Resource Matching in Non-dedicated Multicluster Environments*

J. Ll. Lérida[1], F. Solsona[1], F. Giné[1], J.R. García[2], and P. Hernández[2]

[1] Departamento de Informática e Ingeniería Industrial, Universitat de Lleida, Spain
{jlerida,francesc,sisco}@diei.udl.cat
[2] Departamento de Arquitectura y Sistemas Operativos, Universitat Autònoma de Barcelona, Spain
jrgarcia@aomail.uab.es, porfidio.hernandez@uab.cat

Abstract. We are interested in making use of Multiclusters to execute parallel applications. The present work is developed within the M-CISNE project. M-CISNE is a non-dedicated and heterogeneous Multicluster environment which includes MetaLoRaS, a two-level MetaScheduler that manages the appropriate job allocation to available resources.

In this paper, we present a new resource-matching model for MetaLoRaS, which is aimed at mitigating the degraded turnaround time of co-allocated jobs, caused by the contention on shared inter-cluster links. The model is linear programming based and considers the availability of computational resources and the contention of shared inter and intra-cluster links. Its goal is to minimize the average turnaround time of the parallel applications without disturbing the local applications excessively and maximize the prediction accuracy.

We also present a parallel job model that takes both computation and communication characterizations into account. By doing this, greater accuracy is obtained than in other models only focused on one of these characteristics.

Our preliminary performance results indicate that the linear programming model for on-line resource matching is efficient in speed and accuracy and can be successfully applied to co-allocate jobs across different clusters.

1 Introduction

A Multicluster system has a network topology made up of interconnected clusters, limited to a campus- or organization-wide network. There are collections of several clusters formed by commodity workstations in many laboratories, Universities, and research centers. The main goal of the present work is to make use of wasted computational resources of non-dedicated and heterogeneous Multiclusters to execute parallel applications efficiently without disturbing the local applications excessively.

In order to manage the collective computational power of a Multicluster efficiently, special scheduling mechanisms are required to select and map jobs to available resources. We refer to these schedulers as MetaSchedulers. In general, we consider a MetaScheduler to be the software that decides where, when, and how to schedule jobs in

* This work was supported by the MEyC-Spain under contract TIN2007-64974.

a Multicluster. In previous works [11,12], we presented MetaLoRaS, an efficient MetaScheduler made up of a queuing system with two-level hierarchical architecture for a non-dedicated Multicluster. The most important contribution was the effective cluster selection mechanism, based on the estimation of the job turnaround time. Parallel applications were assigned to the clusters where the minimum turnaround time was obtained. MetaLoRaS was globally aware of the state of the Multicluster and worked in conjunction with each individual cluster's local schedulers.

A Multicluster is distinguished from a traditional computational grid in that the Multicluster utilizes a dedicated interconnection network between cluster resources with a known topology and predictable performance characteristics. This kind of networking infrastructure allows for the possibility of mapping jobs across cluster boundaries in a process known as co-allocation or multisite scheduling. Co-allocation of parallel jobs is considered in this paper, as is minimizing their execution time, this being the desired goal.

Previous work in the area of job co-allocation has tended to characterize jobs based only on communication or computation models. Ernemann and Jones [6,10] describe how schedulers designed to allocate node resources across cluster boundaries can result in rather poor overall performance over a wide range of workload characterizations and Multicluster configurations when co-allocated jobs contend for inter-cluster network bandwidth. In order to overcome these situations, our model is based on co-allocating job tasks to avoid both the communication saturation of inter-cluster links and the overloading of Multicluster nodes, which is not considered in these works. In [5,9,4,10] only communication models are presented, being useful to evaluate the system performance instead of taking online scheduling decisions and accurate predictions about the execution time.

The essence of our MetaScheduling model is to solve the resource matching as an integer-programming problem. Previous work [3,13] illustrates the benefits of using integer programming techniques to solve scheduling problems. However, Naik [13] provides a globally optimal for the system performance assuming that workload is known, and Banino [3] centered on time-sharing scheduling solutions difficult to implement in practice.

The present work aims to extend the works presented in [10,14] and [11,8], creating a MetaScheduling model which takes into account the effect of co-allocation on both computing and communication times. By doing so, we are able to mitigate the negative effect on co-allocated jobs, improving the prediction accuracy of the turnaround time estimation of parallel jobs. This in turn increase the system performance by improving the prediction-like scheduling system. Furthermore, the model takes into account the resource occupancy and capacity of the forming non-dedicated Multicluster nodes. This fact guarantees low impact on the performance of local user applications.

The rest of the paper is organized as follows. In section 2, we present the characterization of both, the Multicluster environment and parallel jobs. In section 3, the integer programming model for matching parallel applications in Multicluster systems is presented. The applicability of the model and the goodness when applied in a real Multicluster system is evaluated in Section 4. Finally, the conclusions and future work are detailed.

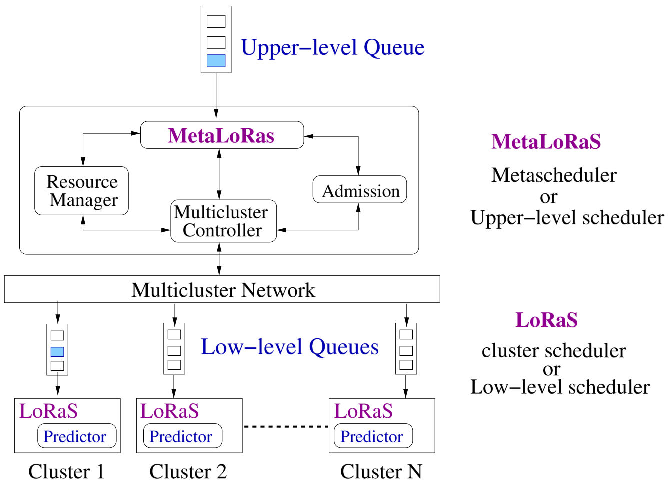

Fig. 1. Multicluster Architecture

2 Multicluster Environment

In [12] we proposed a Multicluster platform. The jobs arriving in the Multicluster enter the Upper-level Queue awaiting scheduling by the MetaScheduler, named *MetaLoRaS*. *MetaLoRaS* assigns jobs to the cluster with the minimum estimate of turnaround time. The estimation is obtained by each local cluster or Low-level scheduler, named *LoRaS* (Long Range Scheduler). *LoRaS* [7] is a space-sharing scheduler with an efficient turnaround predictor [8].

MetaLoRaS is made up of five components (see Fig. 1). These are the *Upper-level Queue* (a queuing system), the Multicluster scheduler (named *MetaLoRaS*), the *Admission system*, the *Resource Manager* and the *Multicluster Controller*.

MetaLoRaS is the Multicluster scheduling system. It is responsible for selecting the next job to be executed from the *Input Queue* (the entry point of parallel jobs), and also the cluster where this job will be executed. The part of *MetaLoRaS* responsible for assigning jobs to clusters is denoted as *Resource Matcher* (RM).

The *Admission System* is responsible for admitting new jobs into the system. This module will accept the new job whenever its required resources are satisfied. If not, the job is discarded. The specified resources are the number of workstations, the Memory size and the per-node bandwidth. It is possible to specify different resource limits in each cluster.

The *Multicluster Controller* collects real time information about the state of each cluster. If an event occurs in one cluster (job start, finish), the *Multicluster Controller* is notified of such a change. The *LoRaS* system is responsible for notifying the *Multicluster Controller* about the cluster state changes.

The *Resource Matcher* (RM) has been designed as an Integer programming approach. The RM is responsible for obtaining a snapshot of the state of the resources

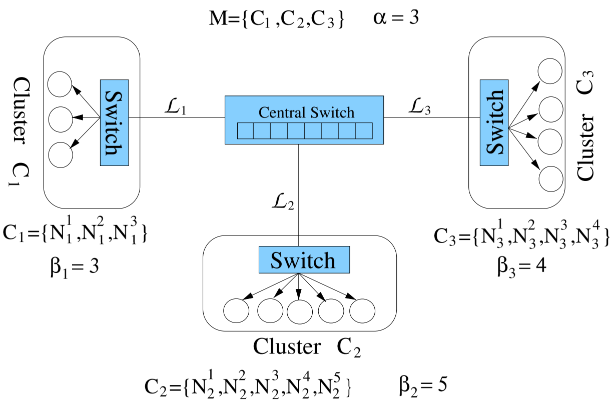

Fig. 2. Multicluster topology

from the *Multicluster Controller* and for generating a mapping solution that will be used by the MetaScheduler *(MetaLoRaS)*. To do this, the RM performs the following functions: (1) it accepts a job matching request through the MetaScheduler, (2) requests the current status of the Multicluster from the *Multicluster Controller*, (3) obtains the parallel application information, (4) submits the parallel application and the Multicluster status information to a mixed integer programming solver and (5) maps the job accordingly to the results obtained in step 4.

Job co-allocation consists of mapping jobs across cluster boundaries. Co-allocation is necessary when a job requires more nodes than the ones available on each particular cluster, but collectively there may be enough available nodes elsewhere in the Multicluster to accommodate such a job. There are situations where despite having enough available nodes in a particular cluster, it may be better to take advantage of remote resources, because they are more powerful or they are the more appropriate for the nature of the parallel job. The *Resource Matcher* (RM) is responsible for deciding if the job will be co-allocated across multiple clusters or mapped exclusively onto one cluster. Scheduling decisions are based on minimizing the job execution time, despite the jobs are exclusively assigned to an unique cluster or across multiple clusters.

2.1 Problem Statement

We are interested on Multiclusters defined 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.

Formally, a Multicluster M=$\{C_1..C_\alpha\}$ can be defined as a system comprised of α heterogeneous clusters interconnected by means of dedicated links (see Fig. 2). Each Cluster C_i ($1 \leq i \leq \alpha$) is also made up by β_i nodes, this is C_i=$\{N_i^1..N_i^{\beta_i}\}$. $\mathscr{L}$ is the set of inter-cluster links ($\mathscr{L}$=$\{\mathscr{L}_1..\mathscr{L}_\alpha\}$), and L=$\{L_i\}$=$\{L_i^k$, $1\leq i \leq\alpha$ 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. We suppose that network bandwidth and latency of inter-cluster links are better than the intra-cluster ones.

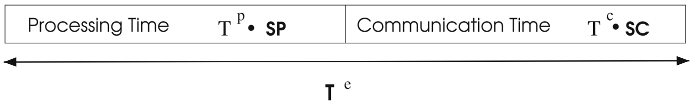

Fig. 3. Execution Slowdown

The model assumes that the jobs follow a BSP (Bulk Synchronous Parallel) model. A BSP job is comprised of coarse or medium grained tasks that require a fixed number of processors (one per task) during their lifetime. The size of their component tasks is generally similar. In addition, each task is comprised of various iterations in which computation alternates with communication and synchronization phases. The job assignment is static, that is, once the job is mapped into a particular set of nodes, no more re-allocations are performed. Additionally, jobs can be co-allocated in a Multicluster by allocating nodes from different clusters to the same job in order to better meet the collective needs across the Multicluster.

We define the job's execution time, T^e (see Fig. 3), 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 times 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 *SP* and *SC*, the processing and communication slowdown respectively.

2.2 Processing Characterization

In a heterogeneous and non-dedicated environment, the computing power and its availability can provoke different processing capabilities of the constituent nodes. The current work presents solutions for measuring these factors and studying their effect on the execution time of the co-allocated jobs.

In a heterogeneous environment, we must take into account the computing power differences between the processor units that form the Multicluster. According to [5], we define the relative Power weight (P_i^k) of the cluster *i* node *k* ($1 \leq i \leq \alpha$ and $1 \leq k \leq \beta_i$), as the computing power ratio of such node with respect to the most powerful node of the Multicluster. The P_i^k range is $0 < P_i^k \leq 1$. $P_i^k = 1$ means that cluster *i* node *k* is the most powerful node in the Multicluster. We obtain the relative computing power of each node by averaging various relative power measurements with different applications.

Local and even parallel jobs executing on the cluster lower the performance of new parallel jobs.The model takes this situation into account by sampling the availability of the computing resources. As was shown in [14], we can obtain an effective measurement of the CPU availability by relating the average of the number of process in the system and the CPU occupancy. We define the Availability of the cluster *i* node *k* (A_i^k) as the percentage of CPU occupancy. $A_i^k \simeq 0$ when 100% of the CPU is occupied and $0 < A_i^k \leq 1$ otherwise.

We define the Effective Power weight of cluster *i* node *k* (Γ_i^k) as the product between the relative Power weight and the Availability of such a node. Formally, Γ_i^k is defined as follows:

$$\Gamma_i^k = P_i^k \cdot A_i^k, \tag{2}$$

where $\Gamma_i^k = 1$ means that cluster i node k has the full capacity to run the tasks at full speed. When $0 < \Gamma_i^k < 1$, the node k of cluster i is unable to execute the task at full speed. Therefore, the processing slowdown of such a node (SP_i^k) is inversely proportional to its Effective Power weight, $SP_i^k = (\Gamma_i^k)^{-1}$.

As in our model we assume that each job task is generally similar in size and they are executing separately, the job execution time is defined as the elapsed execution time of the slowly task. Thus, the processing slowdown can be obtained by taking the node with the lowest Effective Power weight into account, or in other words, the node with the maximum slowdown. According to this, we formally define the slowdown of processing time (SP) in function of the slowdown obtained by each allocated node as follows:

$$SP = max\{SP_i^k, 1 \leq i \leq \alpha\, and\, 1 \leq k \leq \beta_i\} \tag{3}$$

2.3 Communication Characterization

Communication characterization is based on the model described by Jones in [10] for homogeneous and dedicated environments. We provide resource heterogeneity to Jone's model. Furthermore, we add the ability to take into account the effect of the local workload on the co-allocated applications.

We assume that the parallel jobs follow an all-to-all communication pattern periodically throughout their execution, one of the most frequently used in parallel processing. Each task of a given job j is characterized by an average per-node bandwidth metric, $PNBW^j$, consisting of the communication needs for job j.

In co-allocation cases, nodes can communicate across cluster boundaries. This communication will require a certain amount of bandwidth on the inter-cluster network links. Saturation of inter-cluster links reduces job performance drastically. In order to determine when the inter-cluster links become saturated, we must identify how much bandwidth a job will require, and more precisely, each forming task job.

We define BW_i^j (equation 4) as the amount of bandwidth required by job j on inter-cluster link i ($1 \leq i \leq \alpha$). Formally:

$$BW_i^j = \left(n_i^j \cdot PNBW^j\right) \cdot \left(\frac{n_T^j - n_i^j}{n_T^j - 1}\right), \tag{4}$$

where n_T^j is the total number of nodes required by job j, and n_i^j is the number of nodes allocated to job j on the cluster C_i. The first factor of the equation is the total bandwidth required by all the nodes associated with job j on cluster C_i. The second factor represents the communication percentage of job j with other cluster nodes (not in C_i), that will use the inter-cluster link i.

Each communication link i is characterized by a maximum bandwidth rating, BW_i^{max}. We define the saturation degree of an inter-cluster link i (BW_i^{sat}) as the ratio between the maximum bandwidth and the total bandwidth required by the jobs that share the link i. Formally:

$$BW_i^{sat} = \frac{BW_i^{max}}{BW_i^{consumed} + BW_i^j}, \tag{5}$$

where $BW_i^{consumed}$ is the bandwidth occupied by other local or parallel applications in the link i. When $BW_i^{sat} \geq 1$, the link i is *not saturated*. Otherwise, when $0 \leq BW_i^{sat} < 1$ the link i is *saturated*.

A job j using a saturated inter-cluster link i will experience a communicating slowdown inversely proportional to the saturation degree of such a link i. Formally:

$$SC_i = \left(BW_i^{sat}\right)^{-1} \tag{6}$$

If any, the most saturated inter-cluster link will determine the communication slowdown of the co-allocated job. We define the communication slowdown of a job j (SC) as the maximum communication slowdown of such job in each allocated inter-cluster link. Formally:

$$SC = max_i\{SC_i,\ 1 \leq i \leq \alpha\} \tag{7}$$

3 IP Matching Model

Integer Programming (IP) is a technique for solving certain kinds of problems: maximizing or minimizing the value of an objective function subject to some constraints. The objective function and constraints are linear expressions. In the following, we describe our resource-matching approach based on mixed-integer programming techniques.

The problem to be solved in the IP model is the matching of jobs in a Multicluster environment, while avoiding the negative effects of sharing the communication links and processing resources. To do this, the IP model must represent the *job matching request* (specifying their resource requirements) and the state of the Multicluster resources (*Multicluster State*) in order to search for an optimal solution.

The *job matching request* specifies the job requirements as the number of tasks, amount of Memory, per-node bandwidth and the ratio between computation and total execution time. Multicluster nodes without enough Memory are discarded.

The *Multicluster State* comprises the following information of every node: CPU and Memory availability, and both maximum capacity and availability of the intra-cluster communication links. The corresponding inter-cluster information is obtained from the intra-cluster one and the previous job assignments. Only periodic samples of the Multicluster nodes is necessary.

The *Resource Matcher* maps the jobs by minimizing the job execution time. Jobs can be mapped across cluster boundaries. The obtaining of this minimum is performed by means of the Integer Programming solver of CPLEX [1], by using the “Branch and Bound” algorithm. Obviously, this is a well known NP-complete problem. The interest of this work is centered in the definition of heuristics and constraints which delay the exponential time-cost with the number of Multicluster nodes as much as possible.

Input arguments:

1. j: job to be matched.
2. τ^j: number of tasks making up job j.
3. $PNBW^j$: per-node bandwidth requirement for the job j.
4. $M = C..C_\alpha$: Multicluster composition.
5. $\mathscr{L}$ and L=$\{L_i\}$=$\{L_i^k$, $1 \leq i \leq \alpha$ and $1 \leq k \leq \beta_i\}$: set of inter- and intra- cluster links.
6. Γ_i^k: Effective Power weight for the cluster i node k ($1 \leq i \leq \alpha$ and $1 \leq k \leq \beta_i$).
7. BW_i^{av}: available bandwidth for each inter-cluster link $\mathscr{L}_i$, $1 \leq i \leq \alpha$.
8. BW_i^{max}: maximum bandwidth for each inter-cluster link $\mathscr{L}_i$, $1 \leq i \leq \alpha$.

Output parameters:

9. X_i^k, $1 \leq i \leq \alpha$ and $1 \leq k \leq \beta_i$: boolean variable associated to cluster i node k. X_i^k=1 if job j is matched to cluster i node k, and 0 otherwise.
10. SP: processing slowdown. $SP = max\{SP_i^k, 1 \leq i \leq \alpha \, and \, 1 \leq k \leq \beta_i\}$.
11. SC: inter-cluster link communication slowdown. $SC = max_i\{SC_i,\ 1 \leq i \leq \alpha\}$.

Objective Function:

12. $min\{T^e\}$

Constraints:

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

Fig. 4. Model Definition

3.1 Model Definition

An integer-programming model includes input parameters, variables, a set of constraints on the value of the variables, and an objective function. The goal of the model is to find values for every variable so that all constraints are satisfied and the value of the objective function is maximized or minimized.

The input parameters, objective function and constraints of the model presented in this work, are shown in figure 4.

Given a job j, this model finds the best feasible match between the job and the resources taking the heterogeneity and the availability of the resources into account along with the requirements of the job j.

The model accepts as input argument a job j, defined by the number of tasks (τ^j) and the per-node bandwidth ($PNBW^j$). Another group of input arguments are the ones characterizing the Multicluster (M). The variable Γ_i^k defines the Effective Power weight of each node, and the variables BW_i^{av} and BW_i^{max} are the available and maximum bandwidths respectively of the inter-cluster links ($\mathscr{L}$).

The output parameter X_i^k is a boolean variable informing about the mapping of the job j. Other outputs are SP and SC, defined in sections 2.2 and 2.3 respectively. The constraints and the objective function are defined below.

3.2 Constraints

The IP model comprises two constraints, the Gang matching and the non-saturation of inter-cluster links. As major network performance is supposed to inter-cluster links, their constraints also includes the saturation of the intra-cluster ones. Next the two constraints are studied separately.

Gang Matching This constraint ensures that we allocate all the required resources of the parallel job j. In other words, each task is allocated to one processor. The gang matching constraint is formalized with the linear equation 8.

$$\sum_{1\le i\le \alpha, 1\le k\le \beta_i} X_i^k = \tau^j, \tag{8}$$

where τ^j is the number of tasks making up job j and X_i^k is equal to 1 if a task in job j is assigned to cluster i node k. This constraint guarantees the assignment of every task making up job j.

Non Inter-Cluster Link Saturation Non-saturation of the inter-cluster links ensures that the bandwidth consumed by the mapping does not exceed the total available bandwidth capacity of the inter-cluster links. This constraint avoids the saturation of inter-cluster links. We formalize this constraint with the equation 9.

$$SC \le 1, \tag{9}$$

where $SC = max_i\{SC_i,\ 1 \le i \le \alpha\}$ is the maximum slowdown of the inter-cluster links used by job j. The inter-cluster link slowdown, SC_i, was calculated by means of equation 6, explained in section 2.3.

3.3 Objective Function

The objective function defines the quality of a solution when multiple feasible solutions exist. The matching solver uses the objective function to select the best matching solution. In the present work, we are interested in obtaining the minimum execution time for parallel jobs (T^e), defined in section 2.1 equation 1. Accordingly, the objective function is formalized by equation 10.

$$min\{T^e\} \tag{10}$$

4 Experimentation

To study the efficiency of the proposed model we made a great range of tests modifying the amount of resources, their utilization, and the parallel applications characterization. Moreover, we tested the prediction accuracy of the execution time executing parallel applications in a real environment.

The real environment was a Multicluster made up of 2 non-dedicated clusters (CLUSTER1 and CLUSTER2). CLUSTER1 was made up of ten 3-GHz uni-processor workstations with 1GB of RAM, interconnected by a 1-Gigabit network. CLUSTER2 was a heterogeneous cluster made up of ten workstations, five 3-GHz uni-processor with 1GB of RAM and 1-Gigabit network link, and five 3-GHz multiprocessor with 512MB of RAM and 100Megabit network link.

To carry out the experimentation, local and parallel applications need to be defined. The local workload was represented by a synthetic benchmark (named *local_bench*) that can emulate the usage of 3 different resources: CPU, Memory and Network traffic. The use of these resources was parametrized in a real way. According to the values obtained by collecting the user activity in an open laboratory over a couple of weeks, *local_bench* was modeled to use 15% CPU, 35% Memory and a 0.5KB/sec LAN, in each node where it was executed.

We selected two parallel applications, which follows the BSP model, from the NAS parallel benchmarks suite [2]: MG (Multigrid) and IS (Integer Sort). However, the two jobs had different communication patterns and processing/communication needs at each iteration. These parallel jobs were characterized by the number of tasks, the computation time, and the size of their communications.

To study the effect of the constraints on the efficiency of our proposal we defined three different models with different constraint specifications:

Optimal. This approach obtains the optimal solution, looking for the minimum effective slowdown of parallel applications. This model allows the utilization of saturated links. It aims to obtain the best mapping by taking the characterization of parallel jobs and the resource availability into account.

Non-Saturated. In this model the non inter-cluster link saturation constraint was applied. The solver attempts to minimize the execution time of the mapping solutions that will not saturate any inter-cluster link.

Non-Saturated with Non-Optimization. This model does not looks for the optimal solution. Thus, the first solution that avoids the inter-cluster links saturation is returned. This model is thought to be useful with Multiclusters with a high number of resources, where the obtaining of the optimal solution is excessively expansive.

To evaluate the efficiency of our mixed integer programming approach we compare the elapsed time of the *Resource Matcher* to obtain a feasible solution (by means of the CPLEX solver [1]) for different types of parallel jobs and local activity requirements, with different amount of computational resources and inter-cluster links. The per-node bandwidth requirements of the parallel task ($PNBW^j$) was varied from 25% to 75%. The number of workstations with local activity was varied from 0 to 75%.

4.1 Performance Results

First or all, we want to compare the effect of different processing and communication loads on the Optimal and Non-Saturated models.

Figure 5 shows the resulting communication slowdown obtained by the matching solver. As can be seen in Figure 5(right), the Non-Saturated model ensures the non inter-cluster links saturation. Otherwise, in the Optimal model, figure 5(left), the slowdown grows quickly with the network requirements of the parallel application ($PNBW^j$). The local activity has less effect in both models.

Figure 6 shows the effects of $PNBW^j$ and the local activity in the obtaining of the resource matching (by the solver). The behaviour of the models are opposed. The Non-Saturated model is more time-costly than the Optimal one by increasing the nodes with

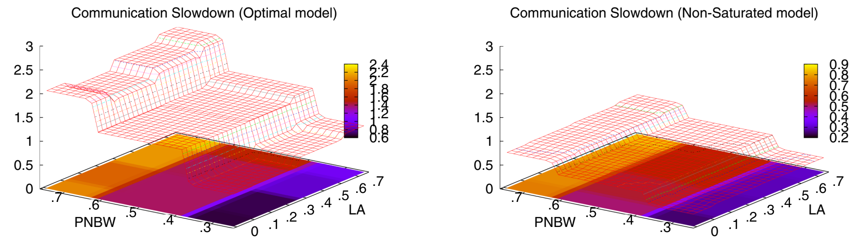

Fig. 5. Communication Slowdown varying $PNBW^j$ and the local activity (LA)

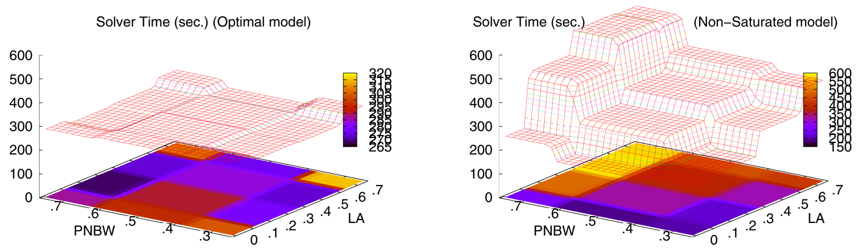

Fig. 6. Solver time varying $PNBW^j$ and the local activity (LA)

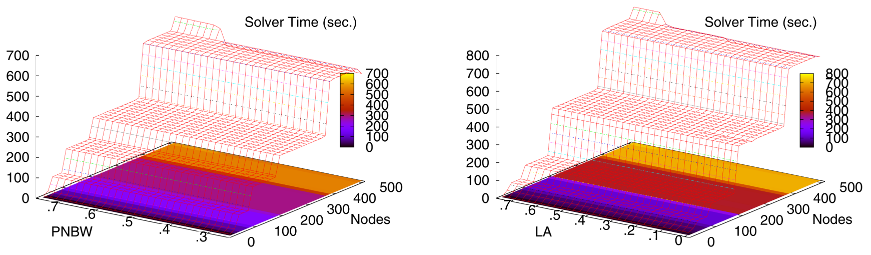

Fig. 7. Solver time. (left) $PNBW^j$ vs. nodes (right) LA vs. nodes.

local activity and the $PNBW^j$. We can observe as in the Optimal model, the bandwidth requirements has a smooth effect on the solver behaviour. Meanwhile, for the Non-saturation model, figure 6(right), the solver response time grows quickly with the bandwidth requirements. This is produced because in the Non-saturated model there are less valid solutions, an the obtaining of one of them is more difficult in time.

Figure 7 shows the solver response time of the Optimal model, by varying the number of nodes jointly with $PNBW^j$ (left) and the local activity (right). It can be appreciated as the predominant parameter in this model is the number of nodes. These results corroborates the ones obtained in Fig. 6(left).

Figure 8 shows the impact of the number of inter-cluster links on the solver response time for different constraints. To study this relationship we fixed the number of nodes per cluster (8 nodes) and ranged between 2 to 64 the number of clusters (inter-cluster

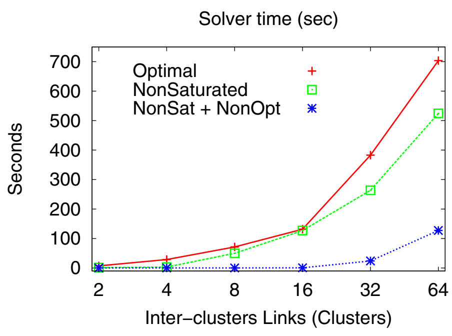

Fig. 8. Resource Matching Time

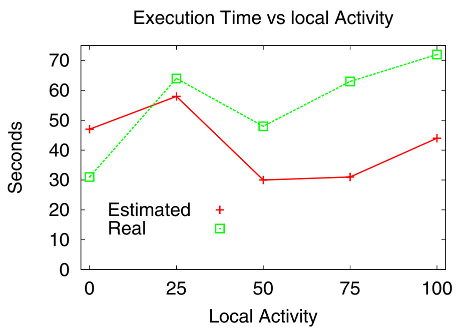

Fig. 9. Estimated vs. Real times

links). The number of constraints in the model have a direct impact on the solver response time. The obtained results indicate that Optimal and Non-Saturated models have a correct behavior for a reasonable number of resources. In our case, below 16 inter-cluster links with 8 nodes per cluster (128 nodes), the response time never overtake one minute. Above this threshold it is advisable the use of the Non-Saturated with Non-Optimization model.

4.2 Prediction Accuracy

In order to evaluate the prediction accuracy of the IP model, we compared the estimations produced by the solver with the real executions of IS and MG. Both benchmarks were executed multiple times with different number of tasks and different local activity situations. Solver times were obtained by using the Optimal model.

The obtained results (see Fig. 9) are very hopeful. Despite the differences between the estimated and real times, we thought that the estimated times can be corrected by applying some sort of correction mechanism, because the two lines have a similar shape. This is the most interesting field to be investigated in the future.

5 Conclusions and Future Work

In the present work we have presented a resource matching mechanism based on integer programing, for non-dedicated and heterogeneous Mulsticluster systems. The model fits efficiently both computation and communication parallel requirements to available Mulsticluster resources by considering the sharing of resources between parallel and local applications.

The results show that, using mixed integer programing, we can model different resource matching situations in a flexible way, and solve them efficiently. As we shows, the number of resources has a great impact on the solver response time. It is important to develop mechanisms to adapt dynamically to inter-arrival job rate, number of resources, etc. The IP model described in the present work allows to adapt the scheduling system to these situations dynamically.

Future work is directed towards the search for a correction factor of the estimates. We also will investigate regression models in the obtaining of the Multicluster State. Due to the intrinsic dynamism of non-dedicated Multiclusters, their state change very quickly, and the on-time monitoring used in this work does not reflect this situation correctly.

In this study, we considered one job at a time. In a further work, we wish to consider the matching problem for multiple jobs, in order to avoid solving large optimization problems achieving a global optimal. Moreover, this matching scheme will allow the matching solver to apply new objective functions based, for example, on throughput or load balancing.

On the other hand, we want to compare the benefits on the system performance obtained with the use of the mixed-integer programing approach, with other meta-scheduling mechanisms based only on partial information about the communications or computation capabilities.

References

1. Cplex (June 30, 2007), `http://www.ilog.com`
2. Bailey, D.H., Dagum, L., Barszcz, E., Simon, H.D.: NAS parallel benchmark results. In: Supercomputing, pp. 386–393 (1992)
3. Banino, C., Beaumont, O., Carter, L., Ferrante, J., Legrand, A., Robert, Y.: Scheduling strategies for master-slave tasking on heterogeneous processor platforms. IEEE Transactions on Parallel and Distributed Systems 15(4), 319–330 (2004)
4. Bucur, A.I., Epema, D.H.: The performance of processor co-allocation in multicluster systems. In: CCGRID 2003: Proceedings of the 3st International Symposium on Cluster Computing and the Grid, Washington, DC, USA, p. 302. IEEE Computer Society, Los Alamitos (2003)
5. Du, X., Zhang, X.: Coordinating parallel processes on networks of workstations. Journal of Parallel and Distributed Computing 46(2), 125–135 (1997)
6. Ernemann, C., Hamscher, V., Streit, A., Yahyapour, R.: Enhanced algorithms for multi-site scheduling (2002)
7. Hanzich, M., Giné, F., Hernández, P., Solsona, F., Luque, E.: Cisne: A new integral approach for scheduling parallel applications on non-dedicated clusters. In: Cunha, J.C., Medeiros, P.D. (eds.) Euro-Par 2005. LNCS, vol. 3648, pp. 220–230. Springer, Heidelberg (2005)

8. Hanzich, M., Giné, F., Hernández, P., Solsona, F., Luque, E.: Using on-the-fly simulation for estimating the turnaround time on non-dedicated clusters. In: Nagel, W.E., Walter, W.V., Lehner, W. (eds.) Euro-Par 2006. LNCS, vol. 4128, pp. 177–187. Springer, Heidelberg (2006)
9. Javadi, B., Abawajy, J.: Performance analysis of heterogeneous multi-cluster systems. In: ICPPW 2005: Proceedings of the 2005 International Conference on Parallel Processing Workshops (ICPPW 2005), Washington, DC, USA, pp. 493–500. IEEE Computer Society, Los Alamitos (2005)
10. Jones, W.M., Ligon, W.B., Pang, L.W., Stanzione, D.: Characterization of bandwidth-aware meta-schedulers for co-allocating jobs across multiple clusters. The Journal of Supercomputing V34(2), 135–163 (2005)
11. Lérida, J.L., Solsona, F., Giné, F., Hanzich, M., García, J.R., Hernández, P.: Metaloras: A re-scheduling and prediction metascheduler for non-dedicated multiclusters. In: Cappello, F., Herault, T., Dongarra, J. (eds.) PVM/MPI 2007. LNCS, vol. 4757, pp. 195–203. Springer, Heidelberg (2007)
12. Lérida, J.L., Solsona, F., Giné, F., Hanzich, M., Hernández, P., Luque, E.: Metaloras: A predictable metascheduler for non-dedicated multiclusters. In: Guo, M., Yang, L.T., Di Martino, B., Zima, H.P., Dongarra, J., Tang, F. (eds.) ISPA 2006. LNCS, vol. 4330, pp. 630–641. Springer, Heidelberg (2006)
13. Naik, V.K., Liu, C., Yang, L., Wagner, J.: Online resource matching for heterogeneous grid environments. Ccgrid 2, 607–614 (2005)
14. Wolski, R., Spring, N., Hayes, J.: Predicting the CPU availability of time-shared unix systems on the computational grid. Cluster Computing 3(4), 293–301 (2000)