



IMHOTEP

AFRICAN JOURNAL OF PURE AND APPLIED MATHEMATICS

Imhotep Mathematical Proceedings Volume 2, Numéro 1, (2015), pp. 88 – 102.

Un système multi-agents de méta-ordonnement distribué de grille

J.S. Wouansi

Université de Ngaoundéré, Faculté
des Sciences, Département de
Mathématiques et Informatique.
jeremie_ws@yahoo.fr

V.C. Kamla

D. Tieudjo


Université de Ngaoundéré, Ecole Nationale
Supérieure des Sciences Agro-Indusrtielles
(ENSAI), Département de Mathématiques et
Informatique, Laboratoire de Mathématiques
Expérimentales (LAMEX).
vc_kamla@yahoo.fr
tieudjo@yahoo.com

Abstract

Dans ce papier, nous proposons un modèle de méta-ordonnement distribué de grille. Ce modèle est basé sur les Système Multi-Agents (SMA). Le méta-ordonneur utilise la Théorie de l'Utilité Multi-Attributs (TUMA) pour l'aide à la décision. Il utilise également un protocole de vente aux doubles enchères et d'appel d'offre, pour l'ordonnement des applications sur les ressources des sites, de façon à optimiser plusieurs métriques. Notamment, l'équilibrage de charge, l'utilité des participants le délai des applications le temps d'attente et de réponse. On s'intéresse au temps de réponse. Il ressort de l'analyse de performance que le système proposé, en plus des métriques optimisées par les modèles à la base, améliore le temps de réponse d'un facteur d'au moins $p/2$, $p > 2$ étant le nombre de sites.

Proceedings of the 4th annual workshop on
CRyptography, Algebra and Geometry
(CRAG-4), 21 - 25 July 2014, University of
Dschang, Dschang, Cameroon.

<http://imhotep-journal.org/index.php/imhotep/>

Imhotep Mathematical Proceedings 

Un système multi-agents de méta-ordonnancement distribué de grille

J.S. Wouansi, V.C. Kamla and D. Tieudjo

Résumé. Dans ce papier, nous proposons un modèle de méta-ordonnancement distribué de grille. Ce modèle est basé sur les Système Multi-Agents (SMA). Le méta-ordonnanceur utilise la Théorie de l'Utilité Multi-Attributs (TUMA) pour l'aide à la décision. Il utilise également un protocole de vente aux doubles enchères et d'appel d'offre, pour l'ordonnancement des applications sur les ressources des sites, de façon à optimiser plusieurs métriques. Notamment, l'équilibrage de charge, l'utilité des participants le délai des applications le temps d'attente et de réponse. On s'intéresse au temps de réponse. Il ressort de l'analyse de performance que le système proposé, en plus des métriques optimisées par les modèles à la base, améliore le temps de réponse d'un facteur d'au moins $p/2$, $p > 2$ étant le nombre de sites.

Keywords. Grille, Méta-ordonnancement, modèles économiques, performance, TUMA, SMA.

I. Introduction

Les grilles sont des infrastructures informatiques distribuées composées des ressources de calcul haute performance telles que les grappes d'ordinateurs ou des supercalculateurs, qui appartiennent à des domaines administratifs divers. Ces derniers sont gérés par des gestionnaires de ressources couplés à des ordonnanceurs locaux. OAR est par exemple un gestionnaire très utilisé dans la grille Grid5000. En général, une grille est gérée par un intergiciel associé à un méta-ordonnanceur dont le rôle est de choisir les meilleurs sites de ressources pour l'exécution des applications candidates. On peut citer Gridway de la grille Globus, eNanos Grid Resource Broker de GT3, Gridbus Broker de Gridbus et Community Scheduler Framework [21]. Les auteurs de [2, 15] présentent quelques technologies de grilles. Dans une grille, le méta-ordonnanceur doit négocier avec les gestionnaires associés aux ordonnanceurs des sites, sans toutefois intervenir dans leur administration. Il doit aussi tenir compte de la nature très dynamique des ressources de la grille, qui sont la plupart du temps utilisées localement. Dans ce contexte, plusieurs applications concurrentes sollicitent des ressources, qui dans la plupart des cas ne suffisent pas pour satisfaire la demande. D'un autre côté, chaque gestionnaire de ressources dispose de ses propres applications locales à placer sur le site. Les premiers méta-ordonnanceurs, comme le spécifient les auteurs de [6, 11, 13, 24], visaient surtout l'optimisation des métriques de performance basées

Communication présentée au 4^{ème} atelier annuel sur la CRYptographie, Algèbre et Géométrie (CRAG-4), 21 - 25 Juillet 2014, Université de Dschang, Dschang, Cameroun / Paper presented at the 4th annual workshop on CRYptography, Algebra and Geometry (CRAG-4), 21- 25 July 2014, University of Dschang, Dschang, Cameroon.

sur le système, telles l'utilisation des ressources, l'équilibrage de charge ou le temps d'exécution des applications des utilisateurs. Aujourd'hui, la croissance des exigences en terme de besoins en QoS des clients et même des fournisseurs nécessite des méta-ordonnanceurs beaucoup plus sophistiqués pouvant y répondre efficacement. En d'autre terme, le méta-ordonnanceur doit prendre en compte dans ce contexte de marché, les métriques de performance application-centrée, basées sur des exigences telles les délais d'exécution des applications, le nombre de CPU requis, et celles ressource-centrée, comme la priorité des sites de ressources des fournisseurs. Il doit en effet offrir des mécanismes économiques lui permettant de reconstituer le scénario de marché réel en place, tel les enchères, le marché des commodités, l'appel d'offre et bien d'autres. Depuis quelques années, comme on peut le constater dans [5, 8, 9, 16, 22], des auteurs s'intéressent à une approche économique pour mettre en oeuvre de tels méta-ordonnanceurs. Certains dans [3, 4, 10, 12, 24] ont utilisé des systèmes multi-agents pour avoir plus de flexibilité. Enfin, d'autres dans [1, 7, 14, 17–20, 24] ont utilisé une approche distribuée pour obtenir plus de performance, et un système plus tolérant. Dans ce travail, nous nous intéressons à trois travaux récents offrant de bonnes performances, à savoir celui d'Aguilar et *al.* (2009) [3], de Mauricio Solar et *al.* (2012) [24], et enfin de Saurabh et *al.* (2013) [16]. En effet, Aguilar utilise une approche multi-agents pour coordonner un processus de méta-ordonnancement centralisé basé sur la vente aux enchères et le protocole "contract-net". Cependant, il fait intervenir explicitement et à plusieurs tours, les participants dans le processus de coordination ; ce qui ne réduit pas le temps de réponse. D'un autre côté, le seul critère de choix est basé sur l'unique valeur qu'est la monnaie. Solar, avec l'approche multi-agents aussi, propose un protocole de négociation basé sur la disponibilité des ressources, pour réaliser un méta-ordonnancement distribué avec surveillance des tâches. Une des insuffisances de Solar est que la priorité est toujours donnée aux tâches dont la date limite est plus proche, sans souci de leur ordre d'arrivée, ce qui force les tâches moins urgentes à toujours les attendre. Par ailleurs, au pire des cas, il faut consulter tous les noeuds de la grille, en mode point à point, ce qui fera attendre très longtemps la tâche concernée. Enfin Saurabh propose un protocole qui utilise la théorie de l'utilité multi-attributs pour évaluer les ressources ainsi que les demandes afin de réaliser un méta-ordonnancement centralisé basé sur la double vente aux enchères. Cependant, la centralisation de la prise des décisions impose de nombreuses communications longues, pour la soumission des tâches et des offres, et nécessite en plus que le méta-ordonnanceur entretienne une base d'information globale de l'ensemble de la grille ; ce qui peut entraîner une perte de cohérence d'information due aux temps de communication et de prise de décision. Par ailleurs l'urgence considérée dans la formule ne permet pas d'extraire à priori les demandes dont le délai ne peut être satisfait.

Le modèle de Saurabh résout donc les limites du modèle d'Aguilar et en partie celui de Solar, mais perd d'une part les avantages qu'offre la distribution que ce dernier propose, et d'autre part augmente la saturation du réseau et le temps de réponse [23]. Dans cet article, nous décrivons un système de méta-ordonnancement distribué de grille, qui améliore le temps de réponse et diminue la saturation du réseau. Ce système combine les politiques de Solar et Saurabh pour offrir de meilleures performances.

II. Revue de la littérature

Saurabh dans [16] définit un système d'évaluation des demandes et des ressources pour un système centralisé, par les fonctions d'utilité. Il évalue à tout instant t l'enchère d'une demande i , notée $b_i(t)$, comme suit :

$$b_i(t) = H_i \times \frac{v_i}{v_{max}} \times \frac{d_{max} - t}{d_i - t} \times \frac{t + 1 - st_i}{t + 1 - st_{min}} \times \frac{Demande}{Offre}; d_i \neq t; \quad (1)$$

Demande : ensemble des demandes d'un type de ressource ; *Offre* : ensemble des ressources

disponibles du type demandé; $d_i - t$: indique l'urgence de la demande D_i à être exécutée à mesure que t augmente, d_i (deadline) est la date limite de D_i donnée par l'utilisateur; st_i : estampille, qui est la date d'arrivée de la demande D_i dans le système; v_i : la valeur initiale de la demande D_i , donnée par l'utilisateur l'ayant soumise; H_i : constante de proportionnalité qui vaut 1 pour la simulation; $v_{max} / d_{max} / st_{min}$: respectivement maximum des v_i / d_i et minimum des st_i . De même, il évalue à tout instant t l'offre d'un site j de ressources $a_j(t)$ comme suit :

$$a_j(t) = O_j \times w_{j,t} \times \frac{c_{j,t}}{c_{max,t}} \times \frac{L_{j,t}}{L_{max,t}} \times \frac{Demande}{Offre}; \quad (2)$$

$w_{u,t}$: temps d'attente de disponibilité de la queue de ressources à partir de l'instant t ; $c_{k,t}$: valeur initiale donnée par le fournisseur du site à t ; $l_{j,t}$: charge de la queue de ressources à t ; O_j : constante de proportionnalité valant 1 pour la simulation.

Dans ce modèle, le méta-ordonnanceur effectue les opérations décrites par le tableau 1. t_{cpu} est le coût en temps d'une opération élémentaire de calcul du CPU, t_{com} le coût en

TABLE 1. Type d'opérations avec leurs consommations

TYPE D'OPERATION	NOM DE LA VARIABLE DE CONSOMMATION EN TEMPS	FONCTION DE CONSOMMATION EN TEMPS
Transfert des demandes	t_{com}	COMMUNICATION : $2nt_{com} + 2pt_{com}$
Transfert des offres	t_{com}	
Transfert des réponses aux demandes	t_{com}	
Transfert des réponses aux offres	t_{com}	CALCUL DES ENCHERES : $\alpha_1 n t_{cpu1} + \beta_1 p t_{cpu2}$ $\alpha_1 > 1, \beta_1 > 1$, des entiers
Calcul des enchères des demandes	t_{cpu1}	
Calcul des enchères des offres	t_{cpu2}	TRI ET ASSIGNATION : $(\frac{n(n-1)}{2} + \frac{p(p-1)}{2})t_{cpu} + TO _p^n$
Tri des enchères des demandes	t_{cpu}	
Tri des enchères des offres	t_{cpu}	
Assignation de n tâches aux ressources de p clusters	$TO _p^n$	

temps d'une opération élémentaire de communication du CPU, et com le coût d'une communication sur la bande passante d'un lien, α_1 le nombre d'opérateurs dans le calcul des enchères des demandes, α_2 celui des offres. On pose $t_{cpu} = \Delta t$ et $t_{com} = a\Delta t$, $a > 1$ et entier, vu que les communications sont plus coûteuses que les calculs locaux en CPU. Soit T_{rep} le temps de réponse à un demandeur de ressources.

- **Meilleur des cas** : décision dans le même cycle, le premier à recevoir.

$$T_{rep} = \left[\frac{n(n-1)}{2} + \frac{p(p-1)}{2} + (\alpha_1 + a)n + (\beta_1 + a)p \right] \times \Delta t + TO|_p^n; \quad (3)$$

- **Pire des cas** : décision à l'expiration du délai après k cycles, le dernier à recevoir.

$$T_{rep} = \sum_{i=1}^k T_{rep}^i; \quad (4)$$

T_{rep}^i étant le temps de réponse dans le cycle i , le demandeur n'ayant pas reçu après toutes les réponses, c'est-à-dire :

$$T_{rep}^i = \left[\frac{N_i(N_i-1)}{2} + \frac{p(p-1)}{2} + (\alpha_1 + 2a)N_i + (\beta_1 + 2a)p \right] \times \Delta t + TO_i|_p^{N_i}; \quad (5)$$

avec N_i le nombre total des tâches présentes dans la grille au cycle i .

Dans le système de Solar [24], il n'existe pas de système d'information central. Les tâches sont directement soumises sur chaque noeud local, et le système multi-agents de méta-ordonnancement gère la commutation entre une queue locale pour les tâches du site et une queue globale pour les tâches de la grille. Quand une tâche est dans la queue locale, elle est susceptible de s'exécuter sur le site ou de retourner dans la queue globale, et dans cette dernière elle peut être exécutée à distance. Le protocole de distribution utilise une négociation point à point entre l'agent de soumission et celui dont ce dernier sollicite le service, jusqu'à consensus. Ici, un ensemble de services distribués collaborent pour ordonnancer les applications des utilisateurs sur les ressources de la grille. On suppose $n_i = \frac{n}{k_i}$, ($k_i > 1$, un réel), une proportion du nombre total des tâches dans la grille n . Le méta-ordonnancement distribué effectue les opérations décrites par le tableau 2.

TABLE 2. Type d'opérations avec leurs consommations

TYPE D'OPERATION		CONSOMMATION EN TEMPS DE L'OPERATION	FONCTION DE CONSOMMATION EN TEMPS
COMPOTEMENT GLOBAL DU META-ORDONNANCEUR : <i>Calcul des priorités des ressources, tri et sélection de site</i>	Calcul du Taux d'Occupation d'un Processeur d'un cluster	$\alpha_1 t_{cpu}, \alpha_1 > 1$	$(2\alpha_1 + \beta_1 + (a+1)(p-1))t_{cpu} + a(p-1)t_{cpu} + (p-1)\text{Calcul local}$ $\alpha_1 > 1, \beta_1 > 1$, des entiers, représentant chacun le nombre d'un type d'opérateurs dans la formule correspondante. On rappelle que $a t_{cpu} = t_{com}, a > 1$ est la consommation d'un transfert.
	Calcul du Temps d'attente de disponibilité de quelques processeurs du noeud	$\alpha_1 t_{cpu}, \alpha_1 > 1$	
	Calcul de la probabilité qu'une tâche atteindra un des noeuds à temps	$\beta_1 t_{cpu}, \beta_1 > 1$	
	Transfert des valeurs calculées	$a(p-1)t_{cpu}, a > 1$	
	Comparaison des paires de noeuds	$(p-1)t_{cpu}$	
	Négociation point à point entre agents	$a(p-1)t_{cpu} + (p-1)\text{Calcul local}$	
COMPOTEMENT LOCAL DU META-ORDONNANCEUR : <i>Calcul des priorités des tâches, tri et assignation de ressources</i>	Calcul de la date de fin souhaitée	$cn_1 t_{cpu}$	Calcul local = $\left(\frac{ni(ni-1)}{2} + 5cn_1\right)t_{cpu} + TO_1^{ni}$ $c > 1$, un entier représentant le nombre d'un type d'opérateurs dans la formule correspondante.
	Calcul du temps d'attente jusqu'au temps présent	$cn_1 t_{cpu}$	
	Calcul de l'estimation de la date de fin à partir du présent	$2cn_1 t_{cpu}$	
	Calcul de la date de livraison anticipée à partir du présent	$cn_1 t_{cpu}$	
	Calcul de la date de livraison au plus tard à partir du présent	$cn_1 t_{cpu}$	
	Tri par comparaison de la liste des tâches selon la priorité	$\frac{ni(ni-1)}{2} t_{cpu}$	
	Assignation de n_i tâches aux ressources de 1 cluster	TO_1^{ni}	

- **Au meilleur des cas** : Comportement global et comportement local sans négociation

$$T_{rep} = \left[\frac{n_i(n_i - 1)}{2} + 5cn_i + (2\alpha_1 + \beta_1 + (a + 1)(p - 1)) \right] \times \Delta t + TO_1^{n_i}; \quad (6)$$

En remplaçant par la relation $n_i = \frac{n}{k_i}$, $k_i > 1$ un réel, on obtient :

$$T_{rep} = \left[\frac{n(n - k_i)}{2k_i^2} + 5c \frac{n}{k_i} + (2\alpha_1 + \beta_1 + (a + 1)(p - 1)) \right] \times \Delta t + TO_1^{\frac{n}{k_i}}; \quad (7)$$

- **Au pire des cas** : Comportement global et comportement local avec négociation point à point avec les $p-1$ autres agents

Le temps consommé pendant la négociation pour une tâche attendant dans un site est donné par la relation :

$$T_{negoce} = a(p - 1)\Delta t + \sum_{i=1}^{p-1} (T_{rep}^i - TO_1^{\frac{n}{k_i}}); \quad (8)$$

où T_{rep}^i est le temps de réponse au meilleur des cas dans le site local i différent du site ayant émis la demande de négociation. La tâche soumise à la négociation au pire des cas doit alors en plus de l'attente locale initiale, attendre la négociation avec le dernier des $p - 1$ autres sites,

c'est-à-dire, T_{negoce} et ensuite le temps d'assignation dans le site choisi. Ce temps est donné par la relation 9 :

$$T_{rep} = \left[\sum_{i=1}^p \frac{n(n-k_i)}{2k_i^2} + 5c \sum_{i=1}^p \frac{n}{k_i} + (2\alpha_1 + \beta_1 + (2a+1)(p-1))p \right] \times \Delta t + TO|_1^{\frac{n}{k_i}} + TO|_1^{\frac{n}{k_{elu}}}; \quad (9)$$

$\frac{n}{k_{elu}}$, $k_{elu} > 1$ réel, étant le nombre de tâches du site élu pour recevoir la tâche en négociation.

III. Système proposé

Les modèles présentés offrent des avantages qu'il est possible d'exploiter en les mettant ensemble, pour produire un système palliant les différents inconvénients de l'un et l'autre. Nous proposons dans le point suivant, un système distribué de méta-ordonnancement qui offre un meilleur temps de réponse. Notre démarche consiste à faire l'analyse des exigences fonctionnelles, ensuite identifier les entités du système multi-agents et décrire les interactions entre elles, après décrire les différentes contraintes et formules d'aide à la prise de décision et enfin nous faire une analyse de performance.

III.1. Exigences fonctionnelles

Ce point nous permet de décrire le système. L'environnement ξ est une grille constituée d'un nombre fini p ($p > 1$, entier) de clusters C_i ($i \leq p$), interconnectées à travers l'Internet.

$$\xi = \{C_1, C_2, \dots, C_p\} \quad (10)$$

Chaque cluster C_i , $i \leq p$, contient un nombre fini m_i de machines M_j^i de mêmes caractéristiques, i étant l'indice du cluster, et j l'indice de la machine.

$$C_i = \{M_1^i, M_2^i, \dots, M_{m_i}^i\} \quad (11)$$

Chaque machine M_j^i , $j \leq m_i$, contient également un nombre fini q_j de processeurs P_k^{ij} de mêmes caractéristiques, i étant l'indice du cluster, j l'indice de la machine, et k l'indice du processeur.

$$M_j^i = \{P_1^{ij}, P_2^{ij}, \dots, P_{q_j}^{ij}\} \quad (12)$$

À tout cycle de méta-ordonnancement, n tâches distinctes t_1, \dots, t_n venant des utilisateurs des différents processeurs des clusters, sont à placer. Le réseau est homogène et on considère les communications uniquement entre les clusters. Dans l'organisation centralisée, les n tâches sont soumises directement au méta-ordonnanceur central de la grille. Par contre dans celle distribuée, à chaque site C_j , $j = 1, \dots, p$, est soumis sa liste de n_j tâches t_{ij} $i = 1, \dots, n_j$, de façon que $n = n_1 + \dots + n_p$.

Le méta-ordonnancement distribué de grille nécessite un ensemble d'éléments pour son fonctionnement :

- les fournisseurs (Manager) et consommateurs (Broker) de ressources ;
- un gestionnaire des profils des demandes (Job Profile Manager - JPM) et un gestionnaire des profils des offres (Offer Profile Manager - OPM) ;
- une table de méta-ordonnancement enregistrant les différentes listes de demandes, offres et contrats (Metaschedule - MS) ;
- une page des contacts du méta-ordonnanceur (Local Yellow Page - LYP) ;
- un directeur des ventes pour la coordination des enchères et des appels d'offre (Coordinator).

La figure 1 décrit l'organisation des participants.

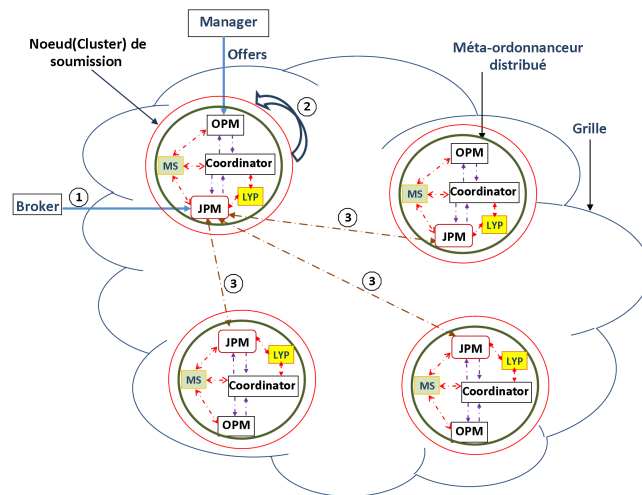


FIGURE 1. Structure organisationnelle du système de méta-ordonnancement proposé

1. : Le client (Broker) soumet sa demande de ressources au JPM du service distribué du méta-ordonnancement (Mo) en spécifiant ses exigences ;
2. : Le méta-ordonnancement du noeud fait un méta-ordonnancement local une fois que la table de méta-ordonnancement est mise à jour ;
3. : L'assignation des ressources n'est pas possible sur le noeud de soumission ; Le JPM du méta-ordonnancement du noeud de soumission engage un appel d'offre pour la demande (Call For proposal : cfp), aux méta-ordonnancements (Mo) répliqués de la même grille, le suit, puis choisit le site approprié pour accueillir et traiter la demande.

III.2. Interaction entre les agents du système

Ici nous décrivons notre système multi-agents avec les différentes communications entre les entités. Les éléments recensés précédemment sont les rôles de notre système, auxquels on ajoute les rôles *tjpm* (rôle fils créé par *jpm* pour la collecte des demandes), *topm* (rôle fils créé par *opm* pour la collecte des offres), le *cfpm* (rôle fils créé par *jpm* pour les appels d'offre) et *monitor* (rôle de surveillance des ressources). Les protocoles d'interaction décrivent les communications par échange de messages entre les différents rôles. La figure 2 décrit les interactions au niveau local, c'est-à-dire de chaque cluster, et la figure 3 les interactions au niveau global, c'est-à-dire de la grille.

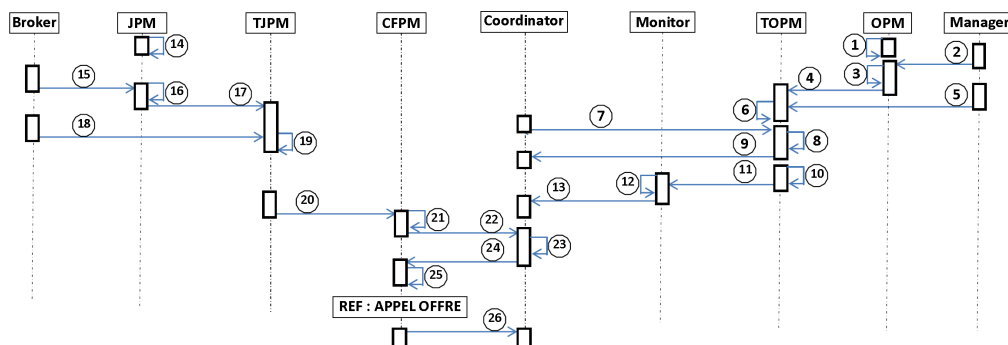


FIGURE 2. Diagramme d'interaction au niveau local

Les messages transmis entre les rôles de la figure 2 sont les suivants : 1 : initialiser *opm()*; 2 : demander connexion; 3 : créer processus *topm()*; 4 : transmettre données connexion du *manager()*; 5 : soumettre offre; 6 : collecter et trier offre selon le temps d'attente; 7 : récupérer offres déjà disponibles dans la table et la verrouiller; 8 : fusionner avec l'offre en cours; 9 : remplacer l'offre de la table par celle fusionnée; 10 : créer le moniteur *monitor()*; 11 : transmettre les dates des tranches de temps disponibles au moniteur; 12 : surveiller le rebours des dates de début et de fin; 13 : mettre à jour les dates des tranches; 14 : initialiser *jpm()*; 15 : demander connexion; 16 : créer processus *tjpm()*; 17 : transmettre données connexion du *broker()*; 18 : soumettre demande; 19 : collecte demande; 20 : transmettre liste demande; 21 : évaluer localement liste demande; 22 : mettre à jour l'utilité des demandes de la table et fusionner avec liste demande évaluée; 23 : associer demandes aux ressources; 24 : collecter demandes non ordonnancées; 25 : engager et suivre un appel d'offre; 26 : mettre à jour la table de méta-ordonnancement, pour les demandes traitées à distance.

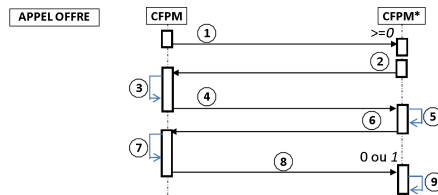


FIGURE 3. Diagramme d'interaction au niveau global

Les messages échangés entre les rôles de la figure 3 sont les suivants : 1 : envoyer SIGNAL_CFP; 2 : envoyer attributs locaux; 3 : calculer les valeurs locales des demandes; 4 : envoyer la liste d'information sur les demandes non ordonnancées (valeurs demandes, délais, ...); 5 : classer les demandes, vérifier le test QoS, calculer la probabilité des demandes; 6 : envoyer la liste des probabilités des demandes; 8 : envoyer l'accord au site sélectionné; 7, 9 : Choisir le site (7), mettre à jour la table de méta-ordonnancement en local.

Quatre principaux agents sont retenus pour les jouer les différents rôles : l'agent *broker* qui va jouer les rôles *broker*, *jobprofiler*, *tjpm*; l'agent *cfpm* qui va jouer le rôle du *cfpm*; l'agent *manager* qui va jouer les rôles *manager*, *offerprofiler*, *topm* et *monitor*; l'agent *coordinator* qui va jouer le rôle *coordinator*.

Les services à offrir par le système étant : Le profilage d'offre, le profilage local des demandes, la coordination des ventes et la surveillance.

III.3. Contraintes de distribution

a- Synchronisation

Le *cfpm()* émetteur est synchronisé avec les *cfpm()* voisins. Quand un *cfpm()* voisin reçoit un signal d'appel d'offre, il retire la liste courante des demandes de la collecte et la traite, après avoir ouvert une nouvelle liste pour les collectes ultérieures. À la fin du traitement de l'appel d'offre, tous les *cfpm()* voisins sélectionnés dans l'appel procèdent à l'ordonnancement, qu'ils soient élus ou non. Ceci assure que la décision prise par un émetteur d'appel d'offre s'appuie sur des données à jour dans le cycle d'ordonnancement, et donc est vraie dans toute la grille pour ce cycle. Le *cfpm()* procède par plusieurs phases :

1. Initialisation de la liste courante des demandes (il la met à vide);
2. Réception d'exception;
3. Copie de liste de demande, réinitialisation et évaluation locale;
4. Mise à jour de la table de méta-ordonnancement locale;
5. Copie des demandes non ordonnancées;
6. Appel d'offre (CFP : Signal, communication, décision et soumission);

7. Mise à jour de la table de méta-ordonnancement locale ;
8. Attente d'exception et passage à l'étape 2.

- Protocole de synchronisation

- L'exception provoquée au niveau d'une part global est ignorée si le $cfpm()$ n'est pas à la phase d'initialisation, de réception ou d'attente, et d'autre part au niveau local est mise en attente si le $cfpm()$ est dans d'autres phases ;
- Un $cfpm()$ voisin non élu est soumis à l'exception suivante qui le libère de l'attente ; il fusionne sa liste avec celle collectée ;
- Un site recevant un appel d'offre ne peut plus l'envoyer à son émetteur ;
- Tout appel d'offre est valide dans un délai inférieur au temps de collecte pour le prochain cycle d'ordonnancement. Toute réponse en dehors de ce délai est ignorée.

- Conditions d'arrêt de collecte de la liste courante des demandes : temps d'attente d'une queue de ressources du site atteint ou maximum de demandes (seuil) que prend la liste courante de collecte de demandes atteint ou signal d'appel d'offre reçu.

b- Terminaison

Un $cfpm()$ ne recevant plus de demandes sur une période fixée procède comme suit :

1. Envoi de signal de terminaison aux $cfpm()$ voisins ;
2. Test si le nombre de signaux de terminaison reçus est égal au nombre de sites : si c'est le cas, alors envoi de signal de terminaison aux autres agents du site et terminaison, sinon attente des demandes et appels d'offre.
3. Si un agent local du site reçoit un signal de terminaison de son agent $cfpm()$, alors il se termine.

III.4. Aide à la décision : fonctions d'utilité proposées

a- Niveau local pour le site de soumission i

$$b_i^{local}(t) = H_i \times \frac{v_i}{v_{max}} \times \left(\frac{d_{max} - t}{d_i - p_i^k - t} \right) \times \frac{t + 1 - st_i}{t + 1 - st_{min}} \times \frac{Demande}{Offre}; d_i \neq p_i^k + t; \quad (13)$$

b- Niveau local pour un site distant j

$$b_{ij}^{local}(t) = H_i \times \frac{v_i}{v_{max}^j} \times \left(\frac{d_{max}^j - t}{d_i - p_i^j - t} \right) \times \frac{t + 1 - st_i}{t + 1 - st_{min}^j} \times \frac{Demande^j + 1}{Offre^j}; d_i \neq p_i^j + t; \quad (14)$$

c- Niveau global

- Utilité d'une demande

$$b_i^{global}(t) = H_i \times \frac{v_i}{\max_j \{v_{max}^j\}} \times \left(\frac{\max_j \{d_{max}^j\} - t}{d_i - p_i^j - t} \right) \times \frac{t + 1 - st_i}{t + 1 - \min_j \{st_{min}^j\}} \times \frac{\sum_j Demande^j}{\sum_j Offre^j}; d_i \neq t; \quad (15)$$

- Utilité d'un site de ressources

$$a_j(t) = O_j \times w_{j,t} \times \frac{c_{j,t}}{\max_k \{c_{k,t}\}} \times \frac{L_{j,t}}{\max_k \{L_{k,t}\}} \times \frac{\sum_k Demande^k}{\sum_k Offre^k}; \quad (16)$$

d- Règle de décision du $cfpm()$

Calcul de Pr_i^k : Probabilité pour qu'une demande D_i soit ordonnancée à l'instant dans un site accointant k .

$$Pr_i^k(t) = \frac{Demande^k - (rg(D_i) - 1)}{Demande^k}; \quad (17)$$

Condition de participation d'un fournisseur à la sélection :

$$\min_{i=1}^n \{b_i^{global}(t)\} > \max_{j=1}^p \{a_j(t)\}; \quad (18)$$

Élection de site : Site élu pour traiter la demande D_i = site de plus forte probabilité.

III.5. Schéma fonctionnel détaillé et organigramme général du système

La figure 4 décrit la structure et le fonctionnement du système proposé et la figure 5, l'organigramme général.

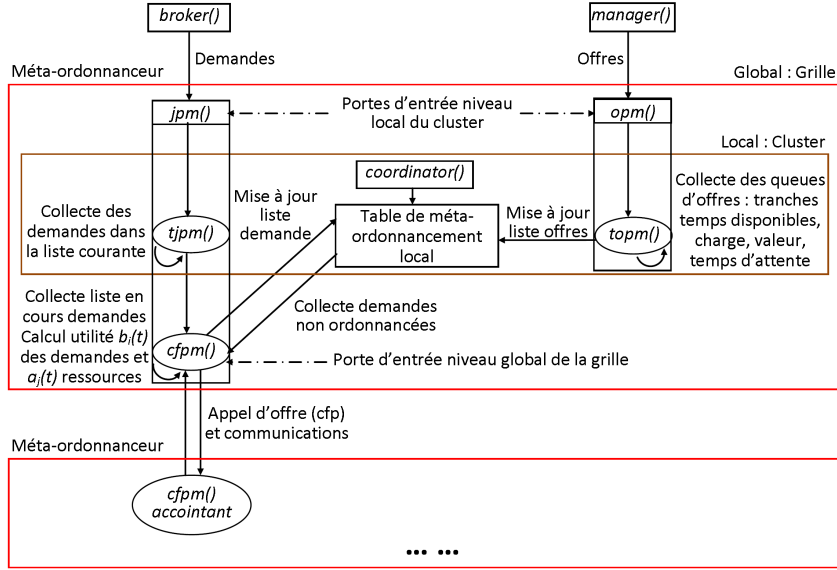


FIGURE 4. Schéma fonctionnel du système

IV. Analyse de performance et discussion

Cette analyse portera sur le temps de réponse, et permettra de comparer le système proposé avec les autres. L'environnement et le contexte sont celui distribué décrit au début de cet article, les mêmes que ceux de Solar, c'est-à-dire p clusters C_j interconnectés, recevant localement chacun n_j tâches t_{ij} ($i = 1, \dots, n_j; j = 1, \dots, p$), avec $n = n_1 + n_2 + \dots + n_p$. On suppose le réseau homogène et on ne considère pas les communications à l'intérieur d'un cluster, moins coûteuses. Cependant celles entre les clusters sont considérées. On suppose aussi $n_i = \frac{n}{k_i}$, ($k_i > 1$ un réel), une proportion de n . On va calculer le temps de réponse au meilleur tout comme au pire des cas. Les opérations sont données dans le tableau 3.

- Au meilleur des cas : Méta-ordonnancement local, sans appel d'offre (CFP)

En remplaçant dans la formule du tableau 3 par la relation $n_i = \frac{n}{k_i}$, $k_i > 1$ un réel, on obtient le temps de réponse pour une tâche soumise dans le cluster C_i :

$$T_{rep}^i = \left(\frac{n(n - k_i)}{2k_i^2} + \alpha_1 \frac{n}{k_i} \right) \times \Delta t + TO \Big|_1^{\frac{n}{k_i}}; \quad (19)$$

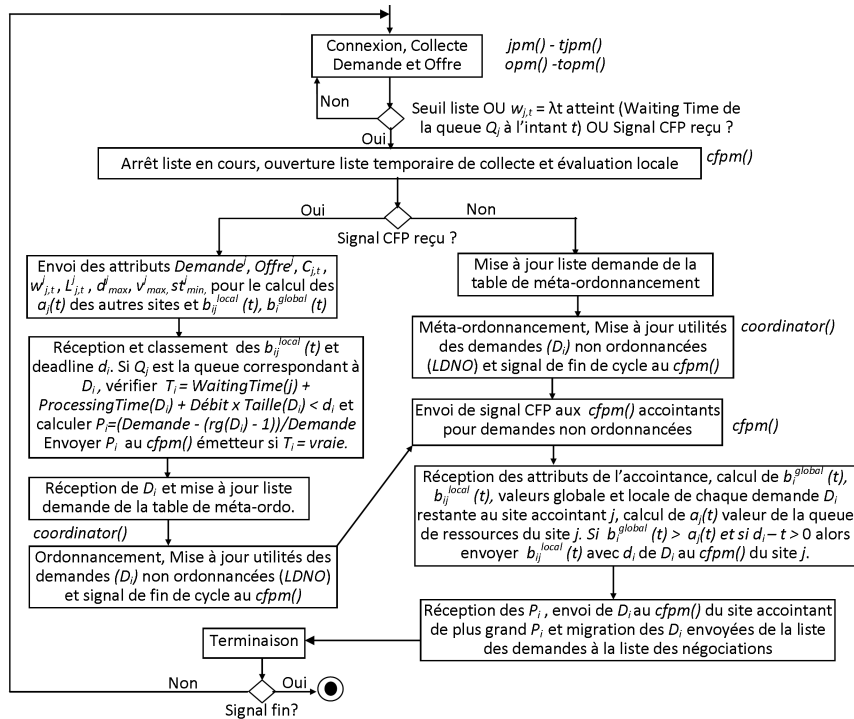


FIGURE 5. Organigramme du système de méta-ordonnancement

TABLE 3. Type d'opérations avec leurs consommations

	TYPE D'OPERATION	CONSOMMATION EN TEMPS DE L'OPERATION	FONCTION DE CONSOMMATION EN TEMPS
META-ORDONNANCEMENT LOCAL : <i>Calcul des enchères des demandes, tri et assignation de ressources</i>	Calcul en parallèle des enchères des demandes	$\alpha_1 n_i t_{cpu}, \alpha_1 > 1$	$\frac{ni(ni-1)}{2} + \alpha_1 n_i t_{cpu} + TO ^{ni}_1$ $\alpha_1 > 1$, entier, le nombre d'opérateurs dans la formule
	Tri en parallèle par comparaison de la liste des demandes	$\frac{ni(ni-1)}{2} t_{cpu}$	
	Assignation de n_i tâches aux ressources de I cluster	$TO ^{ni}_1$	
META-ORDONNANCEMENT GLOBAL : <i>Calcul des enchères des demandes, des enchères des ressources, tri, transferts et assignation de ressources</i>	Transfert de CFP (Call For Proposal)	αt_{cpu}	$\frac{Nj(Nj-1)}{2} + 5\alpha + (\alpha_1(n_1-n'_1) + \alpha_2(n_1-n'_1) + \rho\beta) + \frac{pI(pI-1)}{2} + \alpha_1 \sum_{i=1}^{p-1} (ni - n^i i) + n_p t_{cpu} + TO ^{nj}_1$ avec $N_j = ((n_1-n'_1) + (n_2-n'_2) + \dots + n_{p-1}-n'_{p-1}) + n_p$ p_i , le nombre de sites recevant les CFP. On rappelle que $\alpha t_{cpu} = t_{com}, \alpha > 1$ est la consommation d'un transfert
	Transfert de l'offre par le site distant	αt_{cpu}	
	Calcul des enchères distantes des demandes restantes	$\alpha_1(n_1-n'_1) t_{cpu}$	
	Calcul des enchères globales des demandes restantes	$\alpha_2(n_1-n'_1) t_{cpu}$	
	Calcul des enchères des offres de ressources	$\beta \rho t_{cpu}$	
	Transfert des enchères distantes des demandes restantes	αt_{cpu}	
	Tri sur les enchères des p listes de demandes dans le CFP	$\frac{Nj(Nj-1)}{2} t_{cpu}$	
	Calcul des probabilités des demandes dans le CFP	$\alpha_1((n_1-n'_1) + (n_2-n'_2) + \dots + n_{p-1}-n'_{p-1}) + n_p t_{cpu}$	
	Transfert des demandes avec probabilité aux émetteurs	αt_{cpu}	
	Tri sur les enchères d'offres et sélection des sites	$\frac{pI(pI-1)}{2} t_{cpu}, p_i < p$	
	Transfert de la décision au site élu	αt_{cpu}	
Assignation de n_j tâches aux ressources de I cluster	$TO ^{nj}_1$		

avec $k_i > 1$, et α_1 une constante.

- **Au pire des cas** : Méta-ordonnancement global, et un site j reçoit des CFP des $p - 1$ autres sites

D'après la relation du tableau ci-dessus :

$$N_j = \sum_{i=1}^p n_i - \left(\sum_{k=1}^p n'_k - n'_j \right) = n - \sum_{k=1}^p n'_k + n'_j; \quad (20)$$

n'_k étant le nombre de tâches traitées au premier cycle dans le cluster C_k et n_j le nombre de tâches du cluster C_j recevant les $p - 1$ demandes de CFP. En posant $M_j = n'_1 + n'_2 + \dots + n'_p - n'_j$, c'est-à-dire :

$$M_j = \sum_{k=1}^p n'_k - n'_j; \quad (21)$$

on obtient $N_j = n - M_j$. Avec la relation $n_i = \frac{n}{k_i}$, $k_i > 1$ un réel, T_{rep} devient :

$$T_{rep} = T_{rep}^i + \left[\frac{(n - M_j)(n - M_j - 1)}{2} + 5a + (\alpha_1 + \alpha_2)expr + \beta p_r + \frac{p_r(p_r - 1)}{2} + \alpha_1(n - M_j) \right] \Delta t + TO_1^{\frac{n}{k_{etu}}} \quad (22)$$

avec $expr = \max_{m=1, m \neq j}^p \left(\frac{n}{k_m} - \left(\frac{n}{k_m} \right)' \right)$, p_r le nombre de site recevant les CFP et T_{rep}^i le temps de réponse au meilleur des cas dans le site de soumission C_i . Ici, $p_r = 1$, d'où pour une tâche soumise dans le cluster C_i :

$$T_{rep} = \left[\frac{n(n - k_i)}{2k_i^2} + \alpha_1 \frac{n}{k_i} + \frac{(n - M_j)(n - M_j - 1)}{2} + 5a + (\alpha_1 + \alpha_2)expr + \beta + \alpha_1(n - M_j) \right] \Delta t + TO_1^{\frac{n}{k_i}} + TO_1^{\frac{n}{k_{etu}}} \quad (23)$$

En posant $n - M_j = \frac{n}{k_j}$, $k_j > 1$ un réel, la relation 23 devient :

$$T_{rep} = \left[\frac{n(n - k_i)}{2k_i^2} + \alpha_1 \left(\frac{n}{k_i} + \frac{n}{k_j} \right) + \frac{n(n - k_j)}{2k_j^2} + 5a + (\alpha_1 + \alpha_2)expr + \beta \right] \Delta t + TO_1^{\frac{n}{k_i}} + TO_1^{\frac{n}{k_{etu}}} \quad (24)$$

Comparaison des temps de réponse

Il est vrai que le modèle de Saurabh optimise plus de critères que celui de Solar et prend en prenant en compte l'utilité des participants. Celui de Solar réalise un temps de réponse théorique plus petit, au meilleur tout comme au pire des cas. Tirant déjà profit de l'approche multi-critères de Saurabh, il nous reste alors à montrer que notre temps de réponse est plus petit que celui de Solar, c'est-à-dire $T_{Solar} > qT_{DDAGM}$, avec $q > 1$, ou encore $T_{Solar}/T_{DDAGM} > q$.

- Au meilleur des cas, en observant l'équation 7 il est évident en comparant les termes des deux formules, qu'on a $T_{DDAGM} < T_{Solar}$.

- Au pire des cas, observons ce que donne le rapport T_{Solar}/T_{DDAGM} . Pour minorer cette expression, le résultat serait davantage intéressant si l'on peut minorer T_{Solar} de l'équation 9, et majorer T_{DDAGM} . En exploitant la relation :

$$pn_{k_1} \leq \sum_{k=1}^p n_k \leq pn_{k_2}; \quad (25)$$

avec $n_{k_1} = \min_{l=1}^p \{n_l\}$ et $n_{k_2} = \max_{l=1}^p \{n_l\}$, d'une part, dans T_{Solar} , on a :

$$\sum_{i=1}^p \frac{n(n - k_i)}{2k_i^2} \geq p \frac{n(n - k_j)}{2k_j^2}; \quad (26)$$

avec $\frac{n(n-k_j)}{2k_j^2} = \min_{l=1}^p \left\{ \frac{n(n-k_l)}{2k_l^2} \right\}$, et

$$5c \sum_{i=1}^p \frac{n}{k_i} \geq 5cp \frac{n}{k_j}; \quad (27)$$

avec $\frac{n}{k_j} = \min_{l=1}^p \left\{ \frac{n}{k_l} \right\}$, d'où

$$T_{Solar} \geq p \left[\frac{n(n-k_j)}{2k_j^2} + 5c \frac{n}{k_j} + 2\alpha_1 + \beta_1 + (2a+1)(p-1) \right] \times \Delta t + TO|_1^{\frac{n}{k_i}} + TO|_1^{\frac{n}{k_{etu}}}; \quad (28)$$

D'autre part dans T_{DDAGM} , considérant $n - M_l = \frac{n}{k_l}$, $k_l > 1$, on a :

$$\frac{n(n-k_i)}{2k_i^2} + \frac{n(n-k_l)}{2k_l^2} \leq 2 \frac{n(n-k_r)}{2k_r^2} = \frac{n(n-k_r)}{k_r^2} \quad (29)$$

avec $\frac{n(n-k_r)}{2k_r^2} = \max \left\{ \frac{n(n-k_i)}{2k_i^2}, \frac{n(n-k_l)}{2k_l^2} \right\}$, ensuite

$$\alpha_1 \left(\frac{n}{k_i} + \frac{n}{k_l} \right) \leq 2\alpha_1 \frac{n}{k_r} \leq 2 \times 5c \frac{n}{k_r} \quad (30)$$

à une constante près, avec $\frac{n}{k_r} = \max \left\{ \frac{n}{k_i}, \frac{n}{k_l} \right\}$, et enfin

$$5a + (\alpha_1 + \alpha_2) \max_{m=1, m \neq j}^p \left\{ \frac{n}{k_m} - \left(\frac{n}{k_m} \right)' \right\} + \beta \leq 4a(p-1) + 2(p-1) + 4\alpha_1 + 2\beta_1 \quad \text{quand } p \text{ est grand} \quad (31)$$

$$\leq 2(2\alpha_1 + \beta_1 + (2a+1)(p-1)) \quad (32)$$

doù

$$T_{DDAGM} \leq 2 \left[\frac{n(n-k_r)}{2k_r^2} + 5c \frac{n}{k_r} + 2\alpha_1 + \beta_1 + (2a+1)(p-1) \right] \times \Delta t + TO|_1^{\frac{n}{k_i}} + TO|_1^{\frac{n}{k_{etu}}}; \quad (33)$$

T_{Solar} et T_{DDAGM} ayant des temps d'assignation identiques, c'est-à-dire $TO|_1^{\frac{n}{k_i}} + TO|_1^{\frac{n}{k_{etu}}}$, on peut les négliger dans le rapport T_{Solar}/T_{DDAGM} . En choisissant judicieusement un k_z coïncidant à la borne supérieure de T_{DDAGM} et à la borne inférieure de T_{Solar} , on obtient pour ce k_z :

$$T_{Solar}/T_{DDAGM} \geq \frac{p \left[\frac{n(n-k_z)}{2k_z^2} + 5c \frac{n}{k_z} + 2\alpha_1 + \beta_1 + (2a+1)(p-1) \right]}{2 \left[\frac{n(n-k_z)}{2k_z^2} + 5c \frac{n}{k_z} + 2\alpha_1 + \beta_1 + (2a+1)(p-1) \right]}; \quad (34)$$

d'où

$$T_{Solar}/T_{DDAGM} \geq \frac{p}{2}, \quad \text{avec } \frac{p}{2} > 1, \quad p \text{ le nombre de sites.} \quad (35)$$

IV.1. Discussion

La comparaison des temps de réponse précédente montre que dans tous les cas, le temps de réponse théorique du modèle proposé est meilleur. En plus, la prise en compte multi-critères comme avec le modèle de Saurabh est encore un avantage par rapport au modèle de Solar. L'intérêt de l'utilisation des protocoles double enchères et appel d'offre est l'amélioration de l'utilité des fournisseurs de ressources et des clients, ce que ne fait pas le modèle de Solar. En particulier, le protocole d'appel d'offre permet d'équilibrer d'une façon souple et flexible la charge sur le système, en faisant une sélection basée sur plusieurs critères, du meilleur site; contrairement au modèle de Solar qui fait une négociation point à point pour choisir le site le plus disposé, avec une évaluation principalement basée sur le taux d'occupation des ressources. Notre système de méta-ordonnancement assure qu'une tâche est ordonnancée au plus tôt localement, et qu'elle n'est rejetée que si personne ne peut la satisfaire dans la grille. Les techniques utilisées permettant de réduire les communications, les temps de calcul, et par conséquent le temps de

réponse et la saturation du réseau. La solution est tolérante aux fautes, car même si un noeud tombe en panne, les autres continueront à fonctionner.

V. Conclusion et perspectives

Dans ce travail, après l'étude de quelques modèles de méta-ordonnancement existants, nous avons décrit un nouveau système multi-agents de méta-ordonnancement distribué basé sur la vente aux doubles enchères et l'appel d'offre. Le méta-ordonnanceur utilise des fonctions d'utilité basé sur les métriques à optimiser, pour l'aide à la prise de décision et il coopère avec des services voisins de méta-ordonnancement. Un tel système est tolérant, et nous avons montré qu'il intègre les métriques pris en compte par des modèles considérés à la base. Il diminue la saturation du réseau et présente un meilleur temps de réponse d'un facteur de $p/2$ par rapport à ceux-ci, p étant le nombre de sites.

En terme de perspectives, d'une part des simulations sur Simgrid à partir des traces de charge de travail réelles des grilles, permettront de renforcer ces résultats. D'autre part, la prise en compte des types de ressources variées et l'implantation du système au grilles et clouds existants seraient intéressantes.

VI. Remerciements

Nous adressons nos remerciements aux membres du Laboratoire de Mathématiques Expérimentales (LAMEX) de l'Université de Ngaoundéré, pour leur contribution dans ce travail, ainsi qu'au comité d'organisation du 4-ième séminaire annuel Cryptographie, Algèbre et Géométrie (CRAG-4), qui a permis que cette communication soit effective.

Références

- [1] Stankovic (John A.), Ramamritham (Krithivasan), and Cheng (Shengchang). Evaluation of a flexible task scheduling algorithm for distributed hard real time systems. *Proc. IEEE trans. on Computers*, pages 1130–143, December 1985.
- [2] Martin Adolph, Ewan Sutherland, and Arthur Levin. Distributed computing : Utilities, grids & clouds. *International Telecommunication Union - Technology Watch Report*, 9, 2009.
- [3] Jose Aguilar and Rodolfo Sumoza. A multiagent grid metascheduler. *WSEAS TRANSACTIONS on COMPUTERS*, 8(8) :1388–1397, 2009.
- [4] Zeng B., Wei J., and Liu H. Dynamic grid resource scheduling model using learning agent. *IEEE Int. Conf on Networking, Architecture, and Storage*, 2009.
- [5] Mark Baker, Rajkumar Buyya, and Domenico Laforenza. Grids and grid technologies for wide-area distributed computing. *SOFTWARE :PRACTICE AND EXPERIENCE - John Wiley & Sons, Ltd.*, DOI : 10.1002/spe.488, 2002.
- [6] Tracy D. Braun, Howard Jay Siegel, Noah Beck, Lasislau L. Bólóni, Muthucumara Maheswaran, Albert I. Reuther, James P. Robertson, Mitchell D. Theys, Bin Yao, Debra Hensgen, and Richard F. Freund. A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. *Journal of Parallel and Distributed Computing*, 61(6) :810–837, 2001.
- [7] Franck BUTELLE. *Contribution à l'algorithmique distribuée : arbres et ordonnancement*. Hdr, L'UNIVERSITÉ PARIS 13, Paris - France, Mai 2011.
- [8] Rajkumar Buyya. Qos-based scheduling of e-research application workflows on global grids. Technical report, The University of Melbourne - Dept. of Computer Science and Software Engineering - Grid Computing and Distributed Systems (GRIDS) Laboratory, Australia, 2007.

- [9] Rajkumar Buyya, David Abramson, Jonathan Giddy, and Heinz Stockinger. Economic models for resource management and scheduling in grid computing. Technical report, Monash University, Melbourne - Australia, 2002.
- [10] Chien C., Chang P., and Soo V. Market-oriented multiple resource scheduling in grid computing environments. *19th International Conference on Advanced Information Networking and Applications*, 1 :867–872, 2005.
- [11] Agustin C. Caminero, Omer Rana, Blanca Caminero, and Carmen Carrión. Autonomic network-aware metascheduling for grids : A comprehensive evaluation. Technical report, The National University of Distance Education and Cardiff School of Computer Science and The University of Castilla La Mancha, Spain - UK, 2012.
- [12] Eddy Caron, Vincent Garonne, and Andreï Tsaregorodtsev. A study of meta-scheduling architectures for high throughput computing. Technical Report 5668, CNRS-INRIA-ENS LYON-UCBL, 2005.
- [13] Fangpeng Dong and Selim G. Akl. Scheduling algorithms for grid computing : State of the art and open problems. Technical report, School of Computing, January 2006.
- [14] Shin (K. G.) and Chang (Y. C.). Load sharing in distributed real time systems with state-change broadcasts. *IEEE Trans, on Software Engineering*, 38(8) :1124–1142, 1989.
- [15] Saurabh Kumar Garg. *Meta Scheduling for Market-Oriented Grid and Utility Computing*. Thesis, The University of Melbourne, Australia, June 2010.
- [16] Saurabh Kumar Garg, Sri Kumar Venugopal, James Broberg, and Rajkumar Buyya. Double auction-inspired meta-scheduling of parallel applications on global grids. *Journal of Parallel and Distributed Computing*, 73(4) :450–464, April 2013.
- [17] Chang (Hung-Yang) and Livny (Miron). Distributed scheduling under deadline constraints : a comparison of sender -initiated and receiver initiated approaches. *Proc. IEEE Int. Conf. on Distr. Comp. Syst.*, pages 175–180, 1986.
- [18] Bhattacharjee (Anup K.), Ravindranath (K.), Pal (A.), and Mall (R.). Ddsched : a distributed dynamic real-time scheduling algorithm. *Progress in computer research*, pages 170–184, 2001.
- [19] Ramamritham (Krithi), Stankovic (John A.), and Zhao (Wei). Distributed schedulings of tasks with deadlines and resource requirements. *Proc. IEEE trans. on Computers*, pages 1110–1123, August 1989.
- [20] Bui (Marc), Butelle (Franck), and Lavault (Christian). A distributed algorithm for the minimum diameter spanning tree problem. *Journal of Parallel and Distributed Computing*, 64(5) :571–577, 2004.
- [21] G. Molto, V. Hernandez, and J.M. Alonso. A service-oriented wsrp-based architecture for meta-scheduling on computational grids. *Elsevier*, pages 1–12, May 2007.
- [22] RAJKUMAR BUYYA, DAVID ABRAMSON, and SRIKUMAR VENUGOPAL. The grid economy. *PROCEEDINGS OF THE IEEE*, 93(3), MARCH 2005.
- [23] WOUANSI TOWO Jérémie Serge. *Proposition et modélisation multi-agents d'un méta-ordonnancement distribué de grille*. Mémoire de master, Université de Ngaoundéré, Ngaoundéré - Cameroun, Août 2012.
- [24] Mauricio Solar, Jorge Rojas, Marcelo Mendoza, and Raúl Monge. A multiagent-based approach to the grid-scheduling problem. *CLEI ELECTRONIC JOURNAL*, 15(2), August 2012.

J.S. Wouansi

e-mail: jeremie.ws@yahoo.fr

Université de Ngaoundéré, Faculté des Sciences, Département de Mathématiques et Informatique.

V.C. Kamla and D. Tieudjo

e-mail: vc_kamla@yahoo.fr

e-mail: tieudjo@yahoo.com

Université de Ngaoundéré, Ecole Nationale Supérieure des Sciences Agro-Indusrtielles (ENSAI), Département de Mathématiques et Informatique, Laboratoire de Mathématiques Expérimentales (LAMEX).