / Chirurgien	Gastroscopie		Coloscopie		Coloscopie + Gastroscopie	
	Alpha	Bêta	Alpha	Bêta	Alpha	Bêta
1	18,8534	0,0005	3,3547	0,0072	15,9345	0,0014
2	4,5000	0,0023	4,1999	0,0063	15,8731	0,0022
3	56,3333	0,0003	5,2695	0,0051	70,0215	0,0006
4	333,0625	0,0000	4,0066	0,0048	6,8079	0,0042
5	22,4490	0,0006	11,9713	0,0022	36,6214	0,0010
6	6,5636	0,0020	6,7581	0,0036	9,8851	0,0040
7	Dans ce cas-ci, toute la durée opératoire est égale à 15		5,1963	0,0067	21,3906	0,0020
8	150,0000	0,0000	6,4440	0,0028	38,8204	0,0009

Tableau 37 : Paramètres (α , β) de la loi Pearson III utilisés pour générer les durées opératoires (exprimées en minutes) des interventions en salle d'endoscopie

Etant donné que nous n'avons pas eu assez de données concernant la salle de gastroscopie, nous avons généré nos durées opératoires par chirurgien en nous basant sur les données

relatives au même type d'intervention de la salle d'endoscopie et en y ajoutant[16] une durée générée aléatoirement par une loi uniforme entre 5 et 10. De plus, nous supposons que tous les chirurgiens effectuent tous le même nombre de gastroscopie dans la salle de gastroscopie (12,5 % des interventions chacun).

Nous posons qu'un même paramètre $\beta = 1{,}5$ représente le rapport entre le coût d'une heure supplémentaire et le coût d'une heure normale, quelque soit la salle.

Nous avons testé sept cas différents : 80 interventions dont 30 en salle 1 ; 80 interventions dont 40 en salle 1; 90 interventions dont 40 en salle 1 ; 100 interventions dont 40 en salle 1 ; 110 interventions dont 40 en salle 1 ; 120 interventions dont 40 en salle 1 ; 120 interventions dont 30 en salle 1.

Nous avons généré 20 exemples par cas et avons donc effectué 140 exemples.

4.2 Analyse des résultats

A. Résultats Numériques pour l'étape de planification des interventions

Nous avons fait une comparaison entre la solution approchée du problème de planification trouvée au moyen de la procédure PHBGC (prenant en compte les contraintes d'intégrité) et la borne inférieure (avec relaxation des contraintes d'intégrité) trouvée par la procédure GC.

Tous les résultats numériques importants sont montrés dans le Tableau 38 où nous reprenons les paramètres présentés dans la partie V et ajoutons un nouveau paramètre N_{interv} (N_1 :N_2) représentant le nombre d'interventions où N_1 représente le nombre des interventions qui seront opérées en salle d'endoscopie et N_2 représente le nombre des interventions qui seront opérées en salle de gastroscopie.

Pour 131 cas parmi les 140, une solution réalisable a pu être obtenue et parmi ces 131 cas, 17 ont une solution en nombres entiers c'est-à-dire qu'ils ont pu être résolus directement par la procédure GC.

Pour les 114 exemples (131-17 = 114 exemples) qui ont été résolus par la procédure PHBGC, les résultats numériques sont montrés dans le Tableau 38 :

$N_{interv}(N_1: N_2)$	LP	H	Min(H-LP) /H (en %)	Moyenne (H-LP)/H (en %)	NColGenLP	NColGenH	CPULP (en secondes)	CPUH (en secondes)
80(30:50)	864,75	867,75	0	0,35	112,50	559,75	2,98	8,75
80(40:40)	1128,59	1130,03	0	0,13	108,06	569,00	2,72	9,23
90(40:50)	856,21	857,76	0	0,18	115,26	597,79	3,27	10,62
100(40:60)	606,5	609,53	0	0,49	129,89	653,94	4,31	13,06
110(40:70)	384,05	386,48	0	0,59	142	688,15	5,21	15,35
120(40:80)	140,35	141,28	0	0,61	153	778,7	6,37	19,53
120(30:90)	128,82	131,53	0	2,15	191,53	1326,18	8,54	49,62

Tableau 38 : Résultats numériques obtenus par la procédure PHBGC

Nous voyons que dans tous les cas un taux minimum de distance de 0 a été obtenu, que tous les taux de distance moyens sont inférieurs à 3% et de plus, la plupart d'entre eux sont inférieurs à 1%, ce qui signifie que notre solution approchée constitue pratiquement

[16] Pour un même type d'intervention, la durée opératoire est plus longue dans la salle de gastroscopie que dans la salle d'endoscopie du fait que l'opération est plus lente lorsqu'il n'y a pas d'anesthésie.

l'optimum pour ce modèle de planification. En ce qui concerne le temps de calcul, bien qu'il augmente avec la taille de problème, notre procédure arrive toujours résoudre le problème endéans la minute.

B. Résultats numériques par la combinaison de la procédure PHBGC et l'algorithme Gonzalez-Sahni

Basée sur la solution obtenue par la version 1 de la procédure PHBGC, nous avons utilisé l'algorithme de Gonzalez-Sahni afin construire un programme opératoire réalisable pour le service d'endoscopie pour une semaine. Nous avons comparé la solution obtenue par notre méthode proposée, c'est-à-dire la combinaison de la procédure PHBGC et l'algorithme Gonzalez-Sahni expliqué à la section 3.2 de cette partie, avec celle obtenue par la procédure « Best Fit Descending » proposée par Dexter et al. [1999b]. Etant donné que la procédure « Best Fit Descending » ne prend pas en compte la contrainte du chirurgien, il est donc possible que deux interventions doivent être opérées au même moment par le même chirurgien. Dans ce cas-là, nous décidons que le chirurgien va opérer en premier lieu l'intervention dont la durée opératoire est la plus petite et l'autre intervention devra attendre dans sa salle d'opération jusqu'à ce que le chirurgien se libère.

Concernant les résultats numériques obtenus, nous constatons que les interventions, dont la date limite était inférieure ou égale à la période de planification (5 jours), ont toutes été assignées aux salles d'opération à cent pourcents et ce quelque soit la méthode utilisée. Cependant, ce n'est pas le cas pour les interventions dont la date limite est supérieure à la période de planification.

Les Tableaux 39 et 40 nous montrent les résultats de la comparaison entre la méthode que nous avons proposée et la procédure « Best Fit Descending ». Tous les résultats numériques fournis dans les Tableau 39 et Tableau 40 sont les résultats moyens des 20 exemples pour chaque cas de N_{interv}. La comparaison entre les méthodes a été effectuée selon les indicateurs suivants :

PNb1S : le rapport du nombre des interventions dont la date limite est supérieure à une semaine au nombre total des interventions à assigner pour la salle d'endoscopie (salle 1) ;

PNb2S : le rapport du nombre des interventions dont la date limite est supérieure à une semaine au nombre total des interventions à assigner pour la salle de gastroscopie (salle 2) ;

PNbAsgn1SGS : le rapport du nombre des interventions dont la date limite est supérieure à une semaine assignées pour le programme opératoire obtenu par la combinaison de la procédure PHBGC et l'algorithme Gonzalez-Sahni au nombre total des interventions dont la date limite est supérieur à une semaine pour la salle 1. Dans la suite, cette combinaison « procédure PHBGC et l'algorithme Gonzalez-Sahni » s'appellera le programme opératoire GS;

PNbAsgn1SBFD : le rapport du nombre des interventions dont la date limite est supérieure à une semaine assignées pour le programme opératoire obtenu par la procédure « Best Fit Descending » au nombre total des interventions dont la date limite est supérieure à une semaine à assigner pour la salle 1. Dans la partie suivante, ce programme

opératoire s'appelle le programme opératoire BFD;

PNbAsgn2SGS : le rapport du nombre des interventions dont la date limite est supérieure à une semaine assignées pour le programme GS au nombre total des interventions dont la date limite est supérieure à une semaine à assigner pour la salle 2 ;

PNbAsgn2SBFD : le rapport du nombre des interventions dont la date limite est supérieure à une semaine assignées pour le programme opératoire BFD au nombre total des interventions dont la date limite est supérieure à une semaine à assigner pour la salle 2 ;

TORAsgnGS : la durée totale des salles d'opération qui est affectée dans cette semaine pour le programme opératoire GS (en minutes);

TORAsgnBFD : la durée totale des salles d'opération qui est affectée dans cette semaine pour le programme opératoire BFD ;

PTAsgnGS : le rapport de la durée opératoire totale des interventions assignées par le programme opératoire GS à la durée totale des interventions à assigner ;

PTASgnBFD : le rapport de la durée opératoire totale des interventions assignées par le programme opératoire BFD à la durée totale des interventions à assigner ;

ORUGS : l'utilisation moyenne des salles d'opération par le programme opératoire GS ;

ORUBFD : l'utilisation moyenne des salles d'opération par le programme opératoire BFD.

$N_{interv}(N_1:N_2)$	PNb1SGS (en %)	PNb2SGS (en %)	PNbAsgn1SGS (en %)	PNbAsgn1SBFD (en %)	PNbAsgn2SGS (en %)	PNbAsgn2SBFD (en %)
80(30:50)	48	45	96,65	95,68	100,00	100,00
80(40:40)	45	45	56,34	44,99	100,00	100,00
90(40:50)	44	47	54,18	48,53	100,00	100,00
100(40:60)	44	46	57,26	48,68	100,00	100,00
110(40:70)	45	45	59,40	48,63	100,00	100,00
120(40:80)	43	45	54,17	46,32	100,00	100,00
120(30:90)	46	44	100,00	100,00	92,80	85,63

Tableau 39 : Premiers résultats de la comparaison entre les programmes opératoires GS et BFD

$N_{interv}(N_1:N_2)$	TORAsgnGS	TORAsgnBFD	PTAsgnGS (en %)	PTAsgnBFD (en %)	ORUGS (en %)	ORUBFD (en %)
80(30:50)	2314,20	2412,35	99,32	99,01	92,10	90,96
80(40:40)	2201,59	2259,59	85,98	85,20	92,33	94,32
90(40:50)	2451,40	2528,40	88,17	87,43	94,63	97,16
100(40:60)	2700,50	2802,90	88,92	88,15	95,14	98,17
110(40:70)	2931,70	3018,25	89,95	89,18	92,33	94,68
120(40:80)	3159,35	3352,75	90,63	89,90	96,65	101,95
120(30:90)	3176,63	3337,11	97,93	97,13	95,96	100,53

Tableau 40 : Résultats complémentaires de la comparaison entre les programmes opératoires GS et BFD

En analysant les résultats contenus dans le Tableau 39 et le Tableau 40, nous remarquons que :

- Le taux d'affectation du nombre des interventions, dont la date limite est supérieure à une semaine, du programme opératoire GS est toujours plus élevé que l'autre lorsque le nombre d'interventions est grand (montré par le Tableau 39) ;
- Il n'y a pas une grande différence entre le taux d'affectation des durées des interventions, dont la date limite supérieure à une semaine, du programme opératoire GS et celui du programme opératoire BFD. En fait, le taux d'affectation des durées des interventions, dont la date limite est supérieure à une semaine, du programme opératoire GS est légèrement plus élevé que l'autre (montré par le Tableau 40) ;
- Tous les programmes opératoires GSs ont besoin d'une durée totale moindre que les programmes opératoires BFDs (montré par le Tableau 40) ;
- L'utilisation moyenne des salles d'opération du programme opératoire BFD est généralement plus grande que celle du programme opératoire GS pour le même nombre d'interventions à assigner (montré par le Tableau 38). En combinant ce résultat avec le précédent, nous pouvons constater que le programme opératoire GS est plus « serré » que le programme opératoire BFD et donc qu'il y a moins de temps mort entre les interventions. Cela signifie que le programme opératoire est d'une meilleure qualité que l'autre.

5. Conclusions

Nous avons construit un programme opératoire pour un service d'endoscopie, constitué de deux salles d'opération, en deux étapes : premièrement un modèle hebdomadaire de planification et ensuite un modèle journalier d'ordonnancement. Le modèle de planification est résolu par une procédure heuristique basée sur la génération de colonnes. Le modèle d'ordonnancement est résolu par la méthode de technologie de groupe et l'algorithme de Gonzalez-Sahni.

Les résultats expérimentaux nous montrent que le programme opératoire obtenu est efficace par rapport à celui obtenu par la procédure « Best Fit Descending ».

CONCLUSIONS ET PERSPECTIVES

Un bloc opératoire, comprenant salles d'opération et salle de réveil (SSPI), est un secteur dont la gestion est particulièrement risquée vu les coûts qu'il engendre. Pour en améliorer l'efficacité en termes de coûts d'opération, nous organisons les interventions suivant un planning opératoire satisfaisant les contraintes de disponibilité des salles et les chirurgiens. Ce problème est aussi complexe qu'important.

Nous nous focalisons donc sur la détermination d'un programme opératoire réalisable et efficace. Ce programme se compose de deux étapes : la planification hebdomadaire des interventions et l'ordonnancement journalier du bloc opératoire.

Dans l'étape de planification, nous avons considéré deux modèles généraux, l'un concernant la stratégie du « block scheduling » et l'autre la stratégie de l'« open scheduling ». Ces deux modèles sont les plus fréquemment utilisés dans le milieu hospitalier. La procédure heuristique PHBGC et une méthode exacte Branch-and-Price, basée sur la génération de colonnes, ont été proposées pour résoudre les problèmes de la planification des interventions.

Dans le cadre de la seconde étape, la phase d'ordonnancement, deux modèles liés aux modèles de planification de la première étape ont été proposés. Ces deux modèles d'ordonnancement sont traités comme des variantes du modèle « flow shop » hybride à deux étages, et sont résolus à l'aide d'un algorithme génétique hybride.

Les expériences numériques menées ont pu montrer que les méthodes utilisées permettent la résolution de tels problèmes et offrent en outre des solutions de qualité.

Nous avons également appliqué la procédure heuristique à la résolution du problème de planification du service d'endoscopie de l'Hôpital de la Croix-Rousse dont les distributions des durées opératoires suivent une loi Pearson III. Étant donné que le service d'endoscopie ne contient que 2 salles d'opération, nous avons utilisé une méthode spécifique pour ce cas, pour résoudre le problème d'ordonnancement. Les résultats obtenus sont très bons.

Voici quelques pistes pouvant faire l'objet de recherches futures :

- Prendre en compte des temps de changement entre deux interventions, notamment à savoir les temps nécessaires au remplacement du matériel et à la préparation des instruments ;
- Nous supposons ici que toutes les ressources humaines et matérielles nécessaires, sauf les chirurgiens, les salles d'opération et les lits de réveil, sont toujours disponibles. Or ceci n'est pas toujours les cas dans la vie quotidienne d'un hôpital ;
- Les cas d'urgence ne sont pas traités dans ce travail, et pourraient être incorporés au programme opératoire ;
- Afin d'assurer la cohérence entre les décisions prises au niveau planification (niveau tactique) et l'ordonnancement (niveau opérationnel), il faut trouver un mécanisme qui modifie le modèle de planification en fonction des résultats de l'ordonnancement.

Ce travail pourrait servir de base au développement de modèles plus proches de la réalité afin d'assister au mieux le gestionnaire hospitalier.

BIBLIOGRAPHIE

[Abdul-Razaq *et al.*, 1990] Abdul-Razaq T.S., Potts C.N., Van Wassenhove L.N.: A survey of algorithms for the single machines total weighted tardiness scheduling problem, Discrete Applied Mathematics, 26(2-3), 235-253, 1990.

[Aghezzaf et Artiba, 1998] Aghezzaf E.H., Artiba A.: Aggregate Planning in Hybrid flowshop, International journal of production research, 36(9), 2463-2477, 1998.

[Allahverdi *et al.*, 1999] Allahverdi A., Gupta J., Aldowaisan T.: A review of scheduling research involving setup considerations, Omega, International Journal of Management Science, 27, 219-239, 1999.

[Allaoui et Artiba, 2006] Allaoui H., Artiba A.: Scheduling two-stage hybrid flow shop with availability constraints, Computer & Operation Research, 33(5), 1399-1419, 2006.

[Amor *et al.*, 2005] Amor B., Abdelwaheb R., Taïcir L., Hichem K.: Algorithmes génétique pour le problème d'ordonnancement de type flow shop hybride. 6ème congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'05), Tours, France, 14-16 Février, 2005.

[Arthanari et Ramamurthy, 1971] Arthanari T.S., Ramamurthy K.G.: An extension of two machines sequencing problem, Operation Research, 8, 10-22, 1971.

[Azizoglu *et al.*, 2001] Azizoglu M., Cakmak E., Kondakci S.: A flexible flowshop problem with total flow time minimization, European Journal of Operational Research, 132, 528-538, 2001.

[Baker, 1974] Baker K.R.: Introduction to sequencing and scheduling, John Wiley & Sons, 305p, 1974.

[Baker, 1987] Baker J.E.: Reducing bias and inefficiency in the selection algorithm. Proceeding of 2nd International conference on genetic algorithms, édité par J.J. Grefenstette, 14-21, 1987.

[Barnhart *et al.*, 1998] Barnhart C., Johnson E.L., Nemhauser G.L., Savelsbergh M.W.P., Vance, P.H.: Branch-and-Price: Column generation for solving huge integer programs, Operations Research, 46, 316-329, 1998.

[Bertel et Billaut, 2004] Bertel S., Billaut J.C.: A genetic algorithm for an industrial multiprocessor flow shop scheduling problem with recirculation, European Journal of Operational Research, 159, 651-662, 2004.

[Blake *et al.*, 1995] Blake J.T., Carter M., O'Brien-Pallas L., McGillis-Hall L.: A surgical process management tool, Proceeding of the 8th World Congress on Medical Informatics MEDINFO 95, Greenes R., Vancouver B.C.(Ed.): International Medical Informatics Association, 1995.

[Blake et Donald, 2002] Blake J.T., Donald J.: Mount Sinai hospital uses integer programming to allocate operating room time, Interfaces, 32(2), 63-73, 2002.

[Blake et Carter, 1997a] Blake J.T., et Carter M.W.: Surgical process scheduling: a structured review, Journal of Health systems (I.I.E), 5(3), 17-30, 1997.

[Blake et Carter, 1997b] Blake J.T., et Carter M.W.: Surgical process management: a conceptual framework, Surgical Services Management, 3(9), 31-37, 1997.

[Blazewicz *et al.*, 1983] Blazewicz J., Lenstra J.K., Rinnooy Kan A.H.G.: Scheduling subject to resource constraints, Discrete Applied Mathematics, 5, 11-24, 1983.

[Blazewicz *et al.*, 1986] Blazewicz J., Cellary W., Slovinski R., Weglarz J.: Scheduling under resource constraints – deterministic model, Annals of Operation Research, Baltzer Science Publisher, 7, 1986.

[Blazewicz *et al.*, 1996] Blazewicz J., Ecker K.H., Schmidt G., Weglarz J.: Scheduling computer and manufacturing Process, Springer, 491p, 1996.

[Botta-Genoulaz, 2000] Botta-Genoulaz V.: Hybrid flow shop scheduling with precedence constraints and time lags to minimize maximum lateness, International Journal of Production Economics, 101-111, 2000.

[Brah et Hunsucker, 1991] Brah S.A., Hunsucker J.L.: Branch and Bound algorithm for the flow shop with multiple processors, European Journal of Operation Research, 51, 88-99, 1991.

[Brah et Loo, 1999] Brah S.A., Loo L.L.: Heuristics for Scheduling in a flow shop with multiple processors, European Journal of Operation Research, 113, 113-122 1999.

[Breit *et al.*, 2001] Breit J., Schmidt G., Strusevich V.A.: Two-machine open shop scheduling with an availability constraint, Operations Research Letters, 29, 65-77, 2001.

[Brockmann et Dangelmaier, 1998] Brockmann K., Dangelmaier W.: A parallel Branch and Bound algorithm for makespans optimal sequencing in flow shops with parallel machines, Proceeding of 2nd IMACs, International Multiconference on Computational Engineering in Systems Applications (CESA), 3, 431-436, 1998.

[Carter et Lapierre, 2001] Carter M.W., Lapierre S.D.: Scheduling Emergency Room physicians, Health care Management Science, 4, 347-360, 2001.

[Chaabane, 2004] Chaabane S.: Gestion prédictive des blocs opératoires, Thèse, L'institut National des Sciences Appliquées de Lyon, 172p, 2004.

[Chen, 1995] Chen B.: Analysis of Classes of Heuristics for Scheduling a two-stage Flow shop with parallel Machines at one stage, Journal of Operational Research Society, 46, 234-244, 1995.

[Chen et Powell, 1999a] Chen Z., Powell W.B.: Solving parallel machine scheduling problems by column generation, Journal of Computing, 11(1), 78-94, 1999.

[Chen et Powell, 1999b] Chen, Z., Powell W.B.: A column generation based decomposition algorithm for a parallel machine just-in-time scheduling problem, European Journal of Operation Research, 116, 220-232, 1999.

[Cheng et Gupta, 1989] Cheng T.C.E. and Gupta M.C.: Survey of scheduling research involving due date determination decisions**,** European Journal of Operational Research, 38(2), 2, 156-166, 1989.

[Clergue 1999] Clergue, F.: Gestion du bloc opératoire : Pourquoi une telle préoccupation ?, Informations cliniques en Anesthésie-Réanimation, 93-95, 1999.

[Combes *et al.*, 2004] Combes C., Dussauchoy A., Chaabane S., Smolski N., Viale J.P., Souquet J.C.: Démarche méthodologique d'analyse des données pour la planification des blocs opératoires : une application à un service d'endoscopie, Actes GISEH'04, Mons, Belgique, 2004.

[Cuviller, 1997a] Cuviller P.: Bases d'organisation fonctionnelle d'un bloc opératoire : la charte opératoire, Objectifs Soins, 57: IV-XVI, 1997.

[Cuviller, 1997b] Cuviller P.: Bloc opératoire : procédures, formation et démarche qualité. Objectifs Soins, 57: I-XI, 1997.

[Dantzig et Wolfe, 1960] Dantzig G.B., Wolfe P.: Decomposition principle for linear programs, Operation Research, 8, 101-111, 1960.

[Darwin, 1859] Darwin C.R.: On the origin of species by means of natural selection or the preservation of favoured races in the struggle for life. Murray, London, 1859.

[Delesie, 1998] Delesie L.: Bridging the gap between clinicians and health managers, European Journal of Operational Research, 105, 248-256, 1998.

[De Troyer et Krzeslo, 2004] De Troyer M., Krzeslo E.: Assurance maladie, soins de santé et sécurité sociale : trois éléments indissociables, Chronique internationale de l'IRES, 91, 113-122, novembre 2004.

[Dexter, 2000] Dexter F.: A strategy to decide whether to move the last case of the day in an operating room to another empty operating room to decrease overtime labour cost, Anesthesia & Analgesia, 91, 925-928, 2000.

[Dexter *et al.*, 1999a] Dexter F., Macario A., Qian F., Traub R.: Forecasting surgical groups' total hours of elective cases for allocation of block time : application of time series analysis to operating room management. Anesthesiology, 91(5), 1501-1509, 1999.

[Dexter *et al.*, 1999b] Dexter F., Macario A. and Traub R.D.: « Which algorithm for scheduling add-on elective cases maximizes operating room utilization? », Anesthesiology, 91(5), 1491-1500, 1999.

[Dexter *et al.*, 1999c] Dexter F., Macario A., Traub R.D., Hopwood M., Lubarsky D.A.: An operating room scheduling strategy to maximize the use of operating room bloc time: computer simulation of patient scheduling and survey of patients' preferences for surgical waiting time, Anesthesia & Analgesia, 89, 7-20, 1999.

[Dexter *et al.*, 1999d] Dexter F., Macario A., Lubarsky D., Burns D.: Statistical method to evaluate management strategies to decrease variability in operating room utilization: application of linear statistical modelling and Monte Carlo simulation to operating room management, Anesthesiology, 91(1), 262-274, 1999.

[Dexter *et al.*, 1999e] Dexter F., Macario A., Traub R.D.: Optimal sequencing of urgent surgical cases, Journal of clinical monitoring and computing, 15, 153-162, 1999.

[Dexter *et al.*, 2000a] Dexter F., Macario A., Traub R.D.: Enterprise-wide patient scheduling information systems to coordinate surgical clinic and operating room scheduling can impair operating room efficiency, Anesthesia & Analgesia, 91, 617-626, 2000.

[Dexter *et al.*, 2000b] Dexter F., Macario A., O'Neill L.: Scheduling surgical cases into overflow bloc time – computer simulation of the effects of scheduling strategies on operating room labour costs, Anesthesia & Analgesia, 90, 980-988, 2000.

[Dexter *et al.*, 2001a] Dexter F., Macario A., Lubarsky D.A.: Impact on revenue of incereasing patient volume et surgical suites with relatively high operating room utilization, Anesthesia & Analgesia, 92, 1215-1221, 2001.

[Dexter *et al.*, 2001b] Dexter F., Epstein R.H., and Marsh H.M.: A statistical analysis of weekday operating room anesthesia group staffing costs at nine independently managed surgical suites, Anesthesia & Analgesia, 92, 1493-1498, 2001.

[Dexter *et al.*, 2002] Dexter F., Blake J.T., Penning D.H., Solan B., Chung P., Lubarsky D.A.: Use of Linear programming to estimate impact of changes in a hospital's operating room time allocation on perioperative variable costs, Anesthesiology, 96(3), 718-724, 2002.

[Dexter *et al.*, 2005] Dexter F, Epstein RH, de Matta R, Marcon E.: Strategies to reduce delays in admission into a postanesthesia care unit from operating rooms. Journal of PeriAnesthesia Nursing, 20(2), 92-102, April 2005.

[Dexter et Macario, 1999] Dexter, F., Macario, A.: Decrease in case duration required to complet an additional case during regularly scheduled hours in an operating room suite: a computer simulation study, Anesthesia & Analgesia, 88, 72-76, 1999.

[Dexter et Macario, 2002] Dexter F., Macario A.: Changing allocations of operating room time from a system based on historical utilization to one where the aim is to schedule as many surgical cases as possible, Anesthesia & Analgesia, 94, 1272-1279, 2002.

[Dexter et Tinker, 1995] Dexter F., Tinker J.: Analysis of strategies to decrease postanesthesia care unit costs, Aensthesiology, 82(1), 94-101, 1995.

[Dexter et Traub, 2000] Dexter F., Traub R.D.: Determining staffing requirements for a second shift of anesthetists by graphical analysis of data from operating room information systems, American Association of Nurse Anesthetists, 68, 31-36, 2000.

[Dexter et Traub, 2002] Dexter F., Traub R.D.: How to schedule elective surgical cases into specific operating rooms to maximize the efficiency of use of operating room time, Anesthesia & Analgesia, 94, 933-942, 2002.

[Ding et Kittichartphayak, 1994] Ding F.Y., Kittichartphayak D.: Heuristics for scheduling flexible flow lines, Computers & Industrial Engineering, 26, 27-34, 1994.

[Donham *et al.*, 1996] Donham R.T., Mazzei W., Jones R.: Procedural times glossary of the AACD, American Journal of Anesthesiology, 23(5), 4-12, 1996.

[Dorigo *et al.*, 1991] Dorigo M., Maniewwo V., Colorni A.: Positive feedback as a search strategy (Tech. Rep. 91-016) Milan, Italy: Politecnico di Milano, Dipartimento di Elettronica., 1991.

[DREES, 1999] DREES : Les établissements de santé en 1999, Collection études et statistiques, 36 p, 1999.

[Dréo et al., 2003] Dréo J., Péreowski A., Siarry P., Taillard E. : Métaheuristiques pour l'optimisation difficile, Editions Eyrolles, 356 p, 2003.

[Dutheil, 2005] Dutheil J. F.: Réponse française au questionnaire du comité de la protection sociale, Services d'intérêt général sociaux, Paris, le 24 janvier 2005.

[Duvivier, 2000] Duvivier D.: Etude de l'hybridation des méta-heuristiques, application à un problème d'ordonnancement de type jobshop, thèse, Université du Littoral Côte d'Opale, 289p, 2000.

[Eberhart et Kennedy, 1995] Eberhart R.C., Kennedy J.: A new optimizer using particle swarm theory. Proceedings of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, Japan, Piscataway, NJ: IEEE Service Center, 39-43, 1995.

[Engin et Döyen, 2004] Engin O., Döyen A.: A new approach to solve hybrid flow shop scheduling problems by artificial immune system, Future Generation Computer Systems, 20(6), 1083-1095, 2004.

[Everett, 2002] Everett J.E.: A decision support simulation model for the management of an elective surgery waiting system, Health Care Management Science, 5, 89-95, 2002.

[Fei *et al.*, 2004] Fei H. Artiba A., Chu C.: A memetic algorithm for operating room scheduling, International Conference on Computational & Experimental Engineering and Sciences, Portugal, 2004.

[Fei *et al.*, 2005a] Fei H., Chu C., Artiba A., Meskens N.: Planification des interventions sur une semaine dans un bloc opératoire, 6ème congrès de la Société Française de Recherche Opérationnelle et d'Aide à la Décision (ROADEF'05), Billaut J.C., Esswein C. (Ed.), 359-374, 14-16 Février, France, 2005.

[Fei *et al.*, 2005b] Fei H., Chu C., Meskens N., Artiba A.: Solving surgical cases assignment problem by a branch-and-bound approach, Proceeding of IESM'05, 756-765, Marocco, 2005.

[Fischer, 1999] Fischer M.: Informatique et fonctionnement du bloc opératoire, Informations cliniques en Anesthésie-Réanimation, 109-112, 1999.

[Fogg 2001] Fogg D.M.: Warming cabinets; staffing levels; Staphylococcus aureus; gloving; gastrointestinal endoscopy; OR floors - operating room questions and answers, Association of Operating Room Nurse (AORN) Journal, 2001.

[Gabel *et al.*, 1999] Gabel R.A., Kulli J.C., Lee B.S., Spratt D.G., Ward D.S.: Operating room management. Butterworth-Heinemann, 224p, 1999.

[Gadenne, 2005] Gadenne N.: Programmation et évaluation d'algorithmes d'affectation des niveaux de vol. Rapport de Stage, Université de Technologie Compiègne, 76p, 2005.

[Galinier et Hertz, 2005] Galinier P., Hertz A.: A survey of local search methods for graph coloring, Computers and Operations Research, à venir, 2005.

[Gallivan, 1998] Gallivan S.: Evaluation of priority strategies for hospital admissions, conference presentation ORAHS, 152-161, 1998.

[Garey et Johnson, 1979] Garey M.R., Johnson D.S.: Computers and intractability: a guide to the theory of NP-completeness. W.H. Freeman and Company, New York, 338p, 1979.

[Gerchak *et al.*, 1996] Gerchak Y., Gupta D., Henig M.: Reservation planning for elective surgery under uncertain demand for emergency surgery, Management Science, 42(3), 321-334, 1996.

[Giard, 2003] Giard V. : Gestion de la production et des flux, 3ème édition, Collection Gestion, Série : Production et techniques quantitatives appliquées à la gestion, Economica, 1229p, 2003.

[Glover 1986] Glover F.: Future paths for integer programming and links to artificial intelligence, Computer and Operations Research, 13, 533-549, 1986.

[Glover 1990] Glover F.: Tabu search: A tutorial, Interfaces, 20, 74-94, 1990.

[Glover *et al.*, 1992] Glover F., Taillard E., de Werra D.: A user's guide to Tabu Search, Rapport technique n° ORWP 92101, Département de Mathématique, Ecole Polytechnique Fédéral de Lausanne, Switzerland, 1992.

[Glover et Laguna, 1992] Glover E., Laguna M.: Tabu Search. In: Modern Heuristics Techniques for Combinatorian Problems. University of Colorado at Boulder, Graduate School of Business and Administration, July, 1992.

[Goldberg, 1994] Goldberg D.E.: Algorithmes génétiques : Exploration, optimisation, et apprentissage automatique, Addison-Wesley France, 417p, 1994.

[Gonzalez et Sahni, 1976] Gonzalez T., Sahni S.: Open shop scheduling to minimize finish time, Journal of the Association for Computing Machinery, 23, 665-679, 1976.

[Gordon *et al.*, 1988] Gordon T., Lyles A.P.S., Fountain J.: Surgical unit time utilization review: Resource utilization and management implication, Journal of Medical System, 12, 169-179, 1988.

[Greene et Sadowski 1984] Greene T.J., Sadowski R.P.: A review of cellular manufacturing assumptions, advantages and design techniques, Journal of Operations Management, 4(2), 85-97, 1984.

[Graham *et al.* 1979] Graham R.L., Lawler E.L., Lenstra J.K., Rinnooy Kan A.H.G.(1979). Optimization and approximation in deterministic sequencing and scheduling: a survey, Anneal of Discrete Mathematics, 5, 287-326, 1979.

[Guinet *et al.*, 1996] Guinet A., Solomon M.M., Kedia P. K., Dussauchoy A.: A computational study of heuristics for two-stage flexible flowshops, International Journal of Production Research, 34(5),1399-1416,1996.

[Guinet et Chaabane, 2003] Guinet A., Chaabane S.: Operating theatre planning, International Journal of Production Economics 85, 69-81, 2003.

[Guinet et Solomon, 1996] Guinet A., Solomon M.M.: Scheduling hybrid flowshops to minimize maximum tardiness or maximum completion time, International Journal of Production Research, 34(6), 1643-1654, 1996.

[Guirchoun et Martineau, 2002] Guirchoun S., Martineau P.: Des problèmes à machines parallèles avec serveur aux flowshop hybride, 4ème congrès de la société Française de Recherche Opérationnelle, Paris, 154-155, 2002.

[Gupta, 1988] Gupta J.N.D.: Two-stage hybrid flowshop scheduling problem, Journal of Operation Research Society, 39, 359-364, 1988.

[Gupta *et al.*, 1997] Gupta J.N.D., Hariri A.M.A., Potts C.N.: Scheduling a two-stage hybrid flow shop with parallel machines at the first stage, Annals of Operations Research, 69, 171-191, 1997.

[Gupta *et al.*, 2002] Gupta J.N.D., Krüger K., Lauff V., Sotskov Y.N., Werner F.: Heuristics for hybrid flow shops with controllable processing times and assignable due dates, Computers & Operations Research, 29(10), 1417-1439, 2002.

[Gupta et Kyparisis, 1987] Gupta S.K., Kyparisis J.: Single machine scheduling research, OMEGA, International Journal of Management Science, 15, 207-227, 1987.

[Gupta et Tunc, 1991] Gupta J.N.D., Tunc E.A.: Schedules for a two-Stage hybrid flow shop with parallel machine at the second stage, International Journal of Production Research, 29(7), 1489-1502, 1991.

[Gupta et Tunc, 1994] Gupta J.N.D., Tunc E.A.: Scheduling a two-stage hybrid flow shop with separable setup and removal time, European Journal of Operation Research, 77, 415-428, 1994.

[Hamilton et Breslawki, 1994] Hamilton DM, Breslawski S.: Operating room scheduling. Factors to consider, Association of Operating Room Nurse (AORN) Journal, 59(3), 665-680, 1994.

[Hansen, 1986] Hansen P.: The steepest ascent mildest descent heuristic for combinatorial programming, Congress on Numerical Methods in Combinatorial Optimization. Capri, Italy, 1986.

[Haourai et M'Hallah, 1997] Haouari H., M'Hallah R.: Heuristic slgorithms for the two-Stage hybrid flowshop problem, Operations Research Letters, 21(1), 43-53, 1997.

[Hasegawa *et al.*, 2002] Hasegawa M., Ikeguchi T., Aihara K.: Solving large scale traveling salesman problems by chaotic neurodynamics, Neural Networks, 15(2), 271-283, 2002.

[Havill et Mao, 1998] Havill J.T., Mao W.: On-line algorithms for hybrid flow shop scheduling, Proceedings of the Fourth International Conference on Computer Science and Informatics , Research Triangle Park, North Carolina, 134--137, October 1998.

[HFMA, 2003] Health Financial Management Association: Achieving operating room efficiency through process integration, Healthcare Financial Management, 57(3), suppl 1-7 following 112, March, 2003.

[Hifi et Zissimopoulos, 1996] Hifi M., Zissimopoulos V.: A recursive exact algorithm for weighted two-dimensional cutting, European Journal of Operational Research, 91, 553-564, 1996.

[Hillier et Lieberman, 1980] Hillier F.S., Lieberman G.J.: Operations Research, 3rd ed., Holden-Day, San Francisco, 1980.

[Ho et Haugland, 2004] Ho S.C., Haugland D.: A tabu search heuristic for the vehicle routing problem with time windows and split deliveries, Computers & Operations Research, 31(12), 1947-1964, October 2004.

[Hoesel *et al.*, 1994] Hoesel S.V., Kuik R., SalomonLuk M., Wassenhove N.V.: The single-item discrete lot sizing and scheduling problem: Optimization by linear and dynamic programming, Discrete Applied Mathematics, 48(3), 289-303, 1994.

[Holland, 1962] Holland J.H.: Outline for a logical theory in adaptive systems, Journal of the Association for Computing Machinery, 3, 297-314, 1962.

[Hunsucker et Shah, 1994] Hunsucker J.L., Shah J.R.: Comparative performance analysis of priority rules in a constrained flow shop with multiple processors environment, European Journal of Operational Research, 72, 102-114, 1994.

[Jain et Meeran, 1999] Jain A.S., Meeran S.: Deterministic job shop scheduling: past, present, Future' European Journal of Operation Research, Elsevier Science, 113, 390-434, 1999.

[Janiak *et al.*, 2005] Janiak A., Kozan E., Lichtenstein M., Oguz C.: Metaheuristic approaches to the hybrid flow shop scheduling problem with a cost-related criterion. International Journal of Production Economics, à venir, 2005.

[Jebali, 2004] Jebali A.: Vers un outil d'aide à la planification et l'ordonnancement des ressources dans les services de soins. Thèse, Institut National Polytechnique de Grenoble, 216p, 2004.

[Jebali *et al.*, 2003] Jebali A., Hadj Alouane A.B., Ladet P.: Operating room scheduling, Proceeding of International Conference Industrial Engineering and Production Management, IEPM'03, Portugal, 206-217, 2003.

[Jebali *et al.*, 2006] Jebali A., Hadj Alouane A.B., Ladet P.: Operating room scheduling, International Journal of Productions Economics, 99, 52-62, 2006.

[Jin *et al.*, 2005] Jin Z., Yang Z., Ito T.: Metaheuristic algorithms for the multistage hybrid flow shop scheduling problem, International of Production Economics, à venir, 2005.

[Kellerer et Strusevich, 2003] Kellerer H., Strusevich V.A.: Scheduling parallel dedicated machines under a single non-shared resource, European Journal of Operational Research, 147, 345-364, 2003.

[Kellerer et Strusevich, 2004] Kellerer H., Strusevich V.A.: Scheduling problems for parallel dedicated machines under multiple ressource constraints, Discrete Applied Mathematics, 133, 45-68, 2004.

[Kennedy, 1992] Kennedy M.: Bin-packing, knapsack and chance constrained application to scheduling operating rooms, Thèse, Resselaer Polytechnic Institute, Troy, NY., 1992.

[Kharraja, 2003] Kharraja S. : Outils d'aide à la planification et l'ordonnancement des plateaux-médico-techniques, Génie Industriel, Saint-Etienne : Université Jean-Monnet, 153P, 2003.

[Kharraja *et al.*, 2002] Kharraja S., Chaabane S., Marcon E. : Evaluation de performances pour deux stratégies de programmation opératoire de bloc, Proceeding de la 2ème conférence Internationale Francophone d'Automatique, France, 2004.

[Kharraja et Marcon, 2003] Kharraja S., Marcon E. : Vers une construction automatique du plan directeur d'allocation des plages horaires, Conférence francophone sur la Gestion et Ingénierie des Systèmes Hospitaliers GISEH'03, 54-6 1, France, 2003.

[Kirkpatrick *et al.*, 1983] Kirkpatrick S., Gelatt C.D., Vecchi M.P.: Optimization by simulate annealing, Science, 220, 671-680, 1983.

[Kis et Pesch, 2005] Kis T., Pesch E.: A review of exact solution methods for the non-preemptive multiprocessor flowshop problem, European Journal of Operational Research, 164, 592-608, 2005.

[Kontak-Forsyth et Grant, 1995] Kontak-Forsyth M., Grant A.E.: OR booking policy: development and implementation, Canadian Nursing Journal, 13(1), 1995.

[Kreutz *et al.*, 2000] Kreutz M., Hanke D., Gehlen S.: Solving extended hybrid-flow-shop problems using active schedule generation and genetic algorithms, Parallel Problem Solving from Nature 6th International Conference (PPSN 2000), 293-302, France, 2000.

[Kuo *et al.*, 2003] Kuo P., Schroeder R., Mahaffey S., Bollinger R.R.: Optimization of operating room allocation using linear programming technique, The American College of Surgeons, 197(6), 889-895, 2003.

[Kyparisis et Koulamas, 2006] Kyparisis G.L., Koulamas C.: Flexible flow shop scheduling with uniform parallel machines, European Journal of Operational Research, 168(3), 985-997,2006.

[Lagergren, 1998] Lagergren M.: What is the role and contribution of models to management and research in the health care services? a view from Europe, European Journal of Operational Research, 105, 257-266, 1998.

[Lapierre et al., 1999] Lapierre S.D., Batson C., McCaskey S.: Improving on-time performance in health care organizations: a case study, Health care Management Science, 2, 27-34, 1999.

[Lawer et al., 1993] Lawer E.L., Lenstra J.K., Rinnooy Kan A.H.G., Shmoys D.B.: Sequencing and scheduling: algorithms and complexity, in: Graves S.C., Rinnooy Kan A.H.G., Zipkin P.H. (Eds.), Handbooks in Operations Research and Management Science, 4, Logistics of Production and Inventory, North-Holland, Amsterdam, 455-522, 1993.

[LeBlanc et al., 1999] LeBlanc L.J., Shtub A., Anandalingam G.: Formulating and solving production planning problems, European Journal of Operational Research, 112, 54-80, 1999.

[Lebowitz, 2003] Lebowitz P.: Schedule the short procedure first to improve OR efficiency, Association of Operating Room Nurse (AORN) Journal, 78(4), 651-659, 2003.

[Lee et Vairaktarakis 1994] Lee C.Y., Vairaktarakis : Minimizing makespan in hybrid flowshops, Operations Research Letters, 16(3), 149-158, 1994.

[Lenstra et Rinnooy kan, 1980] Lenstra J.K., Rinnooy Kan A.H.G.: Computational complexity of discrete optimization, In: Lenstra J.K., Rinnooy Kan A.H.G, Van Emde Boas P.(Eds), Interface Between Computer Science and Operations Research, Processing of a symposium held at the Mathematisch Centrum, Amsterdam, Mathematisch Centrum, 64-85, 1980.

[Levecq et al., 2003] Levecq P, Meskens N., Artiba A.: Utilisation d'une approche Data Mining pour la spécification des durées en milieu hospitalier, Santé et systémique, 7 (1-2), 191-203, 2003.

[Lim, 1997] Lim J.M.: A genetic algorithm for a single hoist scheduling in the printed-circuit-board electroplating line, Computers and Industrial Engineering, 33, 789-792, 1997.

[Linn et Zhang, 1999] Linn R., Zhang W.: Hybrid flow shop scheduling: A survey, Computers & Industrial Engineering, 37(1-2),57-61, 1999.

[Logendran *et al.*, 2005] Logendran R., Carson S., Hanson E.: Group scheduling in flexible flow shops, International Journal of Production Economics, 96, 143-155, 2005.

[Lovejoy et Li, 2002] Lovejoy W., Li Y.: Hospital operating room capacity expansion, Management Science, 48(11), 1369-1387, 2002.

[Low, 2005] Low C.: Simulated annealing heuristic for flow shop scheduling problems with unrelated parallel machines, Computers & Operations Research, 32, 2013-2025, 2005.

[Lübbecke et Desrosiers, 2004] Lübbecke M.E., Desrosiers J.: Selected topics in column generation, http://citeseer.ist.psu.edu/551009.html, 2004.

[Macario et al., 1995] Macario A., Vitez T.S., Dund B., McDonale T.: Where are the cost in perioperative case ? Analysis of hospital costs and charges for inpatient care, Anesthesiology, 83(6), 1138-1144, 1995.

[Macario et Dexter, 1999] Macario A., Dexter F.: Estimating the duration of a case when the surgeon has not recently performed the procedure at the surgical suite, Anesthesia & Analgesia, 89, 1241-1245, 1999.

[Magerlein et Martin, 1978] Magerlein J., et Martin J. : Surgical demand scheduling : a review , Health Services Research, 31(4), 418-433, 1978, 1978.

[Malhorta 2001] Malhorta V.: Nuts and bolts of OR management, 52th Annual Refresher cours lectures, Clinical updates and basic science reviews, during the Annual Meeting of the American Society of Anesthesiologists, 13-17, 2001.

[Marcon *et al.*, 2001a] Marcon E., Kharraja S., Simonnet G.: The operating theatre scheduling : an approach centred on the follow-up of the risk of no realization of the planning, Proceeding of the Industrial Engineering and Production Management, Canada, 18-21, 2001.

[Marcon *et al.*, 2001b] Marcon E., Kharraja S., Simonnet G. : Minimization of the risk of no realization for the planning of the surgical interventions into the operating theatre, proceeding of 8th IEEE International Conference on Emerging Technologies and Factory Automation, France, 675-680, 2001.

[Marcon *et al.*, 2003] Marcon E., Kharraja S., Simonnet G.: The Operating theatre scheduling: an approach centered on the follow-up of the risk of no realization of the planning, Special issue of International Journal of Production Economics (IJPE), ed. ELSEVIER, 85(1), 83-90, 2003.

[Martello et Toth, 1990] Martello S., Toth P.: Knapsack problems: algorithms and computer implementation. New York: Wiley, 1990.

[Marty *et al.*, 2001] Marty J., Groupe OMEGA: Organisation des sites opératoires. Conférences d'actualisation, 203-224, 2001.

[Matile *et al.*, 1999] Matile G., Tettamanzi G.B.A., Tomassini M.: Evolutionary design of time-way charts for plating machines, Proceeding of CEC99 Congress on Evolutionary Computation, Etats-Unis, 1999.

[May *et al.*, 2000] May J.H., Strum D.P., Vargas L.G.: Fitting the LogNormal distribution to surgical procedure times, Decision Sciences, 31(1), 129-148, 2000.

[Mcleod et al., 2003] Mcleod H., Ham C., Kipping R.: Booking patients for hospital admissions : evaluation of a pilot programme for day cases, British Medical Journal (BMJ), 327, 1147-1151, 2003.

[Moursli et Pochet, 2000] Moursli O., Pochet Y.A.: Branch and Bound algorithm for the hybrid flow shop, International Journal of Production Economics, 64, 113-125, 2000.

[Murphy et Sigal, 1985] Murphy D.R., Sigal E.: Evaluating surgical block schedules using computer simulation, Winter Simulation Conference Proceeding, Gantz D., Blais G., Solomon S. (Eds.), 551-557, 1985.

[Nawaz *et al.*, 1983] Nawaz M., Enscore E., Ham I.: A heuristic algorithm for the m-machine, n-job flow shop sequence problem, OMEGA, 11, 91-95, 1983.

[Negenman, 2001] Negenman E.: Local search algorithms for the multiprocessor flow shop scheduling problem, European Journal of Operational Research, 128, 147-158, 2001.

[Néron *et al.*, 2001] Néron E., Baptiste P., Gupta J.N.D.: Solving hybrid flow shop problem using energetic reasoning and global operations, OMEGA, 29(6), 501-511, 2001.

[Nowicki et Smutnicki, 1996a] Nowicki E., Smutnicki C.: A fast taboo search algorithm for the job shop problem, Management Science, 42, 797-813, 1996.

[Nowicki et Smutnicki, 1996b] Nowicki E., Smutnicki C.: A fast tabu search algorithm for the permutation flow-shop problem. European Journal of Operational Research, 91, 160-175, 1996.

[Nowicki et Smutnicki, 1998] Nowicki E., Smutnicki C.: The flow shop with parallel machines: A Tabu Search approach, European Journal of Operational Research, 106, 2, 226-253, 1998.

[Ogulata et Erol, 2003] Ogulata S.N., Erol R.: A hierarchical multiple criteria mathematical programming approach for scheduling general surgery operations in large hospitals, Journal of Medical Systems, 27(3), 259-270, 2003.

[Oguz et al, 2003] Oguz C., Ercan M.F., Cheng T.C.E., Fung Y.F.: Heuristic algorithms for multiprocessor task scheduling in a two-stage hybrid flow-shop, European Journal of Operational Research, 149 (2), 390-403 2003.

[Oguez et al., 2004] Oguz C., Zinder Y., Do V.H., Janiak A., Lichtenstein M.: Hybrid flow-shop scheduling problems with multiprocessor task systems, European Journal of Operational Research, 152(1), 115-131, 2004.

[Opit *et al.*, 1991] Opit L.J., Collins R. E. C., Campbell G.: Use of operating theatres: the effects of case-mix and training in general surgery, Annals of the Royal College of Surgeons of England, 73, 389-393, 1990.

[Overdyk *et al.*, 1998] Overdyk F.J., Harvey S.C., Fishman R.L., Shippey F.: Successful strategies for improving operating room efficiency at academic institutions, Anesthesia & Analgesia, 86, 896-906, 1998.

[Ozkarahan, 1995] Ozkarahan I.: Allocation of surgical procedures to operating rooms, Journal of Medical Systems, 19(4), 333-352, 1995.

[Ozkarahan, 2000] Ozkarahan I.: Allocation of Surgeries to Operating room by Goal Programming, Journal of Medical Systems, 24(6), 2000.

[Patrick et Carl., 2005] Patrick G., Carl L.: Cost minimization in Endoscopy Center Scheduling: A case-controlled study, Journal of Clinical Gastroenterology, 39 (4), 268-272, 2005.

[Patterson, 1996] Patterson P.: What makes a well-oiled scheduling system, Journal of OR Manager, 12(9), 19-23, 1996.

[Pirlot et Teghem, 2003] Pirlot M., Teghem J.: Résolution de problèmes de RO par les métaheuristiques, Hermes Science, 173-192, 2003.

[Portmann *et al.*, 1998] Portmann M.C., Vignier A., Dardilhac D., Dezalay D.: Branch and bound crossed with GA to solve hybrid flowshops, European Journal of Operational Research, 107(2), 389-400, 1998.

[Przasnyski, 1986] Przasnyski Z.H.: Operating room scheduling: a literature review, Association of Operating Room Nurse (AORN) Journal, 44, 67-76, 1986.

[Ramis *et al.*, 2001] Ramis F.J., Palma J.L., Baesler F.F. : The use of simulation for process improvement at an ambulatory surgery center, Proceeding of the 2001 Winter Simulation Conference, Peters B.A., Medeiros D.J., Rohrer M.W.(Eds.), 1401-1404, 2001.

[Revel *et al.*, 2003] Revel M., Nicolas G., Bara C., Moreau M.O, Nicolle D., Duret M.: Synthèse des travaux du groupe sur l'implantation et l'organisation des plateaux technique, Ministère de la santé, de la famille, et des personnes handicapées, 35p, 2003.

[Riane *et al.*, 1998] Riane F., Artiba A., Elmaghraby S.E.: A hybrid three-stage flowshop problem: efficient heuristics to minimize makespan, European Journal of Operation Research, 109, 321-329, 1998.

[Riane *et al.*, 2001] Riane F., Artiba A., Iassinovski S: An integrated production planning and scheduling system for hybrid flowshop organizations, International Journal of Production Economics, 74(1), 2001.

[Riane *et al.*, 2002] Riane F., Artiba A., Elmaghraby S.E.: Sequencing a hybrid two-stage flowshop with dedicated machines, International Journal of Production Research, 40(17), 4353-4380, 2002.

[Ross et Soland, 1975] Ross G.T., Soland R.M.: A branch and bound algorithm for the generalized assignment problem, Mathematical Programming, 8, 91-103, 1975.

[Roux, 2001] Roux O. : La mémoire dans les algorithmes à colonie de fourmis : applications à l'optimisation et à la programmation automatique, thèse, Informatique, Université du Littoral Côté d'Opale, 137p, 2001.

[Santos *et al.*, 1995] Santos D.L., Hunsucker J.L., Deal D.E.: Global lower bounds for flowshop with multiple processors, European Journal of Operational Research, 80, 112-120, 1995.

[Sen et al., 2003] Sen T., Sulek J.M., Pileepan P.: Static scheduling research to minimize weighted and unweighted tardiness: A state of art survey, International Journal of Production Economics, 63(1), 1-12, 2003.

[Sevaux et al., 2005] Sevaux M., Jouglet A., Oguz C.: Combining constraint programming and memetic algorithm for the hybrid flowshop scheduling problem, ORBEL 19th annual conference of the SOGESCI-BVWB, Belgium, 2005.

[Shukla et al., 1990] Shukla R. K., Ketcham J.S., Ozcan Y.A.: Comparison of subjective versus data base approaches for improving efficiency of operating room scheduling, Health Services Management Research, 3, 74-81, 1990.

[Sier et al., 1997] Sier D., Tobin P., McGurk C.: Scheduling surgical procedures, Journal of Operating Research Society, 48, 884-891, 1997.

[Sonnenberg, 2000a] Sonnenberg A.: Timing and Scheduling of endoscopic procedures, Gastrointestinal Endoscopy, 52 (2), 204-211, 2000.

[Sonnenberg, 2000b] Sonnenberg A.: Waiting lines in the endoscopy unit, Gastrointestinal Endoscopy, 52(4), 517-524, 2000.

[Sriskandarajah et Sethi, 1989] Sriskandarajah C., Sethi S.P.: Scheduling algorithm for flexible flowshops: worst and average case performance, European Journal of Operational Research, 43(2), 143-160, 1989.

[Sriskandarajah et Wagneur 1991] Sriskandarajah C., Wagneur E.: Hierarchical control of the two processor flow-shop with state dependent processing times: complexity analysis and approximate algorithms, INFOR, Canadian Journal of Information Systems and Operational Research, 29(3), 193-205, 1991.

[Strum et al., 1997] Strum D.P., Vargas L.G., May J.H., Bashein G.: Surgical suite utilization and capacity planning : a minimal cost analysis model, Journal of Medical systems, 22(5), 1997.

[Strum et al., 1998] Strum D.P., May J.H., Vargas L.G.: Surgical procedure times are well modeled by the log normal distribution, Anesthesia & Analgesia, 86, S47, 1998.

[Strum et al., 1999] Strum D.P., Vargas L.G., May J.H.: Surgical subspecialty block utilization and capacity planning: a minimal cost analysis model, Anesthesiology, 90(4), 1176-1185, 1999.

[Strum et al., 2000a] Strum D.P., May J.H., Vargas L.G.: Modeling the uncertainty of surgical procedure times: comparison of the log-normal and normal models, Anesthesiology, 92, 1160-1167, 2000.

[Strum et al., 2000b] Strum D.P., Sampson A.R., May J.H., Vargas L.G.: Surgeon and type of anesthesia predict variability in surgical procedure times, Anesthesiology, 92, 1454-1466, 2000.

[Su, 2003] Su L.: A hybrid two-stage flowshop with limited waiting time constraints, Computers & Industrial Engineering, 44(3), 409-424,2003.

[Suresh, 1997] Suresh V.: A note on scheduling of two-stage flow shop with multiple processors. International Journal of Production Economics, 49, 77-82, 1997.

[Taillard, 1990] Taillard E.D.: Some efficient heuristic methods for the flow shop sequencing problem. European Journal of Operational Research, 47 (1), 65-74, 1990.

[Teghem, 2003] Teghem J.: Programmation Linéaire, Editions de l'Université de Bruxelles, 379p, 2003.

[Ullman, 1976] Ullman J.D.: Complexity of sequencing problems, Coffman E.G. Jr. (ed.), Computer and Job/shop Scheduling Theory, Wiley & Sons, New York, 139-164, 1976.

[Ulusoy et Sivrikarya-Serifoglu, 1998] Ulusoy G., Sivrikarya-Serifoglu: A genetic algorithm approach for hybrid flow-shops, in: 6th International Workshop on Project Management and Scheduling, Turkey, 313p, 1998.

[Valvério de Carvalho, 1998] Valvério de Carvalho, J.M.V.: Exact solution of cutting stock problems using column generation and branch-and-bound, International Transactions in Operational Research Issue, 5(1), 35-44, 1998.

[Vanderbeck et Wolsey, 1996] Vanderbeck, F., Wolsey, L. A.: An exact algorithm for IP column generation, Operation Research letters, 19, 151-159, 1996.

[Viapiano, 1999] Viapiano J.: Determining the cost of surgical operations. Semin Anesthesiol, 18, 310-321, 1999.

[Vignier et al., 1996] Vignier A., Billaut J.C., Proust C.: Solving the k-stage hybrid flowshop scheduling problems, Computational Engineering in systems Applications (CESA96), IEEE-SMC, Lille, France, 1996.

[Vignier et Venturini, 1996] Vignier A., Venturini G.: Comparaison entre une procédure par séparation et évaluation et un algorithme génétique pour la résolution d'un flow-shop hybride. Actes du Congrès d'Automatique et de Génie Industriel (AGI'96), Tours, France 1996.

[Wang, 2005] Wang H.: Flexible flow shop scheduling: optimum, heuristics and artificial intelligence solutions, Expert Systems, 22(2), 78-85, May, 2005.

[Wang et Hunsucker, 2003] Wang w., Hunsucker J.L.: An Evaluation of the CDs Heuristic In flow shop with multiple processors, Journal of the Chinese institute of Industrial Engineers, 20(3), 295-304, 2003.

[Wardono et Fathi, 2004] Wardono B., Fathi Y.: A tabu search algorithm for the multi-stage parallel machine problem with limited buffer capacities, European Journal of Operational Research, 155, 380-401, 2004.

[Weinbroum et al., 2003] Weinbroum A.A., Ekstein P., Ewri T.: Efficiency of the operating room suite, The American Journal of Surgery, 185, 244-250, 2003.

[Wright *et al.*, 1996] Wright I.H, Kooperberg C., Bonar A.B., Bashein G.: Statistical modeling to predict elective surgery time: Comparison with a computer scheduling system and surgeon-provided estimates, Anesthesiology, 85(6), 1235-1245, 1996.

[Yokoyama, 2001] Yokoyama M.: Hybrid flow-shop scheduling with assembly operations, International Journal of Production Economics, 73(2), 2001.

[Zhou et al., 1999] Zhou J., Dexter F., Macario A., Lubarsky D.A.: Relying solely on historical surgical times to estimate accurately future surgical times is unlikely to reduce the average length of time cases finish late, Journal of Clinical Anesthesia, 11, 601-605, 1999.

[Zhou et Dexter, 1998] Zhou J., Dexter F.: Method to assist in the scheduling of add-on surgical case: Upper prediction bounds for surgical case durations based on the log-normal distribution, Anesthesiology, 89 (5), 1228-1232.

ANNEXES

ANNEXE 1 : Principe de génération de colonnes et de décomposition de Dantzig –Wolfe

La méthode de décomposition de Dantzig-Wolfe et la génération de colonnes sont utilisées avec succès pour la résolution de programmes en nombres entiers de grande taille. Dans cette partie, nous allons présenter brièvement le principe classique de génération de colonnes et de décomposition de Dantzig-Wolfe [1960] ci-dessous pour un problème de programmation linéaire.

$$\min \quad cx$$

$$s.t.$$

$$Tx = d \qquad T(m \times n)$$

$$x \geq 0$$

de grande dimension, en particulier si le nombre m de contraintes est beaucoup plus petit que le nombre n de variables, un grand nombre de variables resteront fort probablement hors base durant toute la procédure du simplex. Les colonnes correspondantes du tableau simplexe s'avèrent dès lors totalement inutiles et même coûteuses du point de vue temps de calcul et mémoire à occuper. De plus, la forme révisée du simplexe a l'avantage de ne générer à chaque itération que la seule colonne nécessaire, à savoir celle de la variable hors base x_k rentrant dans la base. Pour appliquer ce principe de génération de colonnes, il faut en fait disposer d'un moyen pour calculer les coûts réduits des colonnes qui ne sont pas dans la base afin de trouver une colonne qui entre dans la base.

Ce principe de génération de colonnes s'avère particulièrement efficace lorsqu'il est cumulé avec une judicieuse exploitation de la structure particulière d'un problème. En effet, très souvent les problèmes de grande dimension possèdent une telle structure : non seulement la matrice de contraintes T possède beaucoup de zéros, c'est-à-dire qu'elle est fort « creuse », mais ceux-ci sont répartis de manière particulière comme une structure « par blocs », où les « blocs » se composent des éléments non-zéros, montré dans la Figure 27.

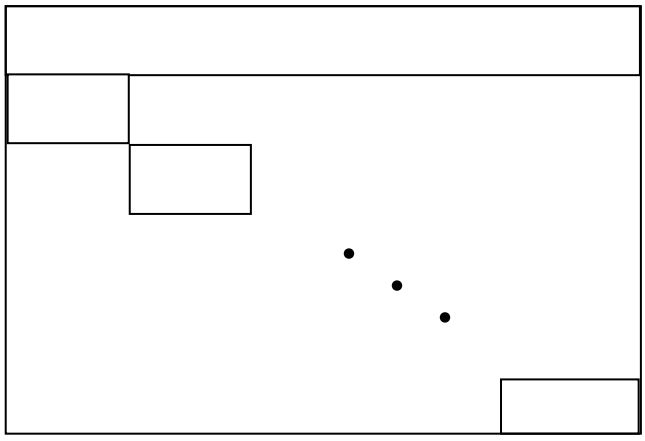

Figure 27 : Structure « par blocs » de la matrice de contraintes

Teghem [2003] nous présente cette structure dans un contexte du type suivant :

Soit une grande compagnie qui dispose de P filiales ($j = 1,..., P$). Chaque filiale j dispose d'une certaine autonomie pour générer ses niveaux d'activité x_j, de façon à minimiser sa fonction de coût $c_j x_j$, en utilisant au mieux ses ressources disponibles d_j et en respectant ses contraintes spécifiques : $T_j x_j = d_j$ avec $T_j = (m_j \times n_j)$.

Toutefois, chaque filiale n'est pas totalement indépendante et la compagnie doit globalement respecter un certain nombre de contraintes, appelées contraintes de liaison, qui lient l'ensemble de ses filiales : $\sum_{j=1}^{P} L_j x_j = d_0$ avec $L_j : (m_0 \times n_j)$.

Le problème général de la compagnie se formule dès lors comme suit:

$$\min \quad \sum_{j=1}^{P} c_j x_j$$

s.t.

$$\sum_{j=1}^{P} L_j x_j = d_0 \qquad \text{avec} \qquad \begin{cases} c_j : (1 \times n_j) \\ x_j : (n_j \times 1) \\ L_j : (m_0 \times n_j) \\ d_0 : (m_0 \times 1) \\ T_j : (m_j \times n_j) \\ d_j : (m_j \times 1) \end{cases}$$

$$T_j x_j = d_j, \quad j = 1, \cdots, P$$

$$x_j \geq 0, \quad j = 1, \cdots, P$$

La matrice globale T de ce problème comprend $\sum_{j=1}^{P} m_j$ contraintes $\sum_{j=1}^{P} n_j$ variables et présente la structure « par blocs » de la Figure 27.

Le principe de décomposition de Dantzig-Wolfe est de diviser le problème en deux parties :

- Problème principal, « Master Problem », constitué des contraintes de liaison ;
- Sous-problème, « sub-problem », composé de tous les blocs de contraintes autres que celles de liaison avec un objectif de trouver une colonne dont le coût de réduit est le plus petit pour entrer dans la base.

ANNEXE 2 : Analyse de la Complexité du Modèle Centré sur une Salle d'Opération et un Lit en SSPI

Comme le problème d'ordonnancement centré sur une salle d'opération et un lit en SSPI est une variante du modèle de « flow shop » à deux machines, nous allons analyser sa complexité ci-dessous :

Premièrement, nous détaillons le processus par la Figure 28 ci-dessous.

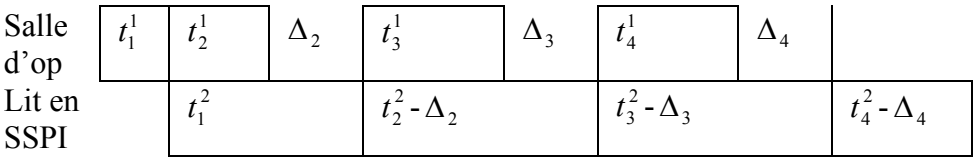

Figure 28 : Processus du modèle centré sur une salle d'opération et un lit en SSPI

t_i^1 représente le temps opératoire de l'intervention *i* dans la salle d'opération ; t_i^2 représente le temps de réveil dans la salle de réveil sans attente dans la salle d'opération. Δ_i représente le temps d'attente dans la salle d'opération après l'intervention *i*. Donc, pour le cas ordinaire, la durée dans la salle de réveil pour le patient *i* est : $\max\{0, t_i^2 - \Delta_i\}$.

Normalement, le coût d'une heure de salle d'opération est beaucoup plus cher que celle de la SSPI à cause des salaires des acteurs et le coût des équipements chirurgicaux. Nous supposons que le rapport entre le coût d'une heure de salle d'opération et celui de la SSPI est α ($\alpha > 1$). Notre objectif est la minimisation du coût total du bloc opératoire, c'est-à-dire, le coût total des salles d'opération et de la SSPI.

Nous mettons T_i^1 représentant le temps d'accomplissement des premières *i* interventions dans la salle d'opération ; T_i^2 le temps d'accomplissement des premières *i* interventions dans la SSPI, c'est-à-dire, le *makespan* des premières *i* interventions du bloc opératoire. Alors, la fonction objectif peut être calculée par la formule suivante : $\alpha T_n^1 + T_n^2$

où

$$T_i^1 = t_1^1 + t_2^1 + \Delta_2 + t_3^1 + \Delta_3 + \cdots + t_i^1 + \Delta_i$$
$$= \sum_{j=1}^{i} t_i^1 + \sum_{j=2}^{i} \Delta_i$$
$$T_i^2 = t_1^1 + t_1^2 + t_2^2 - \Delta_2 + t_3^2 - \Delta_3 + \cdots + t_i^2 - \Delta_i$$
$$= t_1^1 + \sum_{j=1}^{i} t_i^2 - \sum_{j=2}^{i} \Delta_i$$

➔

$$\alpha T_i^1 + T_i^2 = \alpha(\sum_{j=1}^{i} t_i^1 + \sum_{j=2}^{i} \Delta_i) + t_1^1 + \sum_{j=1}^{i} t_i^2 - \sum_{j=2}^{i} \Delta_i$$

$$= \alpha \sum_{j=1}^{i} t_i^1 + t_1^1 + \sum_{j=1}^{i} t_i^2 + (\alpha - 1) \sum_{j=2}^{i} \Delta_i$$

$$= \alpha \sum_{j=1}^{i} t_i^1 + t_1^1 + t_1^2 + \sum_{j=2}^{i} (t_i^2 + (\alpha - 1)\Delta_i) \quad , \alpha > 1$$

Cette formule ci-dessus peut être transformée polynomiale à la formule comme suit:

$$t_1^1 + t_2^1 + \sum_{j=2}^{i} (t_i^2 + (\alpha - 1)\Delta_i), \quad \alpha > 1$$

qui est le *makespan* d'un problème de « flow shop » suivant (montré dans la Figure 29):

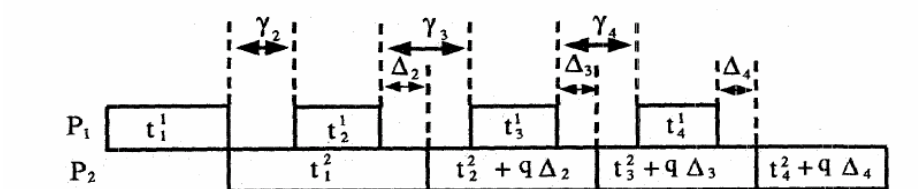

Figure 29 : Modèle de « flow shop » analysé dans [Sriskandarajah et Wagneur 1991]

Où $q = \alpha - 1 > 0$.

Comme ce problème était prouvé NP-complet par Sriskandarajah et Wagneur [1991], nous prouvons aussi que notre problème concerné est NP-complet.

ANNEXE 3 : Lemme 1

NLB $= \omega TST_{\max} + \max\left\{ \dfrac{SPT\ (M_2) + T^{(2)} - \sum_{k=1}^{M_1} TFT_k}{M_2},\ TST_{\max} + \min\{\, t_i^{(2)} \,|\, i \in \Omega_k \text{ dont } TST_k \equiv TST_{\max}\} \right\}$ est une borne inférieure de la valeur optimale de la fonction objectif du problème d'ordonnancement pour la stratégie du « block scheduling ».

Preuve

Pour calculer cette borne inférieure NLB, il faut avant tout calculer la durée opératoire totale pour chaque plage horaire spécialisée k : $T_k^{(1)} = \sum_{i \in \Omega_k} t_i^{(1)}$, le temps d'achèvement de la dernière intervention dans la plage horaire k dans le cas où il n'y a aucun interruption entre les interventions dans la plage horaire k : $TST_k = TS_k + T_k^{(1)}$, la durée reste dans cette plage horaire dans ce cas-là : $TFT_k = TF_k - TST_k$ et la durée totale de réveil pour l'ensemble des patients: $T^{(2)} = \sum_{i=1}^{N} t_i^{(2)}$.

De plus, nous mettons $TST_{\max} = \max\{TST_k \mid k = 1, \cdots, M_1\}$.

Le temps minimal d'achèvement de la dernière intervention au premier étage (les salles d'opération) est obtenu lorsqu'il n'y a eu aucune interruption entre les interventions dans dernière plage horaire de chaque salle d'opération et que tous les patients ont pu être transférés sans attente dans la SSPI. Le temps minimal d'achèvement de la dernière intervention au premier étage est donc égal à TST_{max}.

Du fait que le temps d'achèvement de la SSPI représente aussi le *makespan* du problème, nous étendons la borne inférieure proposée par Haouari et M'Hallah [1997] à un problème de « flow shop » hybride à deux étages où les machines à chaque étage sont identiques (M_1 machines à l'étage 1 et M_2 machines à l'étage 2). Selon Haouari et M'Hallah [1997], si toutes les machines sont identiques à chaque étage et que les durées d'opération à chaque étage sont fixées, la borne inférieure pour le *makespan* peut être :

$$\text{LB\_HM} = \max\left(\frac{SPT(M_2) + \sum_{i=1}^{N} t_i^{(2)}}{M_2}, \frac{SPT(M_1) + \sum_{i=1}^{N} t_i^{(1)}}{M_1} \right)$$

où $SPT(M_2)$ est la somme minimale des temps d'achèvement des M_2 travaux au premier étage dont les durées opératoires au premier étage ($t_i^{(1)}$) sont les plus courtes parmi tous les interventions et $SPT(M_1)$ est égale à la somme des temps d'achèvement des M_1 travaux dont les durées de réveil au deuxième étage ($t_i^{(2)}$) sont les plus courtes.

Étant donné que, dans notre cas, les M_1 plages horaires sont spécialisées par leur interventions affectées, leurs dates d'ouverture et de fermeture données, et de plus les durées de réveil des interventions sont partagées entre les deux étages, LB_HM n'est pas utilisable car la durée totale de réveil dans la SSPI peut être inférieure à $\sum_{i=1}^{N} t_i^{(2)}$. En conséquence, nous avons adapté cette borne en

$$\max \left\{ \begin{array}{l} \dfrac{SPT\ (M_2) + T^{(2)} - \sum_{k=1}^{M_1} TFT_k}{M_2}, \\ TST_{\max} + \min\{\, t_i^{(2)} \,|\, i \in \Omega_k \text{ dont } TST_k \equiv TST_{\max}\} \end{array} \right\}$$

S'il n'y a aucune interruption entre les interventions effectuées dans les plages horaires, le *makespan* des plages horaires est TST_{max}. Mais, étant donné qu'il y a des durées de réveil partagées entre les salles d'opération et la salle de réveil, la durée totale de réveil que peut être passée en salles d'opération sans gêner le *makespan* des plages horaires et leurs heures de fermeture est de $\sum_{k=1}^{M_1} TFT_k$ et la durée totale de réveil restante en SSPI est $T^{(2)} - \sum_{k=1}^{M_1} TFT_k$.

D'autre part, comme chacun des M_2 lits de la SSPI est libre au début de l'ouverture du bloc opératoire, la durée totale d'attente inévitable pour ces M_2 lits est au moins égale à la durée totale opératoire des M_2 interventions, dont les durées opératoires sont les plus courtes parmi toutes les interventions, dans les plages horaires (nommé $SPT(M_2)$ par Haouari et M'Hallah [1997]), alors

$\dfrac{SPT(M_2) + T^{(2)} - \sum_{k=1}^{M_1} TFT_k}{M_2} \leq$ *makespan* est le minimum de notre problème.

De plus, dans le cas où tous les patients se réveillent dans leur salle d'opération sauf un patient i_0 dont la durée de réveil est le petit parmi toutes les interventions ($t_{i_0}^{(2)} = \min\{t_i^{(2)} \,|\, i \in \Omega_k, dont\ TST_k \equiv TST_{\max}\}$) et qui est le dernier à être opéré dans la salle d'opération spécialisé ainsi que le dernier à finir de toutes les salles d'opération, c'est-à-dire $C_{i_0}^{(1)} = TST_{nax}$ et $\Delta_{i_0} = 0$, nous pouvons aussi obtenir un *makespan* le plus petit, alors, $TST_{\max} + \min\{t_i^{(2)} \,|\, i \in \Omega_k, dont\ TST_k \equiv TST_{\max}\} \leq$ m*akespan* est le minimum de notre problème.

En conséquence, nous aboutissons à l'établissement du Lemme 1.

ANNEXE 4 : Lemme 2

Nous notons π^*, un des programmes opératoires optima dont la valeur de la fonction objectif est minimale et Π^*, l'ensemble de tous les programmes opératoires optimums.

$$OLB = \omega^* \max\left\{ t^1_{i^0}, \frac{\sum_{i\in\Omega} t_i^{(1)}}{M_1} \right\} + \max\left\{ t^1_{i^0} + t^{(2)}_{i^0}, \frac{SPT(M_2) + \sum_{i\in\Omega}(t_i^{(2)} + t_i^{(2)})}{M_1 + M_2} \right\}$$

est une borne inférieure de la valeur optimale de la fonction objectif du problème d'ordonnancement pour la stratégie de l'« open scheduling ».

Preuve :

S'il y a un cas où une des salles d'opération ne traite qu'un et un seul patient $i_0 (i_0 \in \Omega)$ sur la journée d'ordonnancement et que l'intervention aura commencer au début de la durée d'ouverture des salles d'opération et que ce patient i_0 est aussi le dernier à quitter le bloc opératoire, il est donc évident que c'est un des programmes opératoires ayant les minimums de *makespan*s sur les deux étages (salles d'opération et la SSPI). Donc, nous constatons que $t^{(1)}_{i_0} \le \min\left\{C^{(1)}_{\max}[\pi^*] \;\middle|\; \pi^* \in \Pi^*\right\}$ et $t^{(1)}_{i_0} + t^{(2)}_{i_0} \le \min\left\{C^{(2)}_{\max}[\pi^*] \,|\, \pi^* \in \Pi^*\right\}$ où $t^{(1)}_{i_0} + t^{(2)}_{i_0} = \max\left\{t_i^{(1)} + t_i^{(2)} \;\middle|\; i \in \Omega\right\}$.

En fait, si nous pouvons obtenir un programme opératoire qui satisfait les conditions que
- il n'y a aucun intervalle entre les interventions dans chaque salle d'opération, c'est-à-dire que les chirurgiens sont toujours disponibles pour leurs interventions ;
- tous les patients peuvent être transférés à la SSPI dès leur intervention finie ;
- les dernières interventions de toutes les salles d'opération sont finies simultanément.

alors le temps minimal d'achèvement de la dernière intervention au premier étage (salles d'opération) est le temps minimal d'achèvement de la dernière intervention au premier étage parmi tous les cas possibles, et donc, $\frac{\sum_{i\in\Omega} t_i^{(1)}}{M_1} \le \min\{C^{(1)}_{\max}[\pi^*] \;|\; \pi^* \in \Pi^*\}$.

De plus, nous constatons qu'au moins *SPT (M_2)* durée d'ouverture de la SSPI ne peut pas être utilisée car tous les patients doivent être d'abord opérés aux salles d'opération avant d'être transférés à la SSPI. *SPT(M_2)* représente la somme minimale des temps d'achèvement des M_2 interventions au premier étage (les salles d'opération) dont les durées opératoires sont les plus petites.

D'autre part, dans notre problème, la durée de réveil de chaque patient peut être partagée entre la salle d'opération et la SSPI et donc, s'il y a un programme opératoire où toutes les salles d'opération et les lits de réveils se libèrent simultanément, il n'y aura pas de temps d'attente entre les salles d'opération et les lits de réveil sauf *SPT(M_2)*. Ce programme opératoire aura sûrement le *makespan* minimal, et donc $\frac{SPT(M_2) + \sum_{i\in\Omega}(t_i^{(2)} + t_i^{(2)})}{M_1 + M_2} \le \min\{C^{(2)}_{\max}[\pi^*] \;|\; \pi^* \in \Pi^*\}$.