Multiple Job Allocation in Multicluster System[⋆]

Héctor Blanco, Damiá Castellá, Josep Lluís Lérida and Fernando Guirado

Dept. of Computer Science, Universitat de Lleida, Spain
e-mail: (hectorblanco, dcastella, jlerida, f.guirado)@diei.udl.cat

Abstract. In multi-cluster systems, job scheduling is delegated to a meta-scheduler that has a global vision of the resources. With this knowledge it determines the best execution cluster to allocate the parallel jobs to. In some cases, resources from more than one cluster could be needed to execute these. This situation, called co-allocation, must ensure that job execution does not affect others present in the shared clusters. Obtaining a lower execution time is a big challenge, even more so when there is an execution queue with multiple jobs in the system.
In this work, the authors present a linear programming model that allows simultaneous scheduling of all the parallel jobs in the system queue, instead of allocating them individually. This model minimizes the average execution time for all of these, and is capable of taking advantage of co-allocation avoiding the saturation of the network links.

1 Introduction

Nowadays the use of clusters of computers is becoming common in all kinds of research laboratories or institutions. Computation problems that would require the use of more computational resources than just one of these can offer can be resolved by the use of multiple clusters in a collaborative manner. These environments are known as multi-clusters and are distinguished from grids by using dedicated interconnection network among the clusters with a known topology and predictable performance characteristics [1][2].

The parallel jobs executed in the multi-cluster system are composed of collaborative tasks. Each of these is individually assigned to the computation nodes of the clusters. There may be no cluster with enough resources to execute all the tasks of a job. In this case, the tasks could be assigned to computation nodes distributed among the clusters. This kind of assignment is known as co-allocation. The main caveat of this approach is that mapping jobs across the cluster boundaries can result in rather poor overall performance when co-allocated jobs contend for inter-cluster network bandwidth [3].

In the literature, some authors have dealt with the co-allocation process, some by developing heuristics that allow the tasks of a job to be co-allocated depending on their computational and communication requirements. This is the case of [4][5], where load-balancing techniques are used to minimize the execution time of the parallel job. In other cases, the solution is based on analytical

[⋆] This work was supported by the MEyC-Spain under contract TIN2008-05913 models, [6][7][3][8]. In [6], there is a design and performance study of four different scheduling policies, based on job queues, local to each cluster or global for all of them. The work presented in [7] applies a linear programming based approach for modeling and solving the allocation of jobs. Another point of view is presented in [3], which characterizes the bandwidth requirements of the parallel jobs that are co-allocated in order to obtain the lowest execution time. This work was extended in [8], where the computation necessities were also taken into account to reduce the execution time of the parallel jobs, while preventing the saturation of the interconnection links. Otherwise, those previous works follow a FCFS scheme to allocate all the jobs in the system queue. This means that the selected technique considers all the available resources to allocate the current job, but as it does not take into account the other jobs in the queue, the current allocation could affect the next assignments negatively.

The main contribution of the present work is to extend the model presented in [8] by adding the capacity for scheduling multiple jobs simultaneously from the system queue, reducing their global execution time and also preventing the saturation of the interconnection net. The model is compared with others in the literature, and the experimental results show that it obtains the lowest global execution times and which are also closer to the times that would be obtained if the jobs were scheduled alone.

The rest of the paper is organized as follows. In Section 2, we present the execution time model of a parallel application based on the slowdown produced by the heterogeneity and non-dedicated nature of multi-cluster resources. In Section 3, there is a detailed description of our proposal for a multiple job allocation model expressed as a Mixed-Integer Programming problem. Section 4 presents our experimentation and the results obtained from comparison with other models in the literature. Finally, the conclusions and future work are presented in Section 5.

2 Problem statement

In [8] we presented a new execution time model for parallel jobs. The goodness of this model was that it defined the execution time by considering both processing resource availability and communication resource utilization. Furthermore, this model was applied in a Mixed-Integer Programming model (MIP) in order to find the allocation of the current job that minimizes its execution time, while avoiding the negative effects of sharing the communication links and processing resources. The drawback of our previous proposal was that jobs in the system queue were selected individually and scheduled in a FCFS manner, without considering the effect on the execution time of the rest of the jobs in the system queue.

We assume that the jobs are not supposed to be malleable, the processing and communicating requirements of every job task are very similar, and the job tasks follow an all-to-all communication pattern. Considering this, we define the job execution time, T^e, as follows:

$$T^e = T^p \cdot SP + T^c \cdot SC \tag{1}$$

where T^p and T^c are the processing and communicating time measured in a dedicated environment. In a real situation, due to the heterogeneity and the non-dedicated property of the resources, T^p and T^c may be lengthened by processing (SP) and communicating (SC) slowdown respectively.

The main aim of this study is to find the best simultaneous allocation for a set of jobs in the system queue that fits on the environment. This new approach overcomes the individual $FCFS$ scheduling where each job is scheduled alone, and thus, the obtained task allocation could negatively affect the performance of the next scheduled jobs. Thus, in this work, the best allocation for a set of jobs is that which minimizes the execution time of the whole.

In section 2.1 we define the execution time model based on the total job slowdown, and therefore, in section 3 we define the allocation problem as a MIP (Mixed-Integer Programming) model where the best solution is the one that minimizes the global execution time of the set of jobs.

2.1 Execution Time model

We define a multi-cluster as a collection of arbitrary sized clusters with heterogeneous resources. Each cluster has its own internal switch. Clusters are connected to each other by single dedicated links by means of a central switch.

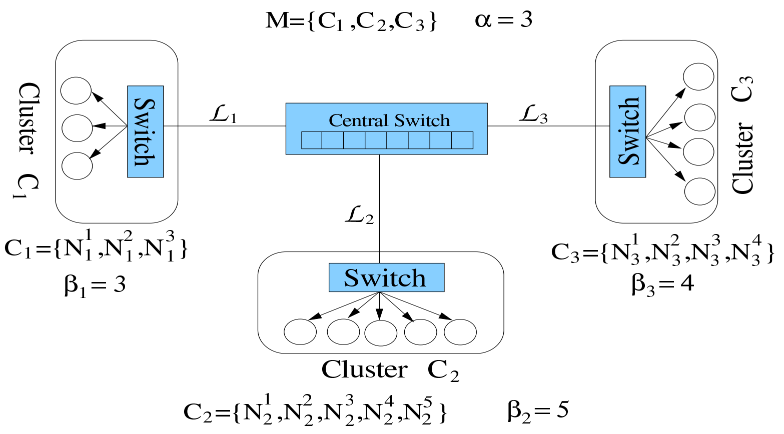

Fig. 1. Diagram of a multi-cluster topology

Formally, a multi-cluster $M = \{C_1..C_\alpha\}$ can be defined as a system comprising heterogeneous clusters interconnected by means of dedicated links (see Figure 1). Each Cluster $C_i (1 \leq i \leq \alpha)$ is also made up of β_i nodes, this is $C_i = \{N_1..N_i^{\beta_i}\}$. $\mathcal{L}$ is the set of inter-cluster links ($\mathcal{L} = \{\mathcal{L}_1..\mathcal{L}_\alpha\}$), and

$L = \{L_i\} = \{\mathrm{L}_i^k, 1 \leq i \leq \alpha \text{ and } 1 \leq k \leq \beta_i\}$, is the set of intra-cluster links, where L_i^k denote the intra-cluster link between node k and the switch of Cluster C_i.

The execution time (T^e) of a parallel job in a heterogeneous and non-dedicated environment can be defined as its execution time in a dedicated environment ($T^{e'}$) delayed by a slowdown factor (SD) produced by the heterogeneity and non-dedicated nature of allocated resources. Based on this, the execution time (T^e) can be expressed by means of equation 2.

$$T^e = T^{e'} \cdot SD \tag{2}$$

where the SD value can be calculated based on the Processing Slowdown (SP) and Communication Slowdown (SC), using equation 3.

$$SD = \sigma \cdot SP + (1 - \sigma) \cdot SC \tag{3}$$

σ being the weighting factor that measures the relevance of the processing time with respect to the communication time of the corresponding job.

As we assume that each job task is generally similar in size and they are executing separately, the job execution time is bounded by the slowest allocated resources. Thus, the job processing slowdown (SP) is obtained from the allocated resource with the maximum processing slowdown, as can be seen in equation 4.

$$SP^j = max\{SP_r | r \in \mathcal{P}^j\} \tag{4}$$

where $\mathcal{P}^j$ is the set of processing nodes allocated to the job j. The SP_r is calculated as the inverse of its Effective Power ($\mathit{\Gamma}_r$) that relates the processing power and the availability of such node.

About the communication slowdown, the co-allocation of jobs consumes a certain amount of bandwidth across inter-cluster network links, as expressed by equation 5.

$$BW_k^j = \left(t_k^j \cdot PPBW^j\right) \cdot \left(\frac{n_T^j - t_k^j}{n_T^j - 1}\right), \quad \forall\, k \in 1..\alpha \tag{5}$$

where n_T^j is the total number of tasks of the job j and t_k^j corresponds to the amount of tasks allocated to cluster C_k. The first term in the equation is the total bandwidth required by all the nodes associated with job j on cluster C_k. The second term represents the communication percentage of job j with other cluster nodes (not in C_k) that will use the inter-cluster link k.

The degree of saturation of inter-cluster links relates the available bandwidth of each link with the bandwidth requirements of the allocated parallel applications, which is calculated by equation 6.

$$BW_k^{sat} = \frac{ABW_k}{\sum_{j,k} BW_k^j} \quad \forall\, k \in 1..\alpha, j \in \mathcal{Q} \tag{6}$$

When the required bandwidth is lower than the available the link is not saturated and the communications could be without contentions. Otherwise, the network link is saturated and then reduces drastically the performance of the jobs sharing the link.

Thus, the job communicating slowdown (SC) is obtained from the slowest, most saturated, communication link used by the job, as the inverse of the saturation bandwidth, 7.

$$SC^j = max\{(BW_k^{sat})^{-1} | k \in 1..\alpha\} \tag{7}$$

3 MIP Model for multiple jobs allocation

Integer Programming (IP) is a technique for maximizing or minimizing the value of an objective function subject to some constraints. In our case, the objective function and its constraints correspond to linear expressions that refer to simultaneous allocation of multiple jobs in a multi-cluster system. This kind of problem can be expressed by a Mixed-Integer Programming model, where the solution is presented as binary values, 1 or 0. In our case those values determine the allocation, or not, of the processing node to the parallel job [7][9]. The objective function corresponds to the lowest global execution time for a set of jobs, giving us the best possible allocation, which may or may not be co-allocated between different clusters.

3.1 MIP model definition

An integer programming model is made up of a set of input parameters, decision variables, a set of constraints which the solution must satisfy and an objective function. The main goal of the model is to find the decision variable values that meet the constraints and also maximize or minimize the objective function.

We named our proposed model $MBPC$, for *Multiple Best Processing and Communication*, as described in Figure 2. The model provides the best possible task allocation for a set of jobs, taking into account the job requirements and the heterogeneous characteristics and availability of the multi-cluster resources.

In order to find the best allocation, information about the jobs and the multi-cluster status are required (lines 1-8). The information about each job j corresponds to the number of tasks (τ^j), the required per-task bandwidth ($PNBW^j$), and the weighting factor (σ^j), which measures the relevance of the processing and communication time in the total job execution time. For multi-cluster resources, its status is specified by the effective CPU power of each node r (Γ_r).

The set of output variables (lines 9-11) consists of an array of binary decision variables $X_{(j,r)}$, with values (1) or (0) when a task of the job j is allocated in node r, or not, respectively. The SP^j and SC^j variables show the processing and communication slowdown for each job j in the provided solution, and thus, define the job Slowdown SD^j by means of equation 3.

Input arguments:

1. $\mathcal{Q}$: Queue of jobs to be matched.
2. τ^j: number of tasks making up job $j \in Q$.
3. $PNBW^j$: per-task bandwidth requirement for each job $j \in Q$.
4. σ^j: weighting factor that relates the processing and communication time.
5. $\mathcal{L}$: set of inter-cluster links.
6. μ: set of multi-cluster resources.
7. Γ_r: Effective Power weight of node $r \in \mathcal{P}$
8. ABW_l: maximum communication capacity for each inter-cluster l link $\in \mathcal{L}$

Output parameters:

9. $X_{(j,r)}$, $j \in \mathcal{Q}$ and $r \in \mathcal{P}$: $X_{(j,r)}$=1 when j is matched to resource r, and 0 otherwise.
10. $\mathcal{P}_j$: Set of allocated resources, when $X_{(j,r)} = 1$, $r \in \mu$
11. SP^j: processing slowdown. $SP^j = max\{SP_{(j,r)} \mid j \in \mathcal{Q}$ and $r \in \mathcal{P}_j\}$.
12. SC^j: communication slowdown. $SC^j = max\{SC_{(j,l)} \mid j \in \mathcal{Q}$ and $l \in \mathcal{L}\}$.

Objective Function:

13. $min\{\sum_{1,j} T^j_{e'} \cdot SD^j\}$

Constraints:

14. Gang matching.
15. Non inter-cluster link saturation.

Fig. 2. MIP model definition for multiple job allocation

3.2 Objective function

When there are many possible solutions, the objective function defines the quality of the solution. Our main aim was the allocation of multiple jobs in heterogeneous and non-dedicated resources over a multi-cluster system, obtaining the lowest execution time for the job set.

In order to deal with multiple jobs obtaining a fair allocation for all of them, we attempted to minimizing the global execution time of the entire job set $\mathcal{Q}$. This is done by the summing of individual obtained execution time for all the jobs involved, as shown in 8.

$$min\{\sum_{1,j} T^{e'}(j) \cdot SD^j\} \tag{8}$$

where $T^{e'}(j)$ is the job execution time measured in a dedicated environment, and SD is the job slowdown obtained by equation 3.

3.3 Constraints

The constraints (lines 14-15) define a feasible matching scheme. In this model, we define two main constraints that must be satisfied, the *gang matching* and the *non-saturation* of the inter-cluster links.

The gang matching constraint ensures that all the tasks in each job are assigned, according to equation 9.

$$\sum_{j \in \mathcal{Q}, r \in \mathcal{P}} X_{(j,r)} = \tau_j \tag{9}$$

where τ^j is the number of tasks for the job j. This ensures that the sum of the resources allocated to the job j corresponds to its number of tasks.

The non-saturation constraint ensures that the bandwidth consumed on the inter-cluster links, once the set of jobs is assigned, does not exceed the total capacity of the links, thus preventing saturation and delay to the jobs that share these. This constraint is formalized in equation 10.

$$SC_j \leq 1, \forall j \in \mathcal{Q} \tag{10}$$

where SC^j is the communication slowdown for all allocated jobs j.

4 Experimentation

The aim of the experimentation was to analyze the execution time of a set of jobs allocated using our *Multiple Best Processing and Communication* (*MBPC*) model. To do so, we compared our model with two models in the literature, [3][7]. The first of these, known as *CBS* for *Chunk Big Small* tries to co-allocate a "large chunk" (75% of the job tasks) into a single cluster in an attempt to avoid inter-cluster link saturation. Also, it uses a First Come First Serve (*FCFS*) policy to allocate the next jobs in the system queue. The second model, named *JPN* for *Job Preferences on Network*, allocates jobs using only the minimization of the network occupation as the ranking criterion, which is similar to minimizing our *SC*. This model takes into account all the jobs in the system queue simultaneously, as we do, to schedule them.

The experimental study was divided into two different cases. The first case study compared the models on a controlled environment, allowing us to determine the effectiveness of the different models. In the second case, we used a complete heterogeneous environment in order to extend the comparison between the models.

4.1 First Case Study - Controlled Environment

The experimental environment was a multi-cluster made up of 4 clusters, $\{C1, C2, C3, C4\}$, each with 16 nodes. In this case the heterogeneity was implemented by assigning different effective power to each individual cluster, being $\{0.5, 0.6, 0.7, 0.8\}$ respectively, from lesser to higher capability. In all cases, the bandwidth availability was assumed to be the same for all the inter-cluster links.

A system queue composed of three jobs, $\{J1, J2, J3\}$, with 18 tasks and different communication requirements was defined. The number of tasks was selected to ensure the co-allocation. The σ factors for these were $\{0.05, 0.2, 0.3\}$

respectively, the first being the most communicative job and less computational, and the third, the least communicative and higher computational. The $PNBW_j$ requirement for all the jobs was set at 0.075 Gbps, which implies a large volume of data transfer.

The experiment consisted of varying the available bandwidth of the inter-cluster links (ABW) with values $\{0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$ Gbps (from lesser to higher), in order to analyze its effect on the job execution time.

The quality of the solutions was measured by comparing the SP_j (processing slowdown) and the SC_j (communication slowdown) obtained for each job j with those obtained in a standalone allocation. The standalone allocation gives us the job slowdowns, SP_j^{st} and SC_j^{st}, when the model evaluates the job individually having all the multi-cluster resources available to obtain the allocation.

Figure 3 shows the results for the processing slowdown values (SP). The X-axis represents the available inter-cluster bandwidth, grouped by the model. In each experiment, the difference to the standalone *Processing Slowdown* ($SP_j - SP_j^{st}$) is calculated for each job j. The Y-axis represents the highest, average and lowest values of these differences. For all the jobs, the distance between the highest and lowest value represents the dispersion range of the results for the set of applications. A short distance means that all jobs receive fair allocations with lower slowdown differences, while a higher distance means that some jobs are more affected than others. Finally the average value, the points joined by a segment, estimates the global slowdown. The lower this is, the better the allocation will be.

As can be seen, $MBPC$ produces the lowest dispersion allocations and also the lowest global processing slowdown, while CBS and JPN have higher dispersions because they do not take the processing characteristics of the cluster nodes into account. CBS also uses an FCFS policy to schedule the jobs, which has negative effects on the next jobs to be allocated when fewer cluster nodes are available and the job must be distributed over the multi-cluster. The fairness of the $MBPC$ allocations produces dispersion values no greater than 5%, while CBS and JPN rise to 15%.

Figure 4 shows the results obtained for the communication slowdown values (SC). All models have low dispersion values. This is because they all try to minimize the network bandwidth occupation. The $MBPC$ and JPN models obtain the lowest communication slowdown, but $MBPC$, in contrast to JPN, also takes care to diminish the SP. The CBS model produces the highest communication slowdown values because it only attempts to reduce the bandwidth usage by grouping at least 75% of the tasks of the job in the same cluster, but does not attempt to explicitly minimize the communication slowdown.

Finally, it must be taking into account that a SC value lower than 1 does not means that the communication could be faster, it just only shows that the intercluster channels are not saturated, thus the communication process could be done at the full speed, without interferences.

Figure 5 compares the job execution time obtained for the three models when the available cluster bandwidth was set to 0.4 Gbps, is the situation with more

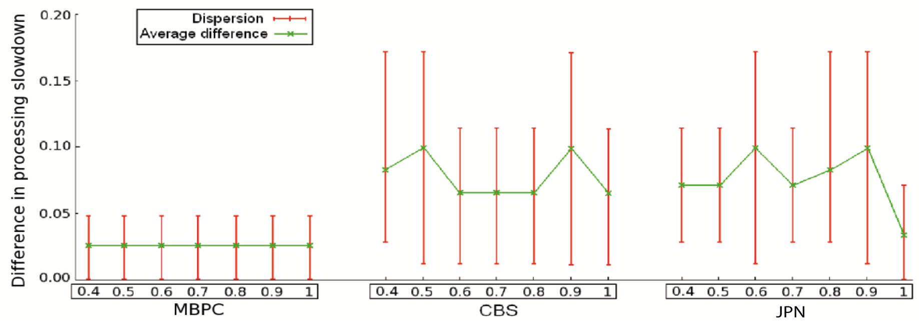

Fig. 3. Comparison of the differences in processing slowdown, varying the ABW value

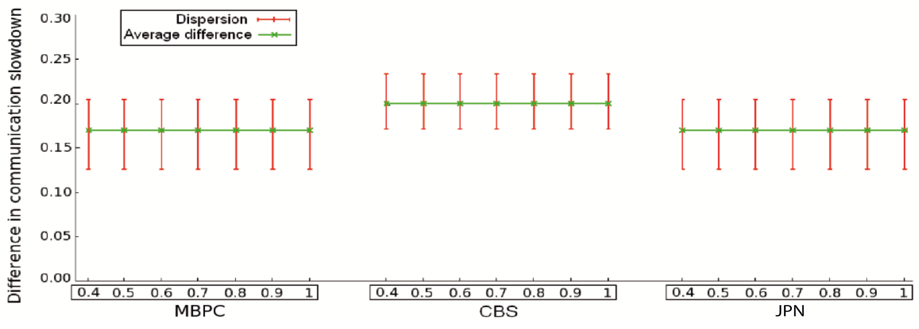

Fig. 4. Comparison of the differences in communication slowdown, varying the ABW value

network restrictions. The obtained execution time is represented by the communication and processing time, in blue and red respectively. As can be observed, the communication time is the same for all three models. This is because they all avoided the saturation of the intercluster channels, and the data was transferred as fast as possible. However, the differences come from the processing time due to the ability of the models to choose better or worst nodes. As can be observed, the best solution comes from the $MBPC$ model, where globally the set of jobs finish earlier. CBS and JPN have obtained worst solutions. This is because both CBS and JPN only evaluate the network occupation, and moreover, CBS allocates the jobs in a $FCFS$ manner. This test showed that $MBPC$ produced an allocation with a global execution time 5% lower than the other models.

In order to understand the differences between the execution times, the obtained allocation is represented in Figure 6. The figure shows which cluster has been selected for each job for each model. In the y-axis, the number of nodes chosen is represented.

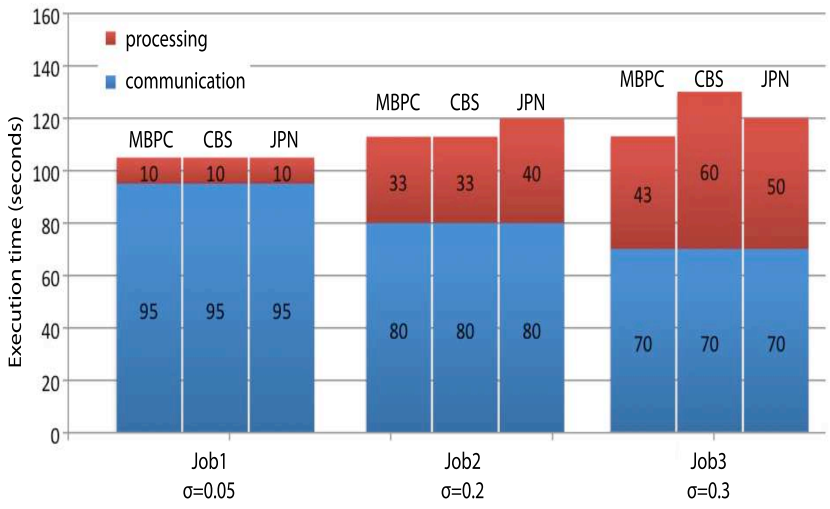

Fig. 5. Time comparison for the three models, on the test with ABW=0.4

As can be observed, $MBPC$ attempts to allocate the set of jobs in a balanced way, satisfying the processing and communication necessities of this. The model fully assigns the cluster C1, the powerless, to job J1, which has the lowest computational requirements. This allocation allows J1 to obtain a good execution time using the minimum computational capabilities. Job J3, which has the highest computational requirements, is mostly assigned to cluster C4, and also nodes from cluster C3. As the execution time is upper bound by the lowest task, it does not need to fulfill cluster C4 because the obtained execution time would be the same. Finally, job J2 is assigned between cluster C2 and C3, obtaining a balanced solution between the processing and communication time.

For the CBS and JPN models, the allocation obtained are based on diminishing the network occupation, by grouping tasks as is the case of CBS or the objective function, for JPN. They did not take into account the processing capabilities of the selected nodes from the clusters. Then, as the communication links were not saturated, the time expended on the communication was equal for both, but as the nodes were badly selected the processing time were higher.

The results for this experiment show that the $MBPC$ model has the ability to obtain the fairness allocation for the job set, as all of them obtain the computational nodes that reduce their processing time, and are distributed over the multi-cluster in order to non saturate the network.

4.2 Second Case Study - Heterogeneous Environment

A second experimentation with a full heterogeneous environment was done in order to check the behavior of the models when an stressed situation is presented.

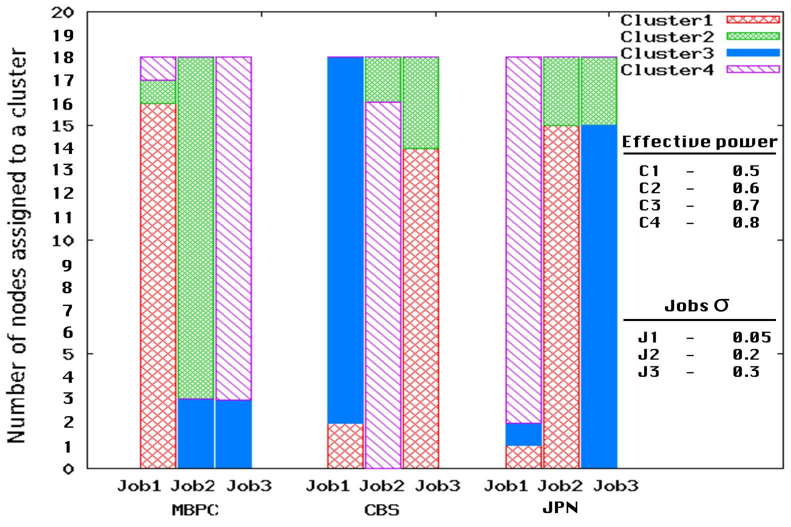

Fig. 6. Allocation map for the three models, on the test with ABW=0.4

The number of jobs and its characteristics were the same than in the previous experiment. However, in this case the effective power of all the nodes for all clusters was established randomly, with values ranging from 0.3 to 0.9. This distribution can represent those cases where the multi-cluster is being shared with local users that influences in the computational capacity of the nodes.

The comparison was performed by varying the ABW values as in the previous study. In this case, the results obtained show the ability of the $MBPC$ model to achieve a balanced allocation between processing and communication.

First of all, we compared the processing and communication slowdowns, taking the standalone allocation for the jobs as reference. The results obtained are shown in Figure 7 and Figure 8 for the SP and SC respectively.

In the case of the SP, Figure 7, $MBPC$ produced solutions near to the standalone processing slowdown, irrespective of the available bandwidth, while CBS and JPN have higher dispersion, due mainly because they didn't take care on the processing resources. The fairness of the $MBPC$ allocations produces dispersion values no greater than 20%, while CBS and JPN rise to 60%.

Otherwise, for the SC values, Figure 8, JPN obtained better solutions because its objective function is exclusively to minimize the network occupation. $MBPC$ and CBS had similar results, and they accomplished the non-saturation of the network links.

Figure 9 shows the comparison for the job execution time for the test with ABW set to 0.4 Gbps. As can be seen, $MBPC$ produces the global slowest execution times for the set of jobs, because it takes into account both processing and communication resources. CBS and JPN, have worse processing times due

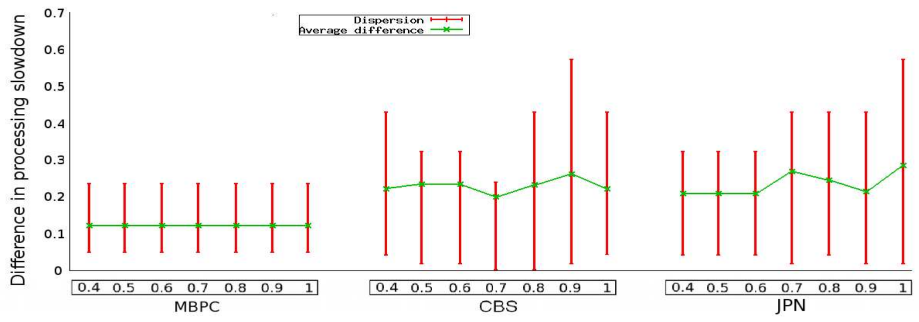

Fig. 7. Comparison of the differences in processing slowdown, varying the ABW value, in an heterogeneous environment

to the heterogeneity of the effective power. The communication time was the same for all of these, as the network links were not saturated and the data transmission could be done at the higher speed. This test showed that $MBPC$ produced an allocation with a global execution time 10% lower than the other models.

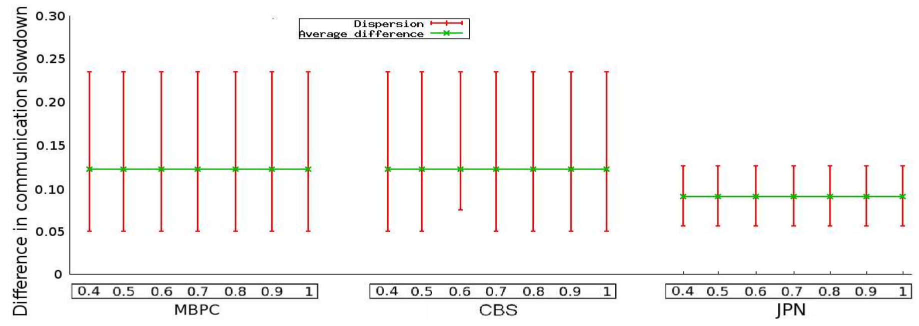

Fig. 8. Comparison of the differences in communication slowdown, varying the ABW value, in an heterogeneous environment

Finally, we can conclude that $MBPC$ has the ability to carefully evaluate the multi-cluster resources, processing and communication, obtaining the allocation that has the greatest benefit for all jobs treated in the allocation process. $MBPC$ produced allocations with a low dispersion degree, around 5% in homogeneous environments, and near 20% in heterogeneous environments, while CBS and JPN produced allocations with a dispersion degree of about 15% in homogeneous environments and 60% in heterogeneous environments.

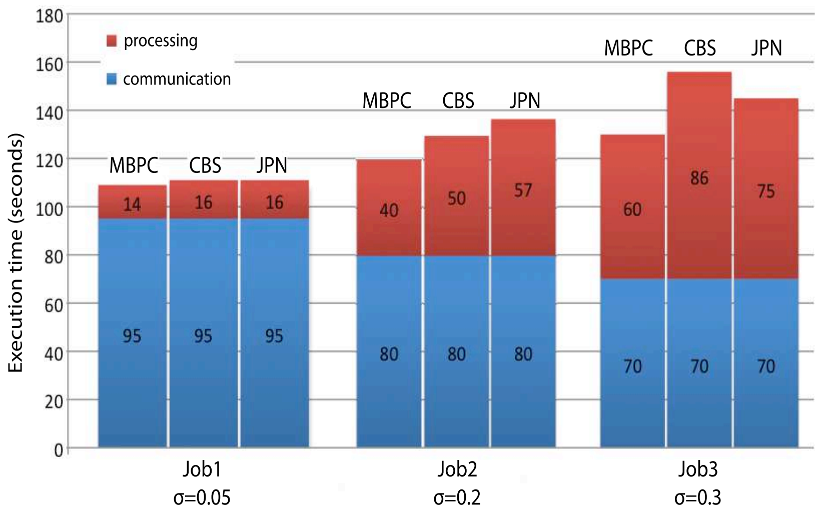

Fig. 9. Time comparison for the three models, on the test with ABW=0.4, in an heterogeneous environment

5 Conclusions and future work

This work presents a linear programming model, named *Multiple Best Processing and Communication* (*MBPC*), which solves the problem of allocating multiple jobs simultaneously in a heterogeneous and non-dedicated multi-cluster system. The model minimizes the global execution time for all the jobs in the system queue, taking into account both processing and communication requirements. Our model was tested against other models in the literature, and the results of the experimentation show that our model produces solutions with the lowest slowdown for all the jobs, and where the jobs are allocated in a way that they affect each other as little as possible.

In the future work, we plan to extend our model in a stochastic, to take into account temporal scenarios where the allocations will be done considering the further jobs in the queue. In temporal scenarios, sets of jobs could be ready for allocation. Given this reason, the capability of the model to treat multiple jobs simultaneously from the system queue is an important target to reach for.

References

1. J.Abawajy, S.Dandamudi, *Parallel Job Scheduling on Multicluster Computing Systems*, CLUSTER'03: Proc. IEEE Int. Conf. on Cluster Computing,pp.11–18, 2003
2. B. Javadi, M.K. Akbari, J.H. Abawajy, *A performance Model for Analysis of Heterogeneous Multi-Cluster Systems*, Parallel Computing,vol.32(11-12),pp.831–851, 2006

3. W. Jones, W. Ligon, L. Pang, D. Stanzione, *Characterization of Bandwidth-Aware Meta-Schedulers for Co-Allocating Jobs Across Multiple Clusters*, The Journal of Supercomputing, vol.34(2), pp.135–163, 2005
4. E.M. Heien, N. Fujimoto, K. Hagihara, *Static Load Distribution for Communicative Intensive Parallel Computing in Multiclusters* In 16th Euromicro Conf. on Parallel, Distributed and Network-Based Processing, pp.321–328, 2008
5. C. Yang, H. Tung, K. Chou, W. Chu *Well-Balanced Allocation Strategy for Multiple-Cluster Computing* In 12th IEEE Workshop on Future Trends of Distributed Computing Systems, pp.178–184, 2008
6. A.I.D. Bucur, D.H.J. Epema, *Schedulling Policies for Processor Coallocation in Multicluster Systems*, IEEE TPDS, vol.18(7), pp.958–972, 2007
7. V.K. Naik, C. Liu, L. Yang, J. Wagner, *Online Resource Matching for Heterogeneous Grid Environments*, In CCGRID'05: Proc. of the 5th Int. Symp. on Cluster Computing and the Grid, vol.2, pp.607–614, 2005
8. J.L. Lérida, F. Solsona, F. Giné, J.R. García, P. Hernández, *Resource Matching in Non-dedicated Multicluster Environments*, In VECPAR'08, pp.160–173, 2008
9. C. Banino, O. Beaumont, L. Carter, J. Ferrante, A. Legrand, Y. Robert, *Scheduling Strategies for Master-Slave Tasking on Heterogeneous Processor Platforms*, IEEE TPDS, vol.15(4), pp.319–330, 2004