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.

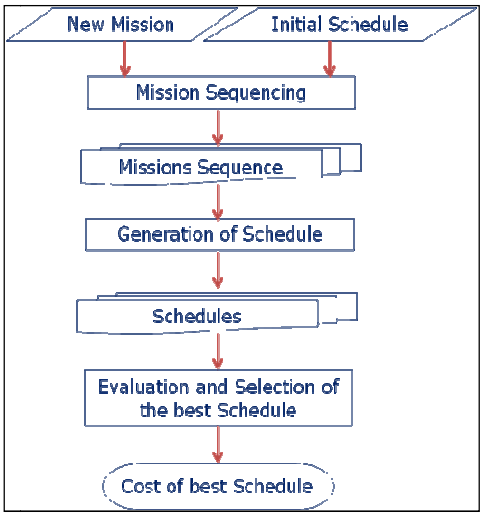

Figure 46: Eléments clé de l'agent de décision et d'évaluation

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.

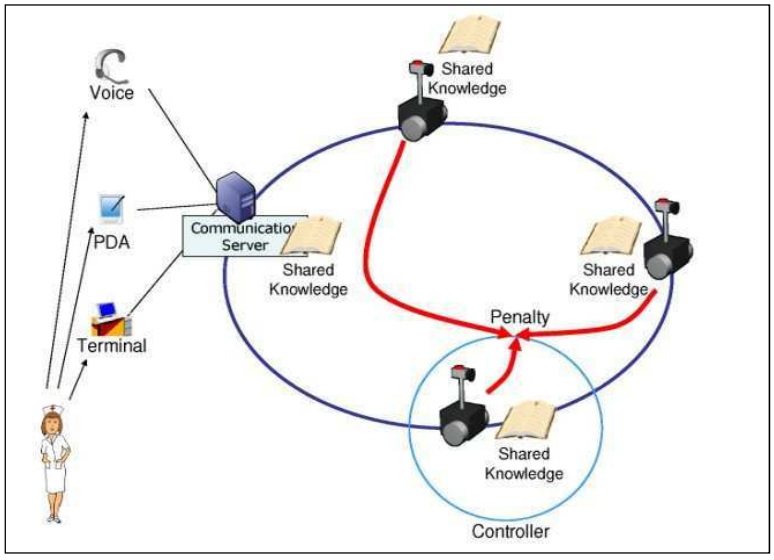

Figure 47: Réception des coûts d'exécution

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.

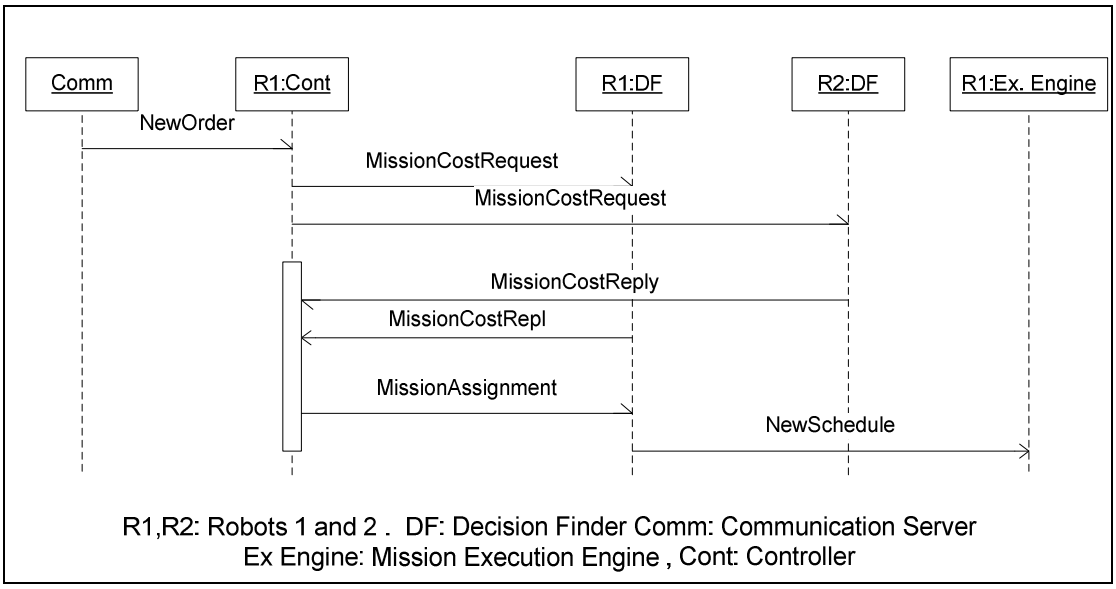

Figure 48 : Négociations et interactions entre les pairs durant une affectation de mission

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)

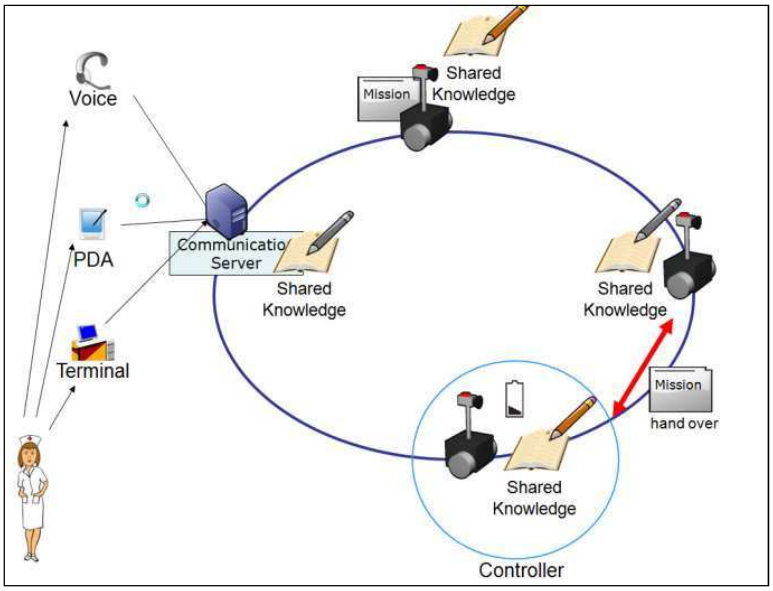

Figure 49 : Passation de missions suite à un problème énergétique

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.

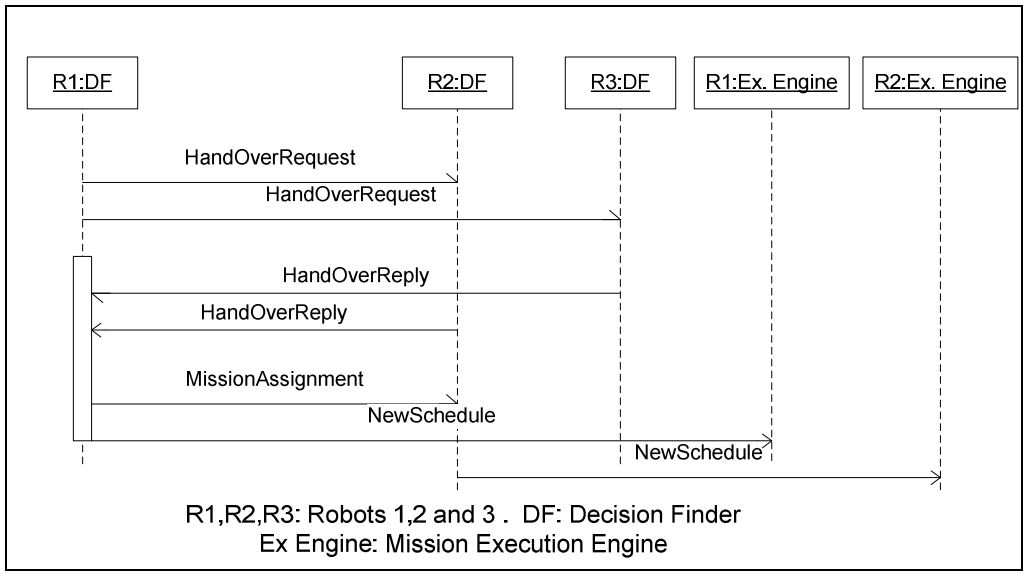

Figure 50 : Négociations lors d'un processus de leg

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.

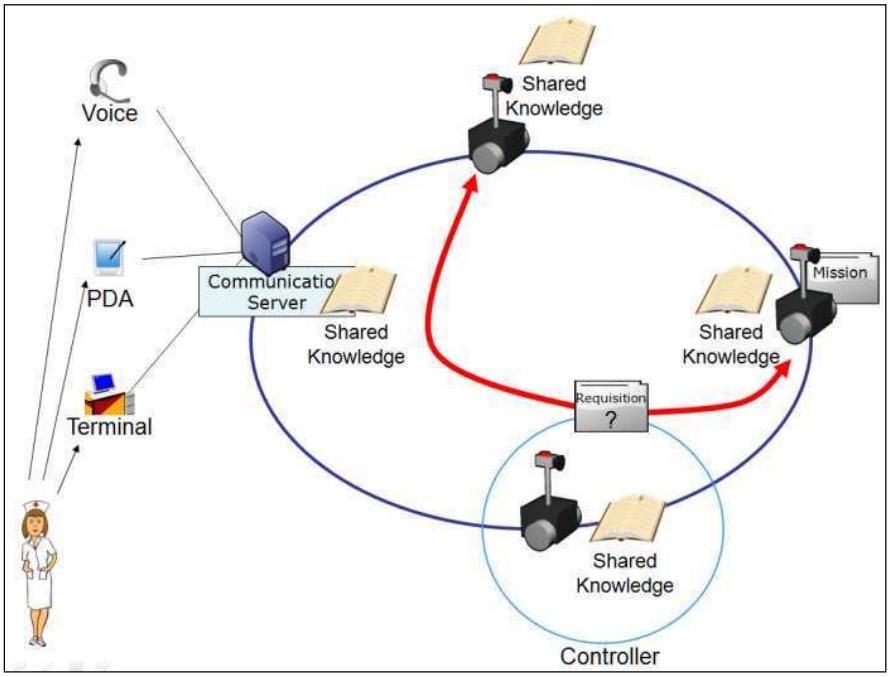

Figure 51 : Requête de réquisition de mission

```
$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

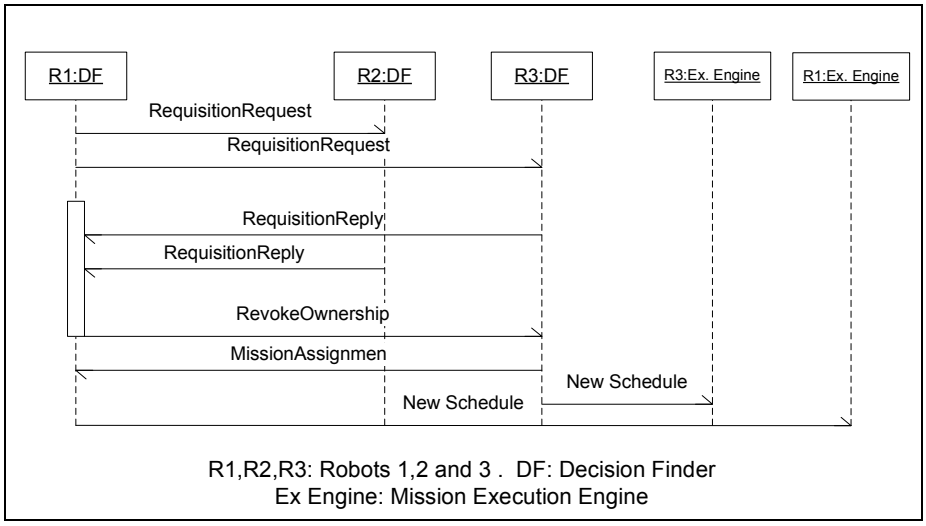

Figure 52 : Interactions entre les différentes entités lors des négociations de réquisition

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.

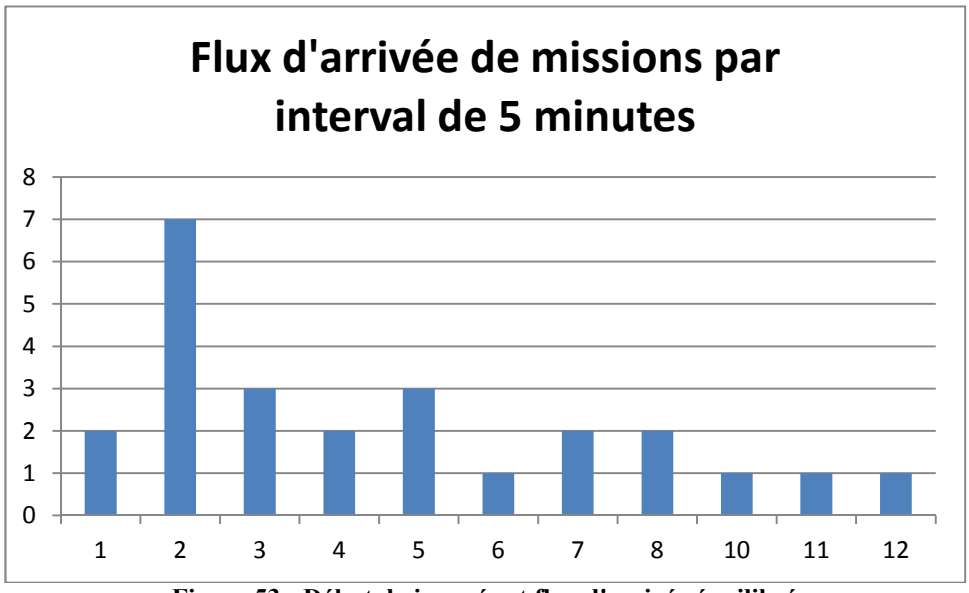

Figure 53 : Début de journée et flux d'arrivée équilibré

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.

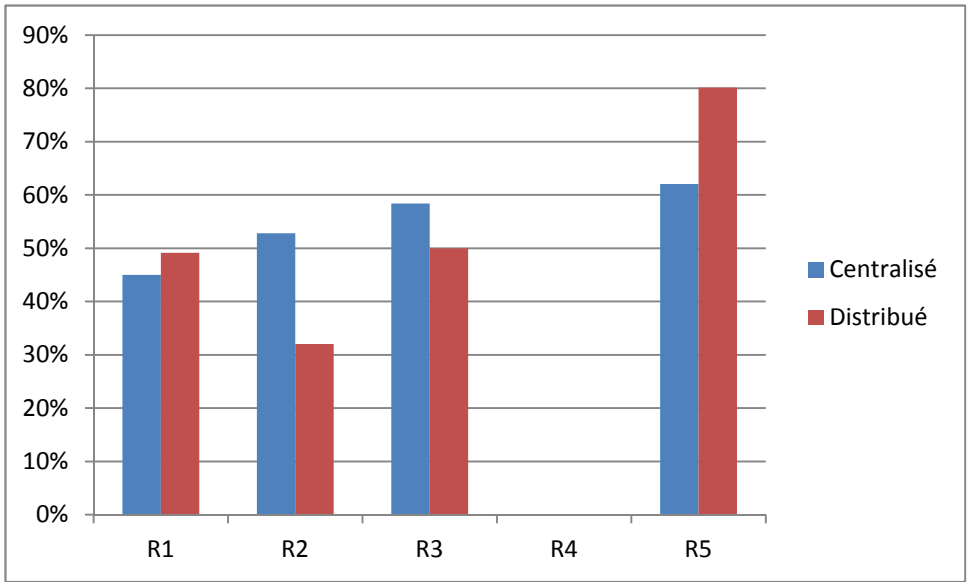

Figure 54 : Taux d'utilisation des robots

La Figure 55 compare les pénalités dues aux retards qui sont utilisées comme critère d'évaluation des plannings de chaque robot.

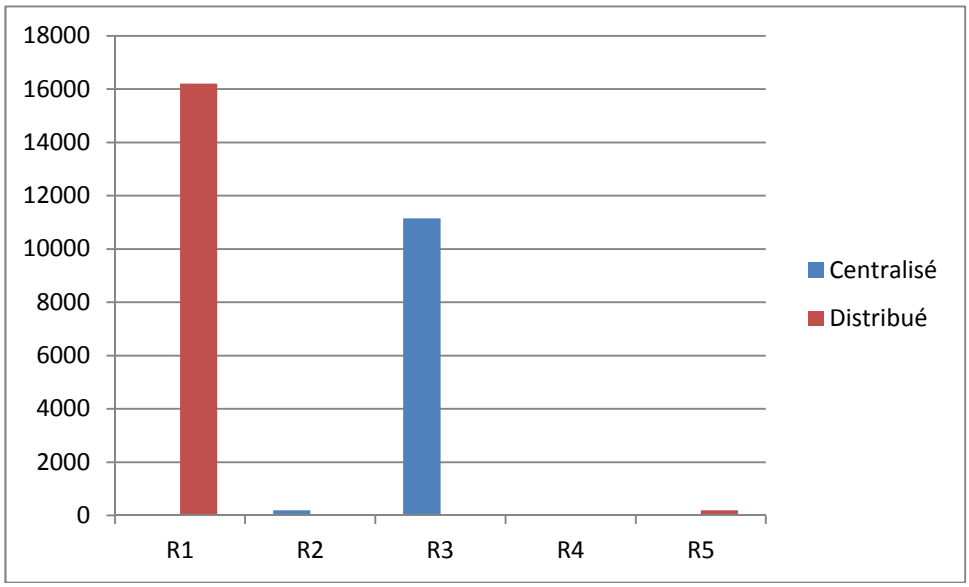

Figure 55 : Fonction Objectif par robot.

	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

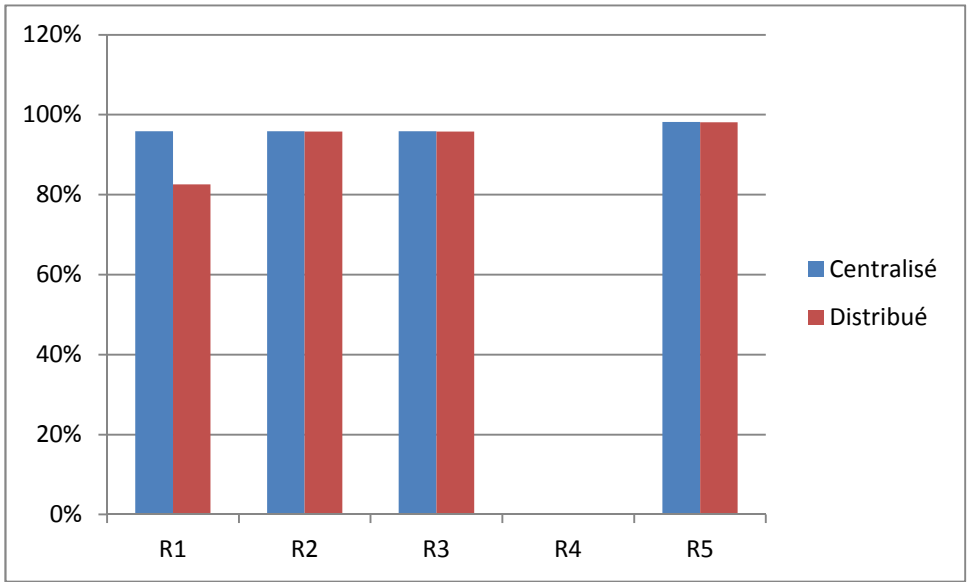

Figure 57: Taux d'utilisation

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.

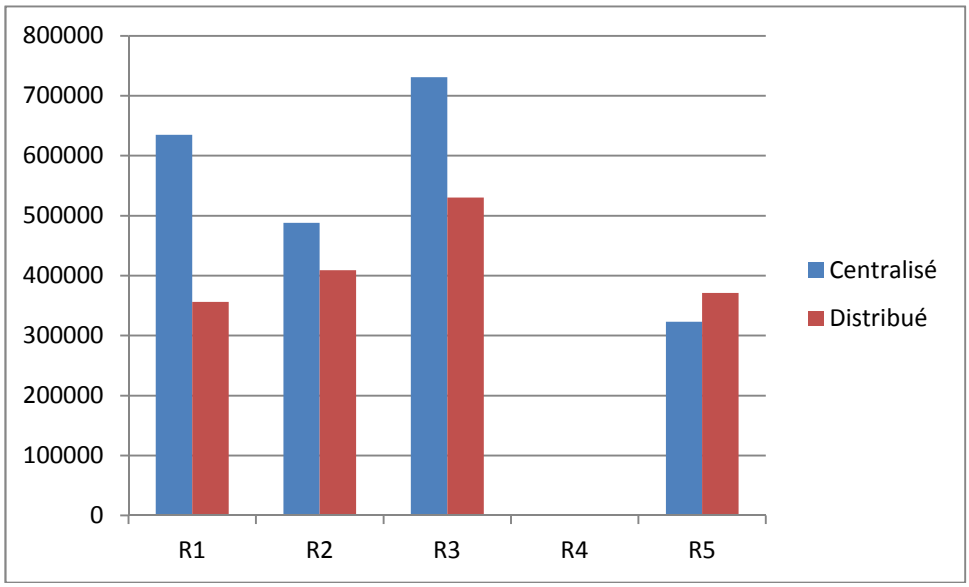

Figure 58: Fonction Objectif

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.