Logistique hospitalière à l'aide de robots mobiles reconfigurables

Hassan Baalbaki

Ecole Nationale Supérieure des Mines
SAINT-ETIENNE

NNT : *2011 EMSE 0618*

THÈSE

présentée par

Hassan BAALBAKI

pour obtenir le grade de
Docteur de l'École Nationale Supérieure des Mines de Saint-Étienne

Spécialité : Génie Industriel

Logistique hospitalière à l'aide de robots mobiles reconfigurables

soutenue à saint Etienne, le 09 septembre 2011

Membres du jury

Président :	
M. Dominique FEILLET	Professeur, Ecole Nationale Supérieure des Mines de saint Etienne
Rapporteurs :	
M. Alain GUINET	Professeur, Institut National des Sciences Appliquées de Lyon
M. Bernard GRABOT	Professeur, Ecole Nationale d'Ingénieurs de Tarbes
Examinateur(s) :	
Mme Gulgun ALPAN	MCF-HDR, Institut National Polytechnique de Grenoble
M. Xavier DELORME	M.A , Ecole Nationale Supérieure des Mines de saint Etienne
Directeur(s) de thèse :	
M. Xiaolan XIE	Professeur, Ecole Nationale Supérieure des Mines de saint Etienne

Spécialités doctorales :	**Responsables :**
SCIENCES ET GENIE DES MATERIAUX	J. DRIVER Directeur de recherche – Centre SMS
MECANIQUE ET INGENIERIE	A. VAUTRIN Professeur – Centre SMS
GENIE DES PROCEDES	F. GRUY Professeur – Centre SPIN
SCIENCES DE LA TERRE	B. GUY Maître de recherche – Centre SPIN
SCIENCES ET GENIE DE L'ENVIRONNEMENT	J. BOURGOIS Professeur – Fayol
MATHEMATIQUES APPLIQUEES	E. TOUBOUL Ingénieur – Fayol
INFORMATIQUE	O. BOISSIER Professeur – Fayol
IMAGE, VISION, SIGNAL	JC. PINOLI Professeur – Centre CIS
GENIE INDUSTRIEL	P. BURLAT Professeur – Fayol
MICROELECTRONIQUE	Ph. COLLOT Professeur – Centre CMP

Enseignants-chercheurs et chercheurs autorisés à diriger des thèses de doctorat (titulaires d'un doctorat d'État ou d'une HDR)

AVRIL	Stéphane	MA	Mécanique & Ingénierie	CIS
BATTON-HUBERT	Mireille	MA	Sciences & Génie de l'Environnement	Fayol
BENABEN	Patrick	PR 1	Sciences & Génie des Matériaux	CMP
BERNACHE-ASSOLLANT	Didier	PR 0	Génie des Procédés	CIS
BIGOT	Jean-Pierre	MR	Génie des Procédés	SPIN
BILAL	Essaïd	DR	Sciences de la Terre	SPIN
BOISSIER	Olivier	PR 1	Informatique	Fayol
BORBELY	Andras	MR	Sciences et Génie des Matériaux	SMS
BOUCHER	Xavier	MA	Génie Industriel	Fayol
BOUDAREL	Marie-Reine	PR 2	Génie Industriel	DF
BOURGOIS	Jacques	PR 0	Sciences & Génie de l'Environnement	Fayol
BRODHAG	Christian	DR	Sciences & Génie de l'Environnement	Fayol
BURLAT	Patrick	PR 2	Génie industriel	Fayol
COLLOT	Philippe	PR 1	Microélectronique	CMP
COURNIL	Michel	PR 0	Génie des Procédés	SPIN
DAUZERE-PERES	Stéphane	PR 1	Génie industriel	CMP
DARRIEULAT	Michel	IGM	Sciences & Génie des Matériaux	SMS
DECHOMETS	Roland	PR 1	Sciences & Génie de l'Environnement	Fayol
DESRAYAUD	Christophe	MA	Mécanique & Ingénierie	SMS
DELAFOSSE	David	PR 1	Sciences & Génie des Matériaux	SMS
DOLGUI	Alexandre	PR 1	Génie Industriel	Fayol
DRAPIER	Sylvain	PR 2	Mécanique & Ingénierie	SMS
DRIVER	Julian	DR 0	Sciences & Génie des Matériaux	SMS
FEILLET	Dominique	PR 2	Génie Industriel	CMP
FOREST	Bernard	PR 1	Sciences & Génie des Matériaux	CIS
FORMISYN	Pascal	PR 1	Sciences & Génie de l'Environnement	Fayol
FRACZKIEWICZ	Anna	DR	Sciences & Génie des Matériaux	SMS
GARCIA	Daniel	MR	Génie des Procédés	SPIN
GIRARDOT	Jean-Jacques	MR	Informatique	Fayol
GOEURIOT	Dominique	MR	Sciences & Génie des Matériaux	SMS
GRAILLOT	Didier	DR	Sciences & Génie de l'Environnement	Fayol
GROSSEAU	Philippe	MR	Génie des Procédés	SPIN
GRUY	Frédéric	MR	Génie des Procédés	SPIN
GUY	Bernard	MR	Sciences de la Terre	SPIN
GUYONNET	René	DR	Génie des Procédés	SPIN
HERRI	Jean-Michel	PR 2	Génie des Procédés	SPIN
INAL	Karim	PR 2	Microélectronique	CMP
KLÖCKER	Helmut	DR	Sciences & Génie des Matériaux	SMS
LAFOREST	Valérie	CR	Sciences & Génie de l'Environnement	Fayol
LERICHE	Rodolphe	CR CNRS	Mécanique et Ingénierie	SMS
LI	Jean-Michel	EC (CCI MP)	Microélectronique	CMP
MALLIARAS	George Grégory	PR 1	Microélectronique	CMP
MOLIMARD	Jérôme	MA	Mécanique et Ingénierie	SMS
MONTHEILLET	Frank	DR 1 CNRS	Sciences & Génie des Matériaux	SMS
PERIER-CAMBY	Laurent	PR 2	Génie des Procédés	SPIN
PIJOLAT	Christophe	PR 1	Génie des Procédés	SPIN
PIJOLAT	Michèle	PR 1	Génie des Procédés	SPIN
PINOLI	Jean-Charles	PR 0	Image, Vision, Signal	CIS
STOLARZ	Jacques	CR	Sciences & Génie des Matériaux	SMS
SZAFNICKI	Konrad	MR	Sciences & Génie de l'Environnement	Fayol
THOMAS	Gérard	PR 0	Génie des Procédés	SPIN
TRIA	Assia		Microélectronique	CMP
VALDIVIESO	François	MA	Sciences & Génie des Matériaux	SMS
VIRICELLE	Jean-Paul	MR	Génie des procédés	SPIN
WOLSKI	Krzysztof	DR	Sciences & Génie des Matériaux	SMS
XIE	Xiaolan	PR 1	Génie industriel	CIS

Glossaire :

PR 0	Professeur classe exceptionnelle
PR 1	Professeur 1ère classe
PR 2	Professeur 2ème classe
MA(MDC)	Maître assistant
DR	Directeur de recherche
Ing.	Ingénieur
MR(DR2)	Maître de recherche
CR	Chargé de recherche
EC	Enseignant-chercheur
IGM	Ingénieur général des mines

Centres :

SMS	Sciences des Matériaux et des Structures
SPIN	Sciences des Processus Industriels et Naturels
Fayol	Institut Henri Fayol
CMP	Centre de Microélectronique de Provence
CIS	Centre Ingénierie et Santé

Dernière mise à jour le : 23 août 2011

Remerciements

Tout d'abord, je souhaite remercier de tout coeur M. Xiaolan XIE pour son encadrement sans faille, son dévouement et tous ses conseils précieux qu'il m'a promulgué .Sa patience et sa sérénité, m'ont permis de surmonter les difficultés que j'ai rencontrées tout au long de ma thèse en m'indiquant les voies à suivre. Je lui serai toujours reconnaissant pour le temps qu'il a consacré sans calcul afin de m'épauler.

Je tiens également à remercier les rapporteurs de cette thèse : M. Bernard Grabot pour ses remarques pertinentes, M. Alain Guinet pour ses conseils judicieux. Mes remerciements vont également à M. Dominique Feillet qui accepté de présider ce jury, sans compter les enseignements intéressants que j'ai pu tirer de nos discussions. Je remercie également Mme Gulgun Alpan d'avoir accepté de participer à ce jury ainsi que M. Xavier Delorme que j'ai eu le plaisir de côtoyer plusieurs fois le long de la thèse, et dont les réflexions avisées ont eu un reflet essentiel dans ce travail.

Tout mon parcours expérimental n'aurait pu avoir lieu sans les efforts et le support de nos partenaires du projet IWARD, plus particulièrement nos partenaires allemands : Simon Thiel, Dagmar Habe et Micha Block (sans leurs nuits blanches et leurs efforts titanesques, nos robots seraient restés immobiles lors de certaines manifestations-clés du projet).

J'ai une pensée particulière, à M. Denis Naddef qui m'a initié sur la voie de la recherche, ainsi que M. Dani Mezher et M. Nicolas Rouhanna qui m'ont fait découvrir le plaisir de la programmation et du raisonnement scientifique.

Avant de clore j'adresse un grand merci, un grand respect et une profonde reconnaissance à notre illustre Ecole des Mines, son personnel sa direction et à tous mes collègues stéphanois, pour les années que nous avons passées ensembles ; nos discussions, nos pauses café, nos repas, nos soirées resteront à jamais gravés dans mon cœur.

Je ne pourrai jamais oublier mes amis : Ali Harb, Bassel El Samad et Abdallah Sadki auxquels je voue une profonde reconnaissance, le les remercie pour leur aide précieuse, pour leurs encouragements, pour les heures et les jours qu'ils ont dépensés sans compter.

Finalement, ma dernière pensée va vers ma famille : mon père médecin qui m'a beaucoup encouragé à suivre la piste qui allie informatique et soins médicaux, mes sœurs qui m'ont toujours soutenu et ma mère qui a tant lu et relu, corrigé et recorrigé les multiples versions antécédentes à cette thèse …. Je vous aime.

Table des matières

Table des Figures

Table des algorithmes

Table des tableaux

Introduction Générale

Le vieillissement de la population européenne a un impact positif sur l'évolution de la médicine et des services médicaux offerts par les hôpitaux. Cependant il est aussi une cause importante de l'explosion de la demande de soins et de la montée des coûts des soins médicaux. Les problèmes liés à la pénurie de services de soins peuvent être réduits par le déploiement de nouvelles technologies. L'objectif principal du projet IWARD, dont cette thèse fait partie, est d'améliorer la qualité de vie des patients hospitalisés ainsi que celle du personnel hospitalier à l'aide de solutions robotiques.

Le projet IWARD intègre des technologies en vue d'assister le personnel médical dans les activités de support, c'est-à-dire des activités logistiques non médicales. L'assistance du personnel médical dans les activités de support leur permet de mieux se concentrer sur les activités de soins et donc de fournir une qualité de soins meilleure.

Le projet IWARD repose sur deux grandes orientations stratégiques du sixième programme cadre de l'Union européenne (FP6), dans le domaine de la science, de la recherche et de l'innovation, dont le but est de "renforcer les bases scientifiques et technologiques de l'industrie afin favoriser la compétitivité internationale tout apportant son aide à la recherche dans d'autres politiques communautaires".

Au début du projet, certains objectifs ont été fixés afin de guider les activités de recherche et de développement du projet IWARD. Tout le long du projet, la majorité de ces objectifs ont été atteints, tout en faisant face aux difficultés techniques et aux limites technologiques existantes. Les thèmes directifs sont :

Sécurité, qualité et fiabilité

Les robots IWARD sont appelés à fonctionner dans des milieux hospitaliers. Pour cette raison, il est essentiel qu'ils soient en mesure de naviguer en toute sécurité et fiabilité dans un environnement peuplé de patients et de personnel médical tout en évitant toutes sortes d'obstacles.

Essaim de robots coopérants

Comme son nom l'indique, un essaim de robots coopérants est un ensemble de robots effectuant ensemble de multiples tâches.

Modularité et système Plug and Play

Les robots IWARD sont tous des robots mobiles simples qui peuvent selon besoin fournir des fonctionnalités supplémentaires par simple insertion ou retrait de modules amovibles grâce à des technologies «*plug and play*» (branche et fonctionne).

Systèmes robotiques flexibles

Le prototype du projet IWARD doit pouvoir apporter son aide au personnel médical dans différentes activités de support comme le transport des médicaments, le nettoyage, le guidage des patients/visiteurs. Il doit également être en mesure de surveiller les activités de la vie

quotidienne (AVQ) des patients (alimentation, toilette, bain, déplacements aux toilettes), les indicateurs de bien-être (état émotionnel, visites / appels reçus, engagement dans des activités de loisirs telles que la télévision / radio, lecture, marche, si possible, etc), les contrôles de santé (contrôle des médicaments, détection d'incidents critiques, chutes, troubles du comportement, épilepsie, crises cardiaques, crises d'anxiété).

Plan de la thèse

Cette thèse est composée de sept chapitres. Le premier chapitre introduit le projet IWARD, les spécificités des robots développés par le consortium, les différents types de tâches exécutées par les robots, les différents membres du consortium, les groupes de travail ainsi que le travail assigné à chaque partenaire.

Dans le deuxième chapitre, nous nous intéressons aux deux problématiques principales développées dans ce manuscrit. La première partie porte sur la coordination des robots. Le modèle adopté pour la thèse est inspiré des techniques de la robotique en essaim, qui sont largement étudiées dans la littérature. La deuxième partie, est la localisation des stations d'attente des robots, qui est une pièce clé du problème stratégique étudié dans cette thèse. Nous présenterons également une revue de littérature des recherches effectuées sur cette problématique, les modèles classiques ainsi que certaines applications de chaque modèle.

Dans le troisième chapitre, nous exposerons le modèle formel du système robotique considéré dans cette thèse ainsi que le plan décisionnel du système robotique. Nous modéliserons les différents composants du système robotique. Ensuite, nous présenterons les différents états dynamiques des robots ainsi que les différentes phases caractéristiques d'une mission. Nous évoquerons également dans ce troisième chapitre la prise de décision et de coordination notamment au niveau stratégique qui traite la gestion de l'autonomie, la reconfiguration, l'affectation des fonctionnalités, et la répartition des robots aux différentes stations d'attentes. Nous passerons alors au niveau tactique qui traite l'affectation des missions aux différents robots du système robotique et l'ordonnancement de ces missions sur chaque robot.

Le chapitre 4 est focalisé sur la planification et la gestion stratégique du système robotique, nous présenterons séparément les trois problèmes stratégiques : (i) le planning de rechargement de chaque robot, (ii) la reconfiguration des robots et l'affection des fonctionnalités sur les différents robots opérationnels, (iii) la localisation des stations d'attente lorsque les robots sont en mode veille. Nous aborderons brièvement l'état de l'art des deux premiers problèmes, sachant le l'état de l'art du troisième problème a été étudié au deuxième chapitre. Ensuite nous modéliserons ces problématiques sous forme de problèmes linéaires, et nous étudierons les différentes possibilités de résolution de ces problèmes séparément ou en les combinant ensemble. De même nous proposerons une méthode de résolution basée sur les algorithmes de générations de colonnes.

La deuxième partie du manuscrit est consacrée à la gestion tactique qui se résume par l'affectation des tâches aux différents membres du groupe des robots et par l'ordonnancement de l'exécution de ces tâches. Afin de prendre en compte les spécificités des différents types de missions, nous proposerons dans le cinquième chapitre deux critères d'évaluation des plannings : le premier est une fonction par escalier et qui se base sur le principe de prime, tandis que le deuxième est plutôt une fonction continue et se base sur le principe de pénalités.

Nous proposons dans le sixième chapitre une approche centralisée pour résoudre le problème tactique d'affectation et d'ordonnancement des tâches. Elle est basée sur les algorithmes évolutionnaires que nous adapterons pour mieux gérer la performance de ces algorithmes et accélérer la convergence vers des instances plus performantes. Une implémentation auto-adaptive est présentée; celle-ci ajustera automatiquement les paramètres et la méthode de croisement par apprentissage à partir des résultats des anciennes générations.

Le septième chapitre est consacré au développement d'une approche décentralisée pour l'affectation et l'ordonnancement des tâches. Elle est basée sur la négociation par des méthodes d'enchères inversées. Différents problèmes rencontrés en temps réel sont étudiés: affectation de nouvelles missions, legs des missions des robots en difficulté, réquisition de missions pour les robots sans missions.

Chapitre 1. Contexte et Problématique

Cette thèse s'inscrit dans le cadre du projet européen IWARD, « *Intelligent Robot Swarm for Attendance, Recognition, Cleaning and Delivery* ». IWARD est un projet de recherche du programme de travail "*Advanced Robotics*" qui fait parti des activités de la société des Technologies de l'Information (IST). Il a été financé par la Commission européenne dans le sixième programme-cadre FP6 de 2007 à 2009.

Le Consortium IWARD est composé de 10 partenaires de 7 pays européens présenté ci-dessous:

- Allemagne: Fraunhofer IAO, Stuttgart (coordinateur)
- Angleterre:
  - Université de Warwick Coventry,
  - Université de Cardiff
  - Université de Newcastle upon Tyne
- Espagne:
  - Instituto Gerontológico Matia, San Sebastian
  - Fundación Fatronik-Tecnalia, San Sebastian
- France: École Nationale Supérieure des Mines de St-Étienne
- Irlande: Dublin City University
- Italie: Université de Naples Federico II
- Turquie: Université de Sakarya

1.1. Robots RP-6 en milieux hospitaliers

IWARD s'appuie sur des études et des expériences de projets antérieurs sur le développement de solutions robotiques en milieux hospitaliers. Le robot médical équivalant le plus connu et disponible dans le commerce [Thacker 05] est RP-6 (Remote Presence 6) développé par InTouch Health Inc, Santa Barbara, États-Unis (voir Figure 1). Ce robot, mesurant environ 1,60 m de hauteur, est équipé d'un système de visioconférence et est capable de transporter des charges légères. Il est déployé avec succès dans plus de 35 hôpitaux du monde entier, parmi lesquels des institutions prestigieuses comme l'hôpital Johns Hopkins à Baltimore, l'Hôpital de l'Université de Californie à Los Angeles, Detroit Medical Center de Londres et l'hôpital St. Mary.

Diverses études de l'acceptation [Harris 05] ont montré des résultats très positifs. L'hôpital Johns Hopkins a réalisé des études détaillées d'acceptation à la fois chez les patients et chez les médecins. Ces résultats ont été très encourageants et ont tracé les objectifs à suivre pour le projet IWARD.

Figure 1 : Le robot Remote Presence 6

1.2. Le projet IWARD

Un robot IWARD est donc composé d'une plate-forme de base simple à faible coût qui peut s'enrichir par le biais de modules fonctionnels. Cela permettrait à un groupe de robots d'effectuer une multitude de tâches logistiques en milieux hospitaliers et de s'adapter aux changements de demandes.

Les caractéristiques et les spécificités de la plateforme robotique développée dans le cadre du projet IWARD sont données dans le tableau suivant :

Caractéristiques et spécifications de la plate-forme	
Longueur (m)	0,65
Largeur (m)	0,6
Hauteur (m)	0,6 à 1,25
Poids (kg)	6
Vitesse de rotation (deg/s)	20
Ouverture angulaire (deg)	10
Poids de charge (kg)	20
Précision (mm)	10
Résolution (mm)	2
Durée (heures)	2
Consommations énergétiques de base (W)	250

Tableau 1: Spécifications mécaniques de la plate-forme de base

Figure 2 : Famille des robots IWARD

.

La Figure 2, illustre les trois robots développés dans le cadre du projet IWARD. On distingue clairement les tiroirs jaunes qui sont des modules fonctionnels amovibles offrant aux robots des fonctionnalités supplémentaires et donc la possibilité d'exécuter une multitude de tâches différentes par un même robot.

1.3. Les groupes de travail du projet IWARD

Afin de diviser la charge de travail au sein du projet, le consortium IWARD a été divisé en plusieurs groupes de travail (GT), et chaque groupe était responsable de développer un aspect spécifique du projet.

Groupe de travail « Exigences et spécifications »

La coordination de ce groupe de travail était confiée à l'Université de Cardiff. Les principaux objectifs de ce GT étaient de :

- définir les exigences et les spécifications nécessaires pour le développement des robots de service capable de superviser et de mener des activités au jour le jour en milieux hospitaliers;
- poser les fondations d'un système robotique qui a pour but de faciliter la participation, la reconnaissance, le soutien et la communication avec les utilisateurs.

Ce groupe de travail avait aussi comme rôle d'établir les scénarios d'usage des robots comme l'assistance aux activités de nettoyage, le guidage des patients, la consultation à distance et le transport des médicaments.

Nous donnons ici deux exemples d'exigences identifiées :

- la capacité de surveiller les services hospitaliers, de reconnaître les patients ou objets qui nécessitent une attention particulière
- la capacité de fournir des informations complètes et immédiates sur la localisation des patients concernés, ainsi que des photos, vidéos ou audios des patients et de leur

environnement, de communiquer des informations au personnel médical compétent ou à d'autres robots.

Groupe de travail « Robotique et plateforme de base »

Ce groupe de travail, coordonné par l'Université de Newcastle, était responsable de la conception et de la réalisation d'un prototype de plate-forme de robot mobile ayant des capacités de navigation en milieu hospitalier à un niveau approprié de sécurité et d'intégrité.

En premier lieu, des plates-formes de base ont été conçues. Ces plates-formes doivent être capables d'accueillir des modules amovibles munis d'un système de détection d'obstacles intégré permettant d'éviter toute collision avec un quelconque obstacle. Le groupe de travail a également mis l'accent sur le développement de l'architecture de la plate-forme de systèmes embarqués permettant d'inclure la fonctionnalité «*plug and play* » (branche et fonctionne) afin de faciliter l'interchangeabilité des fonctionnalités modulaires.

Groupe de travail « Modularisation »

La coordination de ce GT était confiée à l'université de DCU. Le principal objectif était de développer des modules interchangeables permettant au robot d'effectuer diverses activités en milieux hospitaliers. Les fonctionnalités ciblées sont le nettoyage, la livraison, l'orientation, le guidage, la surveillance de lieux, la surveillance des patients et la consultation à distance. Des modules ont été conçus pour être insérés à la plateforme de base par des liaisons mécaniques souples. La connectivité de la couche physique avec les modules était assurée par un système de « *bus* » comme l'USB (*Universal Serial Bus*) ou Ethernet. Cela permet de faciliter le raccordement de modules supplémentaires ainsi que le retrait de ces composants.

Ce groupe de travail avait aussi comme rôle le développement des méthodes de fusion et d'intégration du système sensoriel du robot avec ceux des modules. Pour cela une sélection de capteurs disponibles sur le marché (vision, proximité, son, température, CO, CO2, O2, fumée, humidité) devait être identifiée afin d'être intégré au modules adéquats.

Groupe de travail « Interface Homme – Robot » (HRI)

La coordination de ce groupe de travail était assurée par l'Université de Warwick. Son principal objectif était de développer différents interfaces, notamment entre les utilisateurs finaux et un robot ou bien entre les utilisateurs et le système robotique (l'équipe ou essaim de robots).

Les solutions fournies doivent assurer une simplicité d'utilisation et une certaine souplesse afin de répondre aux besoins d'un large éventail de population ciblée. Cette flexibilité est obtenue grâce à une approche modulaire qui peut s'adapter à l'environnement, à l'utilisateur et aux modules disponibles sur le robot actuel. Ces composants d'interaction doivent permettre aux infirmiers, professionnels de santé et patients de s'adresser aux robots afin de commander et d'effectuer diverses activités en milieux hospitaliers.

Divers types d'interactions entre les robots et les utilisateurs ont été testées notamment l'interaction vocale en demandant l'exécution de nouvelles tâches vocalement.

Un composant assurant l'interaction entre l' « équipe des robots » et les utilisateurs a été conçu pour permettre aux utilisateurs d'émettre de nouvelles tâches et de gérer les tâches en cours. Vu que la gestion est assurée par un système distribué et coopératif, ce composant joue le rôle d'intermédiaire entre les humains et les robots.

Tous ces composants faisaient partie d'une étude d'acceptation afin d'évaluer les attentes des utilisateurs finaux et de proposer de futures améliorations du système proposé.

Groupe de travail « Plate-forme logicielle »

La coordination de ce groupe de travail était confiée à notre partenaire Fatronik. L'objectif était de concevoir et d'implémenter une plate-forme logicielle commune à tous les robots. Cette plateforme comprend une couche de communication « *peer to peer* » pour assurer la communication entre les différentes entités de l'essaim des robots. Cette couche est essentielle pour la prise de décision distribuée, fondée sur des algorithmes de négociation.

Groupe de travail intelligence en essaim « swarm »

La coordination de ce groupe de travail a été confiée à l'Ecole des Mines de Saint Etienne. Son objectif principal était de développer des stratégies d'exploitation de l'essaim de robots. Ces stratégies permettraient de répartir les tâches parmi le groupe des robots afin d'effectuer efficacement, de façon continue, les activités requises dans le domaine des soins.

Plus précisément, ce GT développe des stratégies d'optimisation pour la localisation et la configuration de l'ensemble du système robotique, des stratégies centralisées pour la planification de tâches et leur affectation afin de mieux répondre aux besoins des soins au fil du temps, et des stratégies de contrôle utilisant des méthodes de négociations conduisant à des décisions rapides et appropriées.

Groupe de travail « Évaluation et diffusion »

La coordination de ce groupe de travail a été confiée à l'université de Sakarya. Son objectif principal était d'identifier les domaines de recherches pouvant fournir une preuve que ce concept de système peut être utile à la communauté, notamment par la définition des critères d'évaluation et des scénarios de test d'un tel système robotique. Le deuxième objectif était de sensibiliser le public en démontrant l'intérêt scientifique de ce projet et ses résultats, ensuite, d'établir des accords commerciaux afin d'assurer l'introduction d'une telle technologie sur le marché.

Groupe de travail « Gestion du projet »

La coordination du projet était assurée par l'institut IAO. Ils avaient pour rôle de coordonner la mise en œuvre du plan du travail, d'organiser les réunions, d'établir les rapports sur l'état d'avancement du projet, de les soumettre à la commission européenne, de s'assurer de la qualité des résultats du projet, de fournir des ressources humaines, financières, techniques et comptables pour le volet managérial du projet.

1.4. Les réalisations clés du projet IWARD

A l'issue du projet, en février 2010, la commission européenne a validé le travail de ce consortium. Durant les trois années 2006 – 2009 le consortium a développé une équipe de trois robots de service. La conception modulaire permet d'équiper individuellement chaque robot avec plusieurs modules amovibles.

Les différentes fonctionnalités fournies par les quatre modules amovibles sont:

1. La livraison des médicaments dans un box fermant à clé,
2. Le nettoyage régulier de grandes surfaces prédéfinies, ou le nettoyage de déversements marqués devant être rapidement retirés
3. L'orientation et le guide des patients vers des lieux spécifiques en utilisant la technologie RFID pour mesurer la distance entre le robot et le patient.
4. La surveillance des conditions environnementales (température, humidité).

La consultation virtuelle et la surveillance des locaux sont assurées par l'écran et l'appareil vidéo qui font partie du la plateforme de base.

Un module fonctionnel inséré dans une fente de robot, est automatiquement alimenté par les batteries du robot grâce à un connecteur d'alimentation standard. Une fois sous tension, l'ordinateur embarqué du module démarre et exécute automatiquement le programme de connectivité « plug-and-play ». Ce programme établit un lien avec l'ordinateur du robot via une connexion Ethernet standard et démarre une session de communication avec le composant gestionnaire de matériel en cours d'exécution sur le robot. Le gestionnaire de matériel enregistre le module inséré, et, par conséquent, la connaissance partagée du système de robot est mise à jour. Cela signifie qu'à tout moment l'ensemble du système IWARD est conscient des modules disponibles sur des robots, ce qui rend possible l'exécution des missions.

Les trois robots utilisent un système à deux roues motrices différentielles, sont équipés d'une carte mère et de microprocesseurs de haute performance. Pour faciliter la navigation et la localisation, les trois robots incorporent un anneau de capteurs ultrasons, des capteurs de détection de collision infrarouge et un scanner laser.

La matrice de capteurs et d'interface du contrôleur a été conçue et construite à Newcastle suivant le cahier des charges fourni par Fatronik. Cette matrice comprend des capteurs ultrasons (*Senscomp série 600*) qui se sont avérés être beaucoup plus robustes et tolérants à l'interférence quand plusieurs robots sont exploités. De plus un scanner laser de haute précision de la gamme *Hokoyu finder ULG-04DX*, a été installé sur chacun des robots. Les données laser sont fusionnées avec les données du sonar pour améliorer les algorithmes d'évitement de collision et les algorithmes de localisation utilisant plan composite en format *bmp*.

Le système de gestion de l'alimentation et le contrôle des états des batteries a été mis au point par l'Université de Newcastle. Ce système se compose d'un microcontrôleur *ARM7 AT91SAM7S256*, d'un *Olimex SAM7-MT256* et d'un onduleur *PowerStream* (*Uninterruptible Power Supply*) qui permet d'effectuer un transfert d'énergie sans faille en s'alimentant de courant du secteur lorsque le robot est branché et ne pas épuisée l'énergie stockée à l'intérieur des batteries. Ce procédé permet de prolonger la durée de vie des batteries et d'augmenter la disponibilité des robots. Un tel dispositif permettra aussi d'effectuer des tests sur la

plateforme logicielle durant la période de recharge sans risquer de se faire vider les batteries de bord.

Afin d'améliorer l'efficacité du système robotique, des batteries amovibles ont été conçues et développées. Ces batteries ressemblent aux box de modules. Elles sont interchangeables entre les robots du même type, et sont branchées sur les fentes prévues pour les modules fonctionnels. Ces batteries servent généralement de source d'énergie d'appoint.

Le système de navigation conçu et mis en œuvre par Fatronik pour le projet IWARD suit une architecture classique. Ses principaux modules sont : la navigation locale, la localisation et le plan du chemin. Les deux derniers modules fonctionnent avec une carte géométrique de l'environnement où les caractéristiques statiques de l'environnement sont fournies sous format bitmap.

La coopération est fondamentale dans IWARD et l'aspect coopératif est présent dans toutes les couches et tous les composants du système. Tous les robots IWARD partagent leurs perceptions et les informations recueillies avec la communauté des robots IWARD. Les missions sont réparties grâce à un système de décisions distribué. De cette façon, les décisions sont prises par le système robotique de manière autonome et sont transparentes à l'utilisateur.

Le système de pilotage prend en compte la gestion de l'autonomie d'énergie des robots et leur rechargement grâce un système intelligent de gestion d'énergie capable de prévoir l'autonomie restante d'un robot.

L'interface homme-robot prend en compte l'ambiguïté de la dualité des interactions possibles. D'une part, les utilisateurs interagissent avec un robot spécifique pour accomplir une tâche, d'autre part, un robot pourrait être un membre quelconque de l'équipe, agissant comme une interface explicite entre équipe de robots et l'utilisateur.

1.5. Conclusion

Inspiré des orientations de recherche du sixième programme cadre de la robotique avancée, le projet IWARD vise un système robotique aves les caractéristiques suivantes :

- Ensemble de robots coordonnés pouvant offrir une valeur ajoutée à l'exécution des missions;
- Plate-forme robotique unifiée, basée sur des composants fiables et permettant la reproduction à bas prix;
- Répartition de travail flexible, extensibilité des fonctionnalités du robot, partage des connaissances ont été des points clé pour la polyvalence prévue par ce programme cadre;
- Environnement hospitalier choisi comme espace de fonctionnement pour le système robotique, offrant des capacités d'actions visées par le sixième programme cadre européen;
- Sécurité, robustesse et conformité aux normes d'éthique ont été les principales caractéristiques pour l'acceptation dans un milieu hospitalier, offrant ainsi des capacités d'actions visées par le programme cadre européen.

Chapitre 2. Revue de littérature

2.1. Robotique en essaim

Un essaim de robots est un groupe de robots travaillant ensemble pour exécuter une tâche commune comme la lutte contre les incendies forestiers, ou le sauvetage en mer.

Figure 3 : Seaswarm, le robot nettoyeur de déchets pétroliers par Senseable de MIT

Le but de cette approche est d'étudier la conception d'un ensemble de robots et de faire émerger le comportement collectif souhaité, à travers les interactions entre robots «inter-robots » et entre les robots et l'environnement. Les études sur les robots en essaim s'inspirent souvent des comportements observés des insectes sociaux, et dénommée « intelligence en essaim ».
La robotique en essaim est issue de la bio-inspiration et dans la majorité des études dans ce domaine, les mêmes mécanismes biologiques au sein d'une collective sont adaptés sur des mécanismes de collectivités de robots. Dans certains exemples, les robots communiquent en déposant des signaux dans l'environnement, ce qui fait penser à la *stigmergie* utilisée par les insectes sociaux [Beckers 94],[Agassounon 04].

Figure 4 : Le comportement collectif des fourmis d'Argentine (*Linepithema humile*) formant un pont vivant a servi de source d'inspiration au développement de techniques d'intelligence en essaim. Photo par Guy Theraulaz, CNRS Toulouse

Beni et Wang étaient les premiers à utiliser le terme « intelligence en essaim » (*Swarm Intelligence*) dans leur article [Beni 89], pour décrire le comportement d'un groupe de robots coopérant, en se déplaçant et opérant de façon asynchrone et non séquentielle. A l'époque, ils proposaient une alternative à la terminologie de robotique cellulaire. En 2005 Beni a révisé sa définition initiale [Beni 05] en prenant en comptes les différentes extensions possibles.

D'après [Beni 05] un essaim est un groupe d'entités de plus d'une centaine, mais très inférieur au nombre d'Avogadro (10^{23}) ce qui nécessiterait dans ce cas des approches statiques pour résoudre le problème. Ces entités de comportement simpliste, sont quasi identiques et donc le contrôle de ces entités est décentralisé. Bien évidement, ces entités opèrent en mode asynchrone ce qui leur confère des comportements temporellement indépendants où chaque entité profite de ses facultés de perception pour prendre ses décisions et entreprendre les actions conséquentes. Ces robots sont dotés d'une « aptitude à générer des schémas ordonnés de façon imprévisible, schémas sur lesquels une analyse est faite par chaque robot pour optimiser si besoin ses actions en vue d'atteindre un objectif collectif fixé à l'avance »

Plusieurs applications de robots en essaim ont été répertoriées, ces champs d'applications peuvent être divisés en plusieurs catégories :

A- *L'exploration* :

Dans cette application, les robots en essaim sont introduits dans un environnement inconnu. Ensuite l'essaim utilisera ses capacités collectives pour découvrir l'espace qui l'entoure. Zlot et al. ont utilisé la coordination par enchères pour attribuer les différentes zones aux différents membres de l'essaim [Zlot 06].

B- *La couverture* :

L'essaim est utilisé pour couvrir une zone géographique ou bien effectuer des passages par tous les points qui constituent cette zone. Ce problème est rencontré dans les applications comme les aspirateurs ou bien pour les tondeuses à gazon. Correll et al. comparent différentes méthodes de couverture comme l'échange d'information topologique par radio, la recherche aléatoire et des formes de couverture auto-organisées [Correll 08].

C- *La localisation d'une cible* :

Une multitude d'applications utilisent les essaims de robots pour trouver une cible. Les méthodes diffèrent selon la nature de la cible, sa localisation ainsi que son environnement. Hayes et al. présentent une applications où un essaim de robots est utilisé pour localiser une source d'odeur [Hayes 02]. De même, les essaims de robots aussi utilisés dans les sauvetages en mer pour localiser un naufragé.

D- *La manipulation collective :*

L'avantage essentiel d'un essaim de robots est le collectif, où l'ensemble de robots coopèrent pour exécuter une tâche impossible pour un seul robot. Dans la littérature on peut trouver plusieurs exemples où un essaim de robots s'entraide pour effectuer une manipulation physique comme le transport. Nous citons les applications suivantes : le transport collectif d'un objet [Kube 00], le transport d'une agrégation d'objets [Martinoli 99], le transport d'une ségrégation d'objets [Wilson 04] et l'extraction de tiges de leurs emplacements [Ijspeert 01].

E- *Le fourragement :*

Certains prétendent que les méthodes de colonies de fourmis et des insectes sociaux étaient les précurseurs des méthodes d'intelligence en essaim et de la robotique en essaim plus particulièrement. Le lien entre ces méthodes est bien perceptible, dans les travaux sur le fourragement, où un essaim est appelé à localiser des cibles et à les rasembler en un seul point. Cette application fait penser aux fourmis cherchant leur nourriture et la ramenant à la niche. [Krieger 00] et [Rybski 07] ont comparé différents types de communications appliquées dans une telle situation.

2.2. Discordances entre les caractéristiques des robots en essaim et les spécificités des robots IWARD

2.2.1. Principes de la robotique en essaim

Mark Millonas s'est inspiré des algorithmes des colonies des fourmis pour proposer cinq principes qui caractérisent l'intelligence par essaim [Millonas 93], qui sont également valables pour la robotique en essaim :

Le principe de proximité : les robots/entités doivent pouvoir interagir avec leur environnement, répondre aux signaux/stimuli émis d'autres membres du groupe ou de l'environnement même. Cela implique que les robots doivent avoir une certaine faculté de calcul et de prise de décision, afin de répondre à ces signaux et de favoriser l'action qui sera la plus bénéfique pour le groupe.

Le principe de qualité : la notion de prise de décision, sous entend un objectif recherché et des critères de qualité à respecter. Dans le cas de l'intelligence en essaim ce serait un critère de qualité pour l'essaim (choix et qualité de la source de nourriture dans le cas de recherche de nourriture par un essaim).

Le principe de réponse diversifiée : Afin de mieux faire face aux différents aléas et aux changements environnementaux, l'essaim doit pouvoir diversifier ses réponses et inspecter un ensemble de possibilités.

Le principe de stabilité : Afin de minimiser la perte d'énergie et d'effort, l'essaim doit éviter les basculements systématiques d'un état à un autre, suite à des changements mineurs. Certaines précautions évitant le basculement doivent être mises en place. Ce principe complète le principe précédent.

Le principe d'adaptabilité : L'essaim doit pouvoir changer son mode de fonctionnement, pour s'adapter avec une nouvelle situation dans le cas où le mode de fonctionnement actuel n'est plus satisfaisant.

On Résume les propriétés et caractéristiques des robots en essaim par:

- des unités simples à reproduire, interchangeables, voire même jetables.
- des unités redondantes, dans certains cas quasi-identiques, ce qui permet au système d'être plus fiable face à des perturbations et diverses pannes.
- ces unités possèdent des facultés d'adaptation avec l'environnement.
- ces unités sont dotées de capacités de calcul pouvant résoudre des problèmes complexes.

2.2.2. Caractéristiques du projet IWARD et exigences de la logistique hospitalière.

Le but du projet IWARD est de développer un système de robots mobiles de taille moyenne qui peuvent être équipés de modules additionnels, reliés par un système de communication sans faille, coordonné par des stratégies distribuées/ hybrides pour atteindre les comportements collectifs souhaités, et capable d'effectuer différents types de missions en même temps dans le contexte de l'hôpital afin d'optimiser des indicateurs de performances importants. Les technologies « Plug and play » sont élaborées afin que les modules puissent être facilement insérés et retirés des robots de base par des infirmières ne possédant pas la moindre connaissance de la robotique et des modules.

Partant d'une étude préliminaire des besoins des hôpitaux et des spécifications, le projet IWARD a choisi de cibler six scénarios : (i) la livraison des médicaments, des notes, des films de rayons X et des objets personnels, (ii) le nettoyage régulier ou non planifié lors de déversements, des chambres des patients (iii) la guide des patients et des visiteurs à l'intérieur d'un hôpital, (iv) la surveillance des patients et des intrus, (v) la consultation virtuelle, (vi) la surveillance environnementale.

2.2.3. Les différences entre les robots IWARD et les robots en essaim

Les principales caractéristiques de la robotique en essaim classique ne s'accordent pas avec les exigences de la logistique hospitalière considérées dans le cadre du projet IWARD. Nous montrons dans le tableau ci-dessous les différences entre le concept classique et les exigences adoptées.

Robots en Essaim	Exigences de robots de service en milieu hospitaliers
Travail collectif sur une seule tâche.	Variétés de missions simultanées la livraison, le nettoyage…
Nombre important de robots “physiques” relativement simple.	Nombre limité de robots « physiques ».
Le comportement collectif souhaité est issu des interactions inter-robots et les interactions avec l'environnement.	Les robots doivent se coordonner en respectant les exigences du milieu hospitalier.
Utilisation uniquement d’une communication locale.	Des communications sans faille (sans échec) sont une nécessité.
Tâches demandant une miniaturisation extrême.	Besoins de robots de taille moyenne.
Tâches demandant une conception à prix extrêmement réduit.	Des robots à prix raisonnables.

Tableau 2: Différences entre les robots en essaim et les robots IWARD

Cette analyse dévoile des inadéquations importantes entre le concept classique des robots en essaim et les spécificités des robots de services pour la logistique hospitalière. Ceci nous amène à implémenter notre propre concept de groupe de robots qui répond mieux aux éxigences de l'assistance robotique en milieu hospitalier. Cependant nous nous inspirons des techniques de coordination par enchères souvent utilisées par les robots en essaim pour le pilotage en temps réel de notre système robotique.

2.3. Problèmes de localisation

Les problèmes de localisation ont été largement étudiés. De nombreux chercheurs se sont intéressés aux problèmes de localisation des usines et de configuration des réseaux de distribution au niveau domestique ou global. Alfred Weber était parmi les premiers, à étudier les effets de l’emplacement des usines entre les clients et les fournisseurs de matières premières sur les couts de transport de marchandises [Weber 1909]. Nous donnons ci-après les principaux modèles de location des sites étudiés.

2.3.1. Le modèle P-median

Hakimi était l’un des premiers à formuler un problème de localisation [Hakimi 64] comme un programme linéaire. Il a introduit le modèle P-Median sur l'ouverture de P usines parmi un ensemble J de sites candidats, afin de minimiser le coût total de transport entre les usines et les clients. Les demandes (clients) ont été regroupées par zones de demande I. Soit μ_i le volume de demande du client i et d_{ij} le coût unitaire de transport entre le site j et le client i. Les variables de décisions sont X_j et Y_{ij} où :

$$X_j = \begin{cases} 1 \text{ si le site } j \text{ est choisi} \\ 0 \text{ si non} \end{cases}$$

$$Y_{ij} = \begin{cases} 1, \text{ si le client } i \text{ est affecté à l'usine } j \\ 0, \text{ si non} \end{cases}$$

$$Minimize \sum_{i \in I} \sum_{j \in J} \mu_i d_{ij} Y_{ij} \quad (2\text{-}1)$$

Sous contraintes:

$$\sum_{j \in J} Y_{ij} = 1, \qquad \forall i \in I \quad (2\text{-}2)$$

$$\sum_{j \in J} X_j \leq P \quad (2\text{-}3)$$

$$Y_{ij} \leq X_j \qquad \forall j \in J, \forall i \in I \quad (2\text{-}4)$$

$$X_j, Y_{ij} \in \{0,1\} \qquad \forall j \in J, \forall i \in I \quad (2\text{-}5)$$

La fonction objectif (2-1) minimise les coûts de transport entre les usines et les clients. Les contraintes (2-2) garantissent l'affectation de chaque client à une usine. La contrainte (2-3) limite le nombre des usines à ouvrir. Les contraintes (2-4) assurent que les clients seront affectés à une usine choisie et les contraintes (2-5) sont les contraintes d'intégrité des variables de décisions.

En introduisant le problème, [Hakimi 64] a initialement proposé de résoudre le problème par simple énumération exhaustive. Comme le problème a été prouvé NP difficile [Garey 79], des méthodes dédiées par séparation et évaluation (*Branch and Bound*) ont été proposées ([Maranzana 64], [Efroymson 66], [Jarvinen 72], …). D'autres chercheurs ont proposé des méthodes basées sur des heuristiques. On trouve des heuristiques simples telles que les heuristiques de permutation ([Teitz 68], [Densham 92]). Des heuristiques plus sophistiquées et plus performantes ont également été proposées à l'aide de la relaxation Lagrangienne, la génération de colonnes, la recherche tabou ou les algorithmes génétiques ([Galvao 93], [Beasley 93], …). Une étude de ces heuristiques peut être trouvée dans le livre « Network and Discret Location » [Daskin 95].

2.3.2. Le modèle P-center

Le modèle P-Center est une extension du problème P-Median. Dans ce modèle le but est de minimiser la distance maximale entre les clients et les centres de distribution (usines) au lieu de minimiser la somme des couts de transport. Ce modèle est souvent utilisé pour localiser les ambulances ou bien des centres de soins afin de répondre aux besoins urgents dans les plus brefs délais. Les variables de décision sont celles du modèle P-Median, plus une nouvelle variable réelle W qui représente la distance maximale entre les sites choisis et les clients.

$$Minimize W \quad (2\text{-}6)$$

Sous contraintes :

$$\sum_{j \in J} Y_{ij} = 1, \qquad \forall i \in I \quad (2\text{-}7)$$

$$\sum_{j \in J} X_j \leq P \quad (2\text{-}8)$$

$$Y_{ij} \leq X_j \qquad \forall j \in J, \forall i \in I \quad (2\text{-}9)$$

$$\sum_{j \in J} d_{ij} Y_{ij} \leq W \qquad \forall i \in I \qquad (2\text{-}10)$$

$$W \geq 0; X_j, Y_{ij} \in \{0,1\} \qquad \forall j \in J, \forall i \in I \qquad (2\text{-}11)$$

La fonction objective (2-6) a pour but de minimiser la distance maximale. Les contraintes (2-7), (2-8) et (2-9) sont les même que (2-2), (2-3) et (2-4). Les contraintes (2-10) permettent de déterminer la distance maximale entre les sites choisis et les clients.

Hakimi a proposé une modélisation de ce problème en 1965 [Hakimi 65], Kariv et al. ont démontré la NP-difficulté [Kariv 79]. Les méthodes de résolution proposées dans la littérature sont similaires à celles du modèle p-médian ([Minieka 70], [Mirchandani 79], [Martinich 88], [Handler 90], [Pullan 08]).

2.3.3. Localisation de sites avec coûts fixes

Cette famille de modèles, intitulés « problème de localisation de sites avec coûts fixes » (*Fixed charge Facility Location Problem*), est caractérisée par la prise en compte des coûts fixes pour l'ouverture d'un site correspondant aux coûts de la construction ou de l'acquisition d'un site. Elle se décline en deux sous-familles : « *Uncapacitated Fixed charge Facility Location Problem (UCFLP)* » (Localisation de sites à capacité illimitée avec coûts fixes) et « *Capacitated Fixed charge Facility Location Problem (CFLP)* » (Localisation de sites à capacité limitée avec coûts fixes).

Localisation de sites à capacité illimitée avec coûts fixes

Le problème *UCFLP* aussi connu sous le nom de « *Uncapacitated Facility Location (UFL)* » a été formulé dans [Cornuéjols 90] comme suit :

$$Minimize \sum_{j \in J} f_j X_j + \alpha \sum_{j \in J} \sum_{i \in I} \mu_i d_{ij} Y_{ij} \qquad (2\text{-}12)$$

Sous contraintes :

$$\sum_{j \in J} Y_{ij} = 1, \qquad \forall i \in I \qquad (2\text{-}13)$$

$$Y_{ij} \leq X_j \qquad \forall j \in J, \forall i \in I \qquad (2\text{-}14)$$

$$X_j, Y_{ij} \in \{0,1\} \qquad \forall j \in J, \forall i \in I \qquad (2\text{-}15)$$

où f_j est le coût fixe d'implantation du site j et α est la coût de transport d'une unité de produit pour une unité de distance.

La fonction objective (2-12) a pour objectif la minimisation des coûts d'implantation des sites et des couts de transport. Le paramètre α est lié à l'amortissement des investissements liés aux coûts fixes et permet donc de comparer les coûts fixes et les coûts de transport qui sont des coûts opérationnels. Les contraintes (2-13), (2-14) et (2-15) sont identiques aux contraintes (2-2), (2-4) et (2-5) du modèle P-Median. Dans ce modèle on a repris toutes les contraintes du

modèle P-Median sauf celle limitant le nombre de sites (2-3) qui est remplacée par des coûts fixes.

ADD [Kuehn 63] et DROP [Feldman 66] sont les deux premières heuristiques pour ce modèle UFL. ADD part d'une configuration vide et ajoute chaque fois un seul nouveau site de manière à réduire le coût total le plus possible. DROP part d'une configuration où tous les sites potentiels sont choisis et retire chaque fois un site permettant une diminution la plus importante possible du coût total.

On trouve aussi dans la littérature différentes méthodes applicables au problème comme la relaxation ([Geoffrion 74], [Galvao 93] et [Daskin 95]), la recherche tabou ([Al-Sultan 99], [Sun 06]), des heuristiques de voisinage ([Hansen 98], [Ghosh 03]).

Localisation de sites à capacité limitée avec coûts fixes

Dans ce modèle CPLP, *Capacitated Plant Location Problem*, les capacités de production / stockage des sites sont considérées en plus des contraintes du modèle *UFL.* Ce modèle a été formulé pour la première fois par [Balinski 61] comme suit :

$$Minimize \sum_{j \in J} f_j X_j + \alpha \sum_{j \in J} \sum_{i \in I} \mu_i d_{ij} Y_{ij} \qquad (2\text{-}16)$$

Sous contraintes :

$$\sum_{j \in J} Y_{ij} = 1, \qquad \forall i \in I \qquad (2\text{-}17)$$

$$\sum_{i \in I} \mu_i Y_{ij} \le MAX_j X_j \qquad \forall j \in J \qquad (2\text{-}18)$$

$$Y_{ij} \le X_j \qquad \forall j \in J,\ \forall i \in I \qquad (2\text{-}19)$$

$$X_j, Y_{ij} \in \{0,1\} \qquad \forall j \in J,\ \forall i \in I \qquad (2\text{-}20)$$

où MAX_j est la capacité maximale du site j et les autres notations sont celles du modèle UFL.

Les modèles UFL et CPLP ont la même fonction objectif (2-12) et (2-16). Les contraintes (2-17), (2-19) et (2-20) du CPLP sont équivalentes aux contraintes (2-13), (2-14) et (2-15) du modèle UFL. La seule différence entre ces deux modèles sont les contraintes (2-18) qui limitent la production/ stockage des sites dans le modèle CPLP.

On trouve dans la litérature des méthodes exactes fondées sur la séparation et l'évaluation (Branch and Bound) ([Efroymson 66] et [Spielberg 69]). Sridharan a proposé de résoudre le problème en utilisant la relaxation lagrangienne [Sridharan 95]. Daskin a dressé un état de l'art complet des algorithmes développés pour résoudre le CPLP dans l'ouvrage [Daskin 95].

2.3.4. Les problèmes de recouvrement

Dans le cadre de localisation des services des soins et notamment pour le problème de localisation des ambulances, la notion d'urgence est primordiale par rapport aux coûts de transport. L'objectif principal est de pouvoir satisfaire une demande quelconque en deçà d'une

limite de temps donnée. En présence d'une telle limite de temps, un fournisseur de service en particulier une ambulance ne peut couvrir que les demandes dans un certain rayon afin de respecter le temps de réponse requis.

Problème de la couverture totale

Le problème de couverture totale « *Location Set Covering Model (LSCM*) » a été introduit par [Toregas 71]. Ce modèle consiste à déterminer le nombre de fournisseurs de service nécessaire pour couvrir toutes les demandes. Dans le cadre de notre étude, il est envisageable d'utiliser ce modèle pour calculer le nombre de robots à affecter à un service donné, afin de répondre à toutes les demandes urgente reçues.

Le problème de la couverture totale est le suivant :

$$Minimize \sum_{j \in J} f_j X_j \qquad (2\text{-}21)$$

Sous contraintes :

$$\sum_{j \in J} b_{ij} X_j \geq 1 \qquad \forall i \in I \qquad (2\text{-}22)$$

$$X_j \in \{0;1\} \qquad \forall j \in J \qquad (2\text{-}23)$$

où b_{ij} est un paramètre tel que :

$$b_{ij} = \begin{cases} 1 \text{ si le site } j \text{ peut repondre au besoin de } i \text{ dans l'intervalle de temps requis} \\ 0 \text{ si non} \end{cases}$$

La fonction objectif (2-21) a pour but de minimiser les coûts d'implantation des fournisseurs de service. Les contraintes (2-22) garantissent la couverture de tous les sites. Les contraintes (2-23) sont des contraintes d'intégrité.

Dans certains cas, un client doit être couvert par plusieurs fournisseurs de services. Alors nous remplaçons la borne inférieure des contraintes (2-22) par le nombre requis de fournisseurs.

Garfinkel et al. ont proposé une méthode basée sur l'algorithme de séparation et évaluation (*Branch and Bound)* [Garfinkel 72], Feo et al. ont proposé d'utiliser la méta-heurisitque GRASP (*Greedy Randomized Adaptative Search Procedures*) pour résoudre ce problème [Feo 95]. Beasly et al. ont proposé de résoudre ce problème en utilisant les algorithme génétiques [Beasley 96] ainsi qu'en utilisant la relaxation lagrangienne [Beasley 90]. Et plus récemment Ricardo et al. ont utilisé les colonies de fourmis pour pour localiser les antennes de diffusion [Silva 01]

Ce modèle peut être étendu pour couvrir d'autres applications, Daskin, Jones et Lowe l'ont utilisé pour un problème de sélection d'outils dans les ateliers flexibles [Daskin 90], et Desrochers et al. l'ont appliqué sur le problème de gestion de personnels dans les compagnies aériennes [Desrochers 91].

Problème de couverture maximale

Evidement, le problème de couverture maximale « *Maximal Covering Location Problem (MCLP) »* utilise aussi la notion de couverture. Dans ce modèle le nombre de fournisseurs de service est limité à *N* et le but est de couvrir le plus de demandes possibles. Church et Revelle étaient les premiers à modéliser formellement ce problème [Church 74]. Afin de modéliser ce problème, nous utilisons les mêmes notations que les modèles précédents, et nous introduisons en plus une nouvelle variable de décision Z_i où :

$$Z_i = \begin{cases} 1 \text{ si la zone de demande } i \text{ est couverte} \\ 0 \text{ si aucun fournisseur de service n'est assez proche} \end{cases}$$

Le modèle MCLP peut être formulé de la façon suivante :

$$Maximize \sum_{i \in I} \mu_i Z_i \qquad (2\text{-}24)$$

Sous contraintes :

$$Z_i \leq \sum_{j \in J} b_{ij} X_j \qquad \forall i \in I \qquad (2\text{-}25)$$

$$\sum_{j \in J} X_j \leq N \qquad (2\text{-}26)$$

$$Z_i, X_j \in \{0;1\} \qquad \forall i \in I, j \in I \qquad (2\text{-}27)$$

La fonction objectif (2-24) a pour but de maximiser les zones de demande (clients) qui sont servies (couverts) par un site (fournisseur de service). Les contraintes (2-25) assurent qu'une zone de demande ne peut pas être considérée comme servie (couverte) que si l'un des sites dans le rayon de couverture est ouvert (opérationnel). La contrainte (2-26) limite le nombre de sites opérationnels à *N*, et les contraintes (2-27) sont des contraintes d'intégrité.

A partir de ce modèle plusieurs modèles ont émergés, essentiellement dans le cadre des problèmes de localisation d'ambulance. Dan un modèle dit de couverture avec solution de secours « *Backup coverage problem* », Hogan et Revelle ont introduit les modèles *BACOP1* et *BACOP2*. Dans ces modèles, une zone de demande est doublement couverte par deux ambulances, afin qu'une ambulance puisse répondre si l'autre est déjà occupée [Hogan 86]. Schilling et al. ont introduit les modèles FLEET et TEAM où deux types de véhicules (fournisseur de service) existent avec deux rayons de couverture [Schilling 79]. Gendreau et al. ont introduit le modèle double standard « *Double Standard Model (DSM)* » où chaque fournisseur a deux rayons de couverture et le but est de maximiser le nombre de zones de demande doublement couverts [Gendreau 97]. Une extension stochastique du MCLP a été introduite par Daskin [Daskin 83], pour prendre en compte les probabilités d'indisponibilité des fournisseurs de service (ambulance). Ce modèle est connu sous le nom du « *Maximal Expected covering Location Problem (MEXCLP)* ».

Chapitre 3. Modèle formel et architecture décisionnelle

Le projet IWARD vise à développer une approche robotique capable de répondre dans un même temps à différents besoins logistiques en milieu hospitalier tel le nettoyage, le transport, le guidage, la téléconférence. Nous nous focalisons sur la couche « swarm/ team » (essaim / équipe) dont le but est d'optimiser l'utilisation de l'ensemble des robots pour répondre au mieux à différentes demandes en logistique et réagir à différentes perturbations.

Ce chapitre présente un modèle formel du système robotique considéré. Ce modèle permet d'identifier les différents composants du système, leur caractéristiques, leur comportement statique et dynamique. Il permet aussi de caractériser les différentes activités/et missions demandées aux robots et leur exigences.

Le modèle formel est décliné en deux sous-modèles : le modèle statique et le modèle dynamique. Le modèle statique représente les différents constituants du système ainsi que leurs caractéristiques statiques qui n'évoluent pas au cours du temps. Ce modèle statique permet donc de comprendre la constitution du système robotique que nous allons étudier dans cette thèse. Le modèle dynamique décrit les comportements dynamiques des différents composants du système et les données importantes ou variables d'état de ces constituants dont l'évolution influe fortement sur le comportement du système robotique.

3.1. Modèle Statique

Dans cette thèse, nous considérons un système composé de *N* robots de base et *M* fonctionnalités ou modules permettant d'avoir différentes configurations des robots. L'environnement hospitalier est considéré à travers des points de stationnement pour les robots et des points d'intérêt. Le modèle statique caractérise également les missions pouvant être confiées aux robots. Nous formulons par la suite les caractéristiques de ces composants de notre système robotique, nécessaires pour le pilotage et la coordination du système.

3.1.1. Caractéristiques des robots

Un robot de base que nous appelons simplement « robot » est une plate-forme de base sur laquelle tous les modules fonctionnels seront montés. Les principales caractéristiques de la plateforme de base sont :

- L'identificateur unique de la base, qui est une façon de distinguer les différents robots.
- La vitesse par défaut et la vitesse maximale du robot de base (Le projet IWARD a proposé une méthode issue de l'intelligence artificielle pour estimer la vitesse moyenne d'un robot, mais par manque de ressources cette méthode n'a pu être adoptée.
- L'autonomie de la batterie (en kW) et la consommation d'énergie du robot de base dans les 2 états : en attente et en mission.
- Le temps de recharge du robot.

Exemple en XML :

```
<RobotBase>
```

```
        <idRobot>2</idRobot>
        <Speed> 1.6 </ Speed >
        <MaxSpeed> 1.6 </ MaxSpeed >
        <IdleAutonomy> 480 </ IdleAutonomy >
        <BusyAutonomy > 480 </ BusyAutonomy >
        <RechargingTime> 150 </RechargingTime>
</RobotBase>
```

3.1.2. Fonctionnalités et Modules

Une des caractéristiques du projet IWARD est la modularité des robots : les différents modules peuvent être facilement branchés ou débranchés d'un robot. Chacun de ces modules fournit une ou plusieurs fonctionnalités comme le nettoyage de routine, le nettoyage des déversements, la surveillance de l'environnement, l'orientation des patients et le transport.

Les caractéristiques principales de ces modules et qui doivent être prises en compte par le système robotique (essaim / groupe) sont:

- L'identifiant unique de chaque module.
- Le type ou la catégorie du module.
- Les différentes fonctionnalités fournies par ce module.
- La consommation d'énergie et l'autonomie d'alimentation (le cas particulier de l'autonomie du module de nettoyage doit être pris en compte)
- L'influence sur la vitesse du robot.
- La liste des modules ne pouvant pas être montés sur le même robot.
- La procédure de montage et le degré de difficulté pour le montage et le démontage du module, précisant si la présence d'un technicien qualifié est nécessaire pour l'opération de montage.

Exemple en XML :

```
<Module>
        <idModule> 2 </ idModule >
        <listOfTaskAccomplisable>
                <taskType>2</taskType>
        </listOfTaskAccomplisable>
        <ConflictingModules>
                <idModule>1</idModule>
        </ConflictingModules>
        <MainBatteryConsumption> 50 </MainBatteryConsumption>
        <OwnBatteryAutonomy>500</OwnBatteryAutonomy>
        <PowerSource> Own Battery </ PowerSource >
        <IntelligentModule> Yes </IntelligentModule>
        <CanBeMountedAutomaticly>Yes</CanBeMountedAutomaticly>
        <CanBeDismountedAutomaticly>No</CanBeDismountedAutomaticly>
</Module>
```

3.1.3. Configurations

La configuration d'un robot doit respecter les contraintes suivantes :

- Le nombre maximal de modules pouvant être montés sur le robot.
- La compatibilité des modules.

En raison du petit nombre de modules possibles sur un robot (2 à 4), une énumération de toutes les configurations possibles semble être raisonnable.

Exemple en XML :

```
<Configuration>
      <idConfiguration> 3 </idConfiguration>
      <ConfigurationsElements>
            <idModule>2</idModule>
            <idModule>5</idModule>
      </ConfigurationsElements>
</Configuration>
```

3.1.4. Points de stationnement

A- Station de recharge et de configuration (C/R)

La plupart des robots doivent être reconfigurés afin de répondre aux changements de demandes durant une journée. De même, l'autonomie limitée de la batterie d'un robot oblige un rechargement régulier de sa batterie, le rendant indisponible sur des périodes de rechargement relativement longues (plusieurs heures). Il est alors nécessaire de redistribuer les modules des robots qui se rechargent sur d'autres robots actifs. Cela motive le choix d'un même lieu pour la recharge et la configuration.

Durant le projet, nous avons également envisagé l'utilisation de batteries amovibles, ce qui permettrait d'accroître le nombre de robots disponibles, puis ce qu'il il suffit de remplacer une batterie épuisée par une autre chargée. Les tests techniques avec les batteries amovibles n'étaient cependant pas concluants.

La station de recharge et de reconfiguration (station C /R) est représentée par les données suivantes:

- L'identifiant unique de la station C/R. (Généralement, il y a une seule station C/R par étage)
- L'emplacement de la station (cordonnées x, y ou bien identifiant de l'emplacement)
- Le nombre de chargeurs disponibles
- Le nombre maximal de robots pouvant être hébergés dans une station donnée.

Exemple en XML :

```
<CRStation>
      <idCRStation> 3 </idCRStation>
      <Capacity> 2 </Capacity>
      <NbCharger>1</NbCharger>
      <idLocation > 2 </idLocation>
</CRStation>
```

B- Station d'attente

La station d'attente est l'endroit où les robots retournent après avoir terminé leurs tâches, dans l'attente de l'arrivée d'autres tâches.

Une station d'attente peut être modélisée ou représentée à l'aide des données suivantes:

- L'identifiant unique de la station d'attente
- L'emplacement de la station d'attente
- La capacité maximale de stationnement ou le nombre maximal de robots pouvant y être stationnés

Exemple en XML:

```
<HomeStation>
      <idHomeStation> 3 </idHomeStation>
      <Capacity> 2 </Capacity>
      <idPOI > 2 </idPOI>
</HomeStation>
```

3.1.5. Points d'intérêt

Les points d'intérêt sont des repères connus du système robotiques comme les chambres, les couloirs, la pharmacie, … Ils permettent d'identifier les lieux de missions, les positions et les trajectoires des robots.

La formulation d'un point d'intérêt est décrite par les données suivantes:

- Identification du point d'intérêt
- Coordonnées X et Y du point
- Nom formel du point d'intérêt
- Nom du département ou du service auquel appartient le point d'intérêt

Exemple possible en XML :

```
<POI>
      <idPOI>2</idPOI>
      <POIxCoord>150</ POIxCoord>
      < POIyCoord>230</ POIyCoord>
      <POIdepartement>Pharmacy</POIdepartement>
      <POIName>Home Station 1 – Pharmacy</POIName>
</POI>
```

3.1.6. Caractéristiques des missions

Le projet IWARD considère cinq familles ou types de missions: la livraison, le nettoyage, la téléconférence, la surveillance et le guidage des patients. Ces missions nécessitent des modules différents et donc une configuration adaptée aux missions des robots.

Chaque type de mission peut être modélisé ou représenté en utilisant les données suivantes :

- Identifiant du type de mission
- Nom attribué à de ce type de mission
- Liste des missions incompatibles pour les exécutions en parallèle
- Description formelle des interruptions possibles aussi bien que les procédures de prise en charge de ces interruptions

Exemple en xml d'une classe/ type de missions :

```
<TaskType>
```

```
        <idTaskType >2</idTaskType>
        <Label> cleaning </Label>
        <IncompatibleTasks>
              <idTaskType >2</idTaskType>
              <idTaskType >5</idTaskType>
              <idTaskType >1</idTaskType>
        </IncompatibleTasks>
        </Interruptions>
  </TaskType>
```

Les robots doivent accomplir des missions demandées par le personnel hospitalier. Ces demandes peuvent être régulières comme pour les missions de nettoyage de routine ou imprévues comme une mission de nettoyage suite à un accident qui peut avoir lieu à n'importe quel moment de la journée.

Dans cette thèse, une mission est caractérisée par les informations suivantes :

- Identifiant unique de la mission
- Type de mission
- Fonctionnalités ou modules exigés
- Lieu de départ et lieu de fin de mission
- Date de début au plus tôt, date de début souhaitée et date de début au plus tard
- Date de fin souhaitée et date de fin au plus tard
- Priorité de la mission
- Personnes à contacter en début et fin de mission
- Durée estimée de la mission
- Possibilité d'interruption et d'abandon de la mission
- Trajectoire de la mission (chemin, points d'intérêt à visiter)
- Mission mono-robot ou multi-robots

Exemple en XML :

```
  <Task>
        <idTask> 88 </idTask>
        <taskType>2</taskType>
        <idStartLocation> 2 </idStartLocation>
        <idEndLocation> 5 </idEndLocation>
        <PreferredStartingTime>12:10:02</PreferredStartingTime>
        <LatestStartingTime>12:25:02</LatestStartingTime>
        <DueDate>12:59:59</DueDate>
        <LatestDueDate >13:09:59</LatestDueDate>
        <DueDate>13:59:59</DueDate>
        <Priority>2<Priority>
        <RequestingUser>2</RequestingUser>
        <ReceivingUser>3</ReceivingUser>
        <Schedulled> Yes </Schedulled>
        <CanBeInterruped>Yes</CanBeInterruped>
        <CanCollaborate>No</CanCollaborate>
        <MissionPath>
              <idLocation>12</idLocation>
              <idLocation>5</idLocation>
              <idLocation>2</idLocation>
              <idLocation>10</idLocation>
        </MissionPath>
        <SingleRobotTask>Yes</SingleRobotTask>
        <CanBeDelegated>Yes</CanBeDelegated>
        <ExpectedMissionDuration>45</ExpectedMissionDuration>
        <idStartingEvent>4</idStartingEvent>
```

```
        <idEndEvent>5</idEndEvent>
    </Task>
```

3.2. Modèle dynamique

L'une des particularités de notre projet est la notion de « *Shared knowledge* », (connaissances partagées) sorte de base de données distribuée qui est mise à jour en permanence par la couche de communication. Cette base de données contient les informations statiques du système robotique présenté précédemment mais aussi les données dynamiques du système

Le pilotage du système robotique présenté dans cette thèse repose sur les informations suivantes : état des robots, état des missions, état des modules, plan de navigation des robots, base de connaissances partagées « Shared Knowledge ».

3.2.1. Etats des robots

Le comportement dynamique d'un robot peut être décrit par le diagramme d'états-transitions de la Figure 5 où l'on distingue quatre cycles de fonctionnement. Plus généralement, un robot possède les quatre états suivants :

(i) Etat d'attente. Le robot est alors soit sans missions connues soit en attente de la prochaine mission programmée à une date lointaine.
(ii) Etat de travail, en exécutant une mission ou même plusieurs en parallèle.
(iii) Etat de reconfiguration, soit pour ajouter soit pour supprimer des modules au robot.
(iv) Etat de rechargement de ses batteries pour pouvoir exécuter de futures missions.

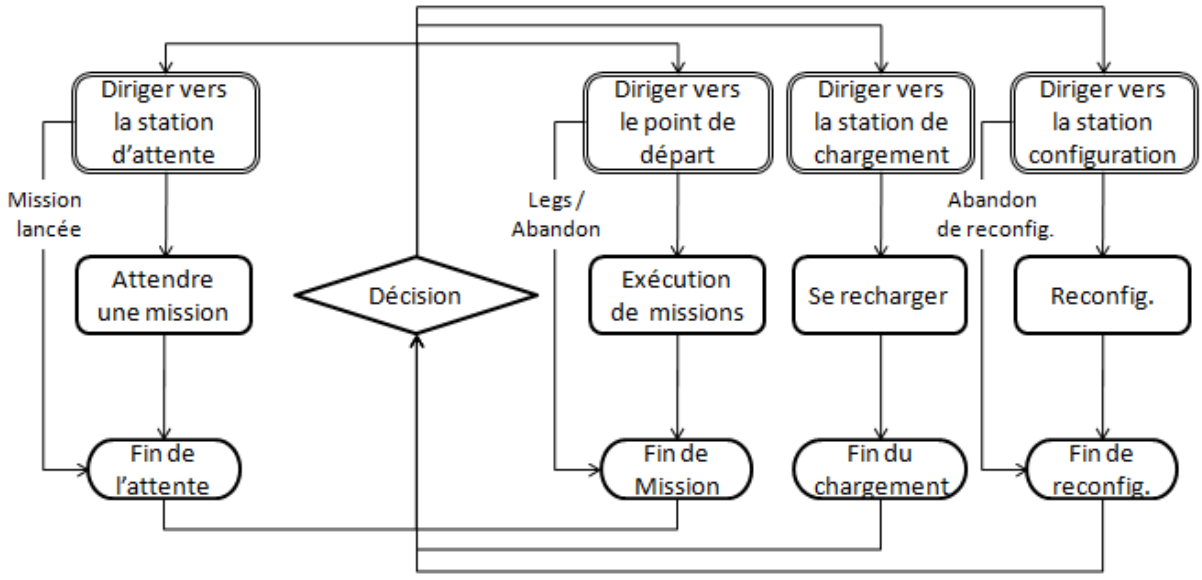

Figure 5 : Diagramme d'états-transitions des robots

En plus de ces états, d'autres données dynamiques concernant les robots sont nécessaires pour trouver la meilleure coordination. Ces informations sont synthétisées par la liste suivante :

- emplacement ou localisation exacte du robot,
- liste des tâches affectées au robot,

- niveau d'énergie restante dans les batteries du robot,
- état de la tâche en cours (En exécution, En conflit, Abandon ...),
- charge de travail affectée au robot,
- modules attribués au robot,
- station d'attente assignée au robot,
- liste des robots à proximité. Une méthode se basant sur la technologie du Bluetooth, devait être implémentée pour déterminer les robots sont dans un certain diamètre autour du robot.

3.2.2. Etats des Missions

La Figure 6 est le diagramme d'états-transitions d'une mission. Le processus démarre par la réception d'une nouvelle mission, ensuite l'agent de décision responsable de l'affectation des missions, en prenant en compte les caractéristiques de la mission ainsi que l'état des robots, affecte la mission au(x) robot(s) le(s) plus approprié(s).
Une fois la mission affectée, le composant responsable de l'exécution des missions ; muni des caractéristiques de la demande ainsi que le type de la mission ; instancie une squelette de missions du type adéquat. Cette instance est mise à jour au fur et à mesure de l'avancement de la tâche.

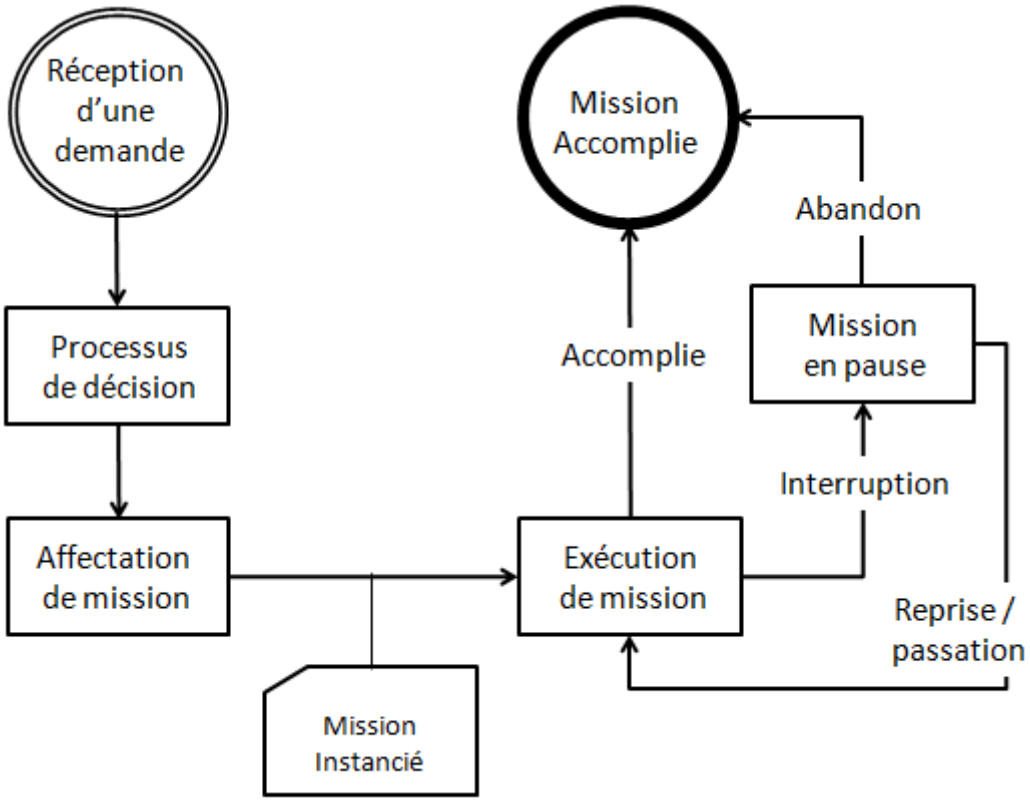

Figure 6 : Graphe d'état-transitions d'une mission

Les principales données dynamiques concernant l'état d'une mission sont ;

- Etat de la mission,
- Etape en cours d'exécution,

- Emplacement actuel du robot exécutant la mission,
- Direction vers laquelle se dirige le robot pour accomplir l'étape suivante,
- Estimation du temps restant de l'étape actuelle, ainsi qu'une estimation de la date de fin de la mission. (Ces informations devaient être mise à jour par un agent d'apprentissage artificiel)

Un exemple d'une instance dynamique d'une mission est illustré par l'entité XML suivante :

```
<MissionRuntimeObject>
      <idMission>88</idMission>
      <idTask>9</idTask>
      <missionType>spillage_clean</missionType>
      <currentMissionStep>spillCleanStartCleaning </currentMissionStep>
      <currentNearestPOI>2</currentNearestPOI>
      <nextPOI>3</nextPOI>
      <StartDate>13:59:59</StartDate >
      <ExpectedMissionDuration>45</ExpectedMissionDuration>
      <EndDate/>
      <MissionState>Mission Execution</MissionState>
      <currentExecutionRobot>3</currentExecutionRobot>
</MissionRuntimeObject>
```

Notons quatre principaux états d'une mission : (i) L'état instancié, où la mission est planifiée et attend la date de début pour être exécuté (ii) L'état d'exécution, où la mission est exécutée par le robot, ensuite soit la mission est interrompue et elle sera en (iii) état de pause, soit elle est achevée et elle sera en (iv) état terminé. On note qu'à partir de l'état de pause, la mission peut, soit être reprise par le même robot, soit être transmise à un autre robot, soit être abandonnée.

3.2.3. Etats des modules

La base de connaissances distribuées « *Shared knowledge* »(connaissances partagées) contient les informations sur l'état des modules. Les données dynamiques d'un module sont synthétisées par la liste suivante :

- Etat du module soit en stock, soit monté sur un robot,
- Emplacement physique du module, s'il n'est pas monté sur un robot,
- Identifiant du robot sur lequel est monté le module, s'il n'est pas stocké.

3.2.4. Plan de navigation dynamique

Le plan de navigation dynamique sert à décrire l'état de l'environnement, des chemins et des liaisons ainsi que l'emplacement actuel de chaque robot. Il est caractérisé par les informations suivantes :

- Etat de chaque chemin (arc reliant deux points d'intérêt POI).
- Durée moyenne estimée pour traverser chaque chemin.
- Emplacement actuel de chacun des robots.
- Destination de chacun des robots.
- Dernier emplacement connu de chaque robot.

3.2.5. Connaissances partagées « Shared Knowledge »

Le tableau suivant, réunit les entités dynamiques ainsi que statiques qui figurent dans la base de connaissance partagée « *Shared Knowledge ».*

Plan:
• *Schedule (Plan) for each robot and*
Robots:
• *Actual_Configuration: modules mounted on each robot*
• *Preferred_Configuration: configuration preferred by plan*
• *Home station: current home station*
• *Scheduled recharging time: next scheduled battery charging time*
• *Missions :*
○ *missions requested*
○ *missions assigned*
○ *missions in progress*
○ *current step / submission*
○ *next target/location of the robot*
• *Location status: actual location of the robot given by the navigation module*
• *Battery status: remaining power level provided by power monitoring device of the base robot*
Missions:
• *List of all missions ordered including their status*
Modules:
• *Location of the module*
• *Busy or idle state of the module*
• *On which robot it is mounted in case of busy status*

3.3. Evénements

Le système robotique implémenté repose sur une architecture de décision réactive, qui inspecte les événements et par conséquence prend des décisions. La compréhension des événements est une étape cruciale. La détection d'un événement incite une série de décisions et déclenche une série d'actions pouvant changer dans certain cas l'état du système robotique.

Nous pouvons classer les événements en 2 catégories : les événements initiés par les acteurs humains (demande d'une nouvelle mission) et les événements initiés par un robot suite à l'interprétation des données recueillies par les différents capteurs du robot (détection d'une personne allongée par terre ou arrivée à une destination).

Nous pouvons aussi classer les événements selon leur impact sur le système robotique. Deux classes d'événements sont considérées: les événements globaux affectant l'ensemble des robots et les événements locaux n'affectant que le robot lui-même.

Nous considérons les événements globaux suivants qui doivent être pris en compte par l'ensemble des robots:

(i) l'arrivée d'une nouvelle demande à exécuter,
(ii) la réaffectation d'une tâche,
(iii) la remise en cause d'une des tâches,
(iv) l'achèvement d'une mission,
(v) le départ d'un robot du système robotique,
(vi) l'intégration d'un nouveau robot,
(vii) la fin d'une période et le début d'une nouvelle période,
(viii) les événements issus des agents d'apprentissage, où un robot informe ses pairs, des informations qu'il a découvert et qui pourrait être utiles pour les autres membres de l'équipe, exemple : un obstacle bloquant un chemin.

Les événements locaux suivants sont considérés :

(i) l'affectation d'une tâche,
(ii) le démarrage d'une mission,
(iii) l'interruption d'une tâche.

Dans certains cas, un événement local peut déclencher une série d'actions pouvant engendrer des événements globaux.

3.4. Architecture de décision

Dans cette thèse, nous nous limitons aux décisions suivantes :

- La gestion des plannings de rechargement des robots,
- La gestion de la configuration des robots et des modules,
- La localisation des stations d'attente des robots,
- L'affectation des missions aux différents robots,
- La planification et l'ordonnancement des missions affectées,
- La gestion et le contrôle de l'exécution des missions,
- La gestion de l'exécution des missions en parallèle,
- La gestion des interruptions de mission,
- La détection de la possibilité de passation de missions,
- La gestion locale d'énergie.

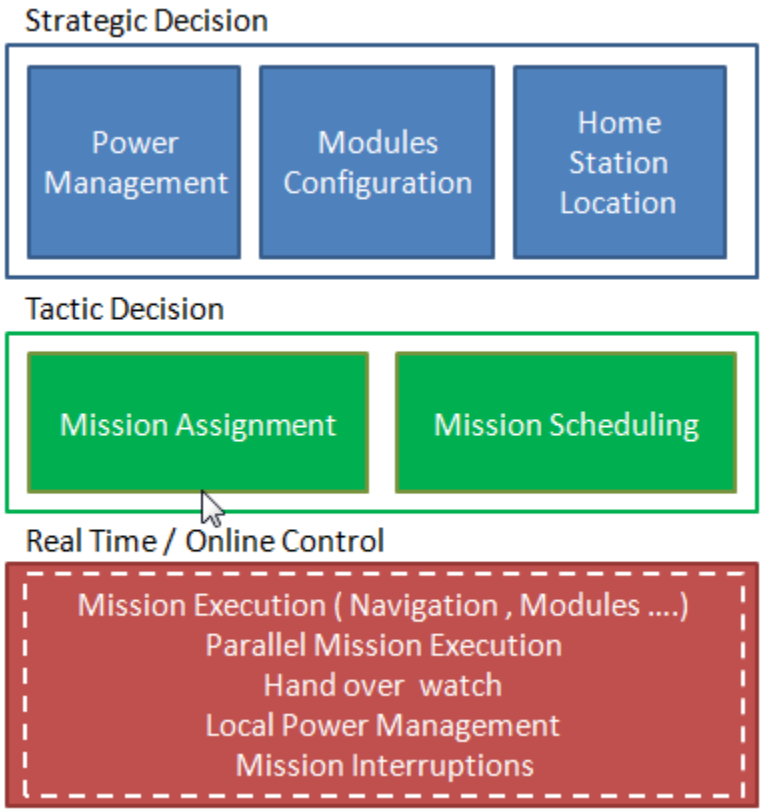

Figure 7 : Trois niveaux de décision

La Figure 7 illustre notre architecture de décision en trois nivaux :

a) Le niveau stratégique,
b) Le niveau tactique,
c) Le niveau opérationnel et temps réel.

3.4.1. Niveau stratégique

Les décisions du niveau stratégique, illustrées par la Figure 8 et considérées dans cette thèse, sont :

- La gestion des plannings de rechargement des robots. Sachant que les robots sont équipés de batteries fixes nécessitant des durées de rechargement assez conséquentes, une bonne coordination des rechargements des robots est nécessaire afin d'assurer une disponibilité suffisante des robots dans le temps.
- La gestion des configurations des robots. Une des particularités du robot IWARD est la modularité permettant de modifier les fonctionnalités d'un robot en ajoutant ou supprimant des modules. Malgré la simplicité de cette opération, ces reconfigurations nécessitent une intervention humaine, d'où la nécessité d'un agent de décisions pour indiquer les configurations des robots actifs, tenant compte des besoins à venir.
- La localisation des stations d'attente. Afin de répondre efficacement et rapidement aux futures demandes, les robots doivent être correctement positionnés dans l'hôpital pour être proches des lieux de missions.

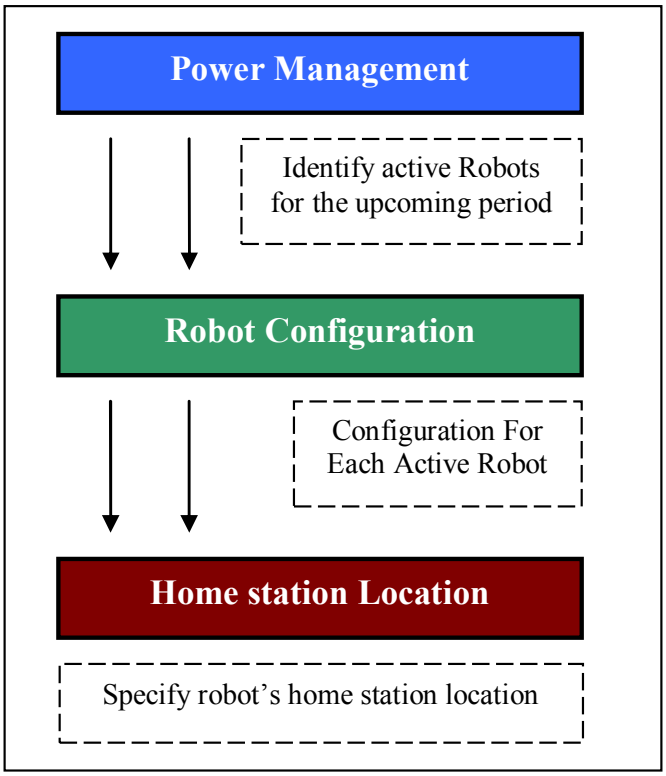

Figure 8 : Niveau stratégique

Un des points communs de ces agents de décisions stratégiques, c'est que le temps et l'horizon de planification, durant lesquels ces décisions sont prises, sont relativement longs et concernent généralement plusieurs périodes consécutives voire même une ou plusieurs journées. Ces décisions ne devront pas être remises en cause fréquemment. Généralement, les changements des décisions stratégiques coïncident avec le changement de postes qui accompagne généralement un changement des demandes de missions.

Afin d'établir ces décisions, l'état global du système robotique doit être considéré : notamment les robots, leur état, leur niveau d'énergie restant, les modules montés ainsi que les modules disponibles en stock, l'état des missions en cours ainsi qu'une prévision des futures missions. Toutes ces informations doivent être prises en considération afin d'établir, pour la prochaine période, quels robots vont se recharger et quels robots sont opérationnels, et établir pour chacun des robots opérationnels la configuration qui lui sera attribuée ainsi que la station d'attente qui lui sera assignée.

3.4.2. Niveau tactique

Au niveau tactique, nous nous intéressons aux problèmes d'affectation des missions aux robots et d'ordonnancement des missions sur chaque robot.

Ces décisions doivent prendre en compte les charges de travail des robots, l'état des robots, les recommandations du niveau stratégique, et les configurations des robots. Ces décisions doivent être prises assez rapidement, surtout pour les missions urgentes, afin de ne pas retarder leur exécution.

Les missions multi-robots ne sont pas prises en compte directement et nous supposons qu'elles sont décomposées en plusieurs sous-missions mono-robot que nous pouvons affecter séparément à différents robots. Néanmoins un système de compatibilité de missions est mis en place afin de ne pas affecter deux sous-missions au même robot.

Deux types de missions sont considérés : les missions régulières et connues en avance qui sont programmées régulièrement (transport régulier de médicaments, nettoyage programmé) et les missions imprévues qui ne sont pas connues en avance.

Deux approches de prise de décisions sont considérées : décisions centralisées et décisions distribuées.

3.4.2.1. Prise de décision centralisée

La caractéristique principale de cette approche est que les décisions sont prises par un agent central. Sachant que ces agents doivent s'occuper de l'affectation et de l'ordonnancement des missions, il est essentiel que ces agents soient informés en permanence de l'état des robots et des missions. La prise en compte de toutes ces informations complique le problème et risque de retarder la prise de décision, ce qui peut se traduire par un retard d'exécution des missions. En revanche, vu que ces agents ont une vue globale, la solution proposée peut être l'optimale.

Ce système peut être avantageux notamment dans le cas où la majorité des missions sont des missions régulières connues en avance, ce qui permet d'établir une affectation et un ordonnancement a priori des missions, et de disposer suffisamment de temps pour la prise de décision.

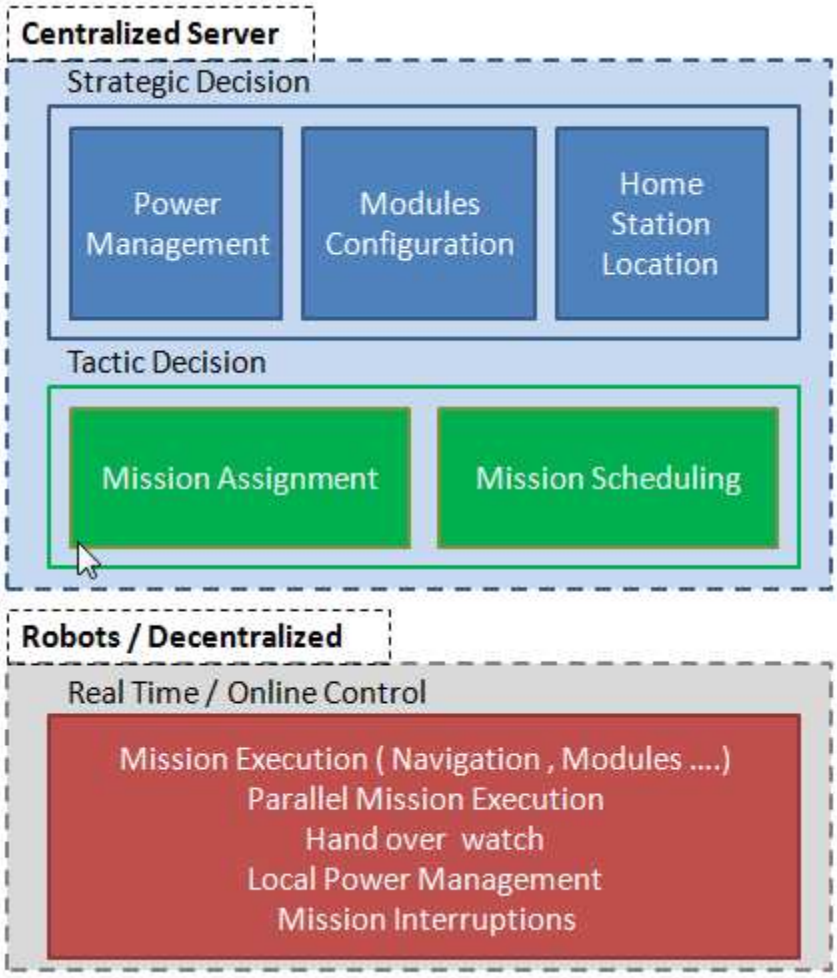

Figure 9: Approche centralisée de décisions tactiques

3.4.2.2. Prise de décision décentralisée

L'utilisation d'un agent central de décisions rend le système sensible aux défaillances des robots et des systèmes de communication. Pour cela, nous proposons également une approche de décision décentralisée. Le système décentralisé profite des capacités de calcul de chaque robot ainsi que de leurs facultés de communication. Dans ce système les robots sont indépendants, et les décisions sont prises en concertation avec les autres robots, le système implémenté repose sur des techniques dites d'enchères inversés et de négociation.

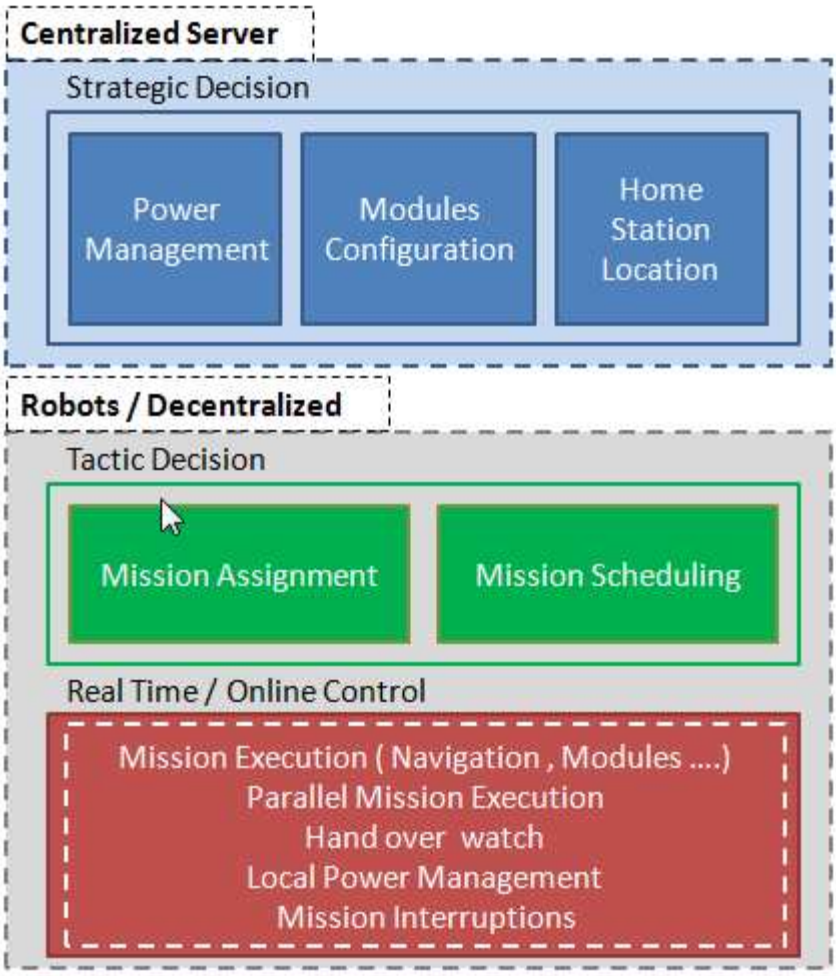

Figure 10 : Décisions tactiques décentralisées.

Afin de tester l'efficacité du système décentralisé, ce système est comparé à l'aide d'une simulation avec un système centralisé basé sur les algorithmes génétiques.

D'autres systèmes hybrides peuvent être envisagés, notamment un système d'affectation centralisé couplé avec un système d'ordonnancement local propre à chaque robot. Une autre possibilité serait d'utiliser le système centralisé pour les missions régulières, lors de l'arrivée de nouvelles missions urgentes ces décisions seront reconsidérées par un système décentralisé. Ce système décentralisé jouera alors un rôle réparateur qui s'adapte aux aléas en temps réel.

3.4.3. Niveau opérationnel et temps réel

Ce niveau gère l'exécution des missions. Le composant principal de ce niveau est l'exécuteur, qui a pour rôle d'exécuter les ordres du niveau tactique, et de veiller au bon déroulement des missions. L'exécuteur doit interagir avec les éléments robotiques, analyser les données reçues par les capteurs, identifier les événements qui doivent être transmis au niveau tactique, et transmettre des ordres aux éléments robotiques afin d'exécuter les différentes étapes des missions. D'autres composants sont conçus pour identifier des opportunités d'optimisation. Le système de prospection de missions parallèles permet d'identifier les missions pouvant être

exécutées en parallèle par le même robot. Un autre système détecte la possibilité de transmettre la mission en-cours d'un robot à un autre aux alentours afin que le premier puisse être libéré pour une mission plus urgente.

Première partie : Niveau Stratégique

Chapitre 4. Planification stratégique du système robotique

Ce chapitre aborde les décisions stratégiques définies dans l'architecture de décisions du chapitre 3, en particulier la gestion d'autonomie des batteries, la configuration des robots et la localisation des robots. Ces décisions sont prises sur un horizon d'un ou plusieurs postes de travail (shifts) afin de prendre en compte la variation des activités dans le temps. Ce chapitre commence par une définition détaillée des problèmes relatifs à ces trois grandes décisions. Nous proposons ensuite trois approches fondées sur des modèles de programmation mathématique. Ces trois approches sont ensuite comparées à travers différentes instances numériques.

4.1. Les trois sous problèmes de la planification stratégique

4.1.1. Gestion d'autonomie des batteries

Un des problèmes majeurs identifiés durant la première réunion d'évaluation du projet est la durée relativement conséquente de rechargement des batteries, et la situation, dans laquelle tous les robots se vident de leur énergie et se rechargent ensembles durant la même période, est à éviter. Cette situation devrait être évitée afin de garder des robots opérationnels pour assurer la continuité d'assistance au personnel en cas de missions urgentes.

Plusieurs solutions ont été proposées afin de remédier à ce problème d'autonomie, certaines par le groupe hardware, en proposant d'augmenter les capacités des robots en ajoutant des batteries supplémentaires afin de prolonger la durée d'activité des robots ou bien d'installer des stations de rechargement automatique qui permettent aux robots de se recharger sans intervention humaine durant les périodes creuses où les robots sont en veille en attendant l'arrivé de nouvelles missions.

Notre groupe « swarm intelligence » s'est également penché sur ce problème et a proposé de planifier les périodes de rechargement de chaque robot afin d'assurer un service minimum nécessaire tout au long de la journée.

L'agent de décision consacré à la gestion de l'autonomie des robots, doit gérer et planifier les cycles de rechargement, et s'assurer de la disponibilité d'un nombre suffisant de robots opérationnels durant les futures périodes pour exécuter les futures missions.

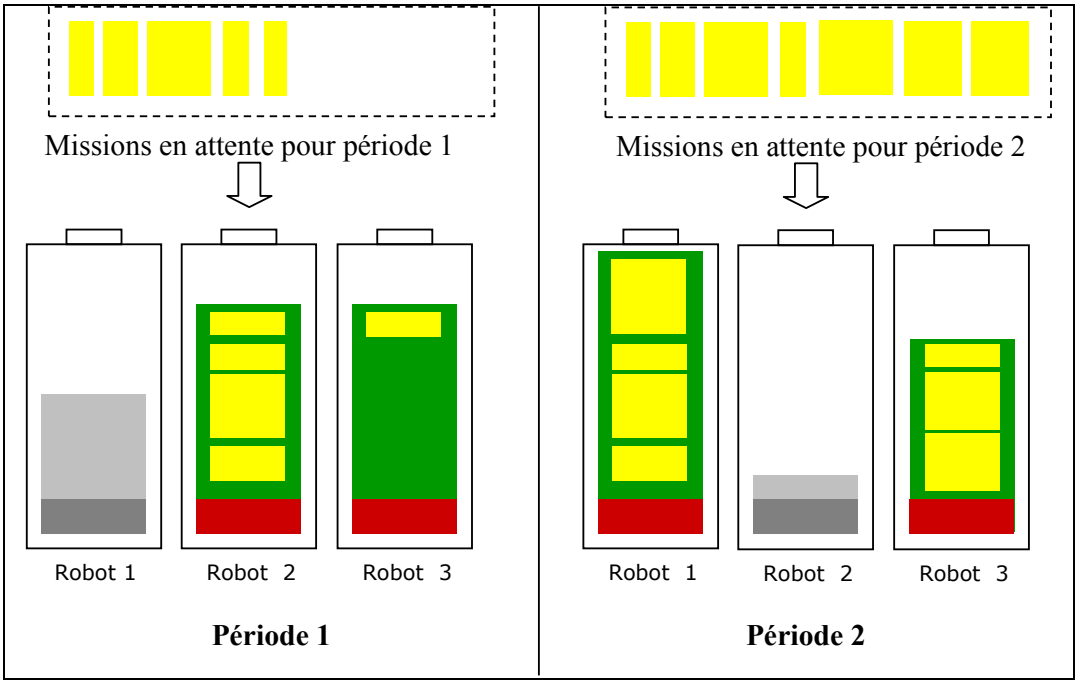

Figure 11 : Gestion de l'autonomie des robots

Les caractéristiques de cet agent de décision sont :

Évènement déclencheur : Le processus de planification est déclenché lorsqu'une différence majeure est constatée entre les demandes prévues et les demandes observées qui rendent les plans calculés obsolètes, ainsi que le début d'une tranche horaire où une nouvelle équipe de travail prend le relais.

Nature du processus de prise de décision : Comme la gestion d'autonomie doit prendre en compte l'état de l'ensemble des robots et la prévision de l'ensemble des demandes, un processus de décision centralisé semble être le plus adapté pour une telle situation.

Horizon de planification: compte tenu des temps de chargement importants (plus de 8h) et de l'objectif d'assurer la disponibilité des robots, l'horizon de planification doit s'étendre sur plusieurs périodes consécutives.

Données nécessaires : Les paramètres qui doivent être pris en compte pour le calcul des plans de chargement sont : le nombre de robots, la capacité de stockage d'énergie des batteries embarquées sur les robots , le niveau d'énergie de chaque robot, la vitesse de rechargement de chaque robot, la vitesse moyenne de déchargement des robots en mode opérationnel et en mode veille, les prévisions de demande et l'énergie nécessaire pour accomplir ces demandes. La configuration des robots et les modules attachés aux robots peuvent être pris en considération pour le calcul de l'affectation des tâches si ces informations peuvent être disponibles à ce stade de calcul.

Décisions : A la fin du calcul, l'agent de décision doit établir, pour chaque période prise en considération, un plan de rechargement spécifiant les robots destinés à se recharger, ainsi que la charge affectée à chaque robot opérationnel.

4.1.2. Configurations des robots

Une des innovations du projet IWARD est l'ensemble des fonctionnalités/ modules « *plug and play* » (brancher et faire fonctionner) qui ressemblent à des tiroirs et qui peuvent être ajoutés/ enlevés à chaud par n'importe quel personnel disponible. Si par exemple il a été décidé de reconfigurer un robot en lui ajoutant la fonctionnalité de transport de médicament, il suffit qu'un(e) infirmièr(e) prenne le tiroir/module qui correspond à cette fonctionnalité et le glisse dans l'un des endroits prévus. Comme les demandes d'assistance varient dans une journée et que certaines demandes sont périodiques, les robots doivent être reconfigurés différemment durant la journée. La configuration d'un robot nécessite une intervention humaine et doit donc être planifiée de préférence à des moments précis comme lors des changements de postes. Puisque la configuration d'un robot détermine sa charge, il est préférable de planifier les configurations des robots en fonction des prévisions de demandes sur plusieurs périodes. Ce problème peut aussi être couplé avec le problème de l'autonomie des robots, présenté précédemment, afin que la capacité énergétique des robots et le type des missions affectées soient pris en considération lors de la configuration ainsi que l'affectation des missions.
Le but de l'agent de décision, responsable de la reconfiguration, est d'analyser la configuration actuelle des robots, et de les reconfigurer en fonction des demandes futures et de la disponibilité des fonctionnalités.

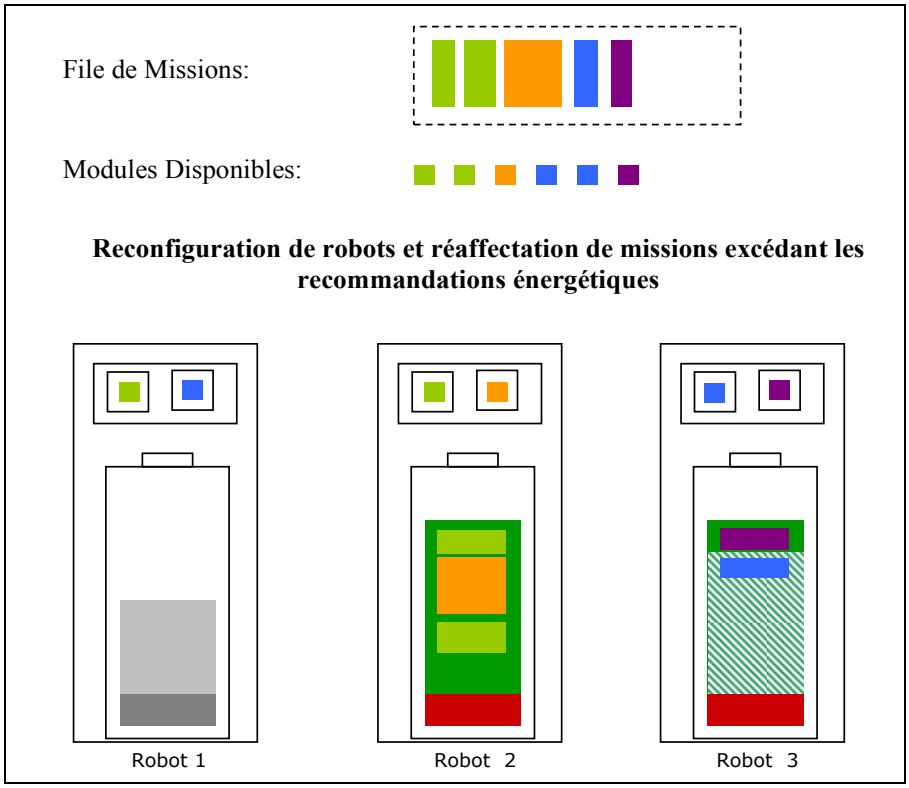

Figure 12 : Configuration et autonomie des robots.

Les caractéristiques de cet agent de décision sont les suivantes :

Evénement déclencheur : Le processus de reconfiguration est déclenché soit au commencement d'une nouvelle période, ce qui indique le changement des équipes de travail ainsi que le changement des demandes du personnel ; soit un changement de ressources

disponibles qui est généralement un changement de robots opérationnels suite à un changement des plans de rechargement des robots.

Horizon de planification : L'horizon de planification peut être limité à une seule période ou étendu à plusieurs périodes. Le processus de reconfiguration sera déclenché à la fin de chaque période.

Nature du processus de prise de décision : Sachant que ce processus doit tenir compte de l'état de tous les robots du système, et que les robots doivent être configurés afin d'assurer une disponibilité globale par rapport à la prévision de demandes, un processus de type centralisé semble plus pertinent pour effectuer cette tâche.

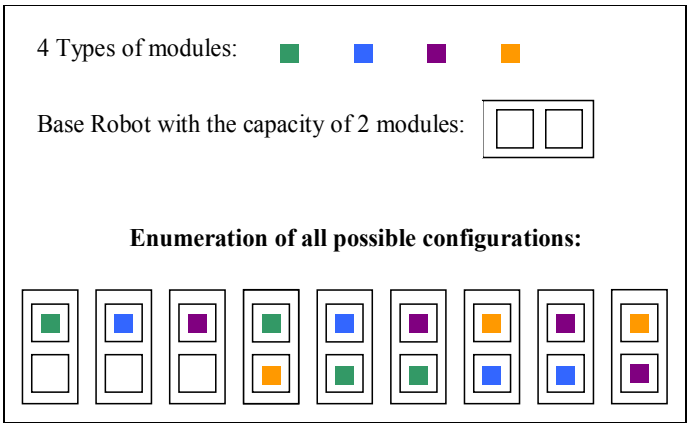

Figure 13 : Enumération des configurations possibles d'un robot ayant 2 tiroirs disponibles

Données nécessaires : Les données essentielles pour effectuer le calcul de reconfiguration sont : le nombre de robots opérationnels ainsi que la capacité énergétique de chaque robot, le nombre des fonctionnalités disponibles ainsi que leur type, l'ensemble de configurations possibles pour chaque robot, les missions prévues, ainsi que leur type et la consommation énergétique nécessaire pour exécuter ces missions.

Décisions : Les décisions prises par cet agent concernent la configuration de chaque robot et les modules qui lui seront affectées.

4.1.3. Localisation des stations des robots

Le positionnement des robots dans l'hôpital influence la réactivité du système robotique. Des stations d'attente doivent être positionnées de manière intelligente dans l'hôpital, ainsi les robots y seront affectés lorsqu'ils sont en état de veille, dans le but de mieux répondre aux demandes urgentes et de minimiser les parcours dans le vide.

Lorsqu'un robot termine toutes les missions qui lui ont été affectées, il repart à sa station d'attente. Cette station doit être choisie en fonction de la proximité des futures missions prévues qui lui seront affectées.
Ces stations d'attente évitent que les espaces vitaux de l'hôpital ne soient encombrés par des robots en état de veille. Ces stations peuvent aussi jouer un rôle de station de rechargement si elles sont équipées de « self docking station » ou station de chargement automatique qui permet aux robots de se ressourcer en énergie sans intervention humaine.

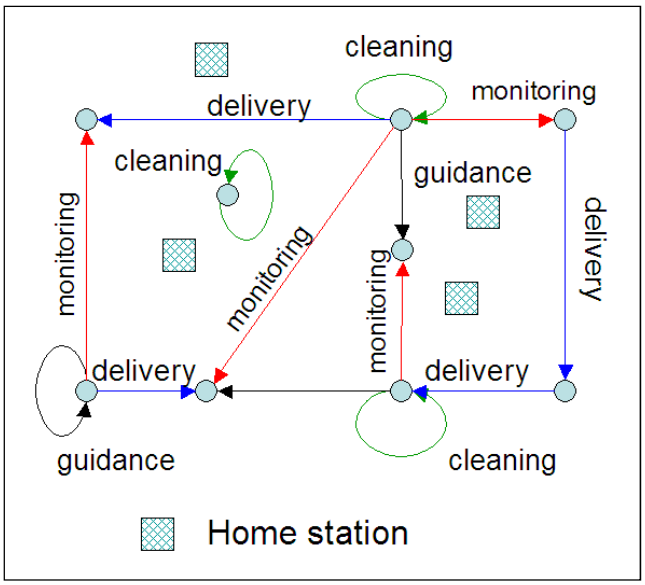

Figure 14: Localisation des stations d'attentes en fonction de futures demandes

Les caractéristiques de l'agent de décisions responsable de la localisation des ces stations d'attente sont :

Evénement déclencheur : Le processus de localisation des stations est déclenché, suite à un changement des ressources existantes (dans notre cas, suite à un changement des robots opérationnels qui peut être engendré par une modification des plans de rechargement des robots). Un changement des fonctionnalités existantes sur les robots peut aussi nécessiter un changement des stations d'attente, par exemple le robot équipé avec le module de transport peut être mieux placé à côté de la pharmacie qui est le premier demandeur de ce module. Également, le processus de localisation peut être déclenché suite à un changement des demandes du personnel, souvent lié à un changement d'équipe durant la journée, ou la passation entre deux équipes qui marque le début d'une nouvelle période et donc nécessite une ré-optimisation du système.

Horizon de planification : Cet agent effectue ses calculs sur une période, en tenant compte des capacités énergétiques des robots.

Nature de l'agent de décision : Vu la nécessité de prendre en compte l'état de l'ensemble des robots et la nécessité de couvrir de manière optimale les demandes, un agent de décision de type centralisé, peut être plus capable d'effectuer l'association entre les robots et les stations d'attente les plus appropriées.

Données nécessaires : l'ensemble des lieux qui peuvent servir de stations d'attente, les futures demandes prévues ainsi que leur nature, les points de départs et d'arrivée de ces missions, la fréquence ou la probabilité d'occurrence de ces missions, les temps d'exécution de ces missions, la matrice du temps nécessaire pour se déplacer d'un point d'intérêt à un autre.

Ce problème peut être couplé avec le problème de configuration et même avec le problème d'autonomie des robots afin de s'assurer que les missions seront affectées à un robot opérationnel proche ayant assez d'énergie stocké pour exécuter sa mission

Décision : Affecter à chaque robot opérationnel une station d'attente vers laquelle il doit se diriger lorsqu'il est en état de veille.

4.2. Planification stratégique intégrée / combinée

L'un des avantages du regroupement des trois problèmes est de pouvoir optimiser de manière intégrée l'ensemble des décisions stratégiques en tenant compte des contraintes d'énergie, de configuration et de localisation. L'inconvénient majeur est la taille importante du problème.

Nous considérons un système robotique composé d'un ensemble N de robots de base, qui sont dotés de batteries ayant une capacité énergétique limitée et dont la durée de rechargement est assez importante. Chaque robot de base peut être configuré différemment en y ajoutant des fonctionnalités. Soit F l'ensemble des types de ces fonctionnalités et N_f le nombre d'exemplaires disponibles de modules du type f.

Désignons par C l'ensemble de configurations possibles pour un robot de base. Cet ensemble peut être déterminé par énumération des différentes combinaisons de fonctionnalités par rapport au nombre d'emplacements possibles sur chaque robot ainsi que la compatibilité des fonctionnalités. Chaque configuration $c \in C$ est définie par un vecteur d'entiers $[a_{cf}]$ indiquant le nombre d'exemplaires de chaque fonctionnalité qui constitue la configuration c.

Soit H l'ensemble des stations d'attente où les robots peuvent attendre l'arrivée de nouvelles missions. En recevant une nouvelle mission, le robot devient opérationnel, quitte la station en se dirigeant vers le point de départ de la mission puis exécute la mission. S'il n a pas d'autres missions affectées, il retourne à sa station d'attente h.

On considère que chaque journée est divisée en plusieurs créneaux $t \in T$. Chaque créneau a une longue T_{max} et un nombre minimal L_t de robots opérationnels pour assurer un service minimal durant la période t.

Chaque robot n est doté d'une batterie. Soit $Q_{n,0}$ le niveau d'énergie initial du robot n, Soit l_n , la quantité d'énergie consommée par un robot n en état de veille durant un créneau t et soit G_n la quantité d'énergie gagnée par le rechargement d'un robot n dans un créneau.

On désigne par $e_{n,m}$ la quantité d'énergie dépensée par un robot n pour exécuter une mission m. Plus précisément, chaque robot n consomme au moins une quantité l_n d'énergie et consomme en plus une quantité $e_{n,m}$ d'énergie additionnelle pour chaque mission exécutée.

Chaque mission m est caractérisée par les paramètres suivants :

- $\square_{m,t}$: fréquence d'occurrence de la mission m durant la période t,
- T_m: durée nécessaire pour exécuter la mission m,
- C_m: ensemble des configurations permettant l'exécution de la mission m,
- A_{hm}: temps nécessaire pour un robot afin d'effectuer le trajet de la station d'attente h jusqu'au point de départ de la mission m,
- R_{hm}: temps nécessaire à un robot pour effectuer le trajet du retour du point de terminaison de la mission m jusqu'à la station d'attente h

Le temps total d'exécution d'une mission est ainsi décomposé en trois temps : temps de réaction A_{hm}, durée d'exécution T_m et temps de retours R_{hm}. Si la mission m est affectée au robot n, la station d'attente qu'il faut considérer pour les calculs de ces durées est la station h à laquelle le robot n est assigné. Ces durées sont calculées en fonction de la vitesse moyenne des robots et de la longueur du trajet.

On désigne par $Q_{n,max}$ le niveau maximal d'énergie que la batterie du robot n peut accumuler, et $Q_{n,mim}$ le niveau minimal d'énergie à ne pas atteindre afin de garder une quantité suffisante d'énergie pour faire face à certains imprévus et pouvoir rentrer se recharger avant que sa batterie ne soit asséchée complètement.

Deux types de charges peuvent être considérés : la charge de travail $\phi_m(A_{hm}+T_m+R_{hm})$ induit par l'exécution d'une mission m et la charge énergétique $\phi_m e_{nm}$. La somme des charges de travail d'un robot pendant une période doit être inferieure à T_{max} qui est proportionnelle à la durée de cette période. La quantité d'énergie restante due à la somme des charge énergétique des missions affecté au robot ne doit pas atteindre le seuil minimal $Q_{n,mim}$ de la batterie.

4.2.1. Modèle mathématique de la planification stratégique intégrée

Les décisions tactiques/opérationnelles comme l'ordonnancement des missions ne sont pas considérés à ce niveau. Les coûts de reconfiguration et de relocalisation des stations d'attente sont supposés nuls. Le rechargement partiel est interdit afin de maximiser la durée de vie des batteries des robots

Le problème présenté consiste à déterminer, pour chaque période, les robots à recharger, et pour les robots opérationnels, une configuration de fonctionnalités attachées au robot ainsi qu'une station d'attente.

Le but ultime est d'exécuter toutes les missions au moindre coût, tout en respectant les contraintes physiques comme le nombre maximal H_h de robots qu'une station h peut accueillir, la disponibilité des fonctionnalités.

Afin de formuler le problème sous forme d'un programme linéaire en nombre entier les variables de décision suivantes sont utilisées:

$$x_{nht} = \begin{cases} 1, & \text{si le robot } n \text{ est stationné en } h \text{ durant la periode } t \\ 0, & \text{sinon.} \end{cases}$$

$$y_{nct} = \begin{cases} 1, & \text{si le robot } n \text{ utilise la configuration } c \text{ durant la periode } t \\ 0, & \text{sinon.} \end{cases}$$

$$z_{nmt} = \begin{cases} 1, & \text{si la mission } m \text{ est affectée au robot } n \text{ durant la periode } t \\ 0, & \text{sinon.} \end{cases}$$

$$w_{nt} = \begin{cases} 1, & \text{si le robot } n \text{ se recharge durant la periode } t \\ 0, & \text{si le robot } n \text{ est opérationnel durant la periode } t. \end{cases}$$

$$u_{mt} = \begin{cases} 0, & \text{si la mission } m \text{ est affectée à un robot durant la période } t. \\ 1, & \text{sinon.} \end{cases}$$

On utilise aussi la quantité $q_{n,t}$ d'énergie à la fin de la période t et la quantité g_{nt} d'énergie acquise par le robot n en se rechargeant durant la période t.

$$Minimize \sum_{t\in T}\sum_{n\in N}\sum_{h\in H}\sum_{m\in M}\phi_{m,t}(\alpha A_{hm} + T_m + \beta R_{hm})r_{nhm,t} + \sum_{t\in T}\sum_{m\in M} p_m \phi_{mt} u_{m,t} \quad (4\text{-}1)$$

sous contraintes:

$$\sum_{h\in H} x_{nh,t} + w_{n,t} = 1, \qquad \forall n \in N; \forall t \in T \quad (4\text{-}2)$$

$$\sum_{c\in C} y_{nc,t} + w_{n,t} = 1, \qquad \forall n \in N; \forall t \in T \quad (4\text{-}3)$$

$$u_{m,t} + \sum_{n\in N} z_{nm,t} = 1, \qquad \forall t \in T; \forall m \in M \quad (4\text{-}4)$$

$$\sum_{c\in C}\sum_{n\in N} a_{cf} y_{nc,t} \le N_f \qquad \forall f \in F; \forall t \in T \quad (4\text{-}5)$$

$$\sum_{n\in N} x_{nh,t} \le H_h, \qquad \forall h \in H; \forall t \in T \quad (4\text{-}6)$$

$$\sum_{h\in H}\sum_{m\in M} \phi_{m,t}(A_{hm} + T_m + R_{hm})r_{nhm,t} \le T_{\max} \qquad \forall n \in N; \forall t \in T \quad (4\text{-}7)$$

$$z_{nmt} \le \sum_{c\in C_m} y_{nct} \qquad \forall m \in M, \forall n \in N, \forall t \in T \quad (4\text{-}8)$$

$$r_{nhm,t} \ge x_{nh,t} + z_{nm,t} - 1 \qquad \forall n \in N; \forall h \in H; \forall m \in M; \forall t \in T \quad (4\text{-}9)$$

$$\sum_{n} w_{n,t} \le N - L_t \qquad \forall t \in T \quad (4\text{-}10)$$

$$Q_{n,t} = Q_{n,t-1} + g_{n,t} - l_n(1 - w_{n,t}) - \sum_{m\in M_t} e_m z_{n,m} \quad \forall n = 1...N; \forall t \in T \quad (4\text{-}11)$$

$$\underline{Q}_{n,\min} \le Q_{n,t} \le \overline{Q}_{n,\max} \qquad \forall n \in N; \forall t \in T \quad (4\text{-}12)$$

$$g_{n,t} \le G_n w_{n,t} \qquad \forall n \in N; \forall t \in T \quad (4\text{-}13)$$

$$x_{nh,t}, y_{nc,t}, z_{nm,t}, w_{n,t}, u_m, r_{nhm,t} \in \{0,1\} \qquad \forall b \in B, \forall m \in M, \forall n \in N, \forall t \in T \quad (4\text{-}14)$$

$$g_{n,t} \ge 0; Q_{n,t} \ge 0 \qquad \forall n \in N; \forall t \in T \quad (4\text{-}15)$$

La fonction objectif (4.1) a pour but de minimiser la charge de travail de l'ensemble des robots en prenant en compte le temps de réaction et le temps nécessaire pour le retour et de minimiser le nombre de missions rejetées ce qui est équivalent à maximiser le nombre de missions affectées aux robots. Les facteurs α et β permettent d'attribuer des priorités aux temps de réaction et de retour et le facteur p_m permet de tenir compte de l'importance globale des missions. Les contraintes (4.2) vérifient que chaque robot opérationnel est affecté à une station d'attente, les contraintes (4.3) vérifient que chaque robot opérationnel est configuré selon une des configurations valides de l'ensemble C des configurations. Les contraintes (4.4) assurent que toute mission est soit affectée à un robot soit rejetée. Les contraintes (4.5)

concernent la disponibilité des fonctionnalités, et que le nombre des fonctionnalités utilisées pour établir les différentes configurations des robots ne dépasse pas la quantité disponible. Les contraintes (4.6) garantissent que le nombre de robots affectés à une station d'attente ne dépasse pas les limites prédéfinies. Les contraintes (4.7) garantissent que la charge maximale de travail d'un robot n'est pas dépassée. Les contraintes (4.8) certifient qu'une mission n'est affectée qu'au robot convenablement configuré.

Les contraintes (4.9) assurent la linéarisation du produit des deux variables de décision $z_{nm,t}$ et $x_{nh,t}$ en utilisant la variable $r_{nhm,t}$. Cette variable de décision $r_{nhm,t}$ est égale à 1 si et seulement durant la période t la mission m est affectée au robot n et que la station d'attente h est désignée comme la station du robot n la. Les contraintes (4.10) vérifient qu'un nombre minimal L_t de robots est disponible dans chaque période t. Les contraintes (4.11) impliquent que le niveau d'énergie à la fin d'une période peut être calculé à partir du niveau d'énergie de la période précédente, de la quantité d'énergie acquise en se rechargeant et de la quantité dépensée soit en état de veille, soit en exécutant les missions affectées au robot durant la période en considération. Les contraintes (4.12) vérifient que le niveau d'énergie emmagasiné sur les robots ne dépasse pas les bornes inférieures et supérieurs recommandées. Les contraintes (4.13) garantissent que les robots opérationnels ne peuvent pas être rechargés partiellement et que la quantité d'énergie acquise par un robot en état de rechargement ne dépasse pas la vitesse de rechargement du chargeur. Les contraintes (4.14) représentent les contraintes d'intégrité des variables de décisions.

La résolution de ce problème en nombres entiers utilisant des solveurs commerciaux de type « Branch and Bound » notamment CPLEX, n'est possible que pour des problèmes de petite taille. Pour cela, nous proposons une méthode de génération de colonne dans le paragraphe suivant.

4.2.2. Génération de colonnes pour la planification intégrée

Les techniques de génération de colonnes ont été introduites par Dantzig et Wolf [Dantzig 60] pour résoudre les programmes linéaires structurés de grande taille. Gilmore et Gomory ont été les premiers à appliquer ces méthodes pour résoudre le problème de cutting stock [Gilmore 61] ; une synthèse des avancées récentes des techniques de géneration de colonnes et de leurs applications peuvent être trouvées dans [Barnhart 98] et [Lubbecke 05].

Afin d'appliquer les méthodes de génération de colonne, une reformulation du problème (1 - 15), permettant de définir le concept de colonne, sera présenté dans cette partie. Dans le cadre de cette étude, une colonne présente l'état d'un robot n durant une période t et spécifie la station d'attente h dans laquelle il sera stationné, sa configuration c et les différentes fonctionnalités qui seront montées sur ce robot, ainsi que l'ensemble des missions qui seront affectées à ce robot.

Chaque colonne p est définie par les variables suivantes :

$$X_h^p = \begin{cases} 1, & \text{si la station } h \text{ est designée comme la station du robot} \\ 0, & \text{sinon.} \end{cases}$$

$$Y_c^p = \begin{cases} 1, & \text{si le robot est configuré selon la configuration } c \\ 0, & \text{sinon.} \end{cases}$$

$$Z_m^p = \begin{cases} 1, & \text{si la mission } m \text{ est affectée au robot pour l'exécuter .} \\ 0, & \text{sinon.} \end{cases}$$

$$W^p = \begin{cases} 1, & \text{si le robot est en état de rechargement .} \\ 0, & \text{sinon.} \end{cases}$$

Evidement, une colonne est faisable si elle est reliée à une seule zone de stationnement, ayant une configuration unique, et que toutes les missions affectées au robot peuvent être effectuées par cette configuration et que les charges de travail et énergétiques ne sont pas dépassées.

Pour une colonne donnée p, la charge/coût peut être définie par l'équation suivante:

$$b^p = \sum_{h\in H}\sum_{m\in M} \phi_m \left(\alpha A_{mh} + T_m + \beta R_{mh}\right) Z_m^p X_h^p \tag{4-16}$$

La nouvelle variable de décision est définie :

$$\lambda^p = \begin{cases} 1, & \text{si la colonne } p \text{ est selectionnée} \\ 0, & \text{sinon.} \end{cases}$$

Ainsi le problème combiné peut être reformulé en un « **problème maître** » :

$$Min \sum_{t\in T}\sum_{n\in N}\sum_{p\in\Omega_{nt}} b^p \lambda^p + \sum_{t\in T}\sum_{m\in M} p_m \phi_{mt} u_{mt} \tag{4-17}$$

où Ω_{nt} est l'ensemble des colonnes faisables du robot n pour la période t et sous contraintes:

$$\sum_{p\in\Omega_{nt}} \lambda^p = 1, \quad \forall n \in N, t \in T \tag{4-18}$$

$$\sum_{n\in N}\sum_{p\in\Omega_{n,t}} Z_m^p \lambda^p + u_{mt} = 1, \quad \forall t \in T, m \in M \tag{4-19}$$

$$\sum_{n\in N}\sum_{p\in\Omega_{nt}} \left(\sum_{c\in C} a_{cf} Y_c^p\right) \lambda^p \le N_f, \quad \forall f \in F, t \in T \tag{4-20}$$

$$\sum_{n\in N}\sum_{p\in\Omega_{nt}} X_h^p \lambda^p \le H_h, \quad \forall h \in H, t \in T \tag{4-21}$$

$$Q_{n,t} - Q_{n,t-1} - g_{n,t} + \sum_{p\in\Omega_{nt}} l_n (1 - W^p)\lambda^p + \sum_{p\in\Omega_{nt}} \left(\sum_{m\in M} \phi_{mt} e_m Z_m^p\right)\lambda^p = 0, \ \forall n \in N, t \in T \tag{4-22}$$

$$g_{nt} \le \sum_{p\in\Omega_{nt}} G_n W^p \lambda^p, \quad \forall n \in N, t \in T \tag{4-23}$$

$$\underline{Q}_n \le Q_{n,t} \le \bar{Q}_n, \ \forall n \in N, t \in T \tag{4-24}$$

$$\sum_{n\in N}\sum_{p\in\Omega_{nt}} W^p \lambda^p \le N - L_t, \quad \forall t \in T \tag{4-25}$$

$$\lambda^p \in \{0,1\}, \forall p \in \Omega; u_{mt} \in \{0,1\}, \forall m \in M, t \in T \tag{4-26}$$

$$g_{n,t} \geq 0; Q_{n,t} \geq 0, \qquad \forall n \in N; \forall t \in T \tag{4-27}$$

De même que dans la formulation précédente, la fonction (4.17) a pour but de minimiser la charge de travail et de maximiser le nombre de missions affectées aux robots. Les contraintes (4.18) vérifient que quelque soit la période et pour n'importe quel robot un état et un seul doit être attribué au robot. Les contraintes (4.19) assurent que toute mission est soit affectée à un robot soit rejetée. Les contraintes (4.20) concernent la disponibilité des fonctionnalités, et assurent que le nombre des fonctionnalités utilisées pour établir les différentes configurations des robots ne dépasse pas la quantité disponible. Les contraintes (4.21) garantissent que le nombre de robots affectés à une station d'attente ne dépasse pas les limites prédéfinies. Les contraintes (4.22) déterminent le niveau d'énergie à la fin d'une période à partir du niveau d'énergie de la période précédente, de la quantité d'énergie acquise en se rechargeant et les la quantité dépensée soit en état de veille, soit en exécutant les missions affectées au robot durant la période en considération. Les contraintes (4.23) garantissent que la quantité d'énergie acquise par un robot en état de rechargement ne dépasse pas la vitesse de rechargement du chargeur. Les contraintes (4.24) vérifient que le niveau d'énergie emmagasiné sur les robots ne dépasse pas les bornes inferieures et supérieurs recommandées. Les contraintes (4.25) vérifient la disponibilité d'un nombre minimal L_t de robots opérationnels. Les contraintes (4.26) représentent les contraintes d'intégrité des variables de décisions.

Le problème maitre (PM) est constitué d'un très grand nombre de colonnes, compliquant la résolution directe de ce problème en nombres entiers. De ce fait, on procède à une relaxation des contraintes d'intégrité (4.26) en les remplaçant par les contraintes suivantes : $0 \leq \lambda_p \leq 1; 0 \leq u_{mt} \leq 1$. Le problème relaxé obtenu est connue sous le nom « Problème Maitre Linéaire » (PML) et ce dernier est résolu en utilisant les méthodes de génération de colonnes. Sachant l'amplitude de l'espace de résolution de problème, on considère une partie de l'espace de calcul en démarrant avec un sous ensemble $\Omega' \subset \Omega$ de colonnes. Ce problème est connu sous le nom du « Problème Maitre Restreints » (PMR) et des nouvelles colonnes sont générées au fur et au mesure, en utilisant des méthodes de Simplex.

Afin de trouver une solution pour le problème PML par génération de colonnes, on démarre avec un PMR avec un nombre réduit de colonnes et le problème PMR est résolu en utilisant la méthode de Simplex qui permet de déterminer la solution optimale du PMR et les valeurs des variables duales du problème PML. Le problème de pricing consiste à déterminer les colonnes ayant les coûts réduits minimaux. Si ces coûts réduits minimaux sont non négatifs, alors la solution optimale du PML est trouvée. Sinon, les colonnes ayant les coûts réduits négatifs sont ajoutées au sous ensemble Ω' et le processus est répété tant qu'on trouve des colonnes avec des coûts réduits négatifs.

Soient $\mu_{nt}, \gamma_{mt}, \pi_{ft}, \eta_{ht}, \nu_{nt}, \tau_{nt}, \delta_t$ les valeurs des variables duales de la solution optimale du PMR correspondant respectivement aux contraintes (4.18, 4.19, 4.20, 4.21, 4.22, 4.23, 4.25).

Le coût réduit d'une colonne relatif au robot *n* en période *t* est donné par la formule suivante :

$$PP(n,t) = \min \sum_{h\in H}\sum_{m\in M} \phi_{m,t}(\alpha A_{mh} + T_m + \beta R_{mh}) X_h^p Z_m^p - \mu_{nt} - \sum_{m\in M} \gamma_{mt} Z_m^p$$
$$-\sum_{f\in F} \pi_{ft} \sum_{c\in C} a_{cf} Y_c^p - \sum_{h\in H} \eta_{ht} X_h^p - \sum_{m\in M} \phi_{mt} e_m Z_m^p \nu_{nt}$$
$$-l_n \nu_{nt} + W_{nt} \nu_{nt} + G_n W_{nt} \tau_{nt} - W_{nt} \delta_t$$

Afin de réduire la complexité du problème de pricing, on le décompose en plusieurs sous problèmes dont chacun concerne une station d'attente h et une configuration c. Pour chaque couple *(h,c)*, on détermine l'ensemble des missions qui génèrent le coût réduit minimum. Plus précisément le sous- problème de pricing consiste à déterminer l'ensemble des missions qui minimisent la formule suivante :

$$SPP(n,t;h,c) = \min \sum_{m\in M} \phi_{m,t}(\alpha A_{mh} + T_m + \beta R_{mh}) Z_m^p - \mu_{nt} - \sum_{m\in M} \gamma_{mt} Z_m^p$$
$$-\sum_{f\in F} \pi_{ft} a_{cf} - \varphi_{ht} - \sum_{m\in M} \phi_{m,t} e_m \nu_{nt} Z_m^p$$

Ce qui est équivalent à:

$$SPP(n,t;h,c) = \sum_{m\in M} [\phi_{m,t}(\alpha A_{mh} + T_m + \beta R_{mh}) - \gamma_{mt} - \phi_{m,t} e_m \nu_{nt}] Z_{nmt}^p - const \qquad (4\text{-}28)$$

Sous les contraintes:

$$\sum_{m\in M} \phi_{m,t}(A_{mh} + T_m + R_{mh}) Z_m^p \le T_{t,\max} \qquad (4\text{-}29)$$

$$Z_m^p = 0 \qquad \text{if } c \notin C_m \qquad (4\text{-}30)$$

$$Z_m^p \in \{0,1\} \qquad \forall m \in M \qquad (4\text{-}31)$$

Notons que les sous- problèmes de pricing *SPP(n,t;h,c)* sont équivalents aux problèmes de sac à dos (knapsack problem), et peuvent être résolus en utilisant les algorithmes de programmation dynamique pseudo-polynomiaux [Martello 90] afin de déterminer pour chaque sous problème *PP(n,t)* le couple *(h,c)* résultant de la colonne ayant le cout réduit le plus bas.

4.3. Planification stratégique en deux phases

Au cours de la deuxième année du projet IWARD, le groupe de travail Hardware a divulgué son intention de développer et d'annexer au robot un système de gestion d'autonomie qui utilise des batteries amovibles assurant la continuité du travail en hôpital. Ce qui a nécessite la dissociation du problème de gestion d'autonomie de celui de la gestion des configurations et de la sélection des stations d'attentes.

Pour cela, nous proposons de planifier le rechargement des robots ; en premier lieu si les robots ne sont pas équipés de batteries amovibles et ensuite de planifier conjointement la configuration des robots et leur localisation.

L'agent de décision responsable de la gestion des configurations et de la sélection des stations d'attente doit être assez flexible. Il doit prendre en compte les décisions prises par l'agent de gestion d'autonomie dans le cas où les robots se rechargent durant des longues périodes. Dans le cas où les robots sont équipés de batteries amovibles, la problématique de l'autonomie ne

sera pas évoquée et n'est pas considérée dans le calcul de gestion des configurations et de localisation.

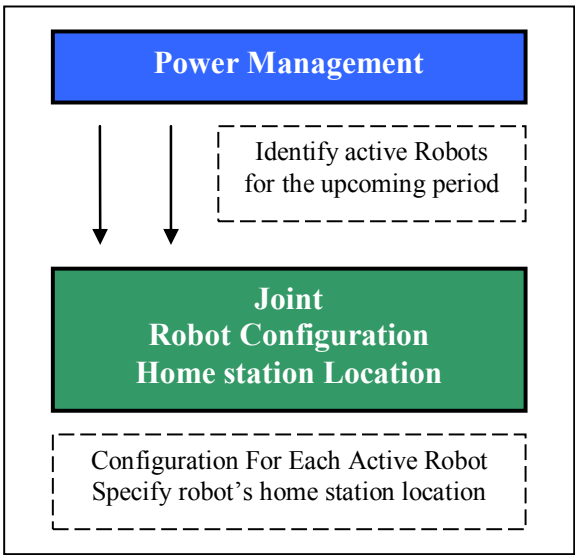

Figure 15 : Planification stratégique en deux phases

4.3.1. Modèle mathématique du problème de gestion d'énergies

Le problème de gestion d'autonomie des robots concerne un système robotique composé d'un ensemble N de robots équipés de batteries fixes ayant une capacité énergétique limitée et dont la durée de rechargement est assez importante.

Les hypothèses suivantes sont considérées :

- Les rechargements partiels sont interdits pour préserver la durée de vie des batteries et les robots ne peuvent pas changer d'état de chargement durant une période. Cette hypothèse exclut le cas où les robots se chargent dans les stations d'attente équipées des chargeurs automatiques.
- Les configurations des robots ne sont pas prises en compte et les missions sont affectées aux robots sans tenir compte des configurations nécessaires.
- La localisation n'étant pas prise en compte, nous ne considérons pas les contraintes des charges de travail des robots.
- Les décisions tactiques comme l'affectation et l'ordonnancement des missions en temps réel ne sont pas prise en compte à ce stade de planification.
- Les courbes de chargement et déchargement sont considérées linéaires entre deux bornes supérieures et inférieures.

Le but de cet agent de décision est d'optimiser le nombre des robots opérationnels durant la journée afin d'avoir des robots de secours pouvant faire face à des pics de demandes occasionnelles, et de maximiser le nombre de missions prises en charge par le système robotique.

On considère que chaque journée est divisée en plusieurs créneaux $t \in T$ et on définie par ϕ_{mt} la fréquence d'occurrence de mission m, durant le créneau t.

Chaque robot n est doté d'une batterie. Soit $Q_{n,0}$ le niveau d'énergie initial du robot n. Une quantité l_n d'énergie est consommée par le robot n en état opérationnel et une quantité additionnelle e_m d'énergie est nécessaire pour chaque mission m exécutée. Le rechargement d'un robot n dans un créneau, permet à ce dernier d'acquérir au maximum une quantité G_n d'énergie.

Afin de modéliser cet agent en un programme linéaire en nombre entiers, les variables de décisions suivantes sont envisagées :

$$v_{n,t} = \begin{cases} 1, \text{ si le robot } n \text{ est opérationnel durant le créneau } t \\ 0, \text{si le robot } n \text{ se recharge durant le créneau } t \end{cases}$$

$$z_{nm,t} = \begin{cases} 1, \text{ si la mission } m \text{ est affectée au robot } n, \text{durant le créneau } t \\ 0, \text{sinon} \end{cases}$$

$$u_{m,t} = \begin{cases} 0, \text{ si la mission } m \text{ pris en charge par un robot, durant le créneau } t \\ 1, \text{sinon} \end{cases}$$

Le problème peut être formulé ainsi

$$Max \sum_{n\in N}\sum_{t\in T} v_{n,t} - \sum_{m\in M} p_m \phi_{m,t} u_{m,t} \qquad (4\text{-}32)$$

Sous contraintes :

$$u_{m,t} + \sum_n z_{nm,t} = 1 \qquad \forall m \in M; t \in T \qquad (4\text{-}33)$$

$$\sum_n v_{n,t} \geq L_t \qquad \forall t \in T \qquad (4\text{-}34)$$

$$z_{nm,t} \leq v_{n,t} \qquad \forall n \in N;\ t \in T;\ m \in M \qquad (4\text{-}35)$$

$$q_{n,t} = q_{n,t-1} - l_n v_{n,t} - \sum_{m\in M_t} \phi_{m,t} e_m z_{nm,t} + g_{n,t}, \forall n \in N; t \in T \qquad (4\text{-}36)$$

$$\underline{Q}_n \leq q_{n,t} \leq \bar{Q}_n \qquad \forall n \in N;\ \ t \in T \qquad (4\text{-}37)$$

$$0 \leq g_{n,t} \leq G_n(1 - v_{n,t}) \qquad \forall n \in N;\ t \in T \qquad (4\text{-}38)$$

$$v_{n,t},\ z_{nm,t} \in \{0;1\}\ , q_{n,t}, g_{n,t} \in \mathrm{R} \quad \forall n \in N;\ t \in T;\ m \in M \qquad (4\text{-}39)$$

La fonction objectif (4.32) a pour but de maximiser le nombre des robots opérationnels, et de maximiser le nombre de missions assignées aux robots ce qui est équivalent à minimiser le nombre de missions rejetées. Les contraintes (4.33) vérifient que toute mission est soit affectée à un robot soit rejetée. Les contraintes (4.34) vérifient que le nombre de robots en état d'opérationnels est supérieur au nombre L_t de robots garantissant un service minimum durant la période t. Les contraintes (4.35) vérifient que les missions sont affectées à des robots opérationnels. Les contraintes (4.36) détermine le niveau d'énergie à la fin d'une période à partir du niveau d'énergie de la période précédente, de la quantité d'énergie acquise en se rechargeant et la quantité dépensée soit en état de veille, soit en exécutant les missions affectées au robot durant la période en considération. Les contraintes (4.37) vérifient que le

niveau d'énergie emmagasiné sur les robots ne dépasse pas les bornes inferieures et supérieurs recommandées. Les contraintes (4.38) garantissent que les robots opérationnels ne peuvent pas être rechargés partiellement et que la quantité d'énergie acquise par un robot en état de rechargement ne dépasse pas la vitesse de rechargement du chargeur. Les contraintes (39) représentent les contraintes d'intégrité des variables de décisions.

Après la résolution de ce modèle à l'aide d'un solveur MIP, on retient pour chaque robot son état au premier créneau ($v_{n,0}$) , en plus pour les opérationnels ($v_{n,0}=1$), on calcule E_n la quantité d'énergie recommandée pour le robot n qui est donnée par la formule suivante : $E_n = \sum_{m \in M} \phi_{m,t} e_m Z_{nm,t}$.

Cependant lors de la seconde phase de calcul, compte tenu du fait que certaines missions ne peuvent pas être exécutées sur certains robots faute de configurations appropriées, on tolère un dépassement par rapport à la recommandation initiale à condition de respecter les limites minimales physiques de la batterie. Et soit $E_{n,max}$ la quantité maximale que le robot n peut dépenser, $E_{n,max}$ est calculée selon la formule suivante : $E_{n,\max} = Q_{n,0} - \underline{Q}_n - l_n$.

4.3.2. Planification intégrée de la configuration et de la localisation des robots

Le modèle mathématique doit prendre en compte les recommandations énergétiques données par le modèle précédent, et relaxer ces contraintes dans le cas où les robots sont équipés par des batteries amovibles.

Le but est de déterminer pour chaque robot n opérationnel, sa station d'attente par défaut et les fonctionnalités qui seront installées sur le robot pour la période à venir.

Rappelons que le système robotique est caractérisé par les notations suivantes :

- F ensemble des fonctionnalités
- N_f nombre d'exemplaires disponibles de la fonctionnalité f
- C ensemble de configurations possibles d'un robot
- a_{fc} nombre d'exemplaires de la fonctionnalité f dans la configuration c
- H ensemble des stations d'attente.

Chaque mission m peut est caractérisée par les paramètres suivants:

- $\phi_{m,}$ fréquence d'occurrence de la mission m
- e_m quantité d'énergie nécessaire pour exécuter la mission m
- T_m temps nécessaire pour exécuter la mission m
- C_m ensemble des configurations permettant l'exécution de la mission m
- A_{hm} temps nécessaire pour un robot d'effectuer le trajet de la station d'attente h jusqu'au point de départ de la mission m
- R_{hm} temps nécessaire pour un robot d'effectuer le trajet du retour du point de terminaison de la mission m jusqu'à la station d'attente h.

Les variables de décisions suivantes sont employées :

$$x_{nh} = \begin{cases} 1, & \text{si le robot } n \text{ est affecté à la station d'attente } h \\ 0, & \text{sinon.} \end{cases}$$

$$y_{nc} = \begin{cases} 1, & \text{si le robot } n \text{ utlise la configuration } c \\ 0, & \text{sinon.} \end{cases}$$

$$z_{nm} = \begin{cases} 1, & \text{si la mission } m \text{ est affectée au robot } n \\ 0, & \text{sinon.} \end{cases}$$

$$u_m = \begin{cases} 0, & \text{si la mission } m \text{ n'est pas prise en charge par un robot} \\ 1, & \text{sinon.} \end{cases}$$

$$r_{nhm} = \begin{cases} 1, & \text{si la mission } m \text{ est affectée au robot } n \text{ stationné en } h \\ 0, & \text{sinon.} \end{cases}$$

La consommation d'énergie recommandée E_n peut être dépassée. Soit s_n le dépassement de cette recommandation et soit χ la pénalité de ce dépassement.

Le problème combiné de configuration et localisation est formulé ainsi :

$$Minimize \sum_{n\in N}\sum_{h\in H}\sum_{m\in M}\phi_m(\alpha A_{hm} + T_m + \beta R_{hm})r_{nhm} + \sum_{n\in N}\chi s_n - \sum_{m\in M} p_m u_m \quad (4\text{-}40)$$

Sous contraintes:

$$\sum_{h\in H} x_{nh} = 1, \qquad \forall n \in N \qquad (4\text{-}41)$$

$$\sum_{c\in C} y_{nc} = 1, \qquad \forall n \in N \qquad (4\text{-}42)$$

$$\sum_{n\in N} z_{nm} + u_m = 1, \qquad \forall m \in M \qquad (4\text{-}43)$$

$$\sum_{c\in C}\sum_{n\in N} a_{fc} y_{nc} \le N_f, \qquad \forall f \in F \qquad (4\text{-}44)$$

$$\sum_{n\in N} x_{nh} \le H_h, \qquad \forall h \in H \qquad (4\text{-}45)$$

$$\sum_{h\in H}\sum_{m\in M} \phi_m(A_{hm} + T_m + R_{hm})r_{nhm} \le T_{\max}, \qquad \forall n \in N \qquad (4\text{-}46)$$

$$\sum_{c\in C_m} y_{nc} \ge z_{nm}, \qquad \forall m \in M, \forall n \in N \qquad (4\text{-}47)$$

$$x_{nh} + z_{nm} \le 1 + r_{nhm} \qquad \forall h \in H, \forall m \in M, \forall n \in N \qquad (4\text{-}48)$$

$$\sum_{m\in M} e_m z_{nm} \le E_n + s_n \qquad \forall n \in N \qquad (4\text{-}49)$$

$$E_n + s_n \le E_{n,\max} \qquad \forall n \in N \qquad (4\text{-}50)$$

$$x_{nh}, y_{nc}, z_{nm}, r_{nhm} \in \{0,1\}, s_n \ge 0; \qquad \forall h \in H, \forall m \in M, \forall n \in N, c \in C \quad (4\text{-}51)$$

La fonction objectif (4.40) a pour but de minimiser le dépassement des recommandations énergétiques des robots ainsi que la charge de travail de l'ensemble des robots. Les contraintes (4.41) vérifient que chaque robot opérationnel est affecté à une station d'attente,

les contraintes (4.42) vérifient que chaque robot opérationnel est associé à une configuration. Les contraintes (4.43) assurent que toute mission est affectée à un et un seul robot. Les contraintes (4.44) garantissent que le nombre des fonctionnalités utilisées et installées sur les robots ne dépasse pas le nombre de fonctionnalités existantes. Les contraintes (4.45) garantissent que le nombre de robots affectés à une station d'attente ne dépasse pas les limites prédéfinies. Les contraintes (4.46) garantissent que la charge maximale de travail d'un robot n'est pas dépassée. Les contraintes (4.47) certifient qu'une mission n'est affectée qu'au robot convenablement configuré. Les contraintes (4.48) assurent la linéarisation du produit $z_{nm}x_{nh}$ en utilisant la variable r_{nhm}. Les contraintes (4.49) déterminent le dépassement de la quantité d'énergie recommandée. Les contraintes (4.50) garantissent les contraintes des limites physiques des batteries. Les contraintes (4.51) représentent les contraintes d'intégrité des variables de décisions.

Dans le cas où ce modèle se révèle sans solution applicable, les contraintes (4.43) sur l'affectation obligatoire des missions sont remplacées par les contraintes (4.33) pour autoriser la non affectation d'une mission. Les coûts de non affectation doivent alors être introduits dans (4-40).

Le cas où les robots sont équipés de batteries amovibles, implique l'omission des contraintes reliées à la gestion de l'autonomie des batteries donc les contraintes (4.49, 4.50) ainsi que les pénalités sur les dépassements énergétiques de la fonction objective (4.40)

Les expérimentations numériques ont révélé certaines limites par rapport à la taille des instances. Nous proposons ainsi une approche de génération de colonnes.

Le modèle précédent peut être rendu plus compact, si pour toute station h, au plus un robot est stationné dans une station d'attente, i.e. $H_h = 1$. Dans ce cas, la variable z_{nm} est remplacée par $z_{hm} = 1$ si la mission m est affectée au robot affecté à la station h et les variables r_{nhm} de linéarisation ne sont pas nécessaires. Le modèle devient alors:

$$Minimize \sum_{h\in H}\sum_{m\in M}\phi_m(\alpha A_{hm} + T_m + \beta R_{hm})z_{hm} + \sum_{h\in H}\chi s_h - \sum_{m\in M} p_m u_m \qquad (4\text{-}52)$$

Sous contraintes:

$$\sum_{h\in H} x_{nh} = 1, \qquad \forall n \in N \qquad (4\text{-}53)$$

$$\sum_{c\in C} y_{hc} \le 1, \qquad \forall h \in H \qquad (4\text{-}54)$$

$$\sum_{h\in H} z_{hm} + u_m = 1, \qquad \forall m \in M \qquad (4\text{-}55)$$

$$\sum_{c\in C}\sum_{h\in H} a_{fc} y_{hc} \le N_f, \qquad \forall f \in F \qquad (4\text{-}56)$$

$$\sum_{n\in N} x_{nh} \le 1, \qquad \forall h \in H \qquad (4\text{-}57)$$

$$\sum_{m\in M} \phi_m(A_{hm} + T_m + R_{hm})z_{hm} \le T_{\max}, \qquad \forall n \in N \qquad (4\text{-}58)$$

$$\sum_{c\in C_m} y_{hc} \ge z_{hm}, \qquad \forall m \in M, \forall h \in H \qquad (4\text{-}59)$$

$$\sum_{m\in M} e_m z_{hm} \le \sum_{n\in N} x_{nh} E_n + s_h \qquad \forall h \in H \qquad (4\text{-}60)$$

$$\sum_{n\in N} x_{nh} E_n + s_h \le \sum_{n\in N} x_{nh} E_{n,\max} \qquad \forall h \in H \qquad (4\text{-}61)$$

$$x_{nh}, y_{nc}, z_{hm} \in \{0,1\}, s_h \ge 0; \qquad \forall h \in H, \forall m \in M, \forall n \in N, c \in C \qquad (4\text{-}62)$$

Les autres modèles de ce chapitre peuvent également être simplifiés de manière similaire.

4.3.3. Génération de colonnes pour la planification intégrée de la configuration et de la localisation

Vue la complexité de ce problème, nous développons des heuristiques et des méthodes permettant de résoudre des instances de taille réelle de ce problème. Dans cette section nous présentons d'abord une formulation de ce problème en génération de colonnes, les méthodes de pricing pour la génération des nouvelles colonnes, permettant de déterminer une borne inférieure du problème. Ensuite on génère la solution entière en utilisant les méthodes d'arrondissage arithmétiques.

Une reformulation du problème utilisant le principe de génération de colonnes a été élaboré pour la planification intégrée et les notations utilisées restent valables pour cette section. Dans cette reformulation, une colonne représente un robot affecté à une station d'attente, configuré selon une configuration donnée et l'ensemble des missions affectées.

Soit Ω l'ensemble des colonnes faisables, sachant qu'une colonne *p* est représentée par les paramètres suivants :

$$X_{ph} = \begin{cases} 1, & \text{si la colonne } p \text{ concerne la station d'attente } h \\ 0, & \text{sinon.} \end{cases}$$

$$Y_{pc} = \begin{cases} 1, & \text{si la colonne } p \text{ est reliée à la configuration } c \\ 0, & \text{sinon.} \end{cases}$$

$$Z_{pm} = \begin{cases} 1, & \text{si la mission } m \text{ est affectée au robot de la colonne } p \\ 0, & \text{sinon.} \end{cases}$$

Une colonne est dite faisable si elle est affectée à une seule station d'attente et une seule configuration telle que les missions affectées au robot puissent être accomplies par les fonctionnalités installées sur le robot, et que la charge de travail respecte les limites prédéfinies T_{max}.

Pour une colonne *p* donnée, la charge/coût peut être déterminée suivant la formule suivante :

$$b_p = \sum_{h\in H} \sum_{m\in M} \phi_m \left(A_{hm} + T_m + R_{hm} \right) X_{ph} Z_{pm} \qquad (4\text{-}63)$$

avec l'hypothèse $\alpha = \beta = 1$.

Une seule variable de décision est utilisée:

$$\lambda_p = \begin{cases} 1, & \text{si la colonne } p \text{ est selectionnée} \\ 0, & \text{sinon.} \end{cases}$$

Ainsi le problème intégré de localisation et de configuration est reformulé comme un «Problème Maître» (PM) :

$$Minimiser \sum_{p \in \Omega} b_p \lambda_p \qquad (4\text{-}64)$$

Sous contraintes:

$$\sum_{p \in \Omega} \lambda_p = N \qquad (4\text{-}65)$$

$$\sum_{p \in \Omega} Z_{pm} \lambda_p = 1, \qquad \forall m \in M \qquad (4\text{-}66)$$

$$\sum_{c \in C} \sum_{p \in \Omega} a_{fc} Y_{pc} \lambda_p \le N_f \qquad \forall f \in F \qquad (4\text{-}67)$$

$$\sum_{p \in \Omega} X_{ph} \lambda_p \le H_h, \qquad \forall h \in H \qquad (4\text{-}68)$$

$$\lambda_p \in \{0,1\} \qquad \forall p \in \Omega \qquad (4\text{-}69)$$

La fonction objectif (4.64) a pour but de minimiser la charge de travail du système robotique. Les contraintes (65) assurent que tous les *N* robot sont considérés et affectés. Les contraintes (4.66) garantissent que toutes les missions sont prises en charge par un robot. Les contraintes (4.67) certifient que le nombre des fonctionnalités à installer sur les robots est inférieur ou égal au nombre d'exemplaires disponibles. Les contraintes (4.68) assurent que le nombre de robots affectés à une station d'attente respecte les limites prédéfinies de ce que la station peut accueillir. Les contraintes (4.69) sont des contraintes d'intégrité.

Le problème maître (PM) contient un très grand nombre de colonnes et ne peut pas être résolu directement. On procède à une relaxation des contraintes d'intégrités (4.69) qui seront remplacées par les contraintes suivantes : $\lambda_p \ge 0$ ($\forall p \in \Omega$) et le problème obtenu, sera désigné « problème maître linéaire » (PML). Vu le grand nombre des colonnes dans ce modèles, une autre variante de ce modèle dénommée «Problème Maitre Restreint» (PMR) limité à un sous ensemble des solutions possibles $\Omega' \subset \Omega$ est considérée, et les colonnes pouvant améliorer la solution sont générées au fur et à mesure.

Génération de colonnes

Afin de trouver la solution optimale du PML, on commence par la résolution du PMR jusqu' à l'optimalité en utilisant l'algorithme du Simplex. Cela conduit à la solution optimale du PMR ainsi qu'aux valeurs des variables duales du problème PML. Les variables duales permettent la détermination des coûts réduits des variables et ainsi la génération des colonnes améliorantes.

Soit u, v_m, w_f et r_h les variables duales du problème PMR correspondant respectivement pour les contraintes (4.65, 4.66, 4.67, 4.68).

Le principe de génération de colonnes (pricing problem) consiste à déterminer les colonnes ayant les plus bas coûts réduits. Si les coûts réduits des colonnes obtenues sont non négatifs, la solution optimale du problème PML est obtenue. Sinon des colonnes avec des coûts réduits négatifs sont ajoutées à l'ensemble Ω' et le processus est répété jusqu'à l'optimalité du PML.

Afin de réduire la complexité du problème de pricing PB, on le décompose en sous problèmes. Chacun de ces sous problèmes est relié à une station d'attente *h* et à une configuration *c*. Chaque sous problème *PB(h,c)* permet de déterminer la colonne ayant le plus bas coût réduit π_{hc}.

Le sous problème de pricing ***SPP(h,c)*** consiste à minimiser le coût réduit parmi les colonnes correspondant à (h, c) :

$$\pi_{hc} = Minimize\ b_p - u - \sum_{m \in M} v_m Z_{pm} - \sum_{f \in F} a_{fc} w_f - r_h$$

Ce qui est équivalent à :

$$\pi_{hc} = Minimize \sum_{m \in M} \left(\phi_m \left(A_{hm} + T_m + R_{hm} \right) - v_m \right) Z_{pm} - u - \sum_{f \in F} a_{fc} w_f - r_h \qquad (4\text{-}70)$$

Sous contraintes:

$$\sum_{m \in M} \phi_m (A_{hm} + T_m + R_{hm}) Z_{pm} \le T_{\max} \qquad (4\text{-}71)$$

$$Z_{pm} = 0, \quad \text{si} \quad c \notin C_m \qquad (4\text{-}72)$$

$$Z_{pm} \in \{0,1\}, \forall m \in M \qquad (4\text{-}73)$$

Le sous problème SPP est un problème de la famille de sac à dos, qui peut être résolu en appliquant un algorithme de programmation dynamique pseudo polynomial [Martello 90].

Solution initiale et variables artificielles

Afin de résoudre le problème PML, il est nécessaire que le premier sous ensemble de Ω' contienne assez de colonnes constituant une solution faisable, ce qui permet de calculer les valeurs duales des contraintes et générer les colonnes.

Deux solutions sont envisageables : soit le développement des heuristiques permettant de construire des solutions faisables, soit la modification du modèle et l'introduction des variables artificielles permettant de trouver une solution pour le modèle même sans solution initiale.
Dans notre étude, la seconde approche a été choisie. Pour chacune des contraintes du modèle PML, une variable artérielle est injectée, et l'usage de ces variables artificielles est pénalisé, par l'ajout d'une pénalité de ($2 \times T_{max}$) à la fonction objectif. Ainsi le PML peut être remplacé par le modèle suivant :

$$Minimizer \sum_{p \in \Omega} b_p \lambda_p + 2 \times T_{\max} \left(P + \sum_{m \in M} Q_m \right) \qquad (4\text{-}74)$$

Sous contraintes:

$$\sum_{p} \lambda_p + P = N \qquad (4\text{-}75)$$

$$\sum_{p \in \Omega} Z_{pm} \lambda_p + Q_m = 1 \qquad \forall m \in M \qquad (4\text{-}76)$$

$$\sum_{c \in C} \sum_{p \in \Omega} a_{fc} Y_{pc} \lambda_p + Q_f \leq N_f, \forall f \in F \qquad (4\text{-}77)$$

$$\sum_{p \in \Omega} X_{pb} \lambda_p + Q_h \leq H_h \qquad \forall h \in H \qquad (4\text{-}78)$$

$$\lambda_p \geq 0, \forall p \in \Omega, P \geq 0, Q \geq 0 \qquad (4\text{-}79)$$

La solution $P = N$, $Q_m = 1$, $Q_f = 0$, $Q_h = 0$, et $\lambda_p = 0$ est toujours faisable, et a une pénalité ($2 \times T_{max}$) des variables artificielles qui est supérieure aux coûts de toutes les colonnes faisables de l'ensemble Ω. Ces variables artificielles seront sorties de base lors des itérations suivantes de générations de colonnes.

Construction d'une solution admissible

Après la convergence de la procédure de génération de colonnes, la solution optimale du problème PML est obtenue. Sachant que le problème PML est obtenu par la relaxation des contraintes d'intégrité (69) du problème maître PM, PML fournit une borne inférieure du problème maitre mais la solution obtenue peut contenir des valeurs fractionnelles des variables λ_p et ainsi une solution non intégrale du PML n'est pas une solution faisable de PM.

Plusieurs approches peuvent être proposées pour déduire une solution au problème PM de la solution du PML. Une approche consiste à appliquer les procédures de Branch and Price pour déterminer une solution du problème maître.

Dans notre étude, nous utilisons une heuristique qui consiste à fixer à 1 les colonnes ayant des valeurs intégrales ($\lambda_p = 1$) et d'arrondir à 1 la colonne ayant une valeur fractionnelle la plus élevée. Une fois que ces valeurs sont fixées, on les retire du problème pour obtenir un problème plus réduit en robots et en missions. Ce processus est répété tant qu'on obtient une solution optimale fractionnelle du PML.

Vue d'ensemble de la procédure de génération de colonnes

En résumé, plusieurs étapes et procédures sont nécessaires pour pouvoir résoudre notre problème de localisation et configuration à l'aide de l'approche de génération de colonne. La *Figure 16* illustre les étapes- clés des techniques de génération de colonnes.

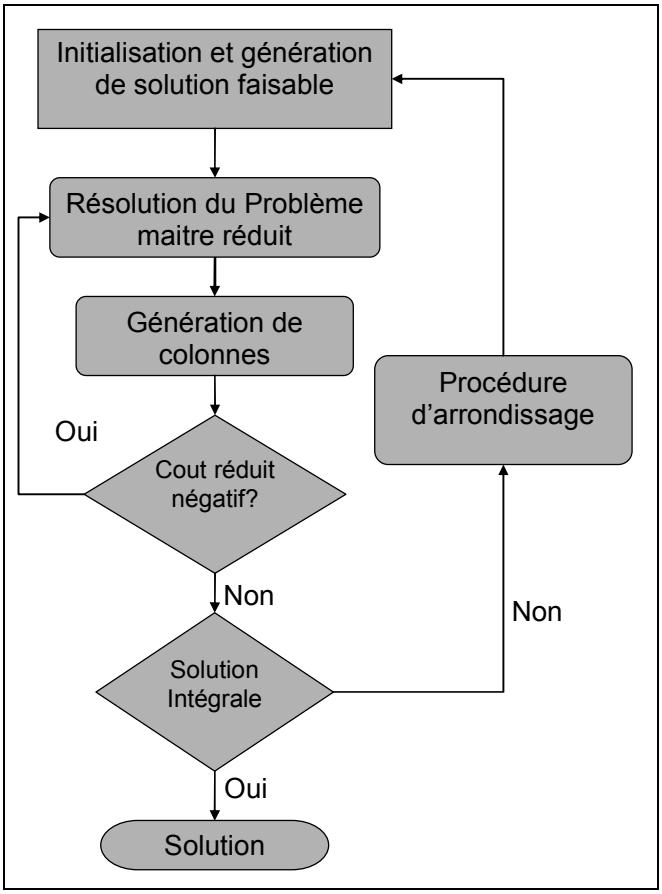

Figure 16 : Approche de génération de colonnes

La première étape consiste à transformer notre problème en un problème Maître Linéaire, ensuite on procède à la génération d'une solution afin de pouvoir construire le premier sous ensemble de colonnes du Problème Maitre Restreint. Dans notre étude on propose d'ajouter des variables artificielles permettant de générer une solution évidente pour le PML.

En deuxième étape, à l'aide d'un solveur standard de programmation linéaire, on résout le Problème Maitre restreint (PMR) qui contient un nombre restreint de colonnes générées précédemment.

En troisième étape, on procède à la génération de colonnes en utilisant la programmation dynamique et les valeurs des variables duales du problème PMR. Les colonnes générées ayant un coût réduit négatif sont ajoutées à l'espace de colonnes du PMR.

Les étapes 2 et 3 sont répétées tant qu'on peut trouver des colonnes ayant un coût réduit négatif. Si toutes les colonnes générées ont des coûts réduits positifs, la solution trouvée du PMR est la solution optimale du PML. Si la solution obtenue est intégrale, la solution optimale du problème Maitre (PM) est obtenue.

La quatrième étape a pour but de trouver une solution intégrale pour le problème maitre. Pour cela on fixe à 1 les colonnes égales à 1 et la colonne ayant la valeur fractionnelle la plus grande et on les retire du problème pour construire un sous problème maitre réduit et les étapes 2, 3 et 4 sont répétées tant qu'une solution intégrale n'est pas trouvée.

Instance illustrative

Dans cette section on illustre l'approche de génération de colonnes par une instance du problème combiné de configuration et de localisation. Le problème considéré est illustré par la Figure 17. Il y a trois stations d'attente *{H1, H2, H3}*, deux robots de base *{R1, R2}* dont la charge de travail maximale T_{max} est de 150 minutes (2.5 heures). Deux types de fonctionnalités sont considérés: les fonctionnalités de transport et les fonctionnalités de nettoyage. On dispose de deux modules de transport N_{transp} = 2 et une module de nettoyage N_{clean} = 1.

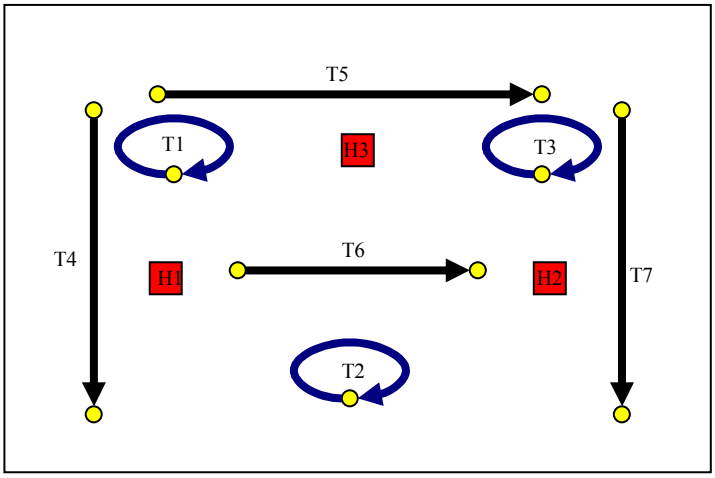

Figure 17 : Instance de 7 missions et 3 stations d'attente

Sept missions sont prévues d'être exécutées. Les caractéristiques de ces missions sont présentées dans le notamment le type de la mission et le temps nécessaires pour accomplir une mission en partant d'une station d'attente donnée. Ce temps comprend les temps de réaction, le temps d'exécution et le temps de retour. La fréquence d'occurrence des missions f_m est égale à 1 pour toutes les missions, et le nombre maximal H_h de robots qu'une station d'attente peut accueillir est 1. Il y a trois configurations possibles pour chaque robot : *C={{clean}, {transp}, {clean, transp}}* ce qui se traduit par qu'un robot est équipé, soit d'une fonctionnalité *{clean}* ou *{transp},* soit des deux fonctionnalités {*clean*, *transp*} en même temps.

Mission	Type	Temps nécessaire pour exécuter la mission en démarrant de:		
		H1	H2	H3
M1	*clean*	20	32	20
M2	*clean*	24	24	30
M3	*clean*	32	20	20
M4	*transp*	26	42	35
M5	*transp*	40	40	31
M6	*transp*	15	15	17
M7	*transp*	42	26	35

Tableau 3: Caractéristiques des missions

Cette instance est résolue par l'algorithme de génération de colonnes présentées précédemment. La solution obtenue est exposée dans le Tableau 4: Solution de l'instance illustrative avec Tmax=150

. On note que cette solution est une solution optimale. Pour chaque robot il existe une

station d'attente à laquelle il est assigné, sa configuration et les missions qui lui seront affectées ainsi que la charge résultante.

Robot	Station	Configuration	Missions	Charge (min)
R1	H2	{*transp*}	{T6, T7}	41
R2	H3	{*clean*, *transp*}	{T1, T2, T3, T4, T5}	136

Tableau 4: Solution de l'instance illustrative avec Tmax=150

A partir des résultats du Tableau 4: Solution de l'instance illustrative avec Tmax=150 , on constate que la charge totale du système robotique est de 177 minutes mais elle n'est pas répartie de manière équilibrée sur les deux robots. La Figure 18 représente la solution présentée par le Tableau 4

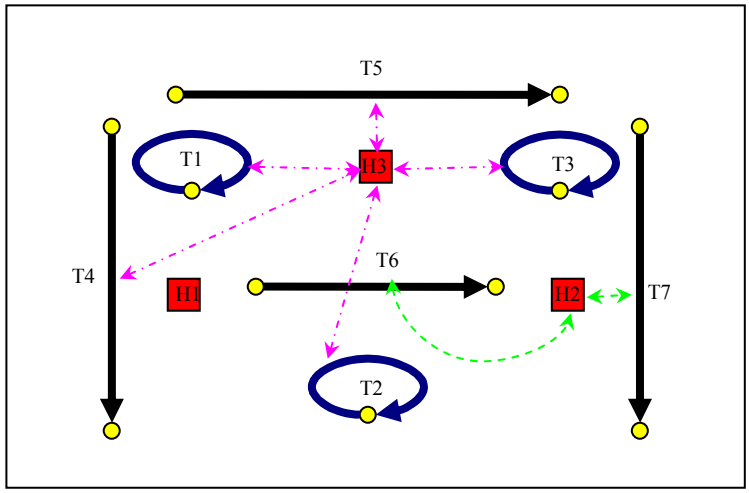

Figure 18 : Solution de l'instance illustrative

En réduisant la charge maximale des robots de 150 minutes à 110 minutes, la solution obtenue par la même procédure de génération de colonne est mieux équilibrée. Le tableau 5 représente la solution de cette instance modifiée.

Robot	Station	Configuration	Missions	Charge(min)
R1	H1	{*transp*}	{T4, T5, T6}	81
R2	H2	{*clean*, *transp*	{T1, T2, T3,	102

Tableau 5: Solution de l'instance illustrative avec Tmax=110

La charge totale du système a augmenté de 177 min à 188 min ce qui représente des parcours plus longs. Par contre les charges de travail sont bien mieux équilibrées.

4.4. *Planification stratégique séquentielle en trois phases*

La troisième approche de la planification stratégique consiste à résoudre les trois problèmes stratégiques de manière séquentielle, en prenant en compte les résultats de l'étape précédente. En premier lieu le problème de l'autonomie des batteries des robots est abordé et on définit pour chaque robot un plan de rechargement, ce qui permet de déterminer les robots opérationnels pour les étapes suivantes. Ensuite, après l'identification des robots opérationnels pour la période suivante, nous abordons le problème de configuration des robots pour déterminer les fonctionnalités de robots afin d'équilibrer la charge des robots. A la fin,

connaissant la configuration des robots pour la période suivante, les robots seront assignés aux stations d'attente les plus proches des missions qu'ils peuvent accomplir avec les fonctionnalités aménagées sur les robots.

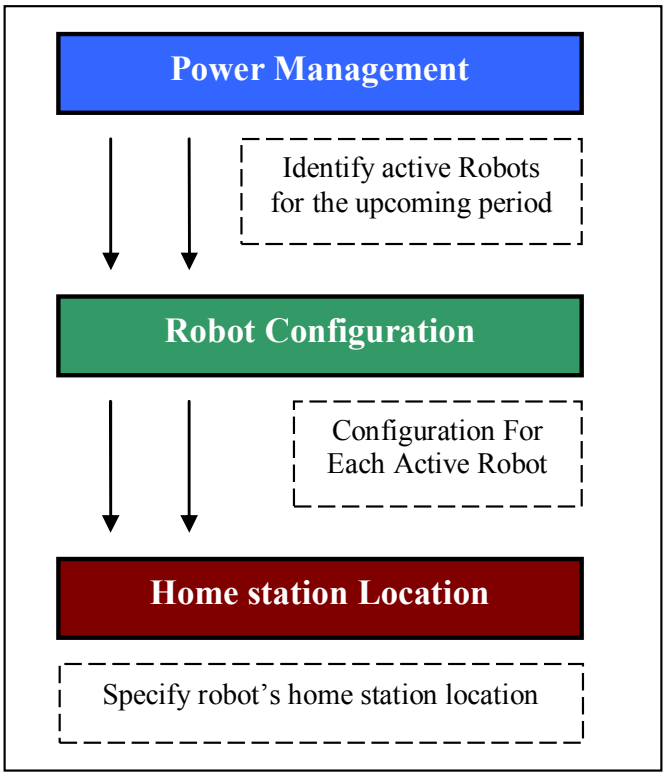

Figure 19 : Planification stratégique en trois phases

Le modèle mathématique de la gestion de l'autonomie est celui de la planification stratégique en deux phases et nous ne répétons pas ce modèle.

4.4.1. Modélisation en programme linéaire du problème de configurations de robots

Une fois que le plan des rechargements des robots est établie, on procède à la configuration des robots. Le but de cette étape est de minimiser les dépassements énergétiques par rapport aux recommandations de consommations E_n d'énergie déterminées précédemment.

Les variables de décisions à prendre en compte dans cette phase sont :

$$y_{nc} = \begin{cases} 1, & \text{si le robot } n \text{ est configuré selon la configuration } c \\ 0, & \text{sinon.} \end{cases}$$

$$z_{nm} = \begin{cases} 1, & \text{si la mission } m \text{ est affectée au robot } n \\ 0, & \text{sinon.} \end{cases}$$

Le problème de configuration du système robots peut être formulé comme suit :

$$Minimize \sum_{n \in N} s_n \tag{4-80}$$

sous contraintes:

$$\sum_{c \in C} y_{nc} = 1, \qquad \forall n \in N \tag{4-81}$$

$$\sum_{n \in N} z_{nm} = 1, \qquad \forall m \in M \tag{4-82}$$

$$\sum_{c \in C} \sum_{n \in N} a_{fc} y_{nc} \le N_f, \qquad \forall f \in F \tag{4-83}$$

$$\sum_{m \in M} \phi_m e_m z_{nm} \le s_n + E_n, \qquad \forall n \in N \tag{4-84}$$

$$\sum_{c \in C_m} y_{nc} \ge z_{nm}, \qquad \forall m \in M, \forall n \in N \tag{4-85}$$

$$s_n + E_n \le E_{n,\max} \qquad \forall n \in N \tag{4-86}$$

$$y_{nc}, z_{nm} \in \{0,1\} \ and \ s_n \ge 0 \quad \forall b \in B; m \in M; n \in N \tag{4-87}$$

où s_n représente le dépassement énergétique par rapport aux recommandations établies dans la phase précédente de gestion d'autonomie. Les contraintes (4.81), (4.82), (4.83), (4.85), (4.86), (4.87) sont identiques à ceux du problème combiné de localisation et de configuration dans la planification stratégique en deux phases. Les contraintes (4.84) déterminent le dépassement énergétique par rapport à la recommandation de consommation d'énergie E_n du robot n.

Dans le cas où la gestion d'autonomie n'est pas nécessaire, on peut prévoir une autre formulation qui a pour but d'équilibrer les charges de travail ou bien de minimiser la charge maximale de travail.

$$Minimize \ \delta \tag{4-88}$$

sous contraintes:

$$\sum_{c \in C} y_{nc} = 1, \qquad \forall n \in N \tag{4-89}$$

$$\sum_{n \in N} z_{nm} = 1, \qquad \forall m \in M \tag{4-90}$$

$$\sum_{c \in C} \sum_{n \in N} a_{fc} y_{nc} \le N_f, \qquad \forall f \in F \tag{4-91}$$

$$\sum_{m \in M} T_m z_{nm} \le \delta, \qquad \forall n \in N \tag{4-92}$$

$$\sum_{c \in C_m} y_{nc} \ge z_{nm}, \qquad \forall m \in M, \forall n \in N \tag{4-93}$$

$$y_{nc}, z_{nm} \in \{0,1\} \ and \ \delta_n, \delta_n^{'} \ge 0 \quad \forall b \in B; m \in M; n \in N \tag{4-94}$$

Les contraintes (4.89), (4.90), (4.91), (4.93), (4.94) sont identiques aux les contraintes présentés dans la formulation. La fonction objectif (4.88) a pour but d'équilibrer la charge de travail des robots. Les contraintes (4.92) définissent la charge de travail minimal et maximal. On note que seulement les temps d'exécution des missions sont pris en compte dans le calcul des charges de travail.

4.4.2. Modélisation en programme linéaire du problème de localisation des stations

Le but de cette phase est de minimiser les temps de parcours en tenant compte les recommandations énergétiques des phases précédentes. Les notations et variables utilisées pour formuler les problèmes combinés de localisation et configurations sont toujours valables.

Les variables de décision sont :

$$x_{nh} = \begin{cases} 1, & \text{si le robot } n \text{ est assigné à la station d'attente } h \\ 0, & \text{sinon.} \end{cases}$$

$$z_{nm} = \begin{cases} 1, & \text{si la mission } m \text{ est affectée au robot } n \\ 0, & \text{sinon.} \end{cases}$$

$$u_m = \begin{cases} 0, & \text{si la mission } m \text{ est prise en charge par un robot.} \\ 1, & \text{sinon.} \end{cases}$$

Le produit $x_{nh}z_{nm}$ est représenté par la variable de décision :

$$r_{nhm} = \begin{cases} 1, & \text{si la mission } m \text{ est affectée au robot } n \text{ et dont la station d'attente est } h \\ 0, & \text{sinon.} \end{cases}$$

Le problème de localisation des stations d'attentes est formulé comme suit:

$$Minimize \sum_{n\in N}\sum_{h\in H}\sum_{m\in M}\phi_m(A_{hm}+T_m+R_{hm})r_{nhm} + \sum_{n\in N}\chi s_n - \sum_{m\in M} p_m u_m \qquad (4\text{-}95)$$

sous contraintes:

$$\sum_{h\in H} x_{nh} = 1, \qquad \forall n \in N \qquad (4\text{-}96)$$

$$\sum_{n\in N} z_{nm} + u_m = 1, \qquad \forall m \in M \qquad (4\text{-}97)$$

$$\sum_{n\in N} x_{nh} \le H_h, \qquad \forall h \in H \qquad (4\text{-}98)$$

$$\sum_{h\in H}\sum_{m\in M}\phi_m(A_{hm}+T_m+R_{hm})r_{nhm} \le T_{\max}, \quad \forall n \in N \qquad (4\text{-}99)$$

$$\sum_{m\in M}\phi_m e_m z_{nm} \le s_n + E_n, \qquad \forall n \in N \qquad (4\text{-}100)$$

$$\sum_{m\in M}\phi_m e_m z_{nm} \le E_{n,\max}, \qquad \forall n \in N \qquad (4\text{-}101)$$

$$r_{nhm} \ge (x_{nh} + z_{nm}) - 1, \qquad \forall m \in M, n \in N, h \in H \qquad (4\text{-}102)$$

$$\sum_{c\in C_m} Y_{nc} \ge z_{nm}, \qquad \forall m \in M, \forall n \in N \qquad (4\text{-}103)$$

$$x_{nb}, z_{nt}, r_{nbt} \in \{0,1\}, \qquad \forall b \in B, \forall t \in T, \forall n \in N \qquad (4\text{-}104)$$

où γ est une variable de décision réelle qui représente la violation maximale des recommandations énergétiques.

Les contraintes (4.95), (4.96), (4.97), (4.98), (4.99), (4.100,) (4.102), (4.104) sont identiques aux contraintes (4.40),(4.41), (4.43), (4.45), (4.46), (4.49), (4.48), (4.51) du problème combiné de configuration et de localisation. Les contraintes (4.101) assurent que la quantité d'énergie requise pour exécuter les missions affectées ne dépasse la quantité maximale stockée sur les batteries. Les contraintes (4.103) garantissent que les missions sont affectées aux robots ayant une configuration adéquate fixée lors de l'étape précédente.

4.5. Comparaison et expérimentations numériques

4.5.1. Exemples illustratifs

Afin de mieux cerner les différences entre les différentes approches et ainsi que les différentes phases de résolution, nous présenterons deux exemples. Le premier est de petite taille, ce qui permet de résoudre à l'optimalité l'approche combinée et nous nous intéressons dans cet exemple à la charge du travail des robots dans le première période après la résolution de toutes les phases de l'approche donnée. Dans le deuxième exemple, nous considérons une instance de taille moyenne, et nous nous intéressons au changement de la charge de travail à travers les différentes phases de résolutions en utilisant l'approche de trois phases. Nous utilisons la loi uniforme pour générer les durées d'exécution de mission ainsi que les points de départ et d'arrivée qui sont générés aléatoirement à partir d'un plan donnée. La vitesse des robots est fixée à 0.5m/sec et la durée d'un trajet est calculée à partir de cette vitesse.

Exemple 1 : Comparaison entre les trois approches.

Afin de comparer l'effet des trois approches (intégrée /combinée, 2 phases et 3 phases) sur la charge du travail des robots, nous considérons un système composé de 4 robots (N=4), disposant de 7 stations d'attentes disponibles (H=7). Chaque robot dispose de 9 configurations par l'ajout de fonctionnalités (|C|=9). Ces robots doivent exécuter un ensemble de 15 missions (|M|=15) sur chacun de deux créneaux (T=2) considérés pour l'optimisation. Chaque station d'attente peut accueillir un robot.

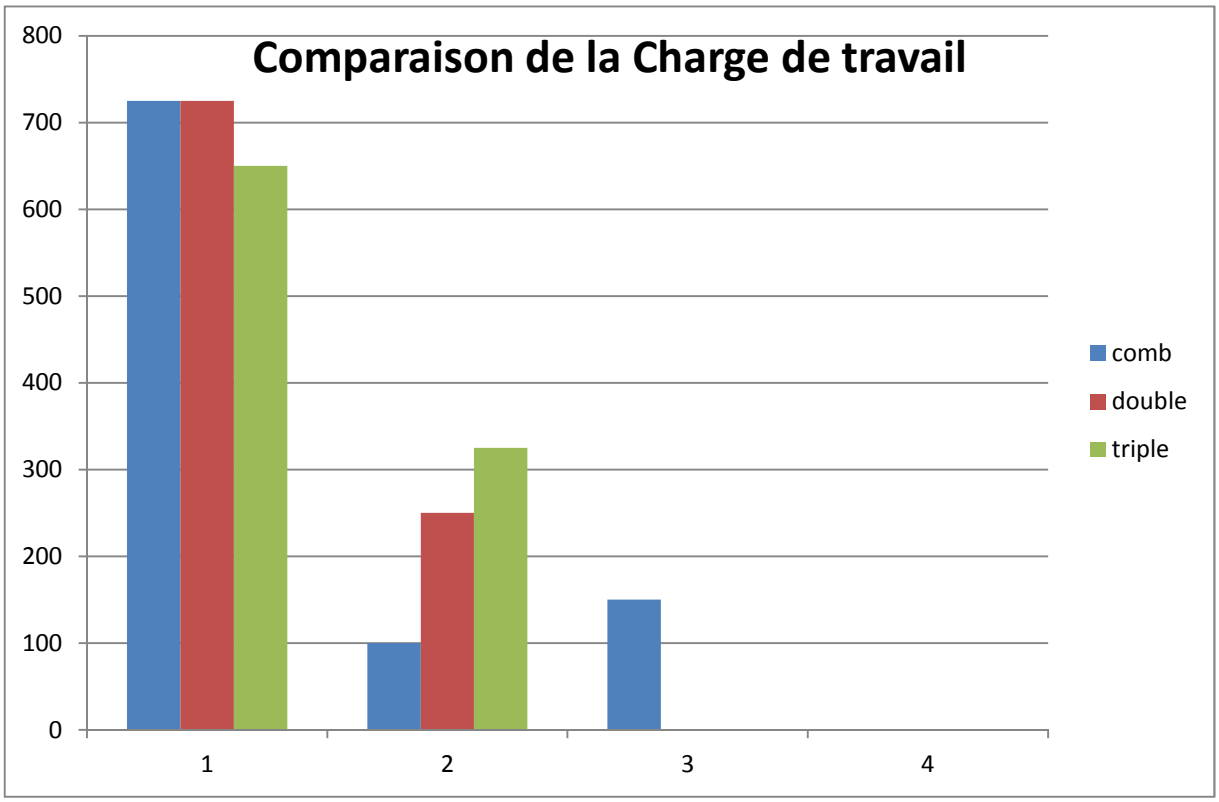

Figure 20 : Charge de travail de chaque robot

La Figure 20 illustrant la charge de travail de chaque robot, montre que l'approche combinée, privilégie la réactivité du système en utilisant un système de 3 robots, tandis que dans les deux autres approches seulement deux robots sont opérationnels, puis ce qu'en première phase de gestion d'autonomie deux robots sont amenés à se recharger, et ce qui entraine que la charge de travail doit être répartie sur ces deux robots disponibles.

Exemple 2 : Charge de travail des robots dans une résolution en trois phases.

Afin d'étudier l'impact de la résolution en trois phases sur la charge de travail, nous considérons un système robotique composé de 8 robots (N=8) pouvant être configurés de 9 façons différentes (|C|=9) et disposant de 12 stations d'attente (|H|=12). Ces robots doivent exécuter 20 missions (|M|=20) sur chacun des deux créneaux considérés (T=2).

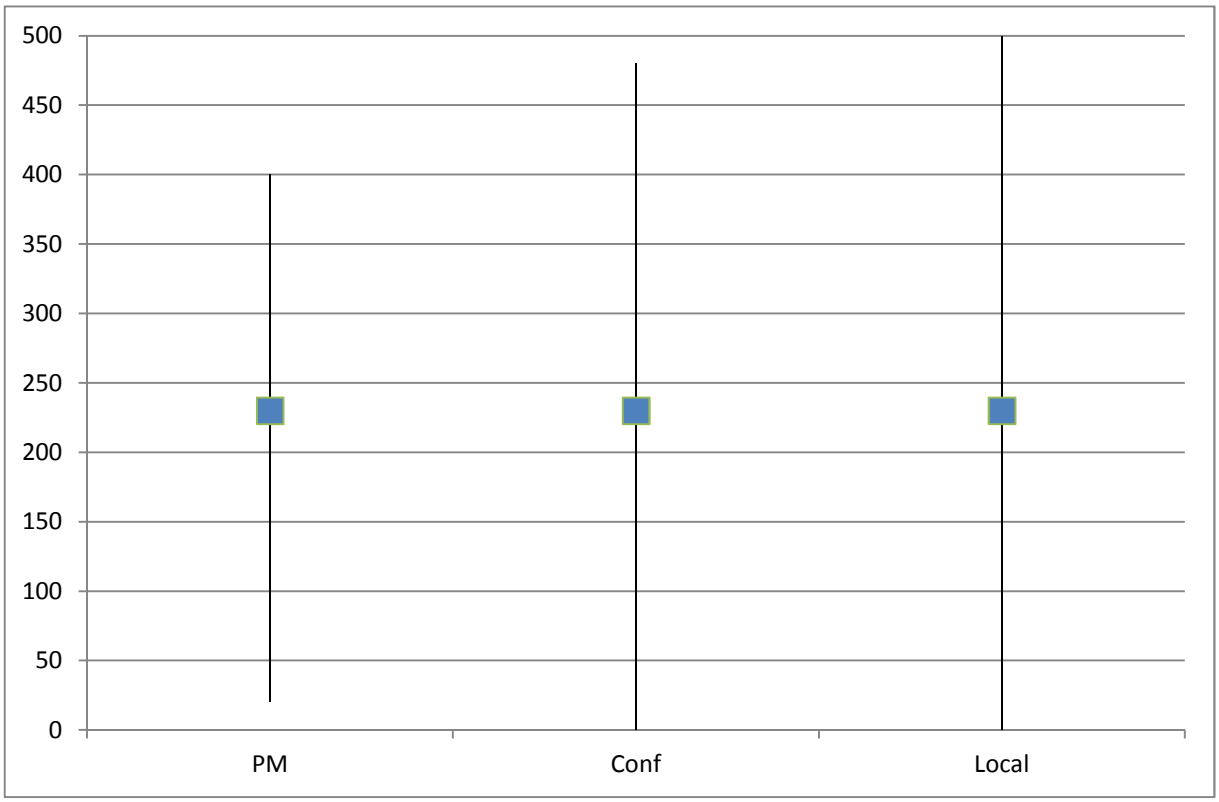

Figure 21 : Charge de travail dans les 3 phases

La Figure 21, illustrant la charge moyenne par robot ainsi que la charge maximale et minimale sur chacune des trois phases, montre qu'en utilisant la résolution MIP en 3 phases, la charge de travail est mieux équilibrée en première phase (PM) entre les différents robots actifs. Ensuite en deuxième phase (Conf) lors de l'attribution des configurations, la charge de travail de certains robots est augmentée et diminuée pour les autres. Le même phénomène est amplifié lors de l'attribution de station d'attente en troisième phase (Local).

4.5.2. Comparaison numériques entre les différentes approches de résolution MIP

Afin de tester la complexité de chaque approche ainsi que la qualité des résultats fournis, nous testons cinq instances présentées dans le tableau suivant

Instance	|N|	|H|	|T|	|M|	|C|
1	4	7	2	15	9
2	4	8	2	14	9
3	5	9	4	17	9
4	4	7	2	15	9
5	15	20	2	40	9

Tableau 6: Caractéristiques des Instances

Où (|N|) représente les nombre de robots, (|H|) représente le nombre de stations d'attente considérées. On note qu'on limite la capacité de chaque station à un seul robot. (|M|) est le nombre de missions à exécuter, (|C|) est le nombre de configurations possibles et finalement (|T|) représente le nombre de créneaux ou de périodes sur lesquels le problème est considéré.

Bien évidement en utilisant l'approche intégrée ou combinée, la résolution se fait en une seule phase. Quant à la résolution en deux phases, en première phase le problème énergétique est seulement considéré, ensuite en deuxième phase le problème de configuration est combiné avec le problème de localisation et ils sont résolus ensemble en en tenant compte des recommandations de la première phase. Afin de comparer la qualité des résultats, nous considérons une deuxième fonction objectif. Cette fonction objectif est calculée, une fois que le problème initial est résolu, et seule la charge effective de travail incluant les durées des trajets d'aller et de retour est considérée.

Instance 1

Méthode	CPU 1ere phase	CPU 2eme phase	CPU 3eme phase	Fonc. Obj
MIP combiné	44			3451
MIP en 2 phases	1<	1<		3603
MIP en 3	1<	1<	1<	3557

Instance 2

Méthode	CPU 1ere phase	CPU 2eme phase	CPU 3eme phase	Fonc. Obj
MIP combiné	7			3450
MIP en 2 phases	1<	1<		3450
MIP en 3 phases	1<	1<	Pas de solution	N /A

Instance 3

Méthode	CPU 1ere phase	CPU 2eme phase	CPU 3eme phase	Fonc. Obj
MIP combiné	2709>			N /A
MIP en 2 phases	1<	1		453.75
MIP en 3 phases	1<	1<	1<	453.75

Instance 4

Méthode	CPU 1ere phase	CPU 2eme phase	CPU 3eme phase	Fonc. Obj
MIP combiné	44			3451
MIP en 2 phases	1<	1<		3603
MIP en 3 phases	1<	1<	1<	3557

Instance 5

Méthode	CPU 1ere phase	CPU 2eme phase	CPU 3eme phase	Fonc. Obj
MIP combiné	Time Out			N/A
MIP en 2 phases	1<	2746		3603
MIP en 3 phases	1<	1<	684	N/A

Ces expérimentations ont été exécutées sur un nœud de cluster linux à 8 processeurs Intel Xeon 64 bits avec une mémoire partagées de 8GB. Le solver Cplex de IBM a été utilisé comme solveur MIP avec une Licence, permettant l'exécution en parallèle sur tous les processeurs.

Ces résultats montrent que la complexité de la méthode combinée la rend inexploitable même pour les petites instances, comme le montre les instances 3 et 5 qui sont relativement des instances de petite tailles mais et que la méthode combiné n'a pas pu résoudre ces problèmes. Tandis que la résolution en deux ou trois phases peut être exécutée en moins d'une seconde pour ces instances de petites tailles. En revanche l'instance 2, met en évidence, un des inconvénients de la méthode de trois phases, où la configuration des robots en deuxième phase, rétréci l'espace des solutions faisables en troisième phase et dans notre exemple aucune solution faisable na pas pu être trouvé.
Et les expérimentations montrent que la résolution du problème de localisation est un peu plus complexe que les autres problèmes traités. Une nouvelle modélisation compacte du problème joint (localisation - configuration) a confirmé ces résultats. Cette nouvelle modélisation permet de résoudre certaines instances en deux phases plus rapidement qu'en trois phases en utilisant l'ancien modèle.

4.5.3. Expérimentations numériques utilisant la méthode de générations de colonnes

Dans cette section, nous considérons que les robots sont équipés de batteries amovibles et le problème de gestion d'autonomie n'est pas considéré. Donc seulement la résolution des deux problèmes combinés : la configuration et la localisation ; en utilisant la méthode de génération de colonnes est considérée dans cette section. Nous proposons de tester les performances de cette méthode, sur les cinq instances présentées dans le tableau suivant :

Instance	|N|	|H|	|C|	|M|	Tmax
1	2	3	3	7	100 min
2	3	7	6	10	100 min
3	4	7	6	20	120 min
4	5	14	12	30	120 min
5	6	14	12	40	120 min

Tableau 7: Caractéristiques des instances de générations de colonnes

Où (|N|,|H|,|C| et |M|) représentent respectivement le nombre de robots, de stations d'attente, le nombre de configurations possibles et le nombre de missions à exécuter.

Ces expérimentations ont été exécutées sur une machine Windows équipée d'un processeur 2Ghz avec 3 Gb de Ram. Le problème maitre est résolu en utilisant le solveur open source CLP de la librairie COIN-OR. Le tableau suivant présente les performances de la méthode de générations de colonnes appliquées sur notre problème

Instance	Fonction Objective	# Col.	# Itérations	#Proced. Arrond.	Gap %	Temps. (sec)
1	1830	26	10	0	0.0	<1
2	2030	157	8	0	0.0	7
3	4090	544	36	1	0.18	74
4	5980	1473	26	0	0.0	435
5	8290	3459	112	4	0.03	889

Tableau 8: Performances de la méthode de générations de colonnes

Où (#col.) représente le nombre de colonnes générées, (# Itérations) est le nombre d'itérations entre problème maitre et esclave avant atteindre le critère d'arrêt. (#Proced. Arrond.) représente le nombre de fois que l'on a eu recours à l'heuristique d'arrondissage des colonnes fractionnelles afin d'atteindre une solution en nombres entiers.

On note que pour les instances de petite taille (#N<4 et #M<20), le modèle compact MIP est plus performant que la génération de colonnes, toutefois cette dernière surpasse le MIP pour les instances de taille moyenne (#N=6 et #M=40). Par exemple la résolution utilisant la génération de colonnes nécessit 889 secondes de temps de calcul. Cependant la résolution MIP en utilisant CPLEX nécessite plus que 4 jours de temps de calcul sur le cluster.

Sachant de 70% du temps de la résolution est consacré au problème esclave, une nouvelle version permettant de paralléliser l'exécution du problème esclave sur le cluster est dans les dernières phases de développement.

Nos collègues de Cardiff ont repris notre modèle joint de localisation et configuration proposé dans [Baalbaki 08 a] et ils ont proposée de le résoudre en utilisant des méta-heuristiques inspirées du comportement des abeilles [Xu 10]. Nous avons pris contact avec eux afin de comparer les performances de nos méthodes.

Un site Web consacré pour ce problème a été mis en place afin de fournir aux chercheurs, une introduction de ce problème, les instances sur lesquels nous avons travaillé ainsi que la meilleure solution trouvée par approche et par instances.

Deuxième partie : Niveau tactique

Chapitre 5. Affectation et ordonnancement des missions

Ce chapitre a pour objectif de définir de manière formelle les problèmes d'affectation et d'ordonnancement des missions au cours d'une journée. Nous commençons par décrire les décisions pertinentes et proposons ensuite les critères d'évaluation des décisions. Deux modèles mathématiques sont ensuite présentés pour une modélisation rigoureuse des décisions, des contraintes et des critères d'optimisation.

5.1. Décisions opérationnelles

Ce chapitre aborde la prise des décisions opérationnelles du système robotique étudié. Elles concernent l'affectation des missions aux robots et l'ordonnancement des missions sur les robots. Les agents de décision traitant ces deux problèmes sont sollicités régulièrement à l'issue de chaque changement des demandes ou des états des ressources. Ces décisions doivent être prises dans les plus brefs délais afin de ne pas gêner le bon déroulement des missions ; de même, les temps d'attente excessifs de prise de décision ne sont pas désirables.

5.1.1. Affectation des missions

Une des décisions opérationnelles les plus importantes est l'affectation des nouvelles missions et la réaffectation des missions en attente aux robots disponibles. Cette décision est évidement contrainte par les configurations, les fonctionnalités installées sur les robots, ainsi que par l'autonomie restante et la localisation des robots.

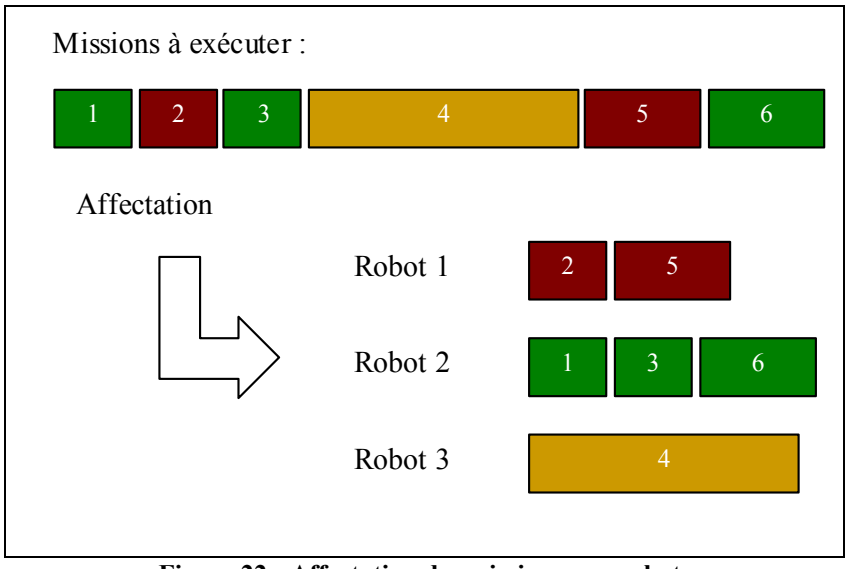

Figure 22 : Affectation des missions aux robots

Evénement déclencheur : deux événements majeurs sont responsables du déclenchement du processus de décision. Le premier est lié aux changements des demandes notamment l'arrivée de nouvelles missions. Le deuxième événement se produit suite à un changement de disponibilité de ressources comme le départ d'un robot ou l'arrivée d'un nouveau robot.

Données : différentes informations sont nécessaire pour une bonne affectation des missions. Ces informations sont essentiellement des indicateurs de l'état des robots, comme la configuration des robots, les fonctionnalités installées sur les robots, le niveau d'énergie restante, ainsi que leur charge de travail due aux missions qui leur sont déjà affectées. Les informations sur les nouvelles demandes comme les missions à affecter ou à réaffecter, sont aussi nécessaires. Les recommandations énergétiques du niveau stratégiques doivent aussi être prises en compte.

Nature de l'agent décision : la prise de décisions peut être centralisée ou décentralisée. Les deux systèmes de prise de décisions seront étudiés et comparés.

Horizon de calcul : l'horizon de calcul s'étend de la date actuelle de la prise de décision à la fin de l'ensemble des missions connues.

Temps de calcul : sachant que certaines missions sont urgentes et doivent être exécutées dans les minutes qui suivent leur demande, l'affectation doit être décidée rapidement. Dans notre cas, le temps le calcul de cet agent est plafonné à une minute.

Décision :l'agent de décision définit pour chaque robot un ensemble de missions qui lui sont affectées.

5.1.2. Planification et ordonnancement des missions

En plus de l'affectation des missions, il faut séquencer les missions sur chaque robot (c'est-à-dire déterminer l'ordre d'exécution des missions) et planifier les dates d'exécution de chaque mission. Des informations plus détaillées sur l'exécution des missions comme les déplacements à vide et les vitesses de déplacements des robots sont nécessaires.

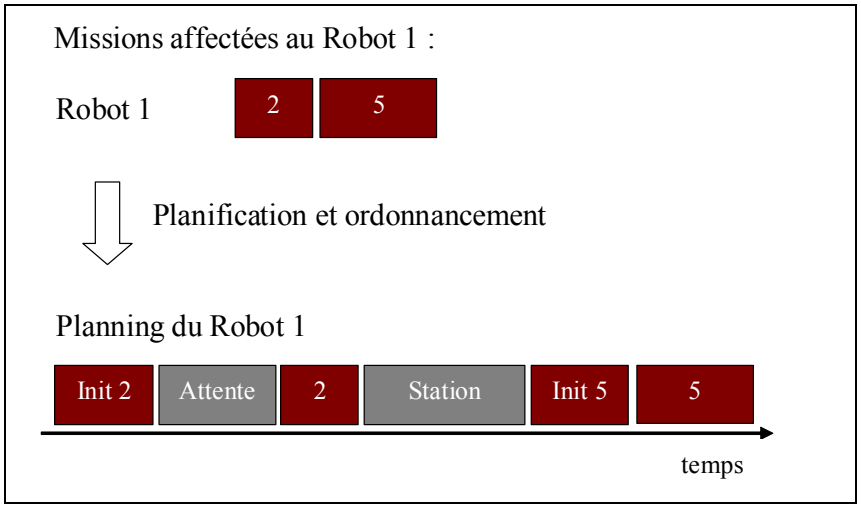

Figure 23 : Planification et ordonnancement des missions

Evénement Déclencheur: Cet agent de décision est sollicité suite à toute nouvelle affectation de missions, mais aussi suite à des circonstances empêchant la bonne exécution d'une mission et provocant ainsi des retards importants sur le planning initial.

Données : Les informations concernant les missions affectées à un robot ainsi que l'état du robot sont nécessaires pour une bonne planification. Pour les missions, nous avons besoins de

la liste de missions affectées à un robot, la mission en cours, les caractéristiques et les contraintes temporelles des missions. Une bonne planification dépend aussi des informations sur l'état des robots telles que la position actuelle du robot, sa vitesse moyenne, le niveau d'énergie restante de la batterie, la vitesse de consommation d'énergie (en veille, en attente et en marche). L'estimation du temps de trajet pour chaque déplacement d'un robot nécessite une trajectoire planifiée par le système de navigation et la vitesse moyenne du robot en fonction de la nature du déplacement (à vide ou chargé).

Nature de l'agent décision : Un agent centralisé peut effectuer l'ordonnancement des missions sur tous les robots, mais cela nécessite une communication continue entre l'agent et les robots pour mettre a jour l'état des robots. Une approche décentralisée avec un agent de planification sur chaque robot peut effectuer la prise de décision plus efficacement et alléger la charge sur les ressources de communication en supprimant les messages de notifications envoyés à l'agent central.

Horizon de calcul : L'horizon de calcul s'étend de la date actuelle de la prise de décision à la fin de l'ensemble des missions connues.

Temps de calcul : Comme pour l'affectation des missions, la planification et l'ordonnancement des missions doivent être décidés rapidement et le temps le calcul de cet agent est plafonné à une minute dans notre application.

Décision : Etablir pour chaque robot un plan d'exécution des missions donnant l'ordre d'exécution des missions et la date de début et de fin de chaque mission, ainsi que la durée des périodes d'attente.

5.2. Evaluation des stratégies de décisions opérationnelles

Cette section a pour objectif de proposer des critères permettant d'évaluer la qualité et les performances des stratégies d'affectation et d'ordonnancement des missions.

Dans la littérature d'ordonnancement des tâches, les critères sont généralement liés aux dates de fin des tâches et peuvent être les suivants : la durée totale d'exécution des tâches, la somme des temps d'achèvement des tâches, le nombre de tâches en retard, la somme des retards, le retard maximal, la somme de pénalités liées aux avances et retards, l'équilibrage des charges des ressources. Dans certaines applications, on cherche à minimiser le nombre de ressources nécessaires. Certaines études prennent en compte les coûts de production, les coûts de transport, les coûts de stockage et les coûts de lancement.

Pour la gestion des missions logistiques hospitalières par des robots, la date de début d'une mission peut être aussi importante que sa date de fin et les exigences diffèrent d'une mission à l'autre. Pour certaines missions, il est important de pouvoir commencer une mission dans une fenêtre de temps donnée afin d'assurer la présence du personnel pour lancer la mission. C'est le cas des missions de nettoyage de vomissements où la présence d'une personne médicale est nécessaire pour définir la zone de nettoyage. Pour d'autres missions, la fin d'une mission dans un intervalle de temps donné est importante. Pour beaucoup de missions comme la livraison urgente de médicaments pour un patient, la date de début et la date de fin dans des fenêtres de temps données sont toutes importantes.

L`importance accordée au respect de ces intervalles en début et fin de missions varie selon l'importance de la mission, le type de mission et selon les exigences de l'utilisateur final. Par exemple, pour les missions de transport et de livraison, il est exigé de mentionner la date de livraison, la date d'approvisionnement ou bien la date de prise en charge du médicament à transporter. Par contre pour les missions de téléconférence, le respect des délais de date de début est primordial afin de ne faire attendre ni le patient, ni le docteur avant d'établir une connexion vidéo entre les deux parties.

Sachant que l'élément- clé des décisions opérationnelles est la mission, les caractéristiques de la mission sont prises en compte pour l'évaluation des décisions opérationnelles. Ces caractéristiques peuvent être déterminées selon les attentes exprimées par l'utilisateur final et les exigences requises par telle ou telle mission.

Dans notre étude, les contraintes temporelles de chaque mission m sont représentées par les critères suivants :

- $[es_m, ps_m]$: fenêtre de temps pour le démarrage de la mission m avec es_m correspondant à la date au plus tôt pour le démarrage et $[es_m, ps_m]$ la fenêtre de temps souhaitée pour démarrer la mission m;
- ls_m : date au plus tard pour le début de la mission m avec $ps_m \leq ls_m$;
- dd_m : date de fin souhaitée de la mission m avec $ls_m \leq dd_m$. Ce critère exige la fin de la mission m avant cette date si cela est possible;
- ld_m : date de fin au plus tard de la mission m avec $dd_m \leq ld_m$;

L'ordonnancement est responsable de la conversion de la séquence des missions affectées à un robot en un planning. Deux dates sont calculées pour chaque mission :

- La date de début de la mission s_m sous contrainte $s_m \geq es_m$;
- La date d'achèvement e_m.

La Figure 24 illustre les différentes dates clé ou jalons d'une mission m, notamment les trois dates attribuées aux dates de début des missions et les deux dates attribuées aux dates d'achèvement.

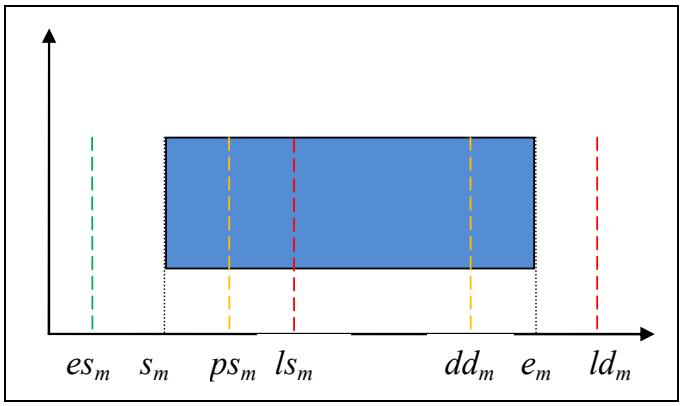

Figure 24 : Dates clés d'une mission

Le non respect des différentes dates clés sera mesuré par différentes pénalités en fonction des priorités des missions. D'autres critères seront également introduits pour l'évaluation de la consommation d'énergie des robots.

Afin de définir les paramètres et les dates clé pour une nouvelle mission, on définit des intervalles type pour chaque famille de missions, sachant que l'utilisateur final, pour des raisons de rapidité, peut fixer une seule des dates clés d'une mission et que les autres caractéristiques manquantes seront calculées à partir de la date fixée par l'utilisateur final et des intervalles type pour chaque famille de missions. Par exemple, un médecin demandant la mise en place d'une session de téléconférence avec un patient à 15h, ce qui sera considéré comme la date de début souhaitée, la durée du premier intervalle de date de début a été définie empiriquement pour les missions de téléconférence d'une valeur de cinq minutes et pour le deuxième intervalle de date de début de dix minutes ce qui nous permet de définir les deux paramètres *es* et *ls* ayant respectivement les valeurs 14:55 et 14:10.

5.2.1. Primes par palier pour l'évaluation des décisions opérationnelles

Nous proposons d'abord une première méthode d'évaluation constituée de deux critères, un critère temporel sur les respects des impératifs temporels des missions et le deuxième critère lié au respect des recommandations énergétiques des robots. Le critère temporel repose sur le principe de gains qui privilégie les solutions ayant respecté les fenêtres de temps recommandées par les utilisateurs finaux.

A la mission m, nous associons une prime RS_m en fonction de sa date de démarrage et une prime RE_m en fonction de sa date de fin. Plus précisément,

$$RS_m = \begin{cases} r_{m1} + r_{m2}, & \text{si } \mathrm{es}_m \le s_m < ps_m; \\ r_{m1}, & \text{si } ps_m \le s_m < ls_m; \\ 0, & \text{si } s_m \ge ls_m. \end{cases}$$

$$RE_m = \begin{cases} r_{m3} + r_{m4}, & \text{si } e_m < dd_m; \\ r_{m3}, & \text{si } dd_m \le e_m < ld_m; \\ 0, & \text{si } e_m \ge ld_m \end{cases}$$

où les gains $r_{mi} > 0$ dépendent du type de la mission et de son degré de priorité. La Figure 25 trace les courbes des gains d'une mission en fonction de sa date de début et de sa date d'achèvement.

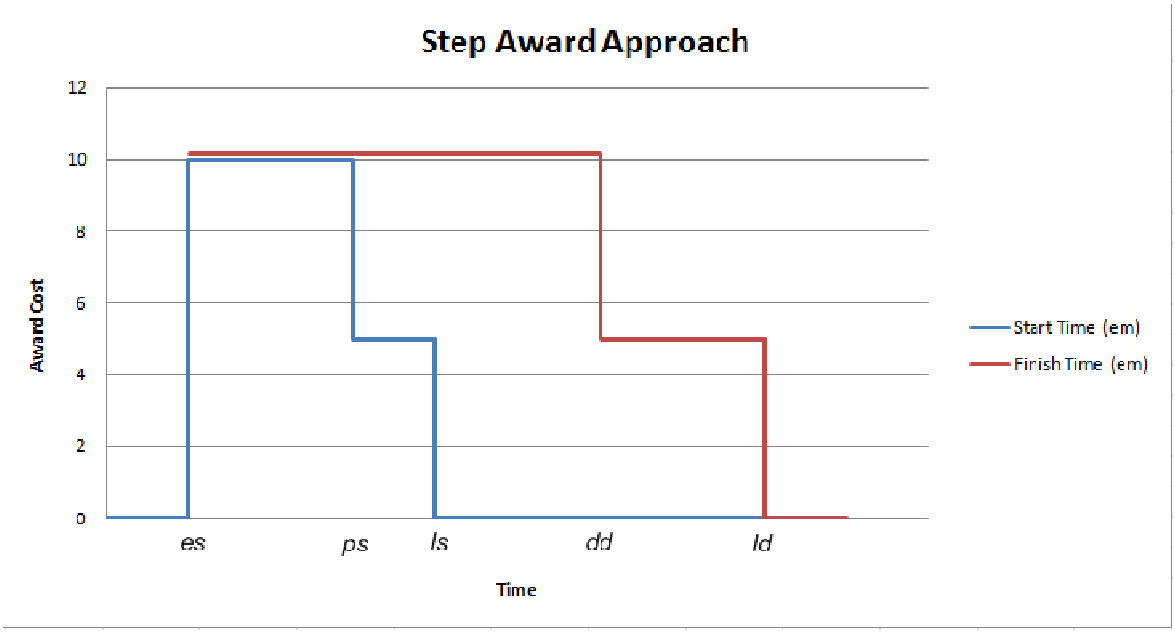

Figure 25 : Primes par palier

Nous proposons maintenant un deuxième critère pour évaluer le respect des consommations énergétiques préconisées par les agents de décisions stratégiques. Il est évident que ce critère concerne seulement les robots avec des batteries rechargeables fixes dont la gestion d'autonomie est importante. Ce critère pénalise les déviations des recommandations de la consommation énergétiques.

Notons d'abord que les recommandations sont établies sur une période assez longue avec des prévisions des différents types de missions estimées sur la totalité de la période. D'un autre côté, les décisions opérationnelles étudiées dans ce chapitre concernent seulement les missions connues et affectées aux robots au fur et à mesure de leur arrivée. Afin de tenir compte des missions futures, nous remplaçons la recommandation énergétique globale du niveau stratégique par une recommandation énergétique uniformément répartie dans le temps durant toute la période concernée.

Plus précisément, soit $Q_{rec,n}$ la recommandation énergétique du niveau stratégique pour l'exécution des missions par le robot n pour une période de longueur T. Soit $Q_{con,n}$ la consommation énergétique du robot n jusqu'à la date actuelle. Soit $Q_{plan,n}$ la quantité d'énergie nécessaire pour exécuter l'ensemble de ces missions du robot n. La pénalité due au dépassement énergétique du robot n est donnée par la formule suivante :

$$CP_n = P \cdot MAX\left(Q_{plan,n} + Q_{con,n} - \frac{t_0}{T} Q_{rec,n}, 0\right)$$

où t_0 peut être la date actuelle.

Et le coût total du planning pour un robot n est donné par la formule suivante:

$$R_n = \sum_{m \in M_n} \left(RS_m + RE_m\right) - CP_n$$

Le but de cette fonction objectif est de maximiser les gains qui découlent des primes et de minimiser les pénalités qui proviennent du dépassement des recommandations de consommation d'énergie.

Exemple illustratif :

Considérons un exemple de trois missions. Les caractéristiques de ces missions sont données par le tableau suivant où $t_0 = 25$ indique la date actuelle:

MISSION	CONSOMMATION Q_m	DATE DE DÉBUT			DATE D'ACHEVEMENT	
		es_m	ps_m	ls_m	dd_m	ld_m
M1	400	t_0+0	t_0+5	t_0+10	t_0+15	t_0+25
M2	200	t_0+0	t_0+10	t_0+15	t_0+17	t_0+23
M3	400	t_0+5	t_0+12	t_0+17	t_0+20	t_0+27

Tableau 9 : Caractéristiques des missions

Les paramètres des gains sont $r_{m1} = r_{m2} = r_{m3} = r_{m4} = 5$. Considérons l'ordonnancement suivant :

MISSION	PLANNING		GAIN	
	s_m	e_m	RS	RE
M1	t_0+0	t_0+9	10	10
M2	t_0+10	t_0+15	5	10
M3	t_0+17	t_0+25	0	5

Tableau 10: Gains par missions

Pour les pénalités liées aux consommations énergétiques, la recommandation du niveau stratégique est $Q_{rec,n}$=4500W établie pour une période de trois heures (T=180 min) et aucune mission n'est encore exécutée, i.e. $Q_{con,n} = 0$. Selon le Tableau 9, le planning (M1, M2, M3) nécessite une énergie de 1000 W pour ces trois missions.

$$CP_n = P \cdot MAX\left(1000 - \frac{25}{180}4500, 0\right) = P.375$$

Avec $P = 0.05$, la somme des deux critères est R_n = (10+10+5+10+5) -18,75=21,25.

5.2.2. Pénalités continues

Cette formulation est une variante de la formulation précédente avec également deux critères d'évaluation : un critère sur le respect des dates clés des missions et l'autre sur le respect de la recommandation énergétique du niveau stratégique.

Comme l'approche des gains par palier, le critère sur le respect des dates clés est la somme des deux critères : l'un sur le respect des dates clés pour le démarrage des missions et l'autre sur les dates d'achèvement.

$$CS_m = \begin{cases} 0, & \text{si } s_m < ps_m; \\ c_{m1}\left(s_m - ps_m\right), & \text{si } ps_m \le s_m < ls_m; \\ c_{m1}\left(s_m - ps_m\right) + c_{m2}\left(s_m - ls_m\right), & \text{si } s_m \ge ls_m. \end{cases}$$

$$CE_m = \begin{cases} 0, & \text{si } e_m < dd_m; \\ d_{m1}\left(e_m - dd_m\right), & \text{si } dd_m \le e_m < ld_m; \\ d_{m1}\left(e_m - dd_m\right) + d_{m2}\left(e_m - ld_m\right), & \text{si } e_m \ge ld_m. \end{cases}$$

où les paramètres (c_{mi} et d_{mi}) sont positifs et définissent les pénalités par unité de dépassement des dates clés de la mission. Dans certains cas ayant une fonction objectif maximisant le gain par rapport à certains critères (voir chapitre 7) ces critères par rapport aux retards peuvent être intégrés en échangeant le signe (c_{mi} et d_{mi}) ce qui permettra d'intégrer les critères de retard dans une fonction maximisant certains gains et en minimisant les critères de retard. La Figure 26 illustre la partie des pénalités temporelles qu'une mission peut procurer selon la date butoir et selon la date d'achèvement.

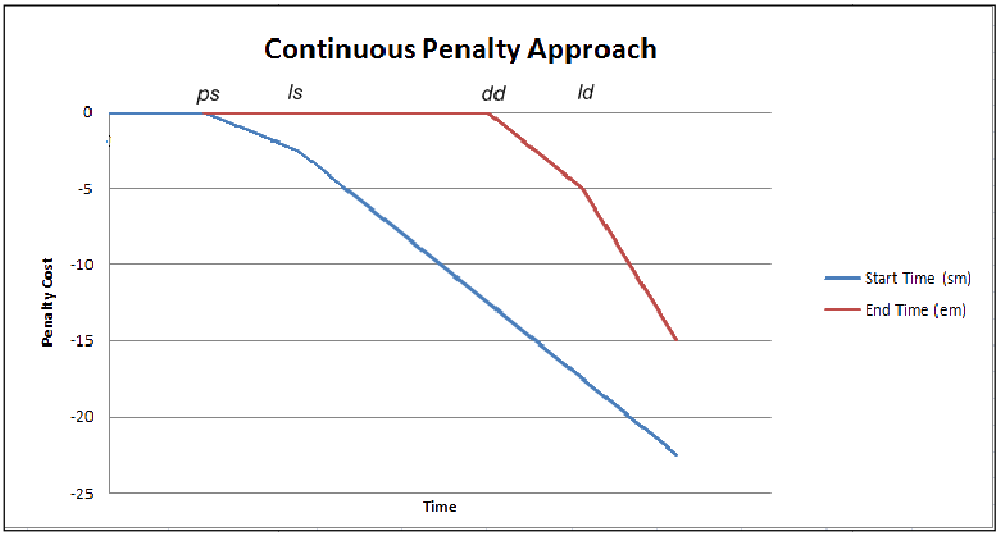

Figure 26 : Pénalités Continues(c_{mi} et d_{mi}) négatives

Le critère sur le respect de la recommandation énergétique est le même que celui de l'approche de gains par palier. Ainsi, le coût total du planning de l'ensemble des missions M_n affecté au robot n est donné par la formule suivante :

$$C_n = \sum_{m \in M_n} \left(CS_m + CE_m\right) + CP_n$$

Exemple illustratif :
Reprenons la même instance illustrée par le Tableau 9 de la section précédente, le tableau suivant permet de dresser les pénalités associées à chaque mission avec $c_{m1} = d_{m1} = 10$ et $c_{m2} = d_{m2} = 10$.

MISSION	PLANNING		PENALITES	
	s_m	e_m	CS	CE
M1	t_0+0	t_0+9	0	0

M2	t_0+10	t_0+15		0	0
M3	t_0+17	t_0+25		-50	-50

Tableau 11: Pénalités continues

La pénalité issue du dépassement énergétique est égale à 18,75 et donc le coût total de ce planning en utilisation la méthode de pénalités continues est égale (-50 -50) -18,75 = -118,75.

5.3. Modèle de décisions opérationnelles utilisant les primes par palier

Le problème d'affectation et d'ordonnancement peut être formulé comme un problème de programmation linéaire en nombres entiers, et les notations suivantes sont utilisées pour modéliser le problème.

Notation

- Soit M l'ensemble de missions connues et à exécuter
- Chaque mission m est caractérisée par :
  - La durée d'exécution T_m
  - La quantité d'énergie nécessaire E_m
  - La date de début au plus tôt es_m
  - La date de début préféré ps_m
  - La date de début au plus tard ls_m
  - La date de fin préférée dd_m
  - La date de fin au plus tard ld_m
  - La fonctionnalité f_m nécessaire
- Soit N l'ensemble des robots
- Chaque robot n est caractérisé par:
  - L'ensemble C_n des fonctionnalités installées sur le robot n
  - La quantité recommandée d'énergie W_n à consommer par le robot n. Selon la Section 5.2.1, $W_n = MAX\left(\frac{t_0}{T}Q_{rec,n} - Q_{con,n}, 0\right)$
  - La quantité maximal d'énergie $E_{n,max}$ que le robot n peut consommer

Les variables de décisions

$$y_m = \begin{cases} 1, \text{ si la mission } m \text{ est affectée et executée par un robot} \\ 0, \text{ sinon} \end{cases}$$

$$x_{n,m} = \begin{cases} 1, \text{ si la mission } m \text{ est affectée au robot } n \\ 0, \text{ sinon} \end{cases}$$

$$k_{n,ij} = \begin{cases} 1, \text{ si les missions } i \text{ et } j \text{ sont affectées au robot } n \\ \quad \text{et } j \text{ est exécutée immédiatement après } i \\ 0, \text{ sinon} \end{cases}$$

$$k_{n,0m} = \begin{cases} 1, \text{ si la mission } m \text{ est la première à être exécutée sur le robot } n, \\ 0, \text{ sinon} \end{cases}$$

$s_m =$ date de début de mission m.

$$a_{1,m} = \begin{cases} 1, \text{ si la mission } m \text{ est lancée avant la date } ps_m \\ 0 \text{ , sinon.} \end{cases}$$

$$a_{2,m} = \begin{cases} 1, \text{ si la mission } m \text{ est lancée avant la date } ls_m \\ 0 \text{ , sinon.} \end{cases}$$

$$b_{1,m} = \begin{cases} 1, \text{ si la mission } m \text{ est achevée avant la date } dd_m \\ 0 \text{ , sinon.} \end{cases}$$

$$b_{2,m} = \begin{cases} 1, \text{ si la mission } m \text{ est achevée avant la date } ld_m \\ 0 \text{ , sinon.} \end{cases}$$

Les contraintes

- Une mission ne peut être affectée qu'aux robots proprement configurés pour exécuter ce type de mission.

$$x_{nm} = 0 \text{ si } f_m \notin C_n$$

- Chaque robot ne peut exécuter qu'une mission à la fois.

$$s_i + k_{n,ij}(T_i + A_{ij}) \le s_j + (1 - k_{n,ij})G$$

où G est un grand nombre.

- Une mission exécutée est, soit la première à être exécutée soit exécutée après une autre mission.

$$\sum_{n \in N} \sum_{i \in M} k_{n,im} + \sum_{n \in N} k_{n,0m} = y_m$$

- Dépassements des recommandations énergétiques des robots.

$$\sum_{m \in M} x_{nm} E_m \le W_n + p_n$$

- Quantités maximales d'énergie des robots.

$$\sum_{m \in M} x_{nm} E_m \le E_{n,\max}$$

- La date de début de la première mission à exécuter par un robot découle du temps d'initialisation du robot et du temps de trajet de la position initiale du robot jusqu'au point de départ du robot.

$$0 + k_{n,0m}(T_{n,0} + A_{n,0m}) \le s_m + (1 - k_{n,0m})G$$

- Une mission est affectée à au plus un robot

$$k_{n,0m} + \sum_{i \in M} k_{n,im} \le x_{nm}$$

$$\sum_{i \in M} k_{n,im} \le x_{nm}$$

- Une mission est considérée comme exécutée si elle est affectée à un robot

$$\sum_n x_{nm} = y_m$$

- Dépassement de la date clé ps_m :

$$s_m + a_{1,m}\varepsilon \le ps_m + (1 - a_{1,m})G$$

où ε est un nombre suffisamment petit.

- Dépassement de la date clé ls_m :

$$s_m + a_{2,m}\varepsilon \le ls_m + (1 - a_{2,m})G$$

- Dépassement de la date clé dd_m :

$$s_m + T_m + b_{1,m}\varepsilon \le dd_m + (1 - b_{1,m})G$$

- Dépassement de la date clé ld_m :

$$s_m + T_m + b_{2,m}\varepsilon \le ld_m + (1 - b_{2,m})G$$

- D'autres contraintes :

$$a_{1,m} \le y_m;\ a_{2,m} \le y_m;\quad b_{1,m} \le y_m;\quad b_{2,m} \le y_m$$

Fonction objectif

Plusieurs objectifs sont considérés, on cherche à exécuter le plus possible de missions en respectant les contraintes de temps et en minimisant les dépassements énergétiques. Plus précisément :

$$Max \sum_{m \in M} (r_{m1}a_{1,m} + r_{m2}a_{2,m} + r_{m3}b_{1,m} + r_{m4}b_{2,m} + y_m H) - \sum_{n=1..N} p_n P$$

où H est le gain de l'exécution d'une mission et P le coût unitaire de dépassement énergétique.

Formulation

Le problème d'ordonnancement peut être formulé ainsi :

$$Max \sum_{m \in M} (r_{m1}a_{1,m} + r_{m2}a_{2,m} + r_{m3}b_{1,m} + r_{m4}b_{2,m} + y_m H) - \sum_{n=1..N} p_n P \qquad (5\text{-}1)$$

Sous contraintes :

$$x_{nm} = 0 \ if \ f_m \notin C_n \qquad \forall n \in 1..N; m \in M \qquad (5\text{-}2)$$

$$s_i + k_{n,ij}(T_i + A_{ij}) \le s_j + (1 - k_{n,ij})G \qquad \forall n = 1..N; \forall i, j \in M \qquad (5\text{-}3)$$

$$0 + k_{n,0m}(T_{n,0} + A_{n,0m}) \le s_m + (1 - k_{n,0m})G \qquad \forall n = 1..N; \forall m \in M \qquad (5\text{-}4)$$

$$\sum_{n=1..N}\sum_{i\in M} k_{n,im} + \sum_{n=1..N} k_{n,0m} = y_m \qquad \forall m \in M \qquad (5\text{-}5)$$

$$\sum_{m\in M} x_{nm} E_m \le W_n + p_n \qquad \forall n = 1...N \qquad (5\text{-}6)$$

$$\sum_{m\in M} x_{nm} E_m \le E_{n,\max} \qquad \forall n = 1...N \qquad (5\text{-}7)$$

$$k_{n,0m} + \sum_{i\in M} k_{n,im} \le x_{nm} \qquad \forall m \in M, \forall n = 1...N \qquad (5\text{-}8)$$

$$\sum_{i\in M} k_{n,mi} \le x_{nm} \qquad \forall m \in M, \forall n = 1...N \qquad (5\text{-}9)$$

$$s_m + a_{1,m}\varepsilon \le ps_m + (1 - a_{1,m})G \qquad \forall m \in M \qquad (5\text{-}10)$$

$$s_m + a_{2,m}\varepsilon \le ls_m + (1 - a_{2,m})G \qquad \forall m \in M \qquad (5\text{-}11)$$

$$s_m + T_m + b_{1,m}\varepsilon \le dd_m + (1 - b_{1,m})H \qquad \forall m \in M \qquad (5\text{-}12)$$

$$s_m + T_m + b_{2,m}\varepsilon \le ld_m + (1 - b_{2,m})G \qquad \forall m \in M \qquad (5\text{-}13)$$

$$\sum_{n=1..N} x_{nm} = y_m \qquad \forall m \in M \qquad (5\text{-}14)$$

$$s_m \ge es_m \qquad \forall m \in M \qquad (5\text{-}15)$$

$$a_{1,m}, a_{2,m}, b_{1,m}, b_{2,m} \in \{0,1\} \qquad \forall m \in M \qquad (5\text{-}16)$$

$$p_n \ge 0 \qquad \forall n = 1..N \qquad (5\text{-}17)$$

$$x_{nm} \in \{0,1\}; y_m \in \{0,1\} \qquad \forall n \in 1..N; \forall m \in M \qquad (5\text{-}18)$$

$$k_{n,ij} \in \{0,1\} \qquad \forall n \in 1..N; \forall i \in M^* j \in M \qquad (5\text{-}19)$$

5.4. Modèle de décisions opérationnelles utilisant les pénalités continues

Cette formulation est une variante de la formulation utilisant les gains par palier, on suppose que les notations de la formulation précédente ainsi que la majorité des variables de décision et des contraintes restent valables et sont conservées pour cette variante.

Nouvelles variables de décisions

Les variables de décision, $a_{1,m}$, $a_{2,m}$, $b_{1,m}$, $b_{2,m}$, ne sont plus d'utilité dans la nouvelle formulation et les variables suivantes les remplaceront :

- Δps_m : retard du début de la mission m par rapport au jalon ps_m
- Δls_m retard du début de la mission m par rapport au jalon ls_m
- Δdd_m retard de la fin de la mission m par rapport au jalon dd_m
- Δld_m retard de la fin de la mission m par rapport au jalon ld_m.

Les contraintes

Les contraintes (2-9,14,15, 17-19) sont conservés et les contraintes liées aux retards sont ajoutés aux contraintes précédentes.

- Retard du début de la mission *m* par rapport au jalon ps_m

$$\Delta ps_m \geq s_m - ps_m$$

- Retard du début de la mission *m* par rapport au jalon ls_m

$$\Delta ls_m \geq s_m - ls_m$$

- Retard de la fin de la mission *m* par rapport au jalon dd_m

$$\Delta dd_m \geq s_m + T_m - dd_m$$

- Retard de la fin de la mission *m* par rapport au jalon ld_m

$$\Delta ld_m \geq s_m + T_m - ld_m$$

Fonction objectif

La fonction objectif est subdivisée en trois parties afin de minimiser :

- Les retards (Δps_m, Δls_m, Δdd_m, Δld_m) par rapport aux impératifs temporels visant à respecter les délais.
- Le nombre des missions repoussées (1- y_m); ceci revient à maximiser le nombre de missions prises en charge y_m.
- Le dépassement p_n de consommation énergétique par rapport aux recommandations.

La fonction objective est la somme de ces pénalités qu'on cherche à minimiser :

$$Min \sum_{m \in M} (c_{m1} \Delta ps_m + c_{m2} \Delta ls_m + d_{m1} \Delta dd_m + d_{m2} \Delta ld_m + (1 - y_m) H) + \sum_{n=1..N} p_n P$$

Où *H* est le coût de rejet d'une mission et *P* est le poids lié au dépassement de consommation énergétique.

Formulation

Le problème d'ordonnancement peut être formulé ainsi :

$$Min \sum_{m \in M} (c_{m1} \Delta ps_m + c_{m2} \Delta ls_m + d_{m1} \Delta dd_m + d_{m2} \Delta ld_m + (1 - y_m) H) + \sum_{n=1..N} p_n P \qquad (5\text{-}20)$$

Sous contraintes :

$$x_{nm} = 0 \ if \ f_m \notin C_n \qquad \forall n \in 1..N; m \in M \qquad (5\text{-}21)$$

$$s_i + k_{n,ij}(T_i + A_{ij}) \leq s_j + (1 - k_{n,ij}) G \qquad \forall n = 1..N; \forall i, j \in M \qquad (5\text{-}22)$$

$$0 + k_{n,0m}(T_{n,0} + A_{n,0m}) \leq s_m + (1 - k_{n,0m}) G \qquad \forall n = 1..N; \forall m \in M \qquad (5\text{-}23)$$

$$\sum_{n=1..N} \sum_{i \in M} k_{n,im} + \sum_{n=1..N} k_{n,0m} = y_m \qquad \forall m \in M \qquad (5\text{-}24)$$

$$\sum_{m \in M} x_{nm} E_m \leq W_n + p_n \qquad \forall n = 1...N \qquad (5\text{-}25)$$

$$\sum_{m \in M} x_{nm} E_m \leq E_{n,\max} \qquad \forall n = 1...N \qquad (5\text{-}26)$$

$$k_{n,0m} + \sum_{i \in M} k_{n,im} \le x_{nm} \qquad \forall m \in M, \forall n = 1...N \qquad (5\text{-}27)$$

$$\sum_{i \in M} k_{n,mi} \le x_{nm} \qquad \forall m \in M, \forall n = 1...N \qquad (5\text{-}28)$$

$$\sum_{n=1..N} x_{nm} = y_m \qquad \forall m \in M \qquad (5\text{-}29)$$

$$s_m \ge es_m \qquad \forall m \in M \qquad (5\text{-}30)$$

$$\Delta ps_m \ge s_m - ps_m \qquad \forall m \in M \qquad (5\text{-}31)$$

$$\Delta ls_m \ge s_m - ls_m \qquad \forall m \in M \qquad (5\text{-}32)$$

$$\Delta dd_m \ge s_m + T_m - dd_m \qquad \forall m \in M \qquad (5\text{-}33)$$

$$\Delta ld_m \ge s_m + T_m - ld_m \qquad \forall m \in M \qquad (5\text{-}34)$$

$$\Delta ps_m, \Delta ls_m, \Delta dd_m, \Delta ld_m, t_m \ge 0 \qquad \forall m \in M \qquad (5\text{-}35)$$

$$p_n \ge 0 \qquad \forall n = 1..N \qquad (5\text{-}36)$$

$$x_{nm} \in \{0,1\}; y_m \in \{0,1\} \qquad \forall n \in 1..N; \forall m \in M \qquad (5\text{-}37)$$

$$k_{n,ij} \in \{0,1\} \qquad \forall n \in 1..N; \forall i \in M^* j \in M \qquad (5\text{-}38)$$

5.5. Expérimentations numériques

Dans cette section, nous proposons de résoudre le modèle MIP présenté ci-dessus, afin de confirmer la complexité de ce problème. Ensuite nous proposons de résoudre un scénario à l'aide d'une approche décentralisée, une fois en utilisant la fonction prime par palier comme critère d'évaluation et une autre fois en utilisant la fonction pénalités continues, afin de comparer les réactions des robots et l'impact sur les décisions prises.

5.5.1. Limites du modèle MIP

Le modèle linéaire [20-38] présenté ci-dessus a été formalisé en un programme linéaire en nombres entiers, développé sous MS Visual 2005 en utilisant les librairies C++ de ILOG Concert. Les expérimentations numériques ont été exécutées sur un ordinateur dont le microprocesseur est de 2.4 GHz Intel Core 2 Quad avec 4GB de mémoire RAM et utilisant le solveur commerciale de ILOG Cplex version 11.

#. Robots	#. Missions	#. Fonctionnalités	Temps calcul (sec)
3	10	3	3
3	12	3	54
3	15	3	5276
4	10	3	3
3	10	5	4
4	12	3	6
5	15	4	∞ Infinie

Tableau 12 : Limites du modèles MIP

Ces expérimentations montrent la complexité de ce problème où même les instances de faible taille (5 robots et 15 missions) ne peuvent être résolues à l'optimalité, e qui nous a poussé à implémenter d'autres méthodes de résolution permettant d'obtenir dans un temps acceptable de bonnes solutions à notre problème d'affectation et d'ordonnancement.

5.5.2. Comparaison des critères d'évaluation

Afin de comparer les deux critères d'évaluation « primes par escalier » et les « pénalités continues », ces critères ont été implémentés et utilisés dans la version distribuée de la prise de décision présentée dans le chapitre 7. Nous comparons le comportement d'un groupe robots dans deux types de scénarios, un où les demandes arrivent l'une après l'autre avec un intervalle d'inter arrivée prédéfini (*Equally Distributed*) et un autre type où les demandes arrivent de manière groupée afin de simuler les pics de demande ou bien des situations d'urgence (*Peek Demands*).

Scenario	# Robots	# Missions	Period	Distribution
1	4	30	3600	Equally Distributed
2	4	30	3600	Peek Demands
3	8	90	3600	Equally Distributed
4	8	90	3600	Peek Demands

Tableau 13 : Scénarios étudiés

Deux critères d'évaluation sont étudiés, la fonction primes par escalier (*Step Reward*) et la fonction pénalités continues (*Continuous Penalty*) .

Scenario	Decision Finder	Nb of Handled Missions	Avg Preferred Start Latency	Min Preferred Start Latency	Max Preferred Start Latency	Avg Preferred Finish Latency
1(eq)	Step Reward	28	12.5	-120	600	-9.7
1(eq)	Continuous Penalty	28	12.5	-120	600	-9.7
2(pk)	Step Reward	24	133,2	-120	900	33,4
2(pk)	Continuous Penalty	24	102.8	-120	1233	-17
3(eq)	Step Reward	76	13.23	-120	600	0,6
3(eq)	Continuous Penalty	76	2.5	-120	656	-0.7
4(pk)	Step Reward	72	211,5	-120	900	178,5
4(pk)	Continuous Penalty	72	185,9	-120	1567	169,5

Tableau 14 : comparaison des critères d'évaluation

Les résultats précédents comparant les effets des critères d'évaluation, montrent que les critères sont équivalents dans les situations normales, tandis que la version « pénalités continues » est plus sensibles en cas de pics de demandes puisque cette version est plus sensible aux retards. Et donc en moyenne les retards en utilisant les pénalités continues sont inférieurs par rapport aux retards en utilisant le critère prime.

On note que les retards maximaux prennent en compte seulement les missions exécutées et que nous remarquons dans la version « primes par escalier », certaines missions dont l'exécution est continuellement repoussée puisque la prime d'exécution d'une mission en retard est inferieure à la prime attribuée à l'exécution d'une mission dans les délais souhaités.

Donc par la suite de ce manuscrit la version pénalités continues sera utilisée comme critère d'évaluation.

Chapitre 6. Gestion centralisée d'affectation et d'ordonnancement des missions

Ce chapitre a pour objectif de proposer des algorithmes efficaces pour la gestion centralisée de l'affectation et l'ordonnancement des missions. Le problème a été modélisé mathématiquement dans le chapitre 5. En raison de la complexité du problème et des limitations de l'approche de programmation linéaire du chapitre 5, d'autres approches sont nécessaires.

Notre problème d'affectation et d'ordonnancement est similaire aux problèmes de tournées de véhicules (*Vehicle Routing Problems, VRP*) où les véhicules sont les robots et les clients sont les missions. Il est particulièrement proche des problèmes de tournées de véhicules avec contraintes de temps (*Vehicle Routing Problems with Time Windows*, VRPTW) tel que ceux décrits au chapitre 5. La nécessité de la gestion d'énergie ajoute une couche de complexité supplémentaire à notre problème.

Une étude bibliographique montre que les algorithmes génétiques sont particulièrement adaptés aux problèmes de tournées de véhicules. Dans la suite de ce chapitre, nous proposons un algorithme génétique pour la gestion centralisée d'affectation et d'ordonnancement des missions.

6.1. Algorithmes génétiques et évolutionnaires

L'algorithme génétique est une approche d'optimisation par évolution d'une population de solutions à travers des opérateurs comme le croisement et la mutation. La structure des algorithmes évolutionnaires utilisée dans ce chapitre est illustrée par le pseudo code suivant :

Algorithme génétique:

1. Générer une population initiale de μ solutions admissibles;
2. Coder en chromosomes les solutions de la population initiale;
3. Evaluer la fontion de fitness de chaque solution;
4. Sélectionner λ couples des chromosomes parents pour la reproduction;
5. Appliquer un croisement à chaque couple des parents pour obtenir des chromosomes enfants;
6. Appliquer une mutation à chaque chromosome enfant avec une probabilité de mutation P_m;
7. Décoder les chromosomes enfants;
8. Evaluer la fontion de fitness de chaque solution;
9. Sélectionner μ chromosomes enfants ou parents pour former la nouvelle génération;
10. Répéter les étapes 4-9 jusqu'à l'arrêt.

Algorithme 1 : Algorithmes Génétiques

Dans un algorithme génétique, chaque solution est représentée par un chromosome qui peut être par exemple une suite de lettres d'un alphabet donné. La représentation des solutions en chromosomes dépend fortement des problèmes étudiés et joue un rôle important dans l'efficacité de l'algorithme génétique.

L'algorithme génétique résout un problème d'optimisation à travers l'évolution d'une population de solutions afin d'identifier les caractéristiques communes des bonnes solutions et de diversifier la recherche. Il commence par une population initiale de solutions souvent générées de manière aléatoires avec différentes heuristiques. Après la génération de la population initiale, l'algorithme génétique fait évoluer la population à l'aide de différents opérateurs génétiques. Chaque population est donc appelée une génération.

A chaque itération / génération, chaque nouvelle solution est évaluée afin de déterminer sa qualité. La fonction d'évaluation connue aussi sous le nom *fitness function* ou fonction de fitness peut être soit la valeur du critère à optimiser soit une fonction de la valeur du critère. Nous associons à chaque solution une valeur de fitness *f*. La fonction de fitness est définie comme suit :

$$f = \begin{cases} Critère, & \text{si le gain par palier est utilisé,} \\ 1/Critère, & \text{si la pénalité continue est utilisée} \end{cases}$$

où *Critère* est défini par l'équation (5-1) pour le cas du modèle de gains par palier et par l'équation (5-20) pour le cas du modèle de la pénalité continue.

Dans ce chapitre, chaque solution est représentée de manière indirecte par un chromosome en utilisant un algorithme de codage pour la conversion. Avant l'évaluation d'un chromosome, un algorithme de décodage est alors nécessaire pour convertir un chromosome en une solution explicite et complète d'affectation et d'ordonnancement des missions. Plus précisément, chaque génération est donnée sous la forme de chromosomes qui peuvent être décodés pour représenter une solution qui est un ensemble de planning pour les robots. La fonction fitness d'un chromosome est la somme de la fonction objectif de cette solution qui peut être directement calculé en additionnant tous les retards et les latences selon les méthodes présentées dans le chapitre précédent.

L'évolution de la population commence par la sélection des chromosomes pour la reproduction de nouveaux chromosomes. Les chromosomes sélectionnés sont ensuite croisés selon un opérateur de croisement et ensuite mutés par un opérateur de mutation. Après la phase de mutation et d'évaluation, on applique les opérations de sélection pour le remplacement qui consiste à choisir un sous ensemble des solutions performantes nouvellement générés (Chromosome enfants) pour remplacer une partie des solutions de la génération parent.

Cette boucle générationnelle est relancée plusieurs fois jusqu'à ce que l'un des critères d'arrêt soit atteint qui peut être soit le nombre maximum d'itérations ou les générations, soit le temps de calcul maximal, ou même si une solution optimale est trouvée.

Nous donnons dans la suite de ce chapitre les différents composants de notre algorithme génétique pour la gestion centralisée d'affectation et d'ordonnancement des missions. La génération de la population initiale dépend fortement du problème d'optimisation concerné et ne sera pas détaillé dans ce chapitre. Dans notre étude, nous nous contentons d'une génération aléatoire de solutions.

6.2. Codage génétique des solutions par chromosomes

Comme nous avons indiqué dans le chapitre précédent, une solution de la gestion centralisée d'affectation et d'ordonnancement des missions est définie par (i) l'affectation des missions à l'ensemble des robots, (ii) l'ordre dans lequel chaque robot exécute ses missions, et (iii) la date de début de chaque mission.

Compte tenu des critères retenus dans le chapitre précédent, les dates de début des missions peuvent être déduites directement de (i) et (ii). Connaissant les missions à exécuter par un robot et l'ordre d'exécution, les dates de début des missions correspondent aux dates au plus tôt d'exécution des missions en respectant l'ordre donné et les dates de disponibilités des missions. Plus précisément,

$$s_{[i+1],n} = MAX\left(s_{[i],n} + T_{[i]} + A_{[i],[i+1]}, es_{[i+1]}\right) \qquad (6.1)$$

où [i] est la i-ième mission du robot n, $s_{[i],n}$ sa date de début, $T_{[i]}$ la durée de la mission [i], $A_{[i],[i+1]}$ le temps nécessaire pour aller de la position de fin de la mission [i] à la position de début de la mission [i+1], $es_{[i+1]}$ la date au plus tôt de la mission [i+1].

Pour résumer, une solution d'affectation et d'ordonnancement est complètement caractérisée par l'affectation des missions aux robots et l'ordre de passage des missions sur chaque robot. Elle est dite **admissible** si chaque mission *m* est affectée à un robot *n* équipé de module fonctionnel nécessaire, i.e. $f_m \in C_n$ et chaque robot dispose suffisamment d'autonomie énergétique pour ses missions, i.e. $\sum_{m \in M} x_{nm} E_m \leq E_{n,\max}$.

Afin de simplifier la représentation génétique, dans ce chapitre, nous utilisons une représentation indirecte d'une solution d'affectation et d'ordonnancement. Cette représentation est faite avec un seul chromosome qui est une suite de symboles appartenant à un alphabet L. L'alphabet L contient l'ensemble des missions à affecter plus des symboles particuliers à préciser. Les positions dans le chromosome des missions d'un même robot doivent respecter l'ordre d'exécution de ces missions.

Le processus de transformation d'une solution d'affectation et d'ordonnancement en un chromosome est appelé le codage. A l'inverse, la transformation d'un chromosome en une solution complète d'affectation et d'ordonnancement est appelé décodage. Trois codages et décodages sont utilisés dans ce chapitre.

Codage parallèle avec séparateurs

Cette représentation utilise un symbole spécial "**f**", appelé séparateur, pour indiquer la fin d'un planning d'un robot. Ce symbole est ajouté à la fin de chaque séquence d'exécution. Le chromosome est obtenu par concaténation des séquences d'exécution des différents robots, chacune terminant par le symbole "**f**". Voir la Figure 27 pour une illustration où le chromosome est {2, 5, **f**, 3, 1, 6, **f**, 4, **f**}.

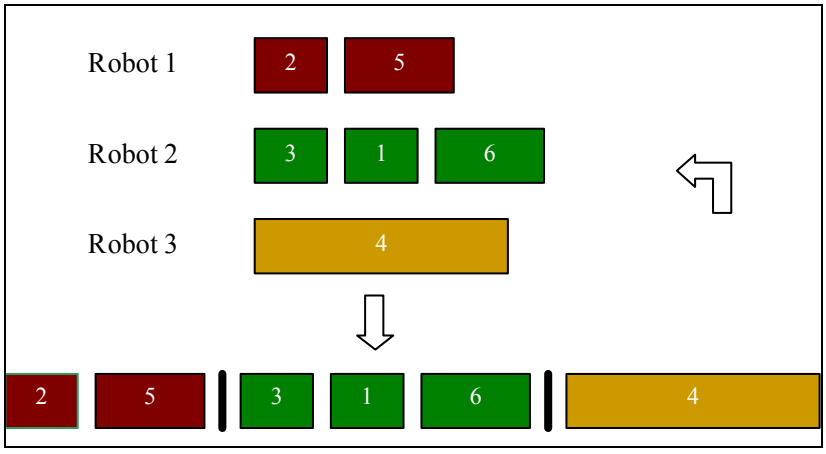

Figure 27 : Chromosome avec séparateurs

Le principal atout de cette représentation est sa simplicité. Un autre avantage de cette représentation est la simplicité de décodage car la fin d'une séquence d'exécution de missions est marquée par un séparateur. Mais d'après la littérature, l'utilisation des séparateurs conduit souvent à des chromosomes de mauvaise qualité.

Codage séquentiel

Dans cette représentation, l'ensemble L de symboles est simplement l'ensemble des missions à planifier. Le codage est très simple, c'est la séquence des missions dans l'ordre de leur date s_m de début et en cas d'égalité dans l'ordre alphabétique du robot l'exécutant. Ce codage est donc une représentation séquentielle dans le temps. La Figure 28 est un exemple de cette représentation que nous appelons codage séquentiel.

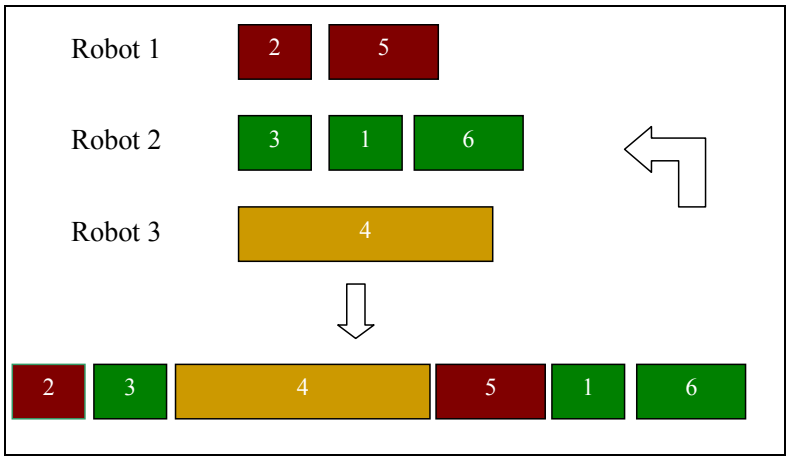

Figure 28 : Codage séquentiel

Le décodage d'un chromosome s'effectue par une heuristique gloutonne. Elle place les missions les unes après les autres dans l'ordre du chromosome. Chaque nouvelle mission est placée sur un robot équipé de module nécessaire qui permet de démarrer la mission au plus tôt tout en respectant les contraintes d'autonomie énergétique. L'inconvénient de cette représentation est que la procédure de décodage ne garantit pas la génération de la même solution initiale du chromosome codé.

Codage séquentiel avec missions fictives

Cette représentation est similaire au codage séquentiel mais permet de préserver l'identité de la solution obtenue par le décodage du chromosome d'une solution initiale. L'ensemble de symboles *L* est l'ensemble des missions à planifier, complété par un symbole "ϕ" représentant des missions fictives. La séquence des missions de chaque robot est complétée par des missions fictives de manière à avoir le même nombre de missions sur tous les robots. Le chromosome est obtenu en triant les missions dans l'ordre: d'abord les premières missions des différents robots dans l'ordre alphabétique des robots, ensuite les deuxièmes missions des différents robots, ainsi de suite. La Figure 29 est un exemple. Les séquences des différents robots sont {2,5,ϕ} pour le robot 1, {3,1,6} pour le robot 2 et {4, ϕ, ϕ} pour le robot 3. Le chromosome final est {2,3,4; 5,1, ϕ; ϕ,6, ϕ}.

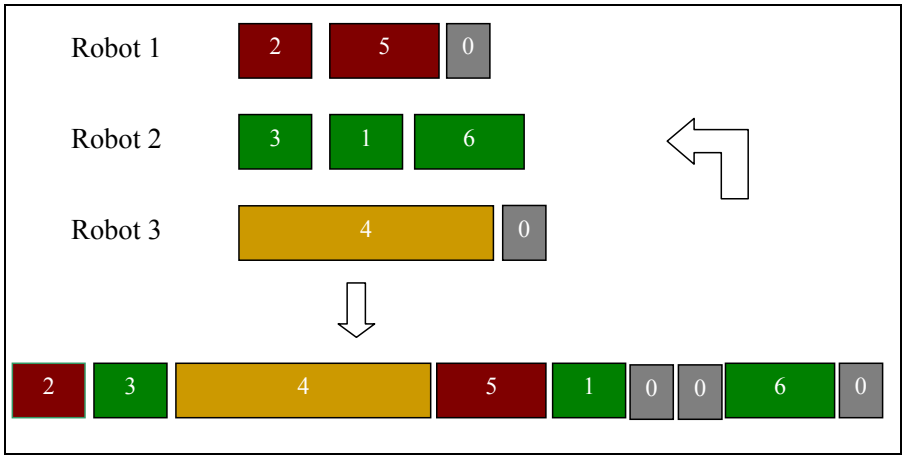

Figure 29 : Codage séquentiel avec missions fictives

Le décodage d'un chromosome de cette représentation est simple. Il suffit d'affecter les missions dans l'ordre du chromosome et chaque fois à un robot différent de manière cyclique. La solution finale est déduite facilement en ignorant les missions fictives.

Notons que, par construction, une solution obtenue par le décodage d'un chromosome séquentiel est nécessairement admissible. Par contre, les solutions obtenues par décodage des chromosomes parallèles avec délimiteurs ou séquentiels avec missions fictives ne sont pas nécessairement admissibles. Il faut pour cela vérifier si chaque mission est affectée à un robot correctement équipé et si les autonomies des robots sont respectées.

6.3. Opérateurs de sélection pour la reproduction

Nous distinguons deux familles d'opérateurs de sélection. Les opérateurs de sélection pour la reproduction sont utilisés pour déterminer les chromosomes parents à croiser pour reproduire la nouvelle génération. Les opérateurs de sélection pour le remplacement servent à éliminer les chromosomes pour former une nouvelle génération.

Parmi les opérateurs de sélection pour la production on distingue deux familles : les sélections proportionnelles et les sélections par tournois.

La roulette

La roulette (*Roulette Wheel Selection* RWS) est une méthode de sélection proportionnelle inspirée du jeu de casino. Cet opérateur a été proposé par Hollande pour les premiers

algorithmes génétiques. Cette méthode consiste à décomposer un disque en μ portions, où chaque portion représente un chromosome parent, en plus la taille de chaque portion est proportionnelle à la performance des individus. Si on choisi un disque dont la circonférence est de taille λ, la circonférence λ_i de la portion d'un individu i de fitness f_i ($i=1..\mu$) peut être donnée par la formule suivante :

$$\lambda_i = \frac{\lambda}{\sum_{i=1..\mu} f_i} f_i$$

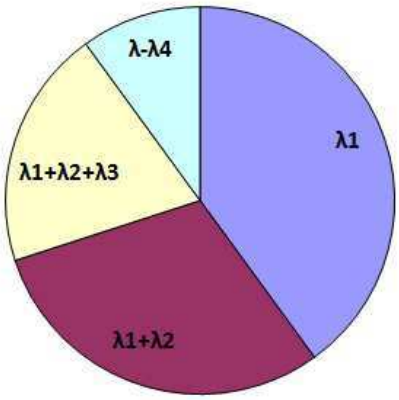

Figure 30 : Roulette

Ensuite, pour effectuer les sélections, on effectue λ tirages de valeurs aléatoires comprises entre 0 et λ. Pour chacun des tirages, la portion pointée est sélectionnée, et ainsi l'individu correspondant est sélectionné pour la reproduction.

Echantillonnage stochastique universel

La méthode RWS nécessite λ tirages, ce qui augmente la probabilité que certains individus soient sélectionnés plusieurs fois au détriment d'autres individus de la population. Pour remédier à cet effet de la roulette, Baker et al. [Baker 87] ont proposé la méthode d'échantillonnage stochastique universel (*Stochastic Universal Sampling SUS*) qui nécessite un seul tirage. Avant de lancer le tirage, la roulette est pointée par λ pointeurs qui sont équidistants les uns des autres, où la distance entre deux pointeurs successifs est de valeur unitaire si la circonférence de la roulette est de valeur λ. Une seule valeur aléatoire λ_0 comprise entre 0 et λ est générée. Cette valeur de λ_0 déterminera la l'angle avec lequel le système de pointeurs pivotera. Après la rotation des pointeurs les λ portions pointées seront sélectionnés et les λ parents correspondants seront sélectionnés pour la reproduction.

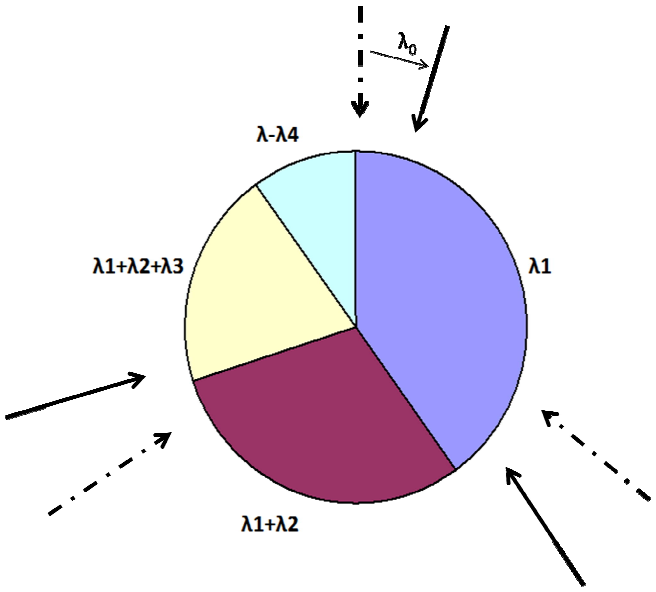

Figure 31 : SUS

Sélection de Boltzmann

Les méthodes RWS et SUS se basent sur les performances de chaque individu pour déterminer la taille de la portion associée, ce qui peut conduire à une convergence prématurée. De la Maza et al. proposent de remplacer les critères de performance f_i utilisés dans les opérateurs de sélection proportionnelle par des critères de performance ajustés selon l'évolution de l'algorithme génétique [de la Maza 93]. Ils proposent un critère de performance ajusté f_i' inspiré par la distribution dite "soft max" de Boltzmann.

$$f_i' = \exp(\frac{f_i}{T})$$

Où T représente la « Température » qui décroit graduellement, passant ainsi d'une sélection équiprobable à une sélection du meilleur individu (avec le plus grand fitness f_i).

Sélection selon le rang des individus

Une autre variante des méthodes de sélection proportionnelle consiste à ranger les individus par ordre croissant de performance, et la fonction de performance ajustée est donnée par la formule suivante :

$$f_i' = \left(1 - \frac{r_i}{\mu}\right)^p$$

où r_i est le rang du chromosome parent i, μ est le nombre d'individus de la population parent et p désigne un paramètre de pression qui détermine la convergence de la population vers les individus ayant les meilleurs performances.

L'avantage de cette méthode est qu'elle ne nécessite pas de calculer la fonction objectif de chaque individu, il suffit de les classer selon des règles prédéfinies, ou bien de les comparer.

Sélection par tournois

Une autre famille des méthodes de sélection, populaire pour sa simplicité d'implémentation, se base sur le principe de tournois. Cette méthode consiste à (i) sélectionner aléatoirement *k* chromosomes dans la population des parents, (ii) sélectionner le meilleur des *k* chromosomes pour la reproduction, et (iii) répéter ce processus jusqu'à la sélection de λ individus pour la reproduction. La sélection par tournois peut être avec remise ou sans remise. Dans la sélection sans remise, un individu sélectionné pour la reproduction est retiré de la population des parents et donc ne peut plus être sélectionné. Dans la sélection avec remise, les individus sélectionnés peuvent toujours être sélectionnés plus tard.

Les tournois stochastiques sont une évolution de ce système. On choisit chaque fois deux chromosomes et on retient aléatoirement l'un des deux individus avec une probabilité *p* plus grande ($0.5 < p < 1$) de sélectionner le meilleur des deux.

6.4. Opérateurs de sélection pour le remplacement

Cet opérateur a pour but de remplacer les μ chromosomes parents de la génération G par un autre ensemble de μ chromosomes enfants. La génération G+1 est constituée d'une part des parents de la génération G et d'autre part des chromosomes enfants générés à la génération G, soit issues de l'ensemble de π individus générés par des heuristiques à l'issue de la boucle générationnelle de l'algorithme génétique.

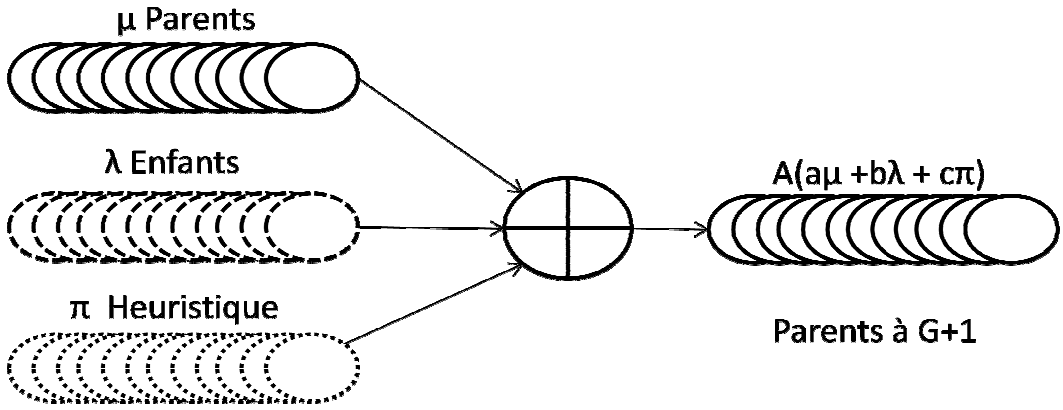

Figure 32 : Sélection pour le remplacement

Remplacement générationnel

Cet opérateur a été utilisé par Holland dans les premiers algorithmes génétiques. Cet opérateur écarte la totalité de la génération des parents, cette dernière est remplacée par la génération des enfants ; en limitant pour cela le nombre λ des chromosomes enfants générés à la taille de la population choisie.

Remplacement déterministe

Pour cette sélection, la population des enfants générés est plus grande que la population des parents ($\lambda > \mu$). Pour cela, les μ meilleurs enfants remplaceront la génération des parents pour former la génération G+1.

Remplacement stationnaire (Steady State)

Cette méthode est utilisée dans le cas où la génération des enfants n'est pas nécessairement meilleure que la génération précédente. Dans cette méthode, la majorité des parents de la

génération *G* est conservée après la reproduction et seulement une minorité des enfants est injectée à la population suivante G+1, ce qui permettra d'atteindre une convergence plus lente sans risquer de perdre la spécificité et la diversité de la génération des parents.

Remplacement élitiste
Dans cet opérateur, la nouvelle génération G+1 est constituée de meilleurs chromosomes parmi l'ensemble des chromosomes enfants et des chromosomes parents. Ainsi, le meilleur individu rencontré dans le passé reste dans la population jusqu'à la détection d'autres solutions strictement meilleures.

Remplacement hybride
Dans cette sélection, la nouvelle génération G+1 est constituée de trois parties : une partie de la population des parents à la génération G, une portion de la population enfants, et des solutions générées par des heuristiques pour compléter la population.

6.5. Opérateurs de Croisements de chromosomes

Les opérateurs de variation consistent à engendrer de nouveaux chromosomes à partir d'un ou de plusieurs chromosomes. L'opérateur de croisement est une variante des opérateurs de variation qui consiste à appliquer une série de transformation à deux chromosomes qu'on appellera chromosomes parents pour engendrer un ou deux chromosomes enfants ayant un mélange des caractéristiques des deux parents.

Ces mécanismes de croisement ont pour but de prendre un sous ensemble de gènes d'un chromosome père et de construire le reste des gènes avec le deuxième chromosome parent de façon à ce que le nouveau chromosome représente une solution réalisable dans notre cas. Le nouveau chromosome devrait contenir toutes les missions et chaque mission peut apparaître une fois et une seule fois dans le chromosome. Différentes techniques de croisement ont été mises en œuvre et la comparaison entre ces techniques peut être trouvée dans les sections suivantes.
Dans notre étude, le croisement dépend du codage utilisé. Pour chaque codage, chaque chromosome est transformé en une permutation d'un ensemble de symboles différents et les opérateurs de croisement pour les problèmes de voyageurs de commerce comme LOX et PMX sont utilisés pour générer de nouvelles permutations et donc de nouvelles solutions.

Pour le codage parallèle avec séparateurs, nous commençons par transformer un chromosome en une permutation d'un ensemble de symboles différents pour différencier les séparateurs. Ainsi le chromosome {2, 5, **f**, 3, 1, 6, **f**, 4, **f**} de la Figure 27 devient {2, 5, $\mathbf{f_1}$, 3, 1, 6, $\mathbf{f_2}$, 4, $\mathbf{f_3}$}.

Algorithme : croisement des chromosomes parallèles avec séparateurs
1. Sélectionner deux parents pour la reproduction;
2. Transformer chaque chromosome en différenciant les séparateurs associés à chaque robot;
3. Appliquer un opérateur de croisement pour générer deux nouvelles permutations;
4. Remplacer les symboles "$\mathbf{f_i}$" par "**f**" pour obtenir deux nouveaux chromosomes;
5. Décoder les nouveaux chromosomes et vérifier l'admissibilité des solutions correspondantes;
6. Ajouter les chromosomes admissibles dans l'ensemble d'enfants.

Algorithme 2 : croisement de chromosomes parallèles avec séparateurs

Pour le codage séquentiel, chaque chromosome est déjà une permutation des symboles différents correspondants à différentes missions. Le croisement est donc plus aisé.

Algorithme : croisement des chromosomes séquentiels

1. Sélectionner deux parents pour la reproduction;
2. Appliquer un opérateur de croisement pour générer deux nouvelles permutations;
3. Décoder les nouveaux chromosomes et retirer ceux n'ayant pas de solution décodée;
4. Ajouter les nouveaux chromosomes restant dans l'ensemble d'enfants.

Algorithme 3 : croisement de chromosomes séquentiels

Pour le codage séquentiel avec missions fictives, les différents chromosomes n'ont pas nécessairement le même nombre de symboles. Le nombre de symboles dans chaque chromosome est multiple de M où M est le nombre de robots. Nous devons ainsi ajouter d'autres missions fictives afin d'avoir des chromosomes de la même longueur. Pour le chromosome {2,3,4; 5,1, ϕ; ϕ,6, ϕ} de la Figure 29, on peut ajouter trois missions fictives, chacune à un robot, pour obtenir un chromosome de longueur 9, i.e. {2,3,4; 5,1, ϕ; ϕ,6,ϕ; ϕ,ϕ,ϕ}. En différentiant les notations des missions fictives dans l'ordre de leur apparition, nous obtenons des permutations d'un même ensemble de symboles. Le chromosome de la Figure 29 devient alors {2,3,4; 5,1, ϕ_1; ϕ_2,6,ϕ_3; ϕ_4,ϕ_4,ϕ_6}.

Algorithme : croisement des chromosomes séquentiels avec missions fictives

1. Sélectionner deux chromosomes parents pour la reproduction;
2. Ajouter des missions fictives au chromosome le plus court pour obtenir deux chromosomes de la même longueur;
3. Transformer les symboles "ϕ" en "ϕ_i" en les numérotant dans l'ordre de leur apparition;
4. Appliquer un opérateur de croisement pour générer deux nouvelles permutations;
5. Remplacer les symboles "ϕ_i" par "ϕ" pour obtenir deux nouveaux chromosomes;
6. Décoder les nouveaux chromosomes en affecter les missions (fictives ou non) aux robots de manière cyclique et vérifier l'admissibilité des solutions correspondantes;
7. Coder en chromosome les solutions admissibles pour supprimer les missions fictives inutiles et pour normaliser les représentations avec les missions fictives à la fin de chaque séquence d'un robot;
8. Ajouter les chromosomes obtenus dans l'ensemble d'enfants.

Algorithme 4 : croisement de chromosomes séquentiels avec missions fictives

Dans la suite, nous présentons les opérateurs que nous utilisons pour le croisement des deux permutations d'un même ensemble de symboles.

Croisement LOX

Le *linéaire Ordre Crossover* (LOX) est l'une des techniques de croisement classiques pour les problèmes de tournées de véhicules. Elle a été introduite par Davis pour le croisement des deux permutations d'un même ensemble de symboles. [Davis 85]

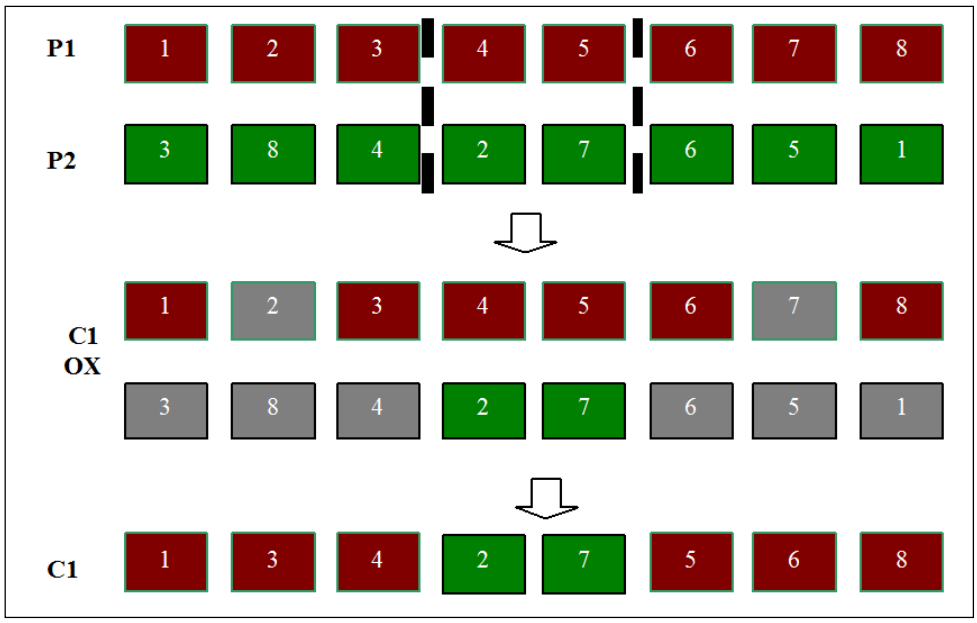

Figure 33 : Croisement LOX

Cette technique illustrée par Figure 33 commence par choisir aléatoirement deux points de coupe. Les symboles figurant de la partie centrale du deuxième parent sont supprimés dans le premier parent. Un nouveau chromosome enfant est obtenu en insérant la partie centrale du parent 2 dans le reste du parent 1. En inversant les deux parents, un autre chromosome enfant peut être généré de manière similaire.

Croisement PMX

La technique de croisement PMX (*Partially Mapped Crossover*) illustrée à la Figure 34, a été présentée par Goldberg et Lingle [Goldberg 85].

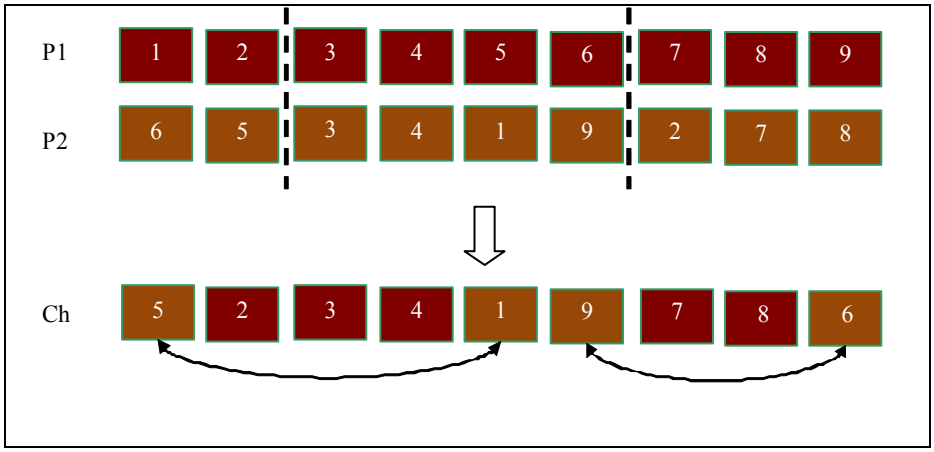

Figure 34 : Croisement PMX

Ce croisement est également défini par deux points de coupe choisis aléatoirement. La partie centrale du deuxième parent est utilisée dans le nouveau chromosome enfant. Le reste du nouveau chromosome vient du chromosome du parent 1 en remplaçant chaque symbole figurant dans la partie centrale du parent 2 par le symbole de la même position du parent 1. Ainsi, dans la Figure 34, le symbole "1" du parent 1 figurant dans la partie centrale du parent 2 à la cinquième position est remplacé par le symbole "5" à la cinquième position du parent 1. De même, en inversant les deux parents, un autre chromosome enfant peut être obtenu.

Croisement Cyclique

Le croisement cyclique [Oliver 87], illustré par la Figure 35, a été proposé par Oliver et al. . Le principe de base de ce croisement est de trouver une série de sous-séquences de gènes, qui seront permutés entre les deux chromosomes parents de façon à ce que les chromosomes enfants ne présentent pas de gènes doublons ni de gènes absents.

Pour cela on détermine un vecteur référence, le vecteur binaire permettant d'identifier les gènes à échanger entre les deux chromosomes.

Afin de déterminer ce vecteur référence, on choisit d'abord aléatoirement un point de départ qui indique un gène quelconque du premier chromosome parent. Ensuite on cherche dans le premier chromosome parent le gène ayant la même valeur que le gène du second parent correspondant au gène de départ. On désigne par gène correspondant, le gène de l'autre chromosome parent, ayant la même position que le gène en question. Chaque fois que l'on effectue un échange, le vecteur référence vaut 1 au gène correspondant. Et le nouveau gène avec qui on vient de faire l'échange est fixé comme le nouveau point de départ. Ce processus est répété jusqu'à retrouver le point de départ.

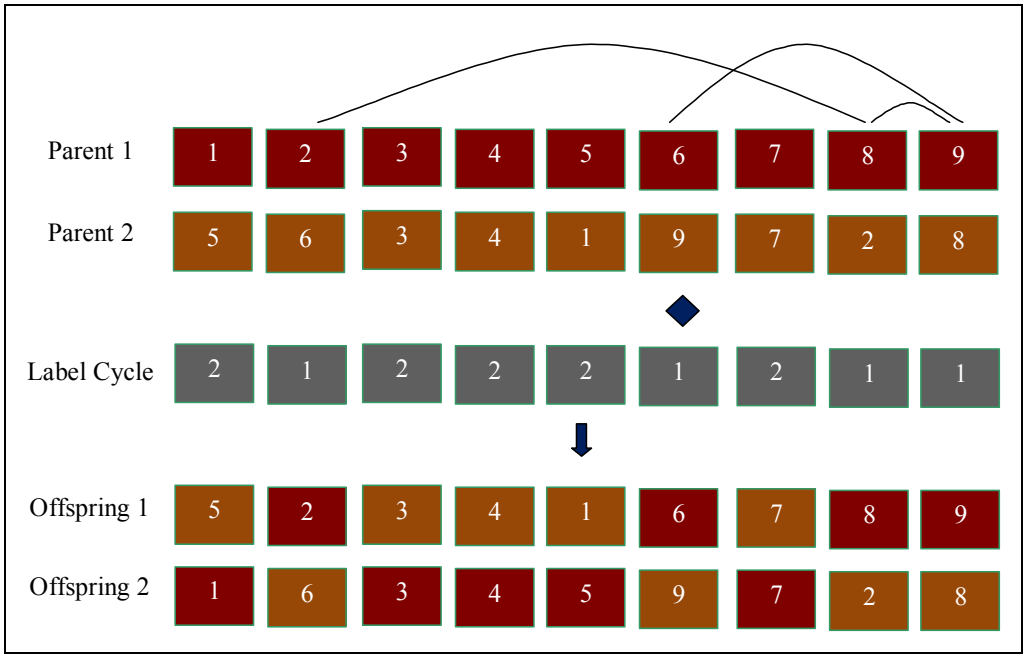

Figure 35 : Croisement cyclique

Par exemple, dans la Figure 35, le point de départ est le 6ème élément ayant la valeur 6 dans le premier chromosome parent et le gène correspondant du second chromosome ait la valeur 9. Donc on cherche le gène du premier chromosome qui a la valeur 9. Dans notre exemple, c'est le 9ème gène et on effectue une permutation entre les $6^{èmes}$ gènes et les $9^{èmes}$ gènes En suivant le même principe on effectue un échange entre les $9^{èmes}$ et les $8^{èmes}$, puis entre les $8^{èmes}$ et les $2^{èmes}$ où on trouve le gène correspondant ayant la valeur de départ 6 ce qui marque le fin du processus d'échange.

Deux chromosomes enfants sont ensuite construits à l'aide du vecteur de référence. Le premier chromosome est obtenu en choisissant pour chaque position le gène du parent 1 (resp. 2) si le vecteur de référence vaut 1. Le deuxième chromosome est obtenu par le choix inverse.

Croisement intelligent

Cette famille de croisements combine les méthodes de croisement précédentes. La première méthode de combinaisons, appelée croisement aléatoire, choisit au hasard à chaque génération un des opérateurs de croisement. Cette opération s'est avérée efficace pour certaines instances et donne des solutions plus performantes.

Une amélioration de cette combinaison aléatoire consiste à attribuer à chaque opérateur de croisement un indicateur de performance et à choisir les opérateurs de croisement avec des probabilités proportionnelles à leur indicateur de performance. Ainsi, plus l'indicateur de performance d'un croisement est élevé, plus il a de chance d'être choisi à la prochaine génération. A la fin de chaque itération, l'indicateur de performance de l'opérateur choisi est ajusté en fonction des résultats de la nouvelle génération par rapport à l'ancienne génération. L'indicateur de performance est borné dans un intervalle donné afin de permettre à tous les opérateurs d'être choisis.

6.6. Opérateur de Mutations

L'opérateur de mutation fait partie des opérateurs de variation, comme l'opérateur de croissement. Ces opérateurs effectuent des modifications sur les chromosomes et donne naissance à de nouveaux chromosomes. L'opérateur de mutation effectue des variations sur un seul chromosome et nécessite la connaissance d'un seul chromosome pour effectuer ces variations. Les variations classiques comme *bit flop* pour les chromosomes binaires et la mutation *2-opt* avec les représentations ordinales du problème de voyageurs de commerce ne semblent pas pertinentes pour notre représentation. Nous proposons les mutations suivantes.

Mutation aléatoire

La mutation aléatoire effectue un échange entre deux gènes choisis aléatoirement du chromosome considéré. Après la mutation, le chromosome est décodé et l'admissibilité vérifiée avant l'introduction dans la population des chromosomes enfants.

Optimisation locale

Une autre variante de cet opérateur consiste à décoder le chromosome et ensuite effectuer une optimisation locale de la solution obtenue. Cette optimisation locale peut s'avérer gourmande en temps de calcul ce qui affecte le temps de calcul d'une boucle génératrice.

Dans notre étude, l'optimisation locale tente d'améliorer le planning d'un robot en appliquant une combinaison des opérations suivantes :

- Déplacer une mission sélectionnée aléatoirement à une position choisie aléatoirement,
- Echanger les positions des deux missions sélectionnées aléatoirement.

Une amélioration serait d'appliquer les heuristiques proposées dans le chapitre précédent afin d'améliorer le planning de chaque robot, dans le cas où seulement quelques missions existent dans le planning du robot (moins de quatre missions), une énumération de toutes les combinaisons possibles peut être envisageable.

6.7. Comparaisons et Résultats Numériques.

6.7.1. Performances des opérateurs de croisement

Nous considérons une instance composée de 7 robots, 49 missions et 4 modules de fonctionnalité. L'algorithme génétique est limité à 1000 générations et à 5 min de temps de calcul.

Quatre approches de croisement sont comparées (LOX, PMX, Cyclique, et le croisement aléatoire). La Figure 36 montre l'évolution de la meilleure solution de la population d'une génération à une autre.

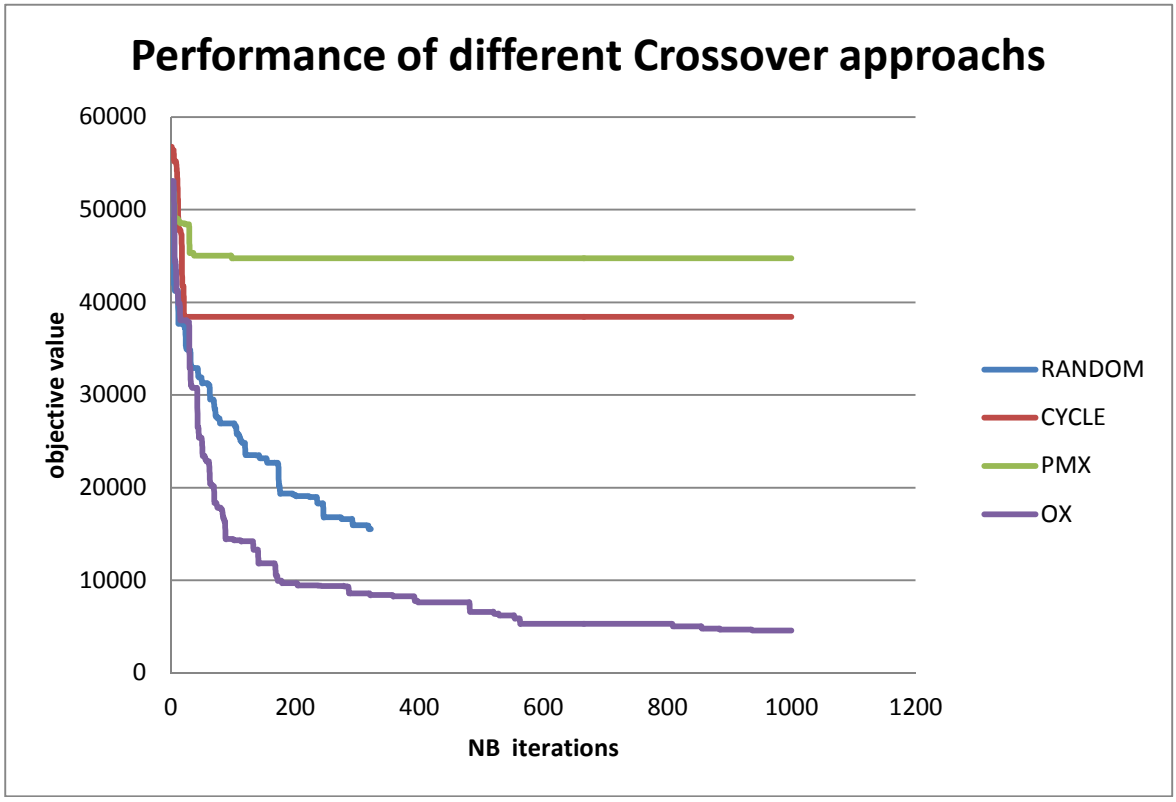

Figure 36 : **Performance des opérateurs de croisement**

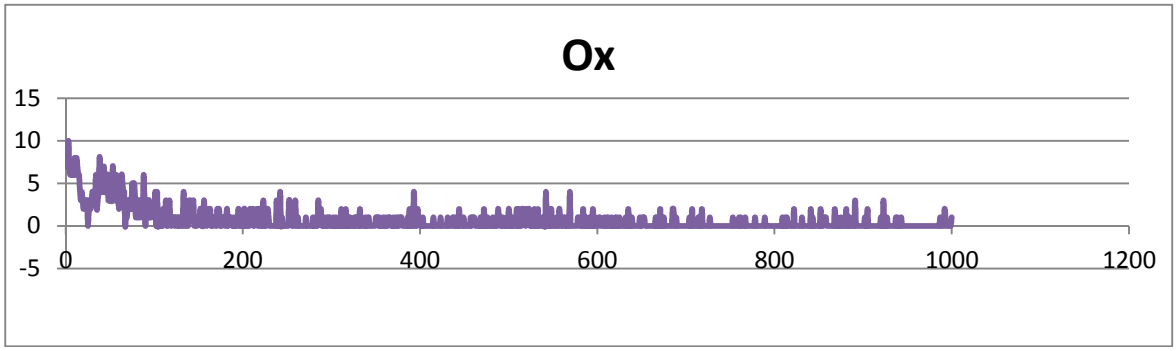

Figure 37 : Nombre de chromosome remplacés par croisement LOX

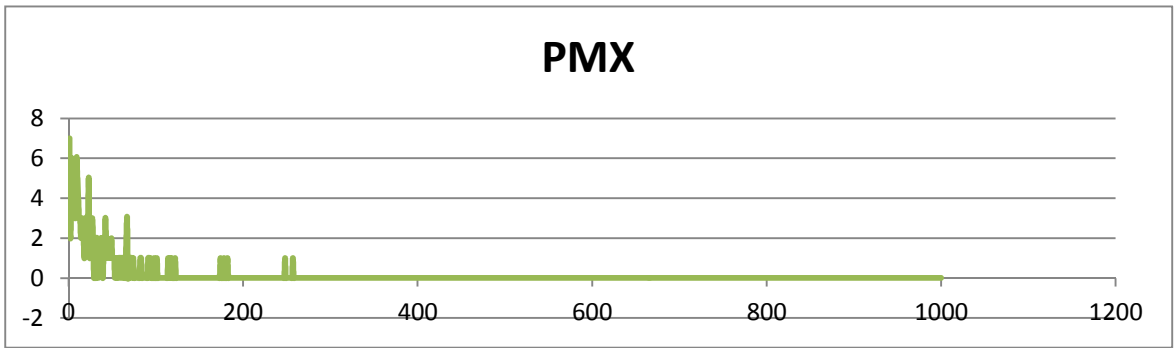

Figure 38 : Nombre de chromosome remplacés par croisement PMX

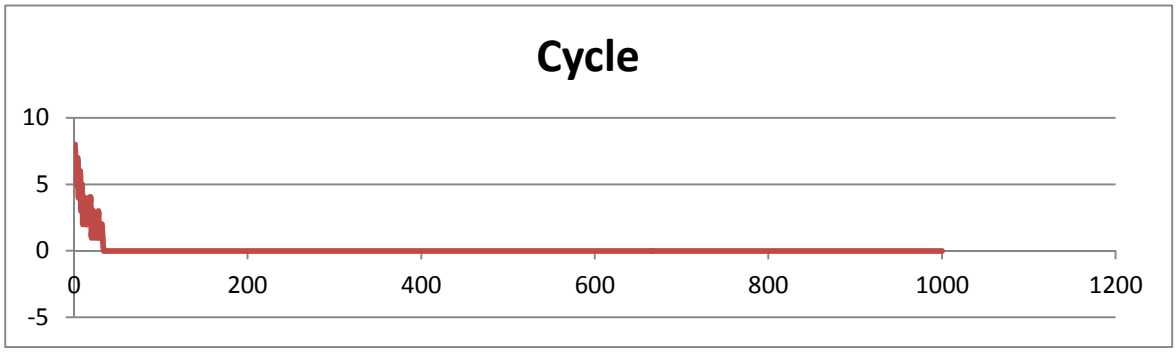

Figure 39 : Nombre de chromosome remplacés par croisement cyclique

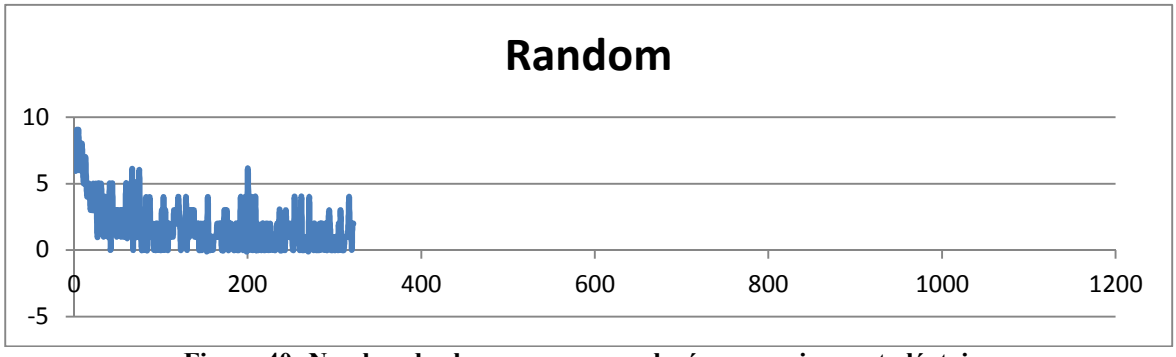

Figure 40: Nombre de chromosome remplacés par croisement aléatoire

La comparaison, illustrée par la Figure 36, entre les performances des différentes techniques de croisement appliquées sur la même instance, montre clairement la prépondérance de l'opérateur de croissement LOX par rapport aux autres opérateurs de croissement. Les graphes (Figure 37 ;Figure 38 ; Figure 39 ; Figure 40) présentent le nombre des descendants générés à chaque itération et qui sont plus performants que certains individus parents.

6.7.2. Performances des opérateurs de mutation optimisés

L'instance à examiner dans cette section se compose de 6 robots, 78 missions et 3 modules fonctionnalité. Cet algorithme à été limité à 1000 itérations au maximum, soit à un temps de calcul de 5 minutes. Les quatre approches de croisement présentées à la section précédente (LOX, PMX, Cyclique, et le croisement aléatoire qui est un mélange des trois précédents) sont testées une fois avec amélioration locale et une fois avec mutation génétique simple.
Le tableau suivant illustre les performances de chaque approche avec la performance de la meilleure solution trouvée pour notre problème de minimisation de coût /retards, le nombre des descendants améliorés et le critère d'arrêt qui a été abouti en premier

Approche	Meilleur Solution	# Iterations	CPU	Nombre Descendants.
LOX	26424	1000	47	1029
LOX + opt	6314	47	304	214
PMX	124392	1000	52	211
PMX+opt	5582	297	300	356
Cycle	148407	1000	53	240
Cycle + opt	8316	398	300	201
Mix	69475	1000	47	1035
Mix + opt	5396	85	303	355

Tableau 15 : Performances des différents opérateurs

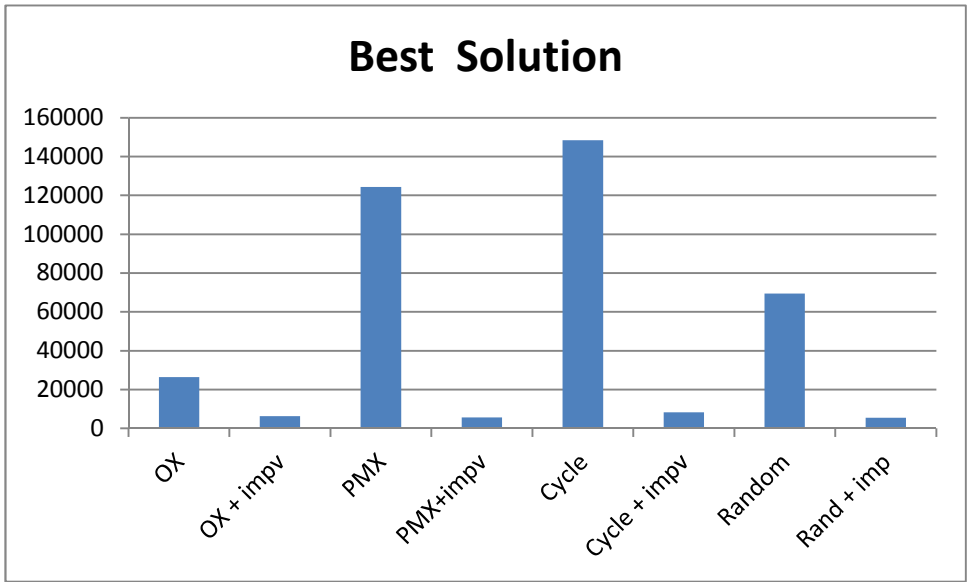

Figure 41 : Meilleure solution par approche

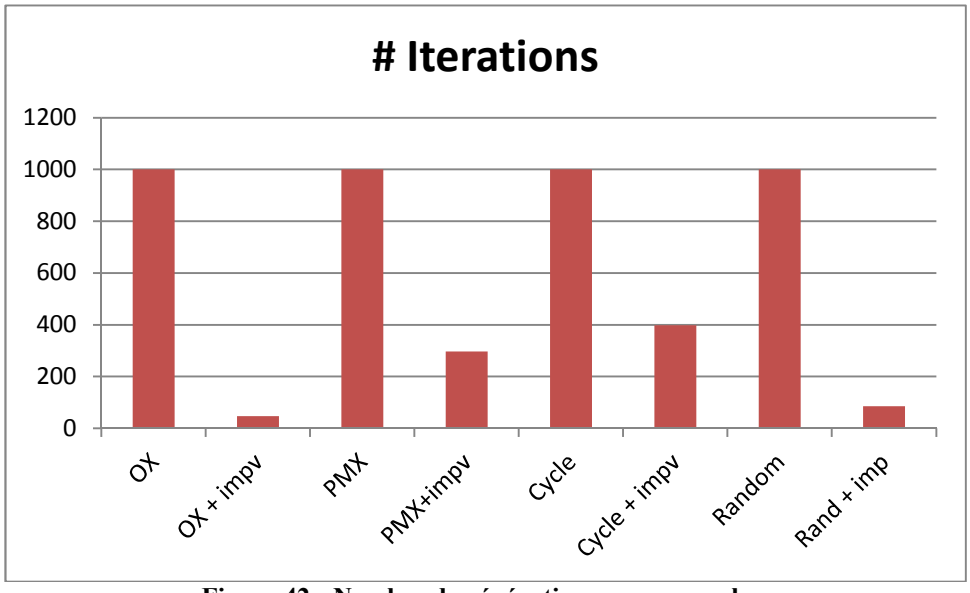

Figure 42 : Nombre de générations par approches

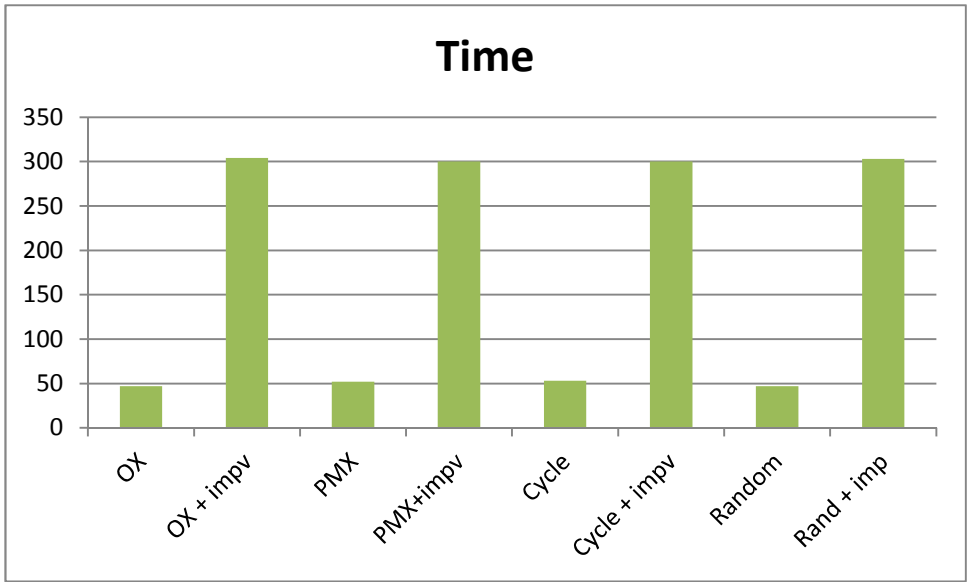

Figure 43 : Temps de calcul par approche

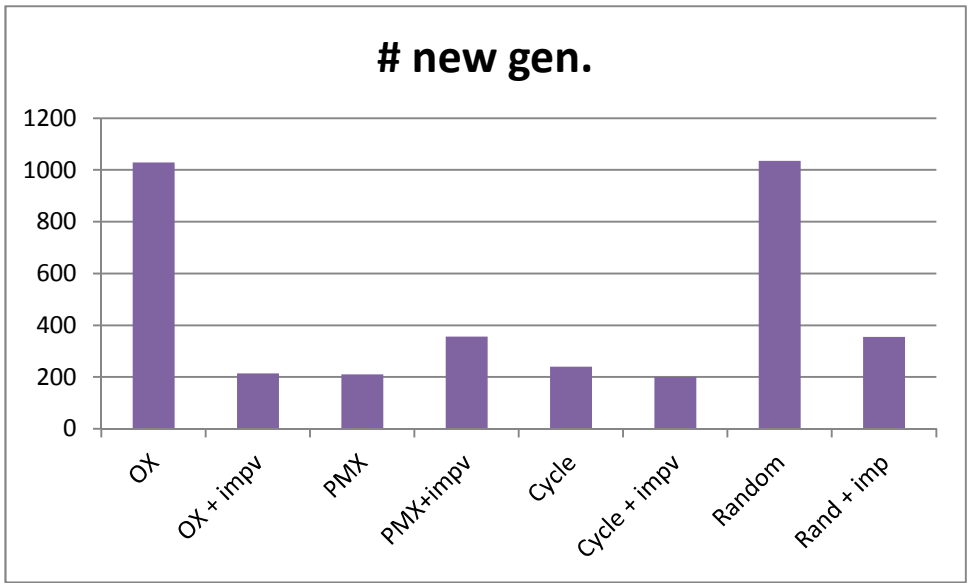

Figure 44 : Nombre total de descendants retenus pour les futures générations.

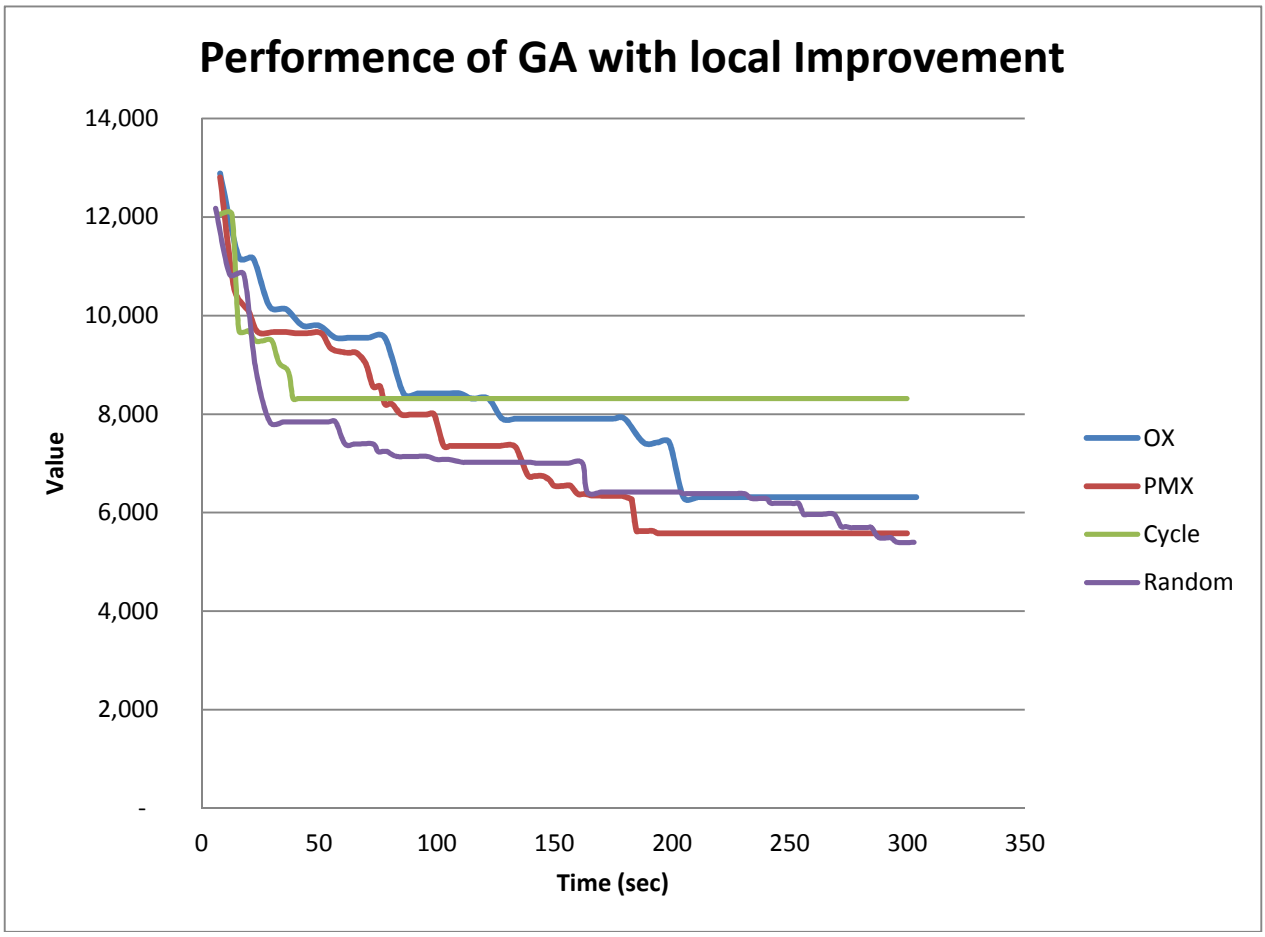

Figure 45 : Convergence des approches de croisements avec optimisation

D'après les résultats ci-dessus, nous pouvons déduire que les algorithmes génétiques couplés avec une optimisation locale offrent de meilleures solutions, mais on note que le processus d'optimisation est gourmand en puissance de calcul. La Figure 41 montre clairement que les instances résolues avec une optimisation locale surpassent les techniques sans optimisation locale par un facteur d'au moins 1 à 5, mais l'inconvénient majeur de cette approche est sa gourmandise en puissance de traitement qui multiplie par six le temps de calcul d'une boucle générationnelle. La Figure 45 présente une comparaison entre les quatre approches de croisements couplés par une optimisation locale, mais de façon surprenante : le croisement aléatoire PMX surpasse l'opérateur de croisement LOX en contradiction avec les résultats de la section précédente. Ces résultats nous ont poussés à mettre en place une technique de croisement intelligent qui est une évolution du croisement aléatoire, qui s'adapte avec l'instance étudiée en fonction des performances de chaque approche.

6.8. Conclusions

Dans ce chapitre on a proposé une implémentation centralisée de l'agent de décision responsable de l'affectation et du planning des missions qui peut être considéré comme une variante de deux problèmes difficiles : le VRP-TW et le MC-VRP. L'implémentation proposée se base sur des algorithmes évolutionnaires couplés par des méthodes d'optimisation locale. Plusieurs opérateurs ont été testés pour identifier les opérateurs les mieux adaptés à notre problème notamment les opérateurs de croisement, les opérateurs de mutation et les opérateurs de sélection. Des approches inspirées des techniques d'apprentissage qui

s’adaptent avec l’évolution des individus et des générations ont été mises en place. Un système de sélection hybride qui permet d’enrichir les générations par des individus générés par des heuristiques lancées en amont avec les algorithmes évolutionnaires a été également mis en place.

Le problème de pré-planification des missions récurrentes est un des problèmes identifiés au début du projet. Il nous semble prometteur de recourir à une résolution par un agent centralisé semblable a celui présenté dans ce chapitre. Ce problème consiste à affecter un semi-planning prédéfini à un système de N robots ayant des configurations différentes et des recommandations prédéfinies de consommation d'énergie. Les missions sont divisées en deux catégories : (i) les missions de routine qui sont régulières et connues à l'avance, comme le nettoyage régulier ou le transport régulier des médicaments (on sait à l'avance le calendrier de cette catégorie de missions). (ii) Les missions inattendues, où seule une estimation approximative de l’ensemble des missions peut être prévue. Le but de ce problème serait de pré-planifier les missions de routine et de réserver des plages sur chaque robot pour les missions inattendues.

Chapitre 7. Gestion distribuée d'affectation et d'ordonnancement des missions

Ce chapitre présente une gestion décentralisée d'affectation et d'ordonnancement en ligne des missions. Chaque robot agit comme un agent indépendant, responsable du planning et de l'ordonnancement des missions affectées au robot. La gestion distribuée allège le système de communication inter-robots en supprimant un agent central de décision qui doit être informé systématiquement de l'état de chaque robot ainsi que de l'état des missions afin d'avoir des données fiables pour l'optimisation des plannings des robots. Sachant que les robots sont mobiles et ne peuvent pas être toujours connectés aux réseaux de communications, la gestion distribuée permet d'augmenter la fiabilité du système et d'alléger le calcul effectué par l'agent central de l'approche centralisée étudiée au chapitre précédent et présenté au [Baalbaki 10 a].

Dans ce chapitre, nous présentons une approche décentralisée et distribuée (publiée dans [Baalbaki 10 b]). Notre travail se base sur les algorithmes d'ordonnancement distribué à l'aide des enchères (*market-based distributed scheduling)*. Cette approche a été utilisée dans Walsh et al. [Walsh 98] , Kiekintveld et al [Kiekintveld 06] et Wellman et al. [Wellman 05] pour des problèmes de chaînes d'approvisionnement. Tous ces travaux sont fondés sur des système d'agents virtuels. L'approche des agents matérialisés (*embodied agents*) de Gerkey et al. [Gerkey 02] nous semblait plus pertinente pour la gestion décentralisée d'un système robotique. Les travaux de Gerkey et al. [Gerkey 02][Gerkey 03] décrivent un système d'allocation dynamique de tâches comme une variante du système négociation (*contract network protocol*)de Smith et al [Smith 83], leur approche est basée sur un système de diffusion de messages utilisant le mécanisme d'abonnement et de publication, ventes aux enchères entre priseurs et entrepreneurs et une structure hiérarchique de tâches.

Dans ce chapitre nous présentons l'architecture de la gestion distribuée, les entités clés ainsi que les événements. Ensuite nous exposerons le séquenceur qui permettra d'établir des plannings pour les robots et de les évaluer. Pour finir, nous détaillerons le processus de négociation, ainsi que la structure de la simulation à événements discrets qui sera utilisée pour comparer les différentes approches.

7.1. Architecture de gestion distribuée

Dans cette section, nous présenterons les différents composants du système de prise de décisions distribuées implémentés sur les robots. L'architecture adoptée est celle des composants (*components*) qui découple les caractéristiques de chaque composant et permet une analyse indépendante des différents niveaux de complexité. Cette approche permet ainsi une compréhension globale du comportement du système. Cette même architecture à été utilisée pour la simulation afin de comparer le système distribué et le système centralisé de prise de décisions.

Architecture de communication

Les robots communiquent entre eux via un réseau de pairs (*peer network*) basé sur une topologie en anneau (*token ring*). Le réseau de communication inter-pair offre trois types de communication : la diffusion (*broadcast),* la diffusion partielle (*multicast*) ainsi que la communication directe (*point to point*).

La gestion distribuée que nous avons adoptée, peut être considéré comme une extension de l'approche de [Gerkey 02] en introduisant la notion des «connaissances partagées» (*Shared Knowledge*) accessibles et stockées sur chaque pair. Les connaissances partagées contiennent des informations sur la configuration et la position de chaque robot. En outre, l'état d'exécution de chaque mission est continuellement mis à jour et les bases de données des connaissances partagées sont automatiquement synchronisées.

Le contrôleur

Le contrôleur, l'équivalent du commissaire priseur dans le système des enchères inversées, gère les demandes de nouvelles missions, engage et contrôle le processus de négociation. En outre, le contrôleur vérifie continuellement l'ensemble de pairs actifs et prend les mesures nécessaires afin de garantir le bon fonctionnement du système en cas de pannes des robots. Pour éviter la défaillance du contrôleur, cette entité n'est pas assurée par un robot prédéfini. Tous les pairs ont la capacité d'assurer ce rôle et le contrôleur est choisi de manière dynamique à chaque fois qu'un pair rejoint ou bien quitte le réseau.

Agent de décisions distribuées

L'agent de décision, l'équivalent de l'entrepreneur dans un système d'enchères inversées, est le composant de décision le plus élevé sur chaque robot. Il assure la communication et la gestion des autres composants sur le robot. Il contrôle l'exécution des missions affectées au robot. Il calcule du coût d'exécution d'une nouvelle mission en se basant sur les informations fournies par les autres composants du robot comme la navigation et le l'exécuteur. Le processus d'évaluation du coût d'une nouvelle mission est détaillé dans la section suivante.

Exécuteur de missions

Ce composant est responsable de l'exécution des missions affectées à un robot. Il est en étroite liaison avec les agents de décision des robots qui établissent le planning. Ce composant s'appuie sur les composants mécaniques (*hardware*) et les drivers adéquats pour exécuter les missions.

Dans la simulation ce composant est replacé par un module qui simule la navigation des robots et les différents états des robots.

Evènements

Le système de prise de décision distribuée est conçu pour réagir à l'occurrence de différents évènements. Ces évènements déclenchent la planification ou bien la replanification des missions actives.

Les principaux évènements considérés dans la prise de décision distribuée sont :

- Arrivée des nouvelles demandes de missions.
- Arrivée d'un nouveau robot, rejoignant le groupe.
- Départ d'un robot.
- Détection des retards majeurs par rapport au planning initial comme les retards causés par des obstacles divers rencontrés par un robot,
- Evénements internes propagés par les composants de bas niveau du robot comme la fin d'une mission, alarme du niveau critique de la batterie du robot émis par le composant de gestion d'autonomie locale.

Procédures de prise en charge des événements

Les événements sont regroupés en trois catégories et pris en charge selon trois procédures de décision distribuée différentes.

Affectation des missions : Cette procédure est initiée lorsqu'un utilisateur final commande une nouvelle mission. Cette commande est acheminée jusqu'au contrôleur via le réseau de communication. A la réception d'une nouvelle commande, le contrôleur prépare une enchère et diffusent les détails de la mission aux différents robots actifs. Ensuite chaque robot évalue localement le coût d'exécution de la mission et envoie cette évaluation au contrôleur. Après avoir reçu les réponses des différents robots, le contrôleur étudie les réponses et désigne un robot gagnant auquel la mission sera affectée.

Legs des missions : Cette procédure est initiée par un robot confrontant à des difficultés d'exécution des missions. Ces difficultés peuvent être techniques (problème d'énergie, panne d'une module) ou bien environnementales (obstacles obstruant leur chemin) et empêchent les robots d'exécuter leurs missions dans les délais raisonnables. On peut également recourir à cette procédure s'il existe un autre robot dans la proximité capable d'exécuter en parallèle deux missions. En cas de difficulté d'exécution des missions, le robot recalcule ses coûts d'exécution des missions et diffuse ces nouveaux coûts aux différents pairs. Ensuite, les autres robots vont jouer le rôle des entrepreneurs. Chaque robot vérifie s'il peut exécuter chaque mission en calculant le coût d'exécution et répond au robot en difficulté en donnant son coût pour chaque mission. A la fin, le robot émetteur de la demande, qui joue le rôle du commissaire priseur, étudie les réponses et prenne la décision de céder une mission si une solution plus avantageuse est identifiée.

Réquisition des missions : Cette procédure est initiée lorsqu'un nouveau robot rejoint le groupe des robots actifs. Il émet une demande de réquisition à ses pairs qui répondent en envoyant les listes des missions qui leur sont affectées ainsi que les détails et le coût d'exécution de chaque mission. Cette procédure peut être déclenchée par un robot terminant toutes ses missions et devenant ainsi libre. Cela permet d'alléger la charge de travail des autres robots.
A la réception des réponses des autres robots, le robot initiateur joue le rôle du commissaire priseur et réquisitionne une ou plusieurs missions qu'il estime judicieux de la(les) réquisitionner et de l'exécuter par lui-même afin d'équilibrer la charge de travail et de minimiser les retards.

7.2. Agent de décision et évaluation

L'agent de décision distribuée est un composant doté de la faculté de communication avec les autres robots afin de collaborer et de prendre une décision collective et de manière distribuée. C'est également cet agent qui transmet les décisions aux composants de bas niveau afin d'être exécutées.
Les éléments clé de cet agent de décision sont : un évaluateur, un séquenceur et un planificateur qui ont pour rôle d'identifier les plannings les plus adaptés et les moins coûteux.

Lors de l'arrivée d'une demande d'affectation / Legs / réquisition, l'agent de décision utilise le séquenceur pour générer plusieurs séquences, le planificateur pour transformer ces séquences en plannings et ensuite l'évaluateur pour déterminer la performance de chacun des plans d'exécution.

En cas d'une affectation effective de nouvelles missions, l'agent de décision transmet le nouveau planning au moteur d'exécution pour établir le plan détaillé le plus performant.

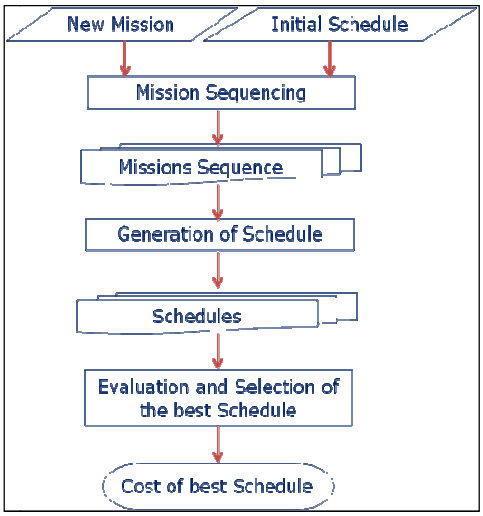

Figure 46: Eléments clé de l'agent de décision et d'évaluation

Nous détaillons maintenant le fonctionnement de chacun des trois éléments.

Le Séquenceur : Lors la demande d'une nouvelle négociation pour l'exécution d'une nouvelle mission, l'agent de décision demande au séquenceur de générer des séquences d'exécution de missions intégrant la nouvelle mission et le planning actuel. Dans de le contexte du projet IWARD, pour des raisons de simplicité de calcul, nous ne remettons pas en cause le séquencement des missions du planning actuel. Il s'agit alors de simple insertion d'une nouvelle mission. Cela donne alors $m+1$ séquences pour un planning actuel de m missions correspondant respectivement à l'insertion au début ou après une mission déjà affectée au robot. Nous autorisons l'interruption de la mission en cours pour exécuter d'abord la nouvelle mission avant de reprendre la mission interrompue. Evidemment cela dépend de la possibilité d'interruption de la mission en cours.

Le planificateur : Pour chaque séquence fournie par le séquenceur, le planificateur établit un planning prévisionnel ou plutôt un ordonnancement. Un planning prévisionnel est construit en estimant la date de début et la date de fin des missions dans la séquence. Le planning prévisionnel doit prendre en compte la configuration actuelle du robot, sa position initiale, la vitesse du robot, les plans de navigation ainsi que les spécifications des missions.

Afin de simplifier l'implémentation de ce planificateur, l'exécution en parallèle des missions n'est pas considérée et les missions sont donc exécutées les unes après les autres dans l'ordre de la séquence. De même les missions multi-robot nécessitant une synchronisation (coopération) entre les robots ne sont pas prises en compte.

La règle d'ordonnancement est une simple variante de la règle d'ordonnancement au plus tôt. A la fin d'une mission, le planificateur estime avec l'aide du système de navigation sa date d'arrivée au point de départ de la mission suivante si le robot y va directement. Si cette date d'arrivée est largement inférieure à la date au plus tôt es_m de la mission, une mission fictive est insérée dans le planning pour faire attendre le robot dans une station d'attente affectée au robot. L'insertion de cette mission fictive permet de ne pas encombrer les lieux avec un robot

inactif. Cette mission fictive prend fin de façon à ce que le robot puisse arriver au point de départ de la mission réelle avant la date au plus tôt es_m.

L'évaluateur : Pour évaluer un planning d'un robot, nous appliquons les mêmes fonctions de performance présentées précédemment dans le chapitre 5. Etant donnée la stratégie réactive adoptée dans ce chapitre, nous mesurons l'effet engendré soit par l'ajout soit par la suppression d'une mission. Pour évaluer ces effets sur un robot *n,* nous comparons la mesure de performance, notée C_n^M , du planning avec un ensemble *M* de missions et la performance, notée $C_n^{M+m'}$, du meilleur planning pour l'ajout d'une mission *m'* parmi l'ensemble des plannings identifiés par le séquenceur.

De ces deux mesures, nous pouvons tirer une troisième mesure $\Delta C_n^{m'} = C_n^{M+m'} - C_n^M$, correspondant au cout différentiel dû à l'ajout d'une mission *m'*. Le couple $(\Delta C_n^{m'}, C_n^{M+m'})$ est essentiel dans la négociation entre les robots pour estimer le coût d'une affectation ou d'une réaffectation d'un robot à un autre.

7.3. Processus de négociation

7.3.1. Affectation des nouvelles missions

À l'arrivée d'une nouvelle mission, le serveur de communication achemine la commande au robot jouant le rôle du contrôleur. Ce dernier commence le processus de négociation en demandant à chaque robot le coût d'exécution de cette nouvelle mission. Si un robot est équipé des modules nécessaires pour l'exécution la mission, il répond au contrôleur, comme le montre la Figure 47, en lui envoyant le coût d'exécution de la mission en question. Si le robot n'est pas doté des fonctionnalités appropriées, il simule une reconfiguration avant l'exécution de la mission et renvoie les coûts liés à la reconfiguration et l'exécution de la mission.

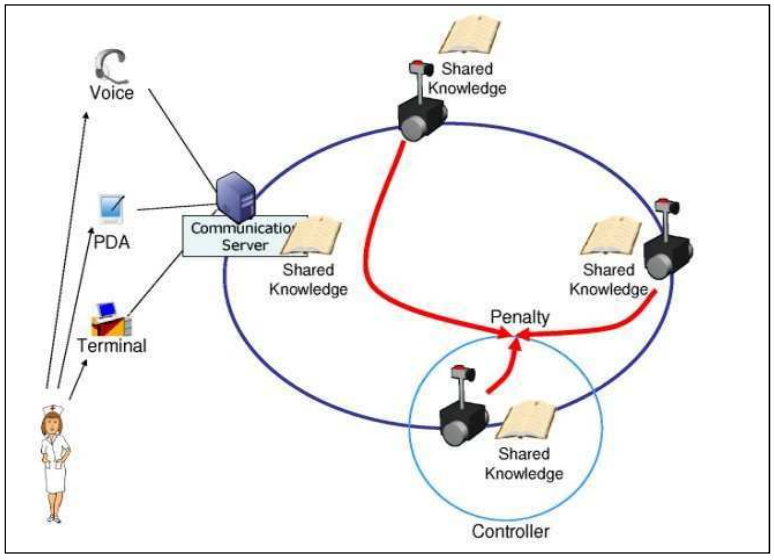

Figure 47: Réception des coûts d'exécution

En fonction des réponses de différents robots au bout d'un temps donné, le contrôleur affecte la mission au robot permettant de procurer le plus grand bénéfice du système selon l'Algorithme 5 ci-dessous.

```
$rq=constructor_mission_cost_request( $mission, $myself);
Broadcast_request($rq,ALL_ROBOTS);
$time_limit=current_time() + 3;
$pl=Create_replies_pile();

//Read replies
While current_time<$time_limit Do
      $answer=get_replies_for_request($rq);
      Stack($answer,$pl);
      Sleep;
End While

//Sort Replies
$best_reply=pop($pl);
While $pl is not empty Do
      $tmp=pop($pl);
      If $tmp is_better_than $best_reply Then
            $best_reply=$tmp;
      End if
End while

//Assign the mission to the winner
Assign_mission($mission,$best_reply->idSender);
```

Algorithme 5 : Affectation d'une mission par le contrôleur

Durant le processus d'affectation de mission, plusieurs messages sont échangés entre les différents pairs, ces messages, illustrés par la Figure 48, peuvent être classifiés en trois types:

- Demande de calcul de coût (*MissionCostRequest*) : L'arrivée d'une nouvelle demande, le contrôleur diffuse, à tous ses pairs robots une *MissionCostRequest* demandant le coût d'exécution de la nouvelle mission. La structure de ce message est composée des éléments suivants: (i) identifiant de l'émetteur (Identifiant du contrôleur), (ii) identifiant du destinataire (fixé à ALL_ROBOTS), (iii) date de lancement du processus de négociation (*timestamp* du moment où la demande a été diffusé), (iv) temps restant alloué à la négociation avant de la prise de décision et la détermination du gagnant, (v) identifiant de la mission, (vi) spécifications détaillées de la mission.
- Répliques du calcul de coût (*MissionCostReply*) : Chaque robot du groupe, après avoir calculé son coût d'exécution de la mission, transmet au contrôleur sa réplique. Ce message est composé des champs suivants : (i) identifiant de la mission, (ii) identifiant du robot émetteur, (iii) identifiant du destinataire (identifiant du contrôleur), (iv) indicateur booléen indiquant si robot est doté des fonctionnalités nécessaires, (v) coût nécessaire pour l'exécution de la mission par ce robot.
- Affectation (*MissionAssignment*) : Une fois que le robot gagnant pour l'exécution de la mission est identifié (celui avec le moindre coût), le contrôleur affecte la mission au gagnant de l'enchère. Le propriétaire de la mission est délégué au nouveau robot après une affectation provisoire au contrôleur durant le processus de négociation. La structure du message d'affectation comprend les domaines suivants: (i) ancien propriétaire de la mission précédente (identifiant contrôleur), (ii) identifiant du destinataire propriétaire de la nouvelle mission (le robot gagnant), (iii) identifiant de la mission, (iv) spécifications détaillées de la mission.

Une quatrième famille de messages pourrait être envisagée pour assurer la bonne passation de la mission en renvoyant une réplique lors de l'affectation (*MissionAssignmentReply*) et qui veille à ce que les missions ne soient pas perdues suite à des problèmes de communication.

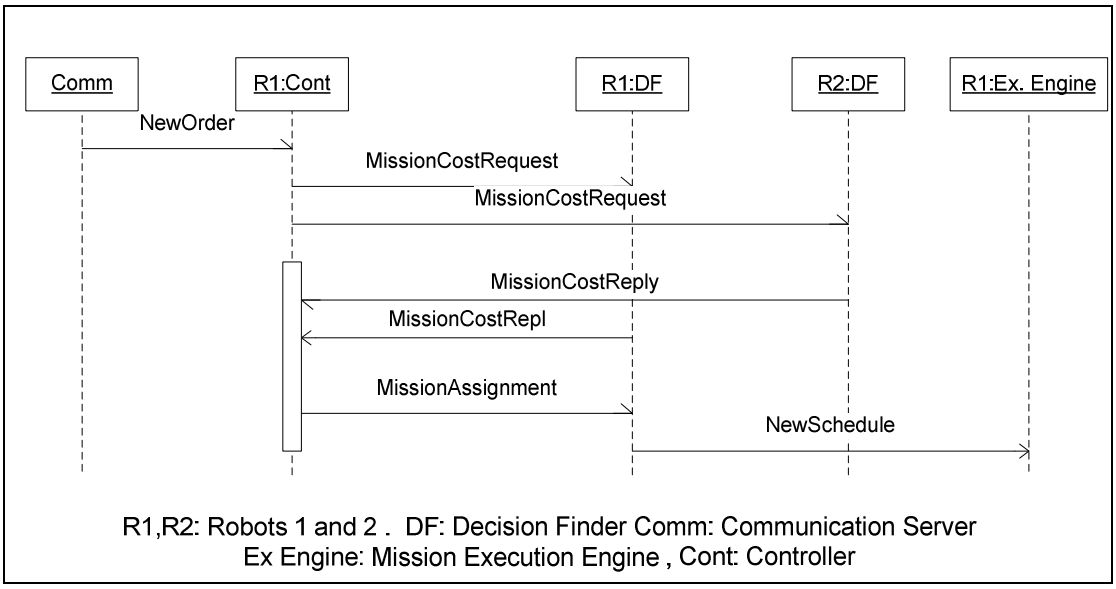

Figure 48 : Négociations et interactions entre les pairs durant une affectation de mission

7.3.2. Passation/ Legs de missions

Cette procédure est déclenchée lorsqu'un robot se trouve dans l'incapacité d'exécuter une mission qui lui a été affectée dans les dates d'échéance prévues. Le robot en question envoie aux autres robots une requête de Legs, en précisant ses missions et ses coûts d'exécution de ces missions. A la réception de cette requête, chaque robot calcule le coût de d'exécution des ces missions et renvoie sous forme de réplique de Legs (*HandOverReply*) le coût d'exécution de la mission rapportant le plus fort gain pour le système robotique. En fonction des répliques reçues, le robot initiateur de la requête détermine le robot et la mission qui allège le plus la charge de travail du système robotique puis délègue cette mission au robot choisi (Figure 49)

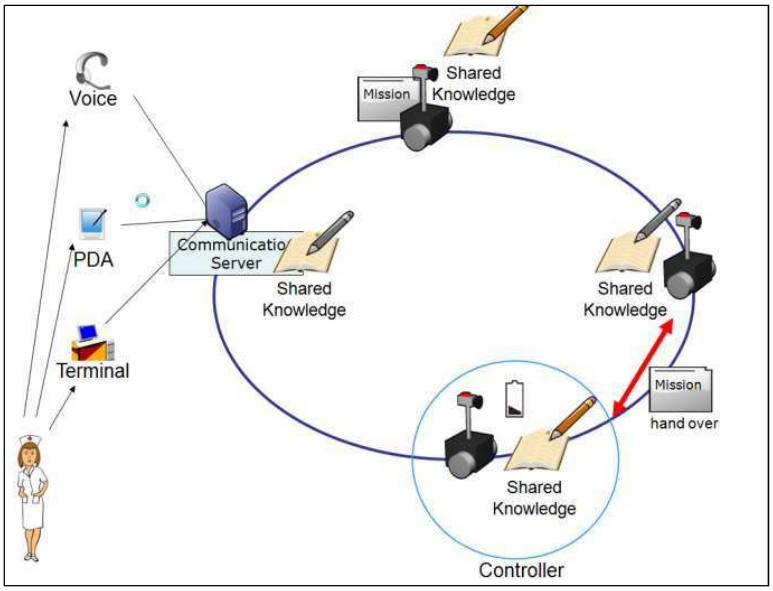

Figure 49 : Passation de missions suite à un problème énergétique

Cette procédure peut être initiée tant que le robot reste en difficulté. En particulier, un robot avec un alarme de batterie cherchera à passer toutes ses missions aux autres robots.

```
$lstMissions = get_list_assigned_missions();
$hor=constructor_handover_request( $ lstMissions);
Broadcast($hor,ALL_ROBOTS);
$end_auction = Current_time() + 3;
$rp = create_replies_pile();

//Read replies
While Current_time()<$end_auction Then
      $answer=get_reply_for($hor);
      Stack($answer,$rp);
      Sleep
End While

//Sort Replies and indentifie best offer
$best_reply=pop($rp);
While $rp is_not_empty Do
      $tmp=pop($rp);
      If $tmp->Delta_gain > $best_reply->Delta_gain Then
            $best_reply=$tmp;
      End if
End while

//Determine the mission to handover and then Assign it to the winner
Assign_mission($best_reply ->mission, $best_reply->idSender);
```

Algorithme 6 : Robot initiateur d'une demande de passation

Au cours du processus de Legs de mission, différents messages sont échangés entre les robots et ces messages peuvent être classés en trois catégories:

- Requête de Legs (*MissionHandOverRequest*): Afin de renoncer à une ou plusieurs missions, un robot en difficulté notifie ses pairs en envoyant une requête de legs. Une requête de legs (*MissionHandOverRequest*) est composée des éléments suivants: (i) identifiant du robot initiateur (robot en détresse), (ii) identifiant du destinataire (dans la version implémenté cette demande est diffusée a tous les robots, et ce champs est fixé à ALL_ROBOTS mais on peut prévoir une variante où cette requête est diffusée à

un sous ensemble restreint), (iii) ses missions et leurs coûts d'exécution, (iv) date du lancement du processus de négociation (*timestamp* du moment où la demande a été diffusée), (v) durée des négociations et la date de clôture du processus de négociation avant d'annoncer le vainqueur (le temps nécessaire requis par tous les pairs afin de répondre à la requête).

- Répliques de legs (*MissionHandOverReply*) : A la réception d'une requête de legs, chaque robot calcule son coût d'exécution de chacune des missions figurant dans la requête. Il détermine la mission pouvant être exécutée avec le moindre coût et un gain maximale puis envoie ses observations sous forme d'une réplique de Legs (*MissionHandOverReply*). La structure du message *MissionHandOverReply* est: (i) identifiant de l'émetteur (robot ayant effectué les calculs), (ii) identifiant du destinataire (robot en détresse initiateur de la demande), (iii) Identifiant de la mission la plus appropriée pour lui déléguer, (iv) coût différentiel d'exécution de la mission, (v) indicateur booléen précisant si le module nécessaire pour l'exécution de la mission est déjà monté sur le robot.
- Message d'affectation (*MissionAssignment*) : similaire au message d'affectation pour l'affectation d'une nouvelle mission.

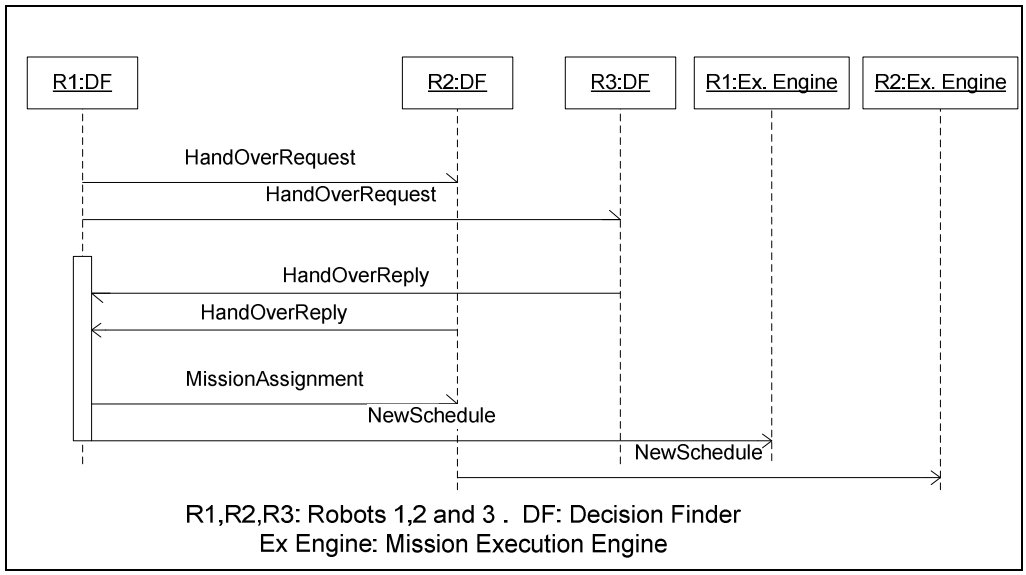

Figure 50 : Négociations lors d'un processus de leg

7.3.3. Réquisition de Missions

Ce processus est déclenché lorsqu'un robot se trouve sans mission. Soit le robot vient de rejoindre le groupe de robot après le rechargement de ses batteries, soit il vient d'achever en avance l'ensemble de ses missions.

Le robot se trouvant inactif, diffuse à ses pairs comme illustré à la Figure 51, une demande de réquisition de missions *MissionRequisitionRequest*. Chaque pair réplique par des messages *MissionRequisitionReply* donnant la liste de ses missions et les coûts différentiels. Au bout d'un temps donné, le robot demandeur examine les répliques reçues, calcule les coûts d'exécution des missions par lui-même et détermine la mission à réquisitionner permettant le plus grand gain au système. Après la détermination de la mission à réquisitionner, le robot

envoie au propriétaire un message *MissionRevokeOwnership* demandant le transfert de la mission.

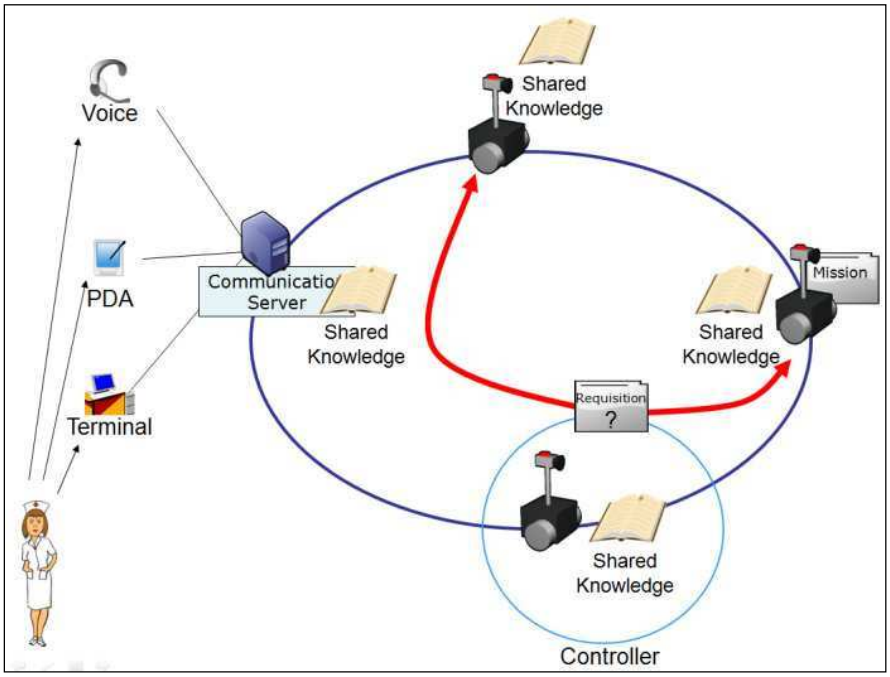

Figure 51 : Requête de réquisition de mission

```
$rqrq=constructor_requisition_request( $myself->Id);
BroadCast_request($rqrq,ALL_ROBOTS);

$auction_end = Current_time() + 3;
$winner = null;
$mission_to_request = null;
$best_gain= -INFINITY

//Read replies
While Current_time()<$auction_end Then
      $answer=get_reply_for_request($rqrq);
      Foreach $mission In $answer->list_Missions() Do
            $gain = simulate_execution_get_cost($mission);
            If $gain > $best_gain Then
                  $best_gain = g$ain;
                  $mission_to_request=$mission;
                  $winner=$answer->Id;
            End If
      End For
      Sleep();
End While

//Ask owner to handover the mission
handover_mission($winner, $mission_to_request, $myself->Id);
```

Algorithme 7 : Algorithme de décision du réquisitionnaire

Pendant le processus de négociation pour les demandes de réquisition, différents types de messages sont échangés entres les robots. Ces messages peuvent être classés en trois catégories:

- Demande de Réquisition (*MissionRequisitionRequest)* : Le robot sans missions informe les autres pairs de sa situation. Un message MissionRequisitionRequest est composé des champs suivants: (i) identifiant de l'émetteur de la demande (identifiant du robot inoccupé), (ii) identifiant du destinataire (fixé à ALL_ROBOTS afin que la demande soit diffusée à tous les robots), (iii) date de lancement du processus de négociation (date du moment où la demande a été diffusé aux autres robots), (iv) durée des négociations et date de clôture (le temps nécessaire requis par tous les pairs pour répondre à la demande afin d'annoncer le gagnant) .
- Répliques des demandes de réquisition (*MissionRequisitionReply*) : A la réception d'une demande de réquisition, chaque pair calcule les coûts différentiels d'exécution de ses missions. Une fois ces coûts calculés, le robot encapsule la liste de ses missions ainsi que les coûts respectifs dans un message adressé au robot inactif. La structure de ce message *MissionRequisitionReply* est: (i) identifiant du de l'émetteur (robot répondant), (ii) identifiant du destinataire (robot inoccupé), (iii) liste des missions et leurs coûts d'exécution.
- Révocation d'une mission (*MissionRevokeOwnership*): Ce message joue le rôle inverse du message *MissionAssignment*, en demandant le transfert d'une mission au robot émetteur de ce message. La structure du message *MissionRevokeOwnership* est : (i) identifiant de l'émetteur (robot inoccupé), (ii) identifiant du destinataire (l'ancien propriétaire de la mission à réquisitionner), (iii) identifiant de la mission à réquisitionner

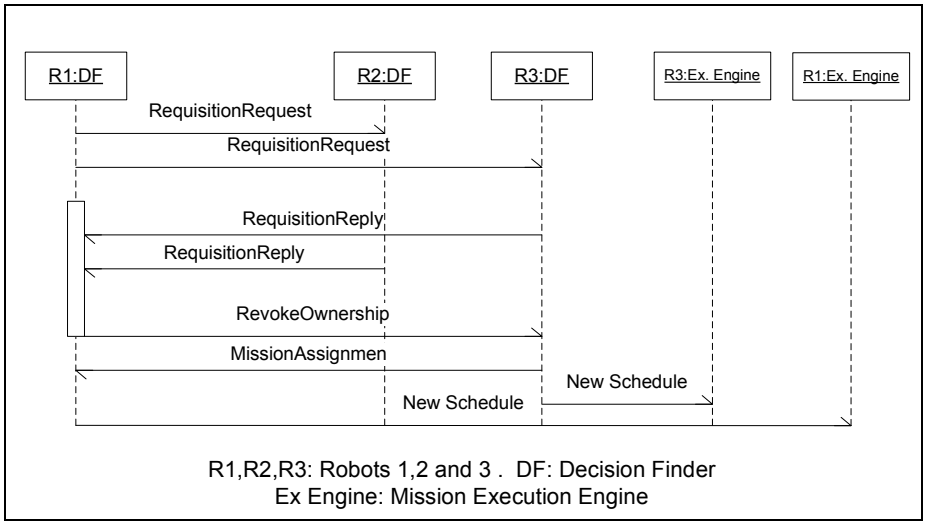

Figure 52 : Interactions entre les différentes entités lors des négociations de réquisition

La Figure 52 illustre le processus de négociation pour une réquisition de missions et l'interaction entre les différents robots ainsi que les interactions entre les différents composants d'un même robot, comme l'exécuteur des missions.

7.4. Simulation

Cette simulation simule le fonctionnement du système robotique. Elle prend en compte les robots et leur configuration, la navigation de robots, l'exécution des missions, la communication entre les robots, la consommation d'énergie. Elle prend en compte de

différents événements comme les échecs, les demandes de nouvelles missions, fin d'une mission et l'arrivée de nouveaux robots.

Afin de simuler les différents composants décisionnels d'un robot qui sont des objets *multi-thread*, leur équivalents dans la simulation héritent d'un objet unique *Timer (horloge)* qui discrétise l'avancement continue dans le temps.

Les paramètres d'entrée de cette simulation sont alimentés via des fichiers *xml* décrivant les robots, les modules, l'environnement mais aussi les scénarios d'arrivée de missions ainsi que les caractéristiques de ces missions. Le composant équivalent à l'exécuteur de missions en temps réel rend le robot inaccessible pour la durée de la mission pour simuler l'exécution d'une mission. La simulation propose deux modes de prise de décision pour l'affectation de taches, le mode centralisé utilisant les algorithmes évolutionnaires et un mode décentralisé basé sur la négociation et l'algorithme d'enchères inversées. Pour le dernier mode, la possibilité de réaffectation de taches peut être activée ou désactivée à la demande. De plus pour les deux modes, les scenarios peuvent être évalués selon l'une des deux fonctions d'évaluation soit la fonction « primes par escalier », soit la fonction « pénalités continue ».

A la fin d'une simulation pour une durée donnée, les états des robots sont fournis, ainsi que les statistiques d'utilisation de chaque robot, les états de toutes les missions, ainsi que les retards récurrents. Pour plus de détails sur cette simulation, l'architecture de la simulation est présentée en annexe de ce document.

7.5. Comparaisons numériques

Afin de comparer les différents mécanismes d'ordonnancement distribué, plusieurs scénarios ont été lancés sur une simulation à événements discrets que nous avons développée dans ce but. Les instances générées dans cette section, utilisent la loi de poisson pour les arrivées et la loi uniforme pour la génération des temps d'exécution de chaque mission.

7.5.1. Avantages de la réaffectation

Plusieurs variantes de prise de décision décentralisée ont été étudiées et comparées, afin de déterminer le plus qu'apporte ces approches présentées dans ce chapitre. Dans cette partie on étudie les effets attribués à chacune de ces approches sur une période de longue durée. La première variante étudié, est un meneur de décision distribué basique sans réquisition ni délégation, cet agent se contente d'affecter les nouvelles missions sans les réaffecter suite aux événements qui puisse intervenir sur le système robotique. Cette approche sera désignée par NO faisant allusion à la réactivé de cette approche. La deuxième approche consiste à enrichir ce système basique par des mécanismes de réactivité suite aux problèmes qui peuvent avoir lieu, donc des mécanismes de délégation et de legs sont joints au système basique d'affectation distribué. Cette deuxième approche sera désignée par le terme HO faisant allusion au terme *Hand Over*. La troisième approche consiste à ajouter au système basique d'affectation distribuée la réquisition des missions qui permettra aux robots d'assister leurs pairs en cas de besoin. Cette approche sera désigné par le terme RQ, faisant allusion à *Requisition* . La quatrième approche consiste à fusionner les deux mécanises de réaffectation avec le mécanisme basique d'affectation distribué, et sera désigné par le terme HO+RQ

L'exemple choisi est le R8M178 qui se compose d'une équipe de 8 robots d'avoir à exécuter 178 missions, qui est plus du double de leur charge de travail moyenne pour la durée de la simulation.

Différents scénarios supplémentaires ont été proposés dans [Baalbaki 10 b] et plusieurs critères de services ont été mesurés. Comme le Nombre de missions exécutées (*#Completed Missions*) durant la simulation, le nombre de missions rejetées (*#Rejected Missions*), le nombre de mission. Le nombre moyen de missions en attentes sur chaque robot (*Avg Schedule*), le file d'attente la plus longue sur un robot (*Max Schedule*), le retard moyen de lancement par rapport a la date de démarrage souhaité (*Avg Start Tardiness*), le retard minimal / maximal par rapport à la date souhaitée de démarrage (*Min Start Tardiness*/*Max Start Tardiness*). De même on mesure le retard algébrique minimal et moyen par rapport à la date de fin souhaitée (*Min Comp Lateness / Avg Comp Lateness*) ainsi que la moyenne de retard absolue par rapport à la date buttoir souhaité et le retard absolue maximal (*Avg Comp Tardiness / Max Comp Tardiness*).

Indicateur	*NO*	*HO*	*RQ*	*HO + RQ*
Avg Schedule	1	1	1	1
Max. Schedule	5	6	5	6
#Completed Missions	87	80	87	80
#Rejected Missions	81	96	82	96
Avg Start. Tardiness	550.9	101.9	515.4	101.9
Max Start. Tardiness	5697	566	5680	566
Min Comp. Lateness	16	20	20	20
Avg Comp. Lateness	691.4	226.4	657.5	227.4
Avg Comp Tardiness	691.4	226.4	657.5	227.4

Tableau 16: Performances des techniques distribuées

Les expérimentations numériques montrent que les algorithmes distribuées prenant en charge la délégation de missions sont supérieurs par rapport aux méthodes d'enchères inversées classiques, et on note aussi une amélioration de performance mais moins importantes avec les méthodes permettant de réquisition de missions et la méthode intégrant ces deux techniques (réquisition et délégation) permet de d'accomplir les missions selon la façon la plus optimale entre ces deux options, ce qui permet de confirmer notre choix d'intégrer les deux à la fois.

7.5.2. Comparaison entre la prise de décision centralisée et la prise de décision décentralisée

Le scenario est réalisé par un groupe de cinq robots, qui sont totalement chargés, donc les aspects énergétiques ne sont pas considérés par ce scénario. On suppose que l'ensemble est initialement en veille pour simuler l'exécution normale au début de journée et on suppose que les arrivées sont équilibrées et ne chargeront pas le système robotique. La figure suivante illustre les arrivées de missions par intervalle de cinq minutes.

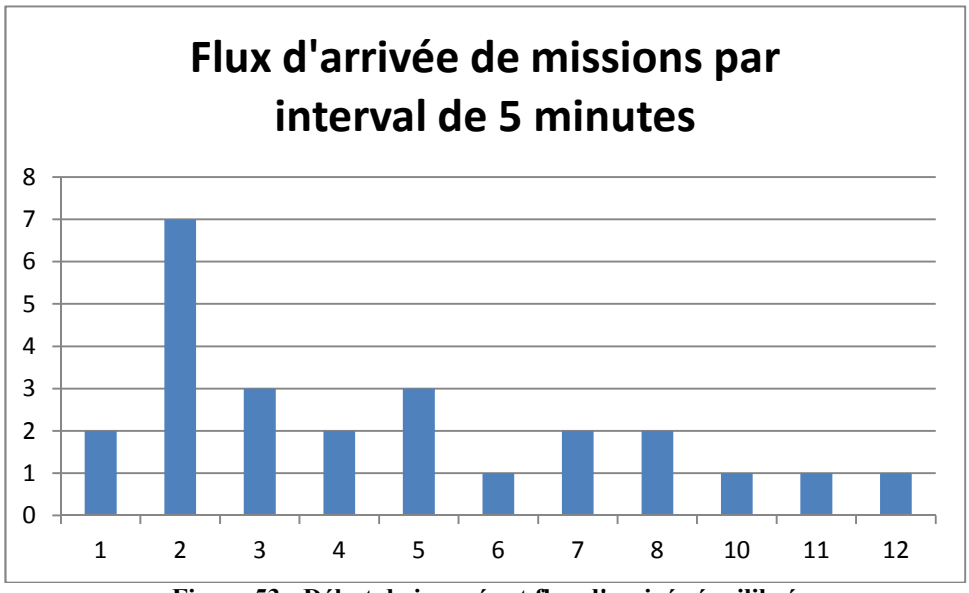

Figure 53 : Début de journée et flux d'arrivée équilibré

La Figure 54 compare les taux d'utilisation des robots, en utilisant soit l'approche centralisé, doit l'approche décentralisé. Ces comparaisons ont été effectuées au bout d'une heure de simulation.

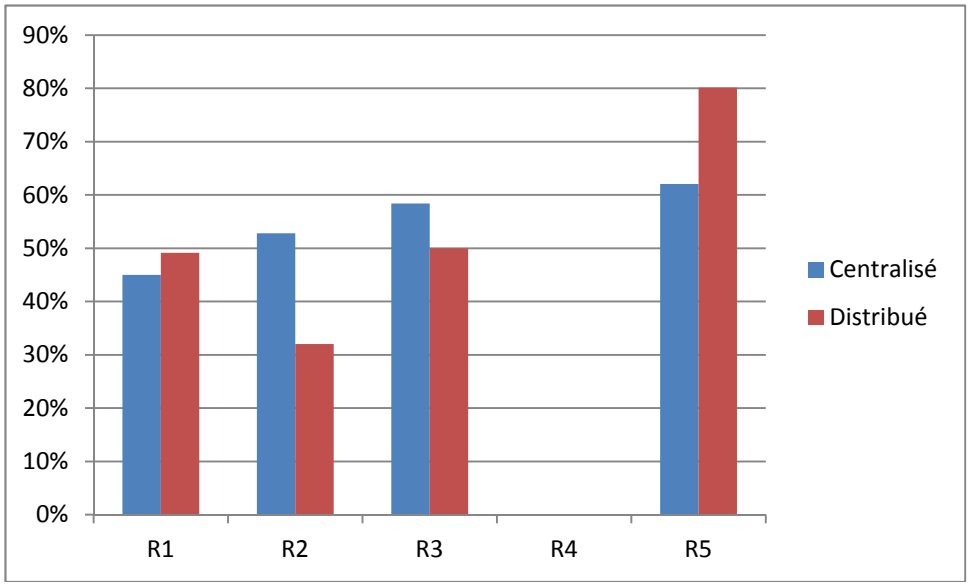

Figure 54 : Taux d'utilisation des robots

La Figure 55 compare les pénalités dues aux retards qui sont utilisées comme critère d'évaluation des plannings de chaque robot.

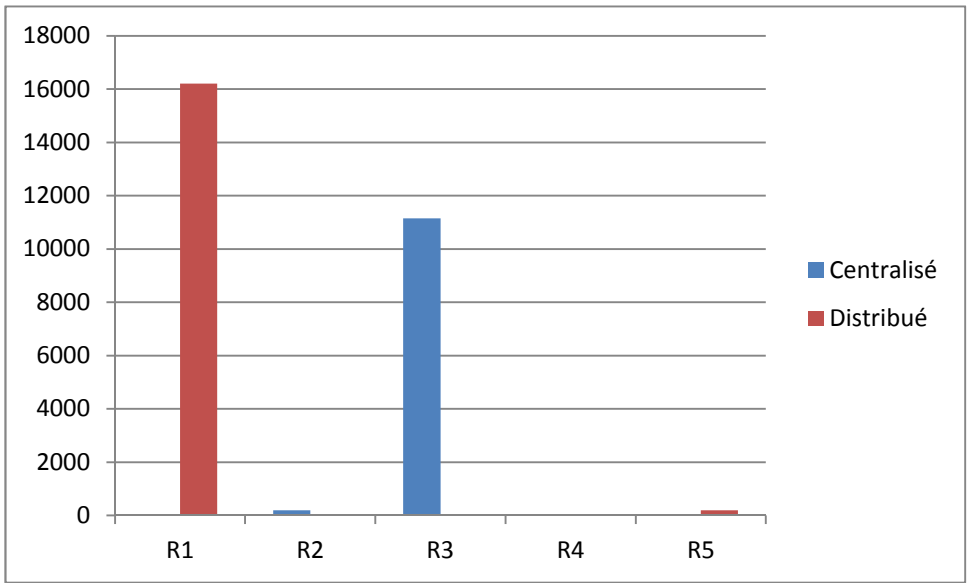

Figure 55 : Fonction Objectif par robot.

	Centralisé	Distribué
Nombre de mission exécutés	22	22
Retard Moyen au démarrage	95,59	105,96
Retard Maximal au démarrage	401	375
Retard Moyen d'achèvement	0	0
Retard Maximal d'achèvement	0	0
Nombre moyen de mission en attende	0	0
Nb Maximal de mission en attente	2	2

Tableau 17: Comparaison entre les deux approches

Le **Error! Reference source not found.**, dresse les critères de performance des deux approches, en comparant ces critères et les figure au dessous nous pouvons déduire que ces deux approches sont équivalant dans les cas ou le système robotique n'est pas chargé et les demandes de missions arrivent d'une manière distribuée.

Pour le deuxième scénario, on prend les mêmes robots mais cette fois ci, le taux d'arrivée des missions est augmenté afin de surcharger le système et de comparer la réponse des deux approches. La distribution des arrivées est illustrée par la figure suivante :

Figure 56: Distribution des arrivée de missions

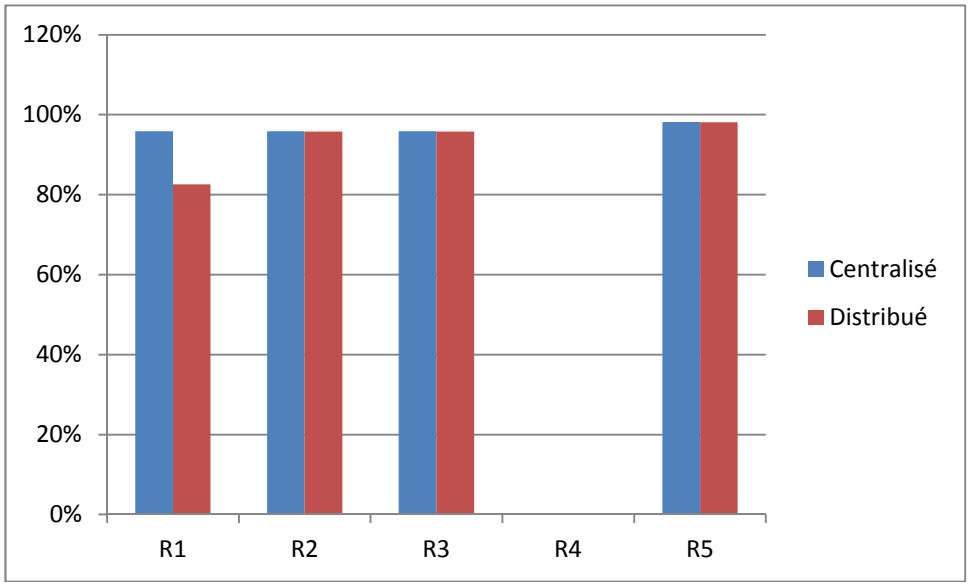

Figure 57: Taux d'utilisation

La Figure 57 montre que l'approche décentralisée arrive à mieux gérer le stress des missions en cas de surcharge et les robots sont utilisés plus efficacement.

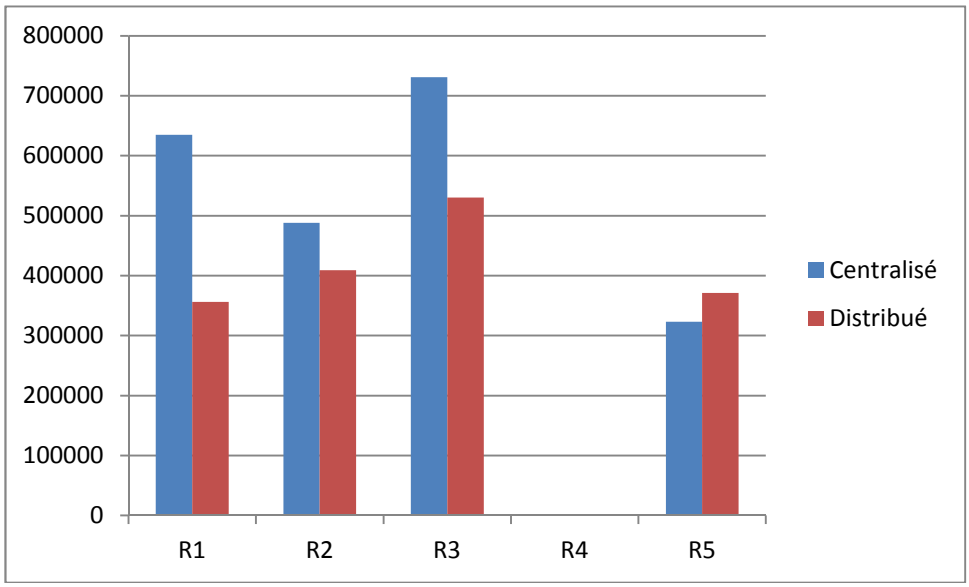

Figure 58: Fonction Objectif

La Figure 58 montre que dans ce scenario, la solution de l'heuristique décentralisée a surpassé celle proposée par l'approche centralisée basée sur les algorithmes évolutionnaires.

	Centralisé	Distribué
Nombre de mission exécutés	46	49
Retard Moyen au démarrage	482,848	360,041
Retard Maximal au démarrage	1451	1550
Retard Moyen d'achèvement	291,935	172,163
Retard Maximal d'achèvement	994	1272
Nombre moyen de mission en attende	3	4
Nb Maximal de mission en attente	11	10

Tableau 18: Critères de performance dans un système surchargé

Dans ce scénario nous montrons que dans certain cas, l'approche décentralisée peut donner des résultats meilleurs que l'approche centralisée.

7.6. Conclusion

Dans ce chapitre, nous avons considéré une approche distribuée d'affectation et d'ordonnancement de missions pour un groupe de robots mobiles reconfigurables. Le problème consiste à déterminer pour chaque robot, une séquence de missions à exécuter selon un calendrier. Comme les robots travaillent dans un environnement incertain et le comportement de l'être humain peut affecter considérablement l'exécution des missions, nous avons proposé une approche réactive, où un robot en tenant compte de ses propres observations des événements et conscient de sa situation, peut déclencher un processus de réaffectation afin de redistribuer la charge de travail au sein du système.

Une première voie de recherche consiste à identifier les occasions d'exécution de missions en parallèle dès la phase d'affectation au lieu d'attendre la phase d'exécution pour identifier telle opportunité.

Une deuxième voie consiste à prendre en compte également la nature collaborative de certaines missions multi-robots nécessitant une synchronisation de plannings et la prévoyance des retards issus de telles synchronisations.

Conclusion Générale et perspectives de recherche

Les modèles développés pour le niveau stratégique sont des modèles statiques. La première perspective de recherche serait de se tourner vers des modèles stochastiques, ou même dynamiques. De plus le modèle basé sur la génération de colonnes est implémenté en utilisant les librairies open source COIN, une autre perspective serait de migrer vers une solution utilisant des solveurs commerciaux.
L'approche centralisée du problème d'affectation est basé sur les algorithmes évolutionnaires, une perceptive de recherche serait de prospecter d'autres méta- heuristiques notamment les colonies de Fourmies.
Egalement, il serait intéressant d'effectuer des recherches en étudiant les interactions et les avantages des recommandations au niveau stratégique sur la simulation qui représente le niveau tactique. De plus, il serait utile de quantifier les gains en énergie et en temps obtenus en appliquant ces recommandations.
En ce qui concerne l'approche distribuée, une implémentation utilisant les réseaux de Pétri peut constituer une autre perspective de recherche pour le futur.
L'intelligence artificielle et les systèmes d'apprentissage présentaient un intérêt majeur dans les recherches de notre projet IWARD, mais suite à certains problèmes technologiques rencontrés, ceux-ci n'ont pu parvenir au but final que nous étions fixé.
Le base de connaissances partagé est diffusée et synchronisée continuellement sur tous les pairs, mais cette approche est applicables seulement aux groupes de robots de tailles moyennes, on propose d'inspecter d'autres types de diffusion de connaissances soit hiérarchiques, soit par cellule, ou même des diffusions partielles par proximité.

****

A la fin nous concluons par deux citations

« *S'il y a plus d'une façon de faire quelque chose, et que l'une d'elles conduit à un désastre, alors quelqu'un le fera de cette façon* »
(Edward A. Murphy Jr / 1918-1990)
et
« *Les espèces qui survivent ne sont pas les espèces les plus fortes, ni les plus intelligentes, mais celles qui s'adaptent le mieux aux changements.* »
(Charles Darwin / 1809-1882)

Par Analogie, si nous appliquons ces citations aux systèmes d'information et de prise de décision, cela signifie que l'efficacité et optimalité des approches appliquées aux systèmes de gestion des systèmes complexes sont les deux aspects les plus recherchés dans les systèmes de prise de décisions actuellement. Cependant il existe une facette peu inspectée, celle l'adaptabilité de ces approches face aux événements improbables.
Nous estimons que, la réactivité et l'adaptabilité des systèmes d'information et de prise de décision, face à des événements imprévues lors de la conception, seront les aspects recherchés dans les systèmes de prise de décisions du futur.

.

Annex I - Simulation architecture

The Objectives and the assumptions

Due to the human and financial cost of construction of each robot, the number of robot constructed for our project is limited to three. This number can be consider a relatively low to inspect the advantages of the distributed decision making process. For this reason a simulation have been developed to surpass such limitation and test the efficiency of the different algorithms developed for this section. This simulation inspects also the reactivity of the system in peek scenarios and to compare the performance of the different decision making approaches

This simulation is linked to a artificial timer simulates the elapse of time and the time base events. In this simulation we simulate the robot , their configuration , a simple navigation, linear power consumption , the negotiation between the robots , the decision logic related the different events like the arrival of a new mission order, the robots failures , the arrival of a new robot to the group and the event related to the execution of the mission.

Performance Indicators

Different indicators can be used , to quantify the quality of the decision taken by the different agents from the strategic level as for the global planner , to the tactical level as fro the different decision finders Several criteria can be identified as performance indicators like be the number of mission executed and the number of mission aborted or unhandled. The tardiness and latency of execution of the missions mainly the sum, the average, the median, the maximum and the minimum, .to measure the reactivity of the system. The number of active and idle robot during each period with the usage of the robots is also an important indicator. The number of message exchanged between the robots and network usage. The average and maximum time to take a decision, mainly the time elapsed between the arrival of a new order or event and assignment of the mission to a robot.

Simulation Architecture

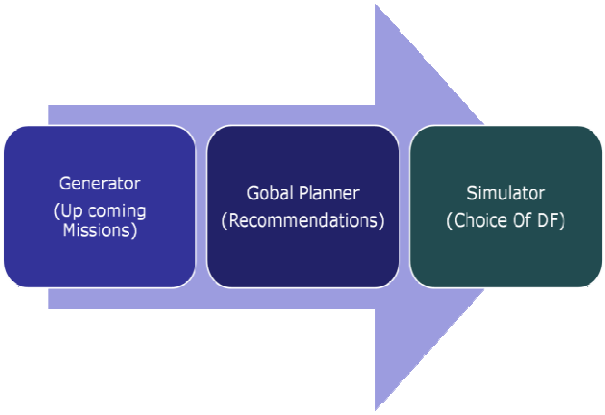

Figure 59: The ongoing multi-robot mission

The simulation architecture can be divided into three main parts, the data generator that creates different scenarios with different characteristics for the simulation. A second set of data is generated for the global planner. This data, considered as historical data, is analysed and used to estimate the future demands. The simulator in addition to the original scenario data; it can take as an optional additional source of data, the recommendations of the global planner. The figure 1 illustrates the three entities of the simulation architecture.

Main components

The simulation inputs

The robots initial State

The simulation was developed to take as input several xml files that characterizes the different scenarios that we need to study. The first mail element is the characteristics of the different robots that we want to simulate, the main chrematistics can be listed as the following:

- Robot Statut { Recharging , Active , Idle}
- Id Robot
- Robot Type
- Average Speed
- Idle Power Consumption Rate
- Active Power Consumption;
- Recharging Speed;
- Current PositionX
- Current PositionY
- Current Power Level
- Current Configuration
- List attached Modules
- Robot Recomendation*
- Max Power Level
- Max Number Of Attached Modules
- Next Recharging Date

The following figure illustrates a snapshot of the robots xml file which contains the initial state of robots at the beginning of the simulation.

```
<!-- List of robot idConfiguration might be removed TODO: Add Arrival/Join Date and leave date  -->
<RobotsList>
    <Robot idRobot="1" type="1" initialPositionX="12.0" initialPositionY="12.0"  initialPowerLevel="1200.5" idConfiguration="2" AverageSp
        <Module idModule="1" />
        <Module idModule="2" />
        <Module idModule="0" />
    </Robot>
    <Robot idRobot="2" type="1" initialPositionX="12.0" initialPositionY="62.0"  initialPowerLevel="1200.5" idConfiguration="1" AverageSp
        <Module idModule="0" />
        <Module idModule="3" />
        <Module idModule="4" />
    </Robot>
     <Robot idRobot="3" type="1" initialPositionX="62.0" initialPositionY="12.0"  initialPowerLevel="1200.5" idConfiguration="1" AverageS
        <Module idModule="2" />
        <Module idModule="5" />
        <Module idModule="6" />
        <Module idModule="0" />
    </Robot>
</RobotsList>
```

Figure 60: snapshot of the robot xml file

One of the important aspect of the robot state is the robots status, where the robot can be active and executing mission, or idle and awaiting the arrival of new order or recharging.

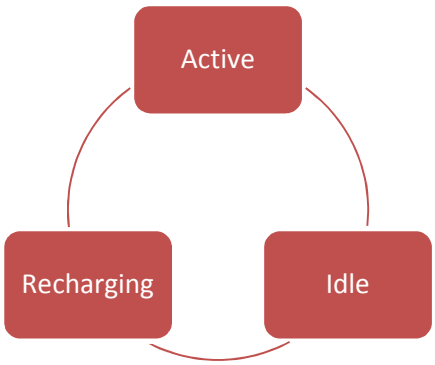

Figure 61: The states of s simulated robots

The Point of interest and navigation

A list of the main POI (Point of Interest) is also inserted as input via an xml file, where the following figure illustrates a sample of such a file

```
<Storage>
    <Module idModule="2" availableInStorage="3"/>
    <Module idModule="3" availableInStorage="0"/>
    <Module idModule="4"availableInStorage="1"/>
</Storage>
<!-- List of POI TODO: add POI name attribue and Z attribute  -->
<Map Length="100" width="100">
    <PointsOfInterestList>
        <POI idPOI="1" POIcoordX="0" POIcoordY="44"></POI>
        <POI idPOI="2" POIcoordX="0" POIcoordY="0"></POI>
        <POI idPOI="3" POIcoordX="33" POIcoordY="0"></POI>
        <POI idPOI="4" POIcoordX="33" POIcoordY="44"></POI>
        <POI idPOI="5" POIcoordX="12.5" POIcoordY="14"></POI>
        <POI idPOI="6" POIcoordX="75" POIcoordY="75"></POI>
        <POI idPOI="7" POIcoordX="60" POIcoordY="70"></POI>
        <POI idPOI="8" POIcoordX="44" POIcoordY="20"></POI>
        <POI idPOI="9" POIcoordX="90" POIcoordY="10"></POI>
    </PointsOfInterestList>
</Map>
```

Figure 62: Snapshot of the POI xml file

The simulated robots use a simple navigation algorithm to simulate their movements, where each robot takes the rectilinear path to travel from one position to another without taking into consideration the obstacle on its path.

Mission orders

The main input file would the mission order file, where in this file the scenarios of mission arrival are predefined and as well as the characteristics of each order. The following figure illustrates a snapshot of an order xml file.

```
<Orders>
  <Order id="1" type="5" OrderDate="760" EarliestStartDate="971" LatestStartDate="971"
    PrefferedFinishDate="1971" DueDate="2571" StartPointX="12.5" StartPointY="12.5" EndPointX="12.5"
    EndPointY="12.5" />
  <Order id="2" type="3" OrderDate="765" EarliestStartDate="765" LatestStartDate="971"
    PrefferedFinishDate="1971" DueDate="2571" StartPointX="70" StartPointY="12.5" EndPointX="33"
    EndPointY="60" />
  <Order id="3" type="1" OrderDate="770" EarliestStartDate="773" LatestStartDate="971"
    PrefferedFinishDate="1971" DueDate="2571" StartPointX="100" StartPointY="55" EndPointX="80"
    EndPointY="15" />
  <Order id="4" type="4" OrderDate="775" EarliestStartDate="974" LatestStartDate="971"
    PrefferedFinishDate="1971" DueDate="2571" StartPointX="99.9" StartPointY="12.5" EndPointX="12.5"
    EndPointY="44" />
  <Order id="5" type="2" OrderDate="790" EarliestStartDate="975" LatestStartDate="971"
    PrefferedFinishDate="1971" DueDate="2571" StartPointX="72.25" StartPointY="47" EndPointX="11"
    EndPointY="12.5" />
</Orders>
```

Figure 63: Snapshot of the orders's xml file

Each order is transformed into a mission instance by the mission inserter component that simulates the role of the users ordering the different mission. The mission executor attached to each robot instance is responsible of executing those mission instances the according to the following diagrams.

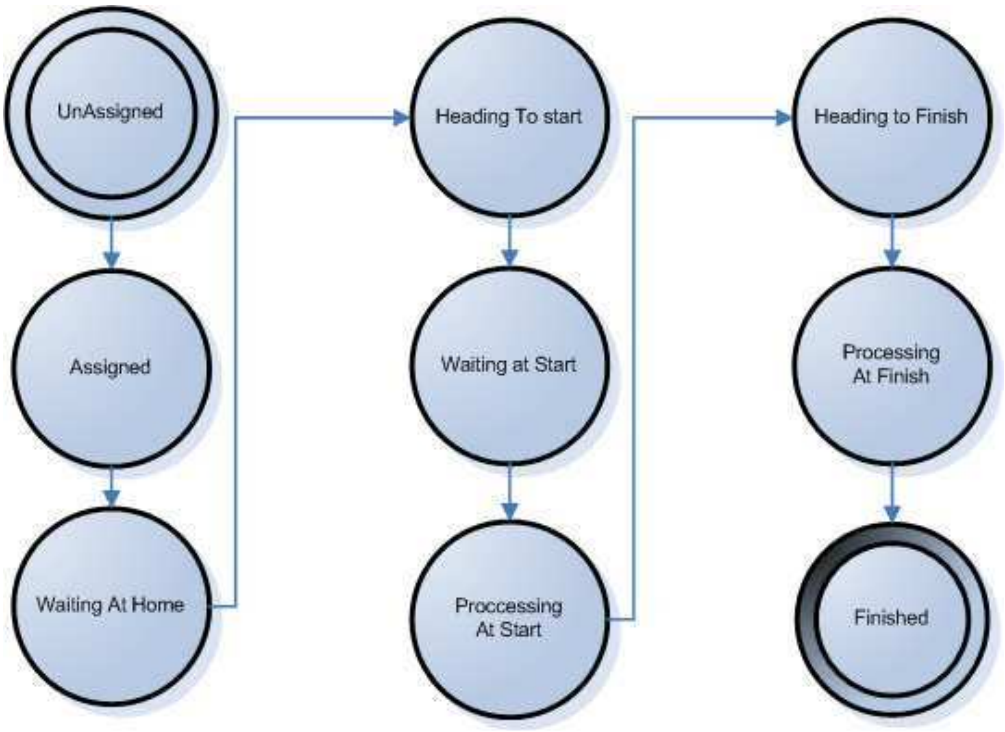

Figure 64: The different states of a mission

Robot Recommendation

An optional input for the simulation could be the results of the global planner. It consists of recommendation for each robot, it configuration, its home station and a recommended recharging schedule. A snapshot of a recommendation file is illustrated in the following figure

```
<RobotsRecommendations>
    <Period idPeriod="1" PeriodStart="1256170000" PeriodEnd="1256190000">
        <Recommendation idRobot="1" recommendedPowerConsumption="450" >
            <HomeStation idHomeStation="3" coordX="12" coordY="11"></HomeStation>
            <Configuration idConfiguration="4">
                <Module idModule="1"/>
                <Module idModule="3"/>
                <Module idModule="4"/>
            </Configuration>
        </Recommendation>
        <Recommendation idRobot="2" recommendedPowerConsumption="250" >
            <HomeStation idHomeStation="3" coordX="12" coordY="11"></HomeStation>
            <Configuration idConfiguration="4">
                <Module idModule="2"/>
                <Module idModule="3"/>
                <Module idModule="4"/>
            </Configuration>
        </Recommendation>
    </Period>
</RobotsRecommendations>
```

Figure 65: Snapshot of the recommendations xml file

The active components

Shared knowledge

All the simulated robots connect to this component to update the date and extract new information, the main information that can be found in the shared knowledge are:

- A simulated timer
- The status of the robots
- The status of the modules
- The status of the missions
- And an event log

Network manager

This component is sort of a buffer that receives all the messages from the different peers and then routes them after certain simulated delay. Three different functionalities can be found on this component the normal message routing mechanism from a peer to another , the message multicast and the broadcast. Only the components that inherit from the peer components can communicate between each other.
Different Message type are used in the simulation for example the MissionCostRequest issued by the controller in order to check out the peers is capable of handling the mission, the initiator peer issues a MissionCostRequest message requesting the delta cost of executing the mission.
The structure of a MissionCostRequest message consists of the following elements:

- Id of the requesting peer.
- Id of the receiving peer
- Start time of the negotiation process (timestamp of when the request was issued)
- Remaining time till winner announced (the necessary time needed by all peer to respond to the request)
- Mission specifications

Another example could be the MissionCostReply issued by a robot receiving a MissionCostRequest and equipped with the necessary capabilities to executing the mission.
The structure of the missionCostReply message is:

- Id of the mission
- Id of the replying robot
- Id of the issuer of the request
- Has the needed capacities to handle the mission (Boolean)
- The delta cost of handling the mission

Mission Executor

The mission executor communicates with the decision finder of the robot as shown in the following figure.

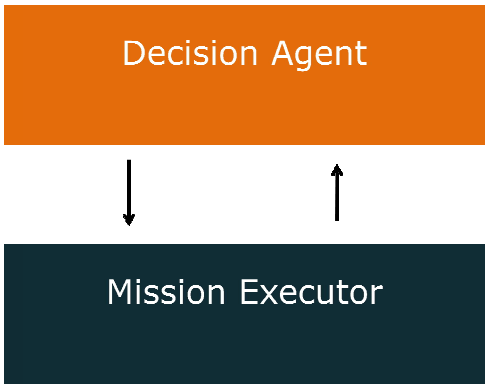

Figure 66: Interaction between the DF and the Execution engine

The main responsibility of the mission executor is to execute the schedule of mission set by the decision finder agent, update the position of the robot and update the status of the mission instance. The simulation of the execution of a mission is done by updating the fields of the mission instance that are characterized as the following:

- MissionCharacteristics (Duedate, earliest start date ….)
- AssignedTo
- Mission Status
- Real Start Date
- Real Finish Date
- Scheduled Start Date
- Scheduled Finish Date
- Remaining Processing Time At Start
- Remaining Processing Time At Finish

The controller

In this study different strategies are examined, therefore different type of controller were envisaged mainly the centralized controller that can be compared to a commander that set the schedule of execution of mission for all the robots. On the other hand the decentralized controller is a light agent that only role is to inform the robots of new missions arrival and managing a bid for each mission.

The decision finder

The decision finder is responsible of finding the best schedule to execute the mission on the robot, as for the controller two different decision finder families can be found mainly the centralized and the decentralized and for each type the cost calculation function can be changed.

The following figure illustrates the architecture of the decision finder and controller; where all the DF components inherit for a basic DF component and can divided into 2 categories the centralized and the decentralized. In the case of the decentralized decision finder they can be divided into different sub categories also.

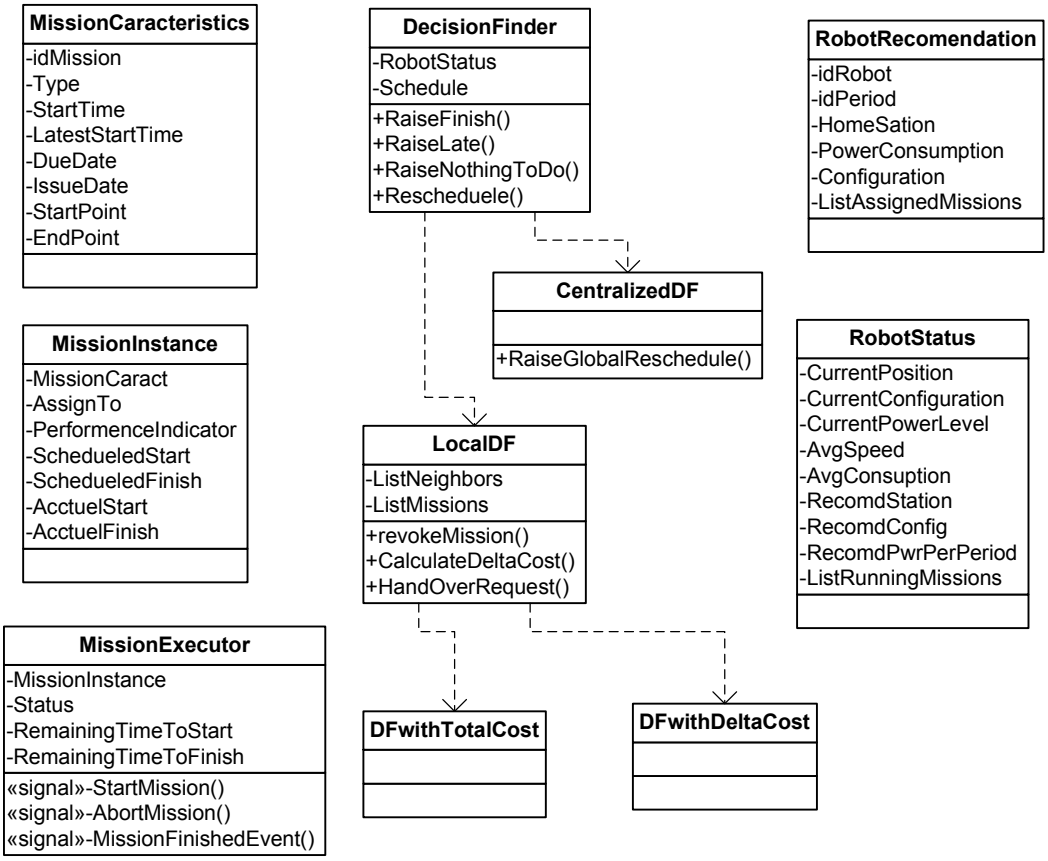

Figure 67: The architecture of the Decision finder

Bibliographie

[Agassounon 04]W. Agassounon, A. Martinoli, and K. Easton, "*Macroscopic modeling of aggregation experiments using embodied agents in teams of constant and time-varying sizes*" Autonomous Robots, vol. 17, no. 2–3, pp. 163–191, (2004).

[Al-Sultan 99] K.S. Al-Sultan et M.A. Al-Fawzan. *A tabu search approach to the uncapacitated facility location problem*. Annals of Operations Research, vol.86, pp. 91-103, 1999.

[Baalbaki 09] Baalbaki, H., Xie, X., "A decision framework for operation management of reconfigurable mobile service robots in hospitals*",* INCOM '09, Moscow, June 3-5, 2009

[Baalbaki 08 a] Baalbaki, H., Lamiri, M., Xie, X. "Joint Location and Configuration of Mobile Service Robots for Hospitals*"*, *Proc. IEEE-CASE* 2008

[Baalbaki 08 b] Baalbaki, H. Lamiri, M, Xie, X "A two-phase approach for configuration and location of mobile robots in a hospital setting*"*. *I*PROMS* – Innovative Production Machines and Systems 2008.

[Baalbaki 10 a]Baalbaki, H., Xie, X., Delorme, X "Mission Assignment and Scheduling for A Team of Service Robots using Evolutionary algorithms*", IEEE Workshop on Health Care Management Venice, Italy* 2010.

[Baalbaki 10 b] Baalbaki,H., Xie, X. "Distributed rescheduling techniques for a group of mobile reconfigurable service robots." ,*GISEH 2010* Clermont-Ferrand 2-4 September 2010

[Baker 87] J.E. Baker. "*Reducing Bias and Inefficiency in the Selection Algorithm*". Proceedings of the Second International Conference on Genetic Algorithms and their Application (Hillsdale, New Jersey: L. Erlbaum Associates) pp. 14–21, (1987).

[Balinski 61] M. L. Balinski. *Fixed-cost transportation problems*. Naval Research Logistics Quarterly, vol. 8, pp. 41–54, (1961).

[Barnhart 98] C. Barnhart, E. L. Johnson, G. L. Nemhauser, M. W. P. Savelsbergh, P. H. Vance. "*Branch-and-price: column generation for solving huge integer programs*." Operations Research, vol.46, pp. 316 – 329, (1998).

[Beasley 90] J.E. Beasley. "*A Lagrangean heuristic for set covering problems."* Naval Research Logistics, vol.41, pp. 151-164, (1990).

[Beasley 93] J.E. Beasley. *Lagrangean heuristics for location problems.* European Journal of Operational Research vol. 65; pp. 383-399, 1993.

[Beasley 96] J.E. Beasley et P.C. Chu. *"A Genetic Algorithm for Set Covering Problem."* European Journal of Operational Research vol.94, pp. 392-404, (1996).

[Beckers 94]R. Beckers, O. Holland, and J.-L. Deneubourg, "*From local actions to global tasks : Stigmergy and collective robotics*" in Proc. of the Int. Workshop on the Synthesis and Simulation of Living Systems, ser. Artificial Life, vol.4, pp. 181–189, (1994).

[Beni 89]G. Beni, J. Wang. "*Swarm Intelligence in Cellular Robotic Systems*", Proceed. NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy, June 26–30 (1989)

[Beni 05]G. Beni. *"From Swarm Intelligence to Swarm Robotics"*. Swarm Robotics, Lecture Notes in Computer Science, 2005, vol. 3342, pp. 1-9, (2005)

[Bonabeau 94]E. Bonabeau; G. Theraulaz. *"Intelligence collective"*.Collection Systèmes complexes. Paris : Hermès, (1994).

[Chang 07] Chang Y., Chen L. *Solve the vehicle routing problem with time windows via a genetic algorithm* Discrite and continuous dynamical systems Supplement, pp.240-249,(2007).

[Church 74] R.L. Church, C.S. ReVelle. *"The maximal covering location problem*," Papers of the Regional Science Association, vol.32, pp. 101-118, (1974)

[Cornuéjols 90] G. Cornuéjols, G.L. Nemhauser et L.A.Wolsey. *The uncapacitated facility location problem.* In: Mirchandani PB, Francis RL, editors. Discrete location theory. NewYork:Wiley; pp. 119–71, (1990).

[Correll 08] N. Correll and A. Martinoli, "*Comparing coordination schemes for miniature robotic swarms : A case study in boundary coverage of regular structures*," in Proc. of the Int. Symp. on Experimental Robotics, ser. Springer Tracts in Advanced Robotics (2008).

[Dantzig 59] G.B Dantzig and R. H. Ramser. *The truck dispatching problem* Management Science, vol. 6, pp. 80-81, (1959).

[Dantzing 60] G. B. Dantzig, P. Wolfe. "*Decomposition principle for linear programs*." *Operations Research,* vol. 8, pp. 101 – 111, (1960).

[Daskin 83]M.S. Daskin. "*A maximum expected location model: Formulation, properties and heuristic solution*". Transportation Science, vol.7, pp. 48–70, (1983).

[Daskin 90] M.S. Daskin, P.C. Jones,. and T.J. Lowe. "*Rationalizing Tool Selection in a Flexible Manufacturing System for Sheet Metal Products*". Operations Research, vol.38, pp. 1104-1151, (1990).

[Daskin 95] M.S. Daskin. Network and Discret Location: Models, Algorithms and Applications. John Willey and Sons Inc., New York 1995.

[Daskin 01] M. Daskin, C. Coullard and Z. Shen. "*An Inventory-Location Model: Formulation, Solution Algorithms and Computational results."* Annals of Operations Research, pp. 83-106, 2001

[Daskin 04] M. S. Daskin and L. K. Dean, "*Location of Health Care Facilities*," chapter 3 in the Handbook of OR/MS in Health Care: A Handbook of Methods and Applications, F. Sainfort, M. Brandeau and W. Pierskalla, editors, Kluwer, pp. 43-76, 2004

[Davis 85] L. Davis. *Applying Adaptive Algorithms to Epistatic Domains* In Proc. International Joint Conference on Artificial Intelligence (1985).

[de la Maza 93] M.de la Maza et B.Tidor. "*An analysis of selection procedures with particular attention paid to proportional and Boltzmann selection*". Proc. 5th Int. Conf. on Genetic Algorithms (Urbana-Champaign, IL, July 1993) ed S Forrest (San Mateo, CA: Morgan Kaufmann) pp. 124–31, (1993).

[Densham 92] P.G. Densham et G. Rushton. *A more efficient heuristic for solving large P-Median problems.* Papers in regional Science, vol. 71(3), pp; 307-329,(1992.)

[Desrochers 91] M. Desrochers,Y. Dumas, F. Soumis, et P. Trudeau . "*Column Generation Approaches to Airline Crew Scheduling Problems*." TRISTAN Conference, Montreal, 1991

[Dou 07] Jianping Dou, Xianzhong Dai and Zhengda Meng. *Optimization for Flow-Line Configurations of RMS Based on Graph Theory*. IEEE International Conference on Mechatronics and Automation August 5 - 8, 2007. Harbin, China. 1261-1266, 2007.

[Efroymson 66] M.A. Efroymson et T.L. Ray. *A branch and bound algorithm for plant location.* Operations Research vol.14 pp. 361-368.

[El Fallahi 08] El Fallahi A, Prins C, Calvo RW.*A menetic algorithm and a tabu search for the multi_copartment vehicle routing problem.* Computers and Operation Research vol.35, pp.1725-1741, (2008).

[Farritor 01]: Shane Farritor and Jun Zhang. *A RECONFIGURABLE ROBOTIC SYSTEM TO SUPPORT PLANETARY SURFACE OPERATIONS.* Self-Reconfigurable robotics Workshop at the IEEE conference on robotics and automation 2001.

[Falkenauer 91] E. Falkenauer, & Bouffouix, S. *A genetic algorithm for job shop*. Proceedings 1991 IEEE International Conference on Robotics and Automation, 824–829, (1991).

[Feldman 66] E. Feldman, F.A. Lehrer et T.L. Ray. "*Warehouse locations under continuous economies of scale*", Management Science, vol. 12, pp. 670-684, 1966.

[Feo 95] T.A. Feo, M.G.C. Resende. "*Greedy randomized adaptive search procedures*". Journal of Global Optimization, vol.6, pp.109–133 (1995)

[Galvao 93] R.D. Galvao. *The Use of Lagrangean Relaxation in the Solution of Incapacitated Facility Location Problem.* Location Science, pp. 57-79, 1993.

[Garey 79] M.R. Garey et D.S Johnson. *Computers and intractability : a guide to the theory of NP-completeness.* W.H. Freman and Co, San Francisco

[Garfinkel 72] R.S. Garfinkel et G.L. Nemhauser. *"Integer Programming"* by John Wiley and Sons. (1972°

[Gendreau 97] Gendreau , M., Laporte, G., Semet, F, "*Solving an ambulance location model by Tabu search*," Location Science, vol. 5, pp. 75-88, (1997)

[Gendreau 98] M. Gendreau M, G. Laporte,J.Y Potvin. *Metaheuristics for the vehicle routing problem.* GERAD research report G-98-52, Montreal, Canada. (1998).

[Gendreau 01] Gendreau , M., Laporte, G., Semet, F, "*A Dynamic model and parallel Tabu search heuristic for real-time ambulance relocation*," Parallel computing vol. 27, pp. 1641-1653, (2001).

[Geoffrion 74]A. M. Geoffrion, G. W. Graves. *Multicommodity Distribution System Design by Benders Decomposition.* MANAGEMENT SCIENCE vol. 20, pp. 822-844, 1974.

[Geoffrion 95] A. Geoffrion and R. Powers, Twenty *years of strategic distribution system design: an evolutionary perspective*, Interfaces, vol. 25, pp. 105-128, (1995).

[Gerkey 02] Gerkey, BP and Mataric, MJ; "Sold!: Auction methods for multirobot coordination"; *IEEE Transactions on Robotics and Automation,* 2002

[Gerkey 03] Gerkey, B.P. and Mataric, M.J.; "Multi-robot task allocation: Analyzing the complexity and optimality of key architectures"; *IEEE International Conference on Robotics and Automation*, 2003

[Gilmore 61] P.C. Gilmore, R.E. Gomory. "*A linear programming approach to the cutting stock problem*." *Operations Research,* vol. 9, pp. 849 – 859, (1961).

[Ghosh 03] D. Ghosh. *Neighborhood search heuristics for the uncapacitated facility location problem.* European Journal of Operational Research vol.150, pp. 150–162, (2003).

[Goldberg 85] D.Goldberg et R. Lingle. *Alleles, loci and the Traveling Salesman Problem* In Proc. International Joint Conference on Artificial Intelligence, (1985).

[Golden 98] B.L. Golden, E.A. Wasil,J.P Kelly, I.M. Chao. *The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem sets and computational resultus.* Crainic TG, Laporte G, editors. Fleet management and logistics. Dordrecht: Kluwer pp. 33-56, (1998).

[Hakimi 64] L. Hakimi. "*Optimum Locations of Switching Centers and the Absolute Centers and the Medians of a Graph."* Operations Research, vol.12, pp. 450-459, (1964).

[Hakimi 65] L. Hakimi. *Optimum Distribution of Switching Centers in a Communication Network and Some Related Graph Theoretic Problems.* Operations Research vol.13, pp. 462-475, (1965).

[Handler 90] G.Y. Handler. *P-Center Problems: Discrete Location Theory.* eds. P.B. Mirchandani and R.L. Francis pp. 305-347, 1990.

[Hansen 98] P. Hansen et N. Mladenovic. *An Introduction to Variable Neighborhood Search*. In S. Voss, S. Martello, I. H. Osman, and C. Roucairol, editors, *Meta-heuristics, Advances*

and Trends in Local Search Paradigms for Optimization, pp. 433–458. Kluwer Academic Publishers, (1998).

[Harris 05] S. Harris. *Here Come the Robots*. Association of American Medical Colleges, (2005). Available at http://www.aamc.org/newsroom/reporter/oct05/robots.htm.

[Hayes 02]A. T. Hayes et P. Dormiani-Tabatabaei, "*Self-organized flocking with agent failure : Off-line optimization and demonstration with real robots*" in Proc. of the IEEE Int. Conf. on Robotics and Automation, pp. 3900–3905, (2002).

[Hogan 86] K. Hogan, C.S. ReVelle "*Concepts and applications of backup coverage*," Management Science, vol. 34, pp. 1434-1444, (1986).

[Ijspeert 01]A. J. Ijspeert, A. Martinoli, A. Billard, et L. Gambardella, "*Collaboration through the exploitation of local interactions in autonomous collective robotics : The stick pulling experiment*" Autonomous Robots, vol. 11, no. 2, pp. 149–171, (2001)

[Jarvinen 72] P.J. Jarvinen et J. Rajala. *A branch abound algorithm for seeking the p-median.* Operations Research, vol.20, pp. 173-178, (1972).

[Kariv 79] O. Kariv et S.L. Hakimi. "*An Algorithmic Approach to Network Location Problems". The P-Medians.* SIAM Journal of Applied Mathematics 37, pp. 539-560, (1979).

[Kiekintveld 06] Kiekintveld, Christopher and Miller, Jason and Jordan, Patrick R. and Wellman, Michael P.;: "Controlling a supply chain agent using value-based decomposition"; P*roceedings of the 7th ACM conference on Electronic commerce,* (2006)

[Klose 05] A. Klose and A. Drexl (2005). *Facility location models for distribution system design*, European Journal of Operational Research, Vol. 162, pp. 4-29, 2005

[Kube 00]C. R. Kube and E. Bonabeau, "*Cooperative Transport by Ants and Robots*" Robotics and Autonomous Systems, vol. 30, no. 1–2, pp. 85–101, (2000).

[Kuehn 63] A.A. Kuehn, et M.J. Hamburger, "*A heuristic program for locating warehouses*", Management Science, vol. 9, pp. 643-666, 1963.

[Krieger 00]M. J. B. Krieger et J.-B. Billeter, "*The call of duty : Self-organised task allocation in a population of up to twelve mobile robots*" Robotics and Autonomous Systems, vol. 30, no. 1-2, pp. 65–84, (2000).

[Lubbecke 05] M.E. Lubbecke and J. Desrosers, "*Selected topics in column generation*," *Operations Research*, 53/6, pp. 1007-1023, (2005).

[Martello 90] S. Martello and P. Toth, *Knapsack problems: algorithms and computer implementations,* John Wiley & Sons, (1990)

[Martinoli 99]A. Martinoli, A. J. Ijspeert, et F. Mondada, "*Understanding collective aggregation mechanisms : From probabilistic modelling to experiments with real robots*" Robotics and Autonomous Systems, vol. 29, no. 1, pp. 51–63, (1999)

[Maranzana 64] F.E. Maranzana. *On the location of supply points to minimize transport costs.*Operations Research Quaterly, vol 15 pp 261-270

[Martinich 88] J. Martinich. *A vertex-closing approach to the p-center problem.*Naval Res. Log. 35, pp. 185-201, 1988.

[McMillen03] : Colin McMillen, Kristen N. Stubbs, Paul E. Rybski, Sascha A. Stoeter, Maria Gini and Nikolaos P. Papanikolopoulos. *Resource scheduling and load balancing in distributed robotic control systems.* Robotics and Autonomous Systems 44 (2003) 251–259

[Millonas 93]M. Millonas "*Swarms, Phase Transitions, and Collective Intelligence*" in eprint arXiv:adap-org/9306002. (1993).

[Minieka 70] E. Minieka. *The m-center Problem.* SIAM Review 12, pp. 138-139, (1970).

[Mirchandani 79] P.B. Mirchandani et A.R. Odoni. *Locations of medians on stochastic networks.* Transportation Science vol. 13(2) pp 85-97, (1979).

[Teitz 68] M.B. Teitz et P. Bart. *Heuristic methods for estimating the vertex median of a weighted graph.* Operations Research vol. 16, pp. 955-961, (1968)

[Thacker 05] P. D. Thacker. *Physician-Robot Makes the Rounds*. Journal of the American Medical Association Vol. 293 No. 2, (2005).

[Thangiah 93] Thangiah SR. *Vehicle routing with time windows using genetic algorithms.* Report SRU- CpSc-TR-93-23, Slippery Rock University, Slippery Rock , (1993).

[Toregas 71]C.S.R Toregas, C. ReVelle and L. Bergman. "*The location of emergency service facilities*," Operations Research, vol. 19, pp. 1363-1373, (1971).

[Oliver 87] I. Oliver, D. Smith, et J. Holland. *A study of permutation crossover opereation on the travelling salesman problem.* In Proc. International Joint Conference on Artificial Intelligence (1987).

[Potvin 96] Potvin J-Y, Bengio S. *The vehicle routing problem with time windows part II: genetic search.* INFORMS journal on computing; vol.8, pp.165-172, (1996).

[Prins 04] C. Prins. *A simple and effective evolutionary algorithm for the vehicle routing problem.* Computers and Operations Research; vol. 31(12): pp.1985-2000, (2004).

[Pullan 08] W. Pullan *A memetic genetic algorithm for the vertex p-center problem.* Journal Evolutionary Computation vol. 16, pp. 417-436, (2008).

[ReVelle 05] C.S. ReVelle and H. A. Eiselt. "*Location analysis: A synthesis and survey*," European Journal of Operational Research, Vol. 165, pp. 1-19, (2005).

[Rybski 07]P. Rybski, A. Larson, H. Veeraraghavan, M. LaPoint, and M. Gini, "*Performance evaluation of a multi-robot search & retrieval system : Experiences with MinDART*" Int. Journal of Intelligent and Robotic Systems, (2007).

[Schilling 79] D.A. Schilling, D.J. Elzinga, J. Cohon, R.L. Church, C.S. Revelle. "*The TEAM/FLEET models for simultaneous facility and equipment sitting,*" Transportation Science, vol. 13, pp. 192-200, (1979).

[Schmitt 94] L.J. Schmitt. *An empirical study computational study of genetic algorithms to solve order problems: an emphasis on TSP and VRPTC.* PhD Dissertation, University of Menphis ,(1994).

[Schmitt 95] L.J. Schmitt. *An evaluation of a genetic algorithm approach to the VRP*.Working paper, Departement of Information Technology Management, Christian Brothers University Memphis, (1995).

[Setchi 04]: Rossitza M. Setchi, and Nikolaos Lagos. *Reconfigurability and reconfigurable manufacturing systems: state-of-the-art review*. Industrial Informatics, 2004. INDIN '04. 2004 2nd IEEE International Conference . pp.529-535, (2004).

[Silva 01] R.M. De A Silva et G.L. Ramalho. "*Ant system for the set covering problem*". Systems, Man, and Cybernetics, IEEE International Conference,vol.6, pp.3129-3133, (2001).

[Smith 83] Smith, R.G. and Davis, R.; "Negotiation as a metaphor for distributed problem solving" *Artificial Intelligence*, (1983)

[Spielberg 69] K. Spielberg. "*Algorithms for the Simple plant Location Problems with some side constraints*". Operational Research, vol. 17, pp. 85-111, (1969).

[Sridharan 95] R. Sridharan. "*A Lagrangean Heuristic for the Capacitated Plant Location Problem with single sourcing constraints*." European Journal of Operational Research vol. 66, 305-312, (1995).

[Sun 06] M. Sun. Solving *the uncapacitated facility location problem using tabu search.* Computers & Operations Research vol. 33 pp. 2563–2589, (2006).

[Walsh 98] Walsh, W.E. Wellman, M.P. Wurman, P.R. MacKie-Mason, J.K. Michigan Univ., Ann Arbor, MI; "Some economics of market-based distributed scheduling", *Distributed Computing Systems*, (1998).

[Weber 1909] A. Weber. Uber den Standort der Industrien . Tubingen, J.C.B. Mohr. Traduction en anglais: *The Theory of the location of Industries* . Chicago University Press.(1909).

[Wellman 05] Wellman, M.P. and Estelle, J. and Singh, S. and Vorobeychik, Y. and Kiekintveld, C. and Soni, V.; "Strategic interactions in a supply chain game"; *Computational Intelligence*, (2005).

[Wilson 04] M. Wilson, C. Melhuish, A. B. Sendova-Franks, et S. Scholes, "*Algorithms for building annular structures with minimalist robots inspired by brood sorting in ant colonies*" Autonomous Robots, vol. 17, no. 2–3, pp. 115–136, (2004).

[Xu 10] Shuo Xu, Ze Ji, Duc Troung Pham, Fan Yu, Bio-Inspired Binary Bees Algorithm for a Two-Level Distribution Optimisation Problem, Journal of Bionic Engineering, vol. 7, Issue 2, p. 161-167, June 2010.

[Yim 03]: Mark Yim, Kimon Roufas, David Duff, Ying Zhang, Craig Eldershaw and Sam Homans. *Modular Reconfigurable Robots in Space Applications*. Autonomous Robots 14, 225–237, 2003

[Youssef 07] Ayman M. A. Youssef and Hoda A. ElMaraghy. *Optimal configuration selection for Reconfigurable Manufacturing Systems.* International Journal of Flexible Manufacturing Systems, vol.19, pp.67-106, (2007).

[Zhang 07] W. Zhang et J. Hu. *Low Power Management for Autonomous Mobile Robots Using Optimal Control.* The 46th IEEE Conference on Decision and Control New Orleans, LA, USA, Dec. 12-14, (2007).

[Zlot 06] R. Zlot and A. Stentz, "*Market-based multirobot coordination for complex tasks,*" Int. J. of Robotics Research, vol. 25, no. 1, pp. 73–102, (2006).

École Nationale Supérieure des Mines
de Saint-Étienne

NNT : *2011 EMSE 0618*

Hassan BAALBAKI

Logistics in hospitals using mobile reconfigurable robots

Speciality : Industrial Engineering

Keywords : Scheduling, Robots, Team Behaviour, Mission Assignment, Genetic Algorithms, Column Generation Approach, Collaborative Decision Making, Swarm …

Abstract :

Due to the expansion of the life duration and the shortage of medical personal in hospitals the EU funded IWARD project as part of the IFP6 program. The aims of this project were to assist the medical personnel in logistic and non medical tasks (transport, cleaning, environmental monitoring, guidance and tele-monitoring) through the usage of mobile, reconfigurable, rechargeable robots, thus letting the Medical staff to concentrate on medical aspects of their work.
This thesis was part of this project, and our work consisted on developing a decision making framework for the team of robots.
In the first part of the thesis, we address the strategic decisions essentially the: (i) the robots' home station location problem, (ii) Robot‘s reconfiguration problems and (iii) Robots recharging scheduling. We formulate those problems as a linear problems and we propose to solve them using Mixed Integer Programming (MIP). We also present a formulation using a column generation approach to solve those problems.

In the later part we address the tactical problems, mainly the mission assignment, the mission scheduling and rescheduling. We present two different approaches; a centralized decision finder implemented using genetic algorithms. And a decentralized approach using auction like and market based algorithms in order to provided collaborative decision making framework.
Finally we compare those two approaches using a custom made discrete event simulation (DES).

École Nationale Supérieure des Mines
de Saint-Étienne

NNT : *2011 EMSE 0618*

Hassan BAALBAKI

Logistique hospitalière à l'aide de robots mobiles reconfigurables

Spécialité: Génie Industriel

Mots clefs : Ordonnancement, Prise de décision Collaborative, Affectation de missions, Algorithmes génétiques, Génération de colonnes, Robots, Swarm

Résumé :

Ce manuscrit expose notre travail dans le cadre du projet IWARD et détaille la couche de gestion et de décision du groupement de robots. Ce projet avait comme objectif d'assister le personnel médical dans leur travail, ceci est réalisé en utilisant des robots mobiles, reconfigurables, et rechargeables. Ces robots sont conçus pour effectuer des taches logistiques comme : Le transport de médicaments, le nettoyage, le guidage des patients, la surveillance et la téléconsultation.

Dans la première partie de la thèse nous présenterons le problème stratégique qui consiste à déterminer les plannings de rechargement des robots, la configuration des robots opérationnels ainsi que la localisation des stations d'attentes des robots lorsqu'ils sont en état de veille. Différentes hiérarchies à plusieurs niveaux de décisions, sont formulées comme des programmes linéaires en nombres entiers. Des formulations utilisant l'approche de génération de colonnes sont aussi développées pour résoudre ces problèmes.

Dans la deuxième partie, le problème tactique est exposé, ceci consiste à affecter les taches arrivantes aux différents robots et d'ordonnancer dynamiquement l'exécution ces missions. Deux approches sont inspectées une version centralisée utilisant les algorithmes évolutionnaires et une autre version distribuée utilisant les algorithmes d'enchères inversées. Afin de mettre à l épreuve ces deux approches, une simulation a événements discrets a été conçue et développée spécifiquement pour le projet, permettant ainsi d'évaluer ces deux approches.