DOI:10.5392/IJoC.2011.7.2.001

Cost-Aware Dynamic Resource Allocation in Distributed Computing Infrastructures

Gianni M. Ricciardi
University of Science and Technology,
Korea Institute of Science and Technology Information
Daejeon, 305-806, Republic of Korea

Soonwook Hwang
Korea Institute of Science and Technology Information
Daejeon, 305-806, Republic of Korea

ABSTRACT

Allocation of computing resources is a crucial issue when dealing with a huge number of tasks to be completed according to a given deadline and cost constraints. The task scheduling to several resources (e.g. grid, cloud or a supercomputer) with different characteristics is not trivial, especially if a trade-off in terms of time and cost is considered. We propose an allocation approach able to fulfill the given requirements about time and cost through the use of optimizing techniques and an adaptive behavior. Simulated productions of tasks have been run in order to evaluate the characteristics of the proposed approach.

Keywords*: Distributed Computing, Resource Allocation, Optimization*

1. INTRODUCTION AND RELATED WORKS

The efficient use of a set of computing infrastructures for a large production of independent tasks raises several scheduling issues that need to be addressed. Each computing resource has its own features in terms of computing power and cost that have to be considered when a goal comprising a deadline to meet and a total cost to afford is given to be satisfied; hence for each task submission a decision has to be taken about which resource the task should be submitted to.
As *computing infrastructures* we consider:

- a *grid* infrastructure: a set of different computing sites accessed through a common user interface and security infrastructure, providing heterogeneous machines from the point of view of computing power;
- a *supercomputer*: a set of homogeneous computing nodes managed by a local batch scheduler;
- A *cloud* computing infrastructure: a set of Virtual Machines(VM) provided either by a private cloud or a commercial cloud provider.

Each resource is described by several parameters including the *cost* and the number of available CPUs; the *cost* is indicated as the number of *credits* per second spent to allot one CPU on a given resource. A *credit* is a conventional unit of cost that can be converted to a real currency. We consider the problem of allocating a certain number of CPUs (each one executing a single task) on each resource in order to complete all tasks according to a given deadline or a given limit for the total cost to be paid.
In the literature related to resource allocation the *economic* model is used in a multi-user environment to take decisions about the amount of computing power assigned to each user; different market-based approaches have been proposed as described in [1], and some of them are specific for *grid* infrastructures [2]. An extensive survey [3]of economic and market-based scheduling and resource allocation approaches shows the strength and weakness of this kind of approaches. However these models use a *virtual* currency to optimize scheduling decisions and make an efficient use of resources: they are not intended to model a real cost of usage to be paid by infrastructures' users, and they usually do not consider a deadline to be fulfilled. An interesting analysis specific for *cloud* resources [4] describes an approach aiming to solve the optimization problem of outsourcing tasks to several cloud providers considering deadline constraints and cost optimization; in this case an internal data center is considered along as different cloud infrastructures, but no other kinds of computing resources are taken into account. A more general approach [5] delineates a design for distributed computing systems focused on cost effective resource allocation, and

* *Corresponding author. E-mail : hwang@kisti.re.kr*
Manuscript received Mar 06, 2011 ; accepted Jun.10, 2011

compare grid and cloud systems to this kind of design: it is a multi-users case and the cost used refers to an economic model implemented to optimize resource provisioning.
In the approach we propose a single production of tasks belonging to a single user is considered, to whom the possibility of a choice about a deadline to be fulfilled or a total *real* cost to be paid is given: different kinds of computing resources are optimally scheduled accordingly.

2. PROBLEM STATEMENT

In order to formulate the problem with a proper formalism we consider N computing resources and a set of independent tasks making up a *production*: the total number of tasks is called N_T^{tot}. Each of the N resources can execute a single task in a time measured in seconds and equal to $T_{ST}^i[s]$ with a cost per second of $C^i[credit/s]$ per single task, and it is able to run concurrently a maximum number of tasks equal to $N_{Rmax}^i (i = [1 \dots N])$.
At any given time t the number of task yet to be completed is $N_T(t) (N_T(t=0) = N_T^{tot})$, and there are $N_R^i(t)$ tasks running on each resource i (i.e. there are $N_R^i(t)$ CPUs allotted for that resource, each of them executing one task); since any resource can complete all assigned tasks in T_{ST}^i seconds, the corresponding throughput at the time *t* is:

$$P^i(t) = \frac{N_R^i(t)}{T_{ST}^i} \quad i = [1 \dots N][task/s] \quad (1)$$

and the total throughput of the system comprising the N resources:

$$P_{tot}(t) = \sum_{i=1}^{N} P^i(t)[task/s] \quad (2)$$

so that the residual time required to complete the production is:

$$T_{prod}(t) = \frac{N_T(t)}{P_{tot}(t)}[s] \quad (3)$$

The cost to run one task on one resource is:

$$C_{ST}^i = C^i \times T_{ST}^i \quad i = [1 \dots N][credit] \quad (4)$$

while the cost per second per each resource at time *t* is:

$$C_{tot}^i(t) = N_R^i(t) \times C^i \quad i = [1 \dots N][credit/s] \quad (5)$$

hence the total cost per second at time *t*:

$$C_{tot}(t) = \sum_{i=1}^{N} C_{tot}^i(t) = \sum_{i=1}^{N} \left[N_R^i(t) \times C^i\right][credit/s] \quad (6)$$

and the residual cost to complete all the tasks assuming to keep the same allocation of resources till the end of the production is:

$$C_{prod}(t) = T_{prod}(t) \times C_{tot}(t)[credit] \quad (7)$$

The problem is therefore described by the following two equations:

$$\begin{cases} T_{prod}(t) = \dfrac{N_T(t)}{\sum_{i=1}^{N} \dfrac{N_R^i(t)}{T_{ST}^i}} \\ C_{prod}(t) = N_T(t) \times \dfrac{\sum_{i=1}^{N} \left[N_R^i(t) \times C^i\right]}{\sum_{i=1}^{N} \dfrac{N_R^i(t)}{T_{ST}^i}} \end{cases} \quad (8)$$

Where N_T^{tot}, T_{ST}^i, C^i are fixed values describing the production and resources' features, while $N_R^i \in [0, N_{Rmax}^i] (i = [1 \dots N])$ define the number of CPUs allocated on each resource and their values can be decided and modified by a scheduling algorithm.

3. APPROACHES TO RESOURCE ALLOCATION

Considering the beginning of production (*t = 0*), we can fix a trade-off about the total time and cost allowed to execute all the tasks. Obviously the minimum value for T_{prod} is achieved considering the maximum available throughput,
i.e.:

$$P_{tot}^{max} = \sum_{i=1}^{N} \frac{N_{Rmax}^i}{T_{ST}^i} \quad (9)$$

hence the corresponding value of total cost:

$$C_{prod}^{T_{min}} = N_T \times \frac{\sum_{i=1}^{N} \left[N_{Rmax}^i \times C^i\right]}{\sum_{i=1}^{N} \frac{N_{Rmax}^i}{T_{ST}^i}} \quad (10)$$

The minimum value of the production cost C_{prod}^{min} is obtained minimizing the cost function $C_{prod}(t)$: among all values of production time corresponding to this value of production cost we choose the minimum one $T_{prod}^{C_{min}}$. In the general case the production manager is given two possibilities:

1. select a required value for T_{prod} in the range $\left[T_{prod}^{min}, T_{prod}^{C_{min}}\right]$ and let the system minimize the value of C_{prod} considering the subset of $N_R^i (i = [1 \dots N])$ values that gives the wished value of T_{prod};
2. select a required value for C_{prod} in the range $\left[C_{prod}^{min}, C_{prod}^{T_{min}}\right]$ and let the system minimize the value of T_{prod} considering the subset of $N_R^i (i = [1 \dots N])$ values that gives the wished value of C_{prod};

Once the *time vs cost* trade-off is fixed the production is started using the values of $N_R^i (i = [1 \dots N])$ computed in the previous

step. The described case is a *static* one since he values of all parameters describing the resources do not change during the production; to get closer to a real world scenario we have to consider a *dynamic* case implying a variation of quantities like $T_{ST}^{i}(i = [1 \dots N])$, while the values of N_T^{tot}, N_{Rmax}^{i} and C^i are considered constant.

During the production we collect statistics about T_{ST}^{i} in order to update scheduling decisions according to the variation of this parameter, following this approach:

- the production is started after the trade-off time-cost has been fixed and the resulting values of N_R^i computed according to those of T_{ST}^{i} known at $t = 0$;
- while tasks are executed by computing resources statistics are collected about T_{ST}^{i};
- every Δt (equal to an integer multiple of the longest T_{ST}^{i}) the present values of T_{ST}^{i} are evaluated using the collected statistics and used to compute again the values of N_R^i in order to fulfil the fixed value of trade-off, considering in the computation the number $N_T(t)$ of remaining tasks at the current instant of time and the amount of time or credits still available according tothe chosen trade-off;
- the scheduling is then carried on according to the new values of N_R^i till the next time $t + \Delta t$, when a new set of N_R^i is computed again (according to statistics collected till that time) and used to update the scheduling of tasks to resources.

This is clearly a best effort approach, trying to fulfill the chosen trade-off despite variations in the used parameters, and an implicit approximation is due to the integer values of N_R^i so that goals in terms of time and cost are expectedto be accomplished with a certain error.

4. SIMULATION METHODS AND RESULTS

Given the described problem and solving approach we simulated a variable set of resources with different characteristics and a production of independent tasks to be completed by them. The main goal of simulation is to prove the feasibility of the delineated algorithm and figure out what kind of results can be achieved in terms of achieved percentages of time and cost targets. The *dynamic* case has been simulated using different values for the number N of resources and a number of independent tasks $N_T^{tot} = 2000$. The time-cost trade-off has been fixed in all cases choosing the middle value of ranges $[T_{prod}^{min}, T_{prod}^{C_{min}}]$ and $[C_{prod}^{min}, C_{prod}^{T_{min}}]$ respectively. During the simulated executionof tasks by different resources the values of T_{ST}^{i} havebeen changed in a pseudo-random way using normal distributions.Each kind of simulation has been run five timeswith different seeds used to initialize pseudo-random sequences:the same sets of seeds have been used to run simulations with different values of resources number, in order to evaluate the behavior of optimizer functions in similar conditions. Each resource has been modelized by several parameters:

- the maximum number of CPUs that can be allotted on each of them (i.e. the maximum number of tasks they can execute at the same time) N_{Rmax}^{i};
- the execution time T_{ST}^{i} of one task on each resource known at time $t = 0$ (this value is updated during the production collecting statistics about running time oftasks on each resource);
- cost per second C^i of each resource;
- μ and σ of the pseudo-random normal distribution used to model the changing value of T_{ST}^{i}.

The simulation has been implemented as a Discrete-Event Simulation (DES) using the SimPy [6] Python module. Each resource has been modeled as a *Process* executing the number of tasks assigned to it by the *Optimizer Process*, and collecting statistics about variable running times T_{ST}^{i}. TheOptimizer Process makes use of several methods tooptimizedifferent parameters according to the given values of other ones, considering the characteristics of available resources, the number of tasks yet to be completed and the fixed tradeoff: one method minimizes the production time given the target value of cost, the number of tasks and of course the characteristics of resources, while another method minimizes the production cost in a similar way. As return values these functions give a set of values $N_R^i \in [0, N_{Rmax}^{i}](i = [1 \dots N])$ to be assigned to each of the N resources inorder to reach the values of time or cost given as argument to the functions or optimized by them. At the beginning of the production the total number of tasks N_{tot} is considered:then during the production every Δt seconds of simulatedtime the Optimizer is called to perform resource reallocation considering as parameters of minimizing methods the current value of the number of tasks yet to be completed $N_T(t)$ and the remaining time and budget available to complete the production, i.e.:

$$T_{prod}^{trade-off} - t \qquad (11)$$

and

$$C_{prod}^{trade-off} - C_{prod}(t) \qquad (12)$$

where $C_{prod}(t)$ is the number of credits paid till currentinstant of time t.Given a set of resources and their characteristics we computed the average values of production time and cost of simulations related to this set: these values have been compared to the time-cost trade-off ($T_{prod}^{trade-off}$ and $C_{prod}^{trade-off}$) decided before the production.Two different set of resources have been modeled in the simulation runs:

- a *stable* set of resources with a reasonable variability (i.e. a smaller value of σ compared to the second set) of T_{ST}^{i};
- a set of resources with greater values of σ of their probability density functions conveying a higher variability compared to the previous set;

in this way the behavior of Optimizer and its related methods have been tested in different conditions and its adaptive approach evaluated.

Table 1: Resources parameters of first simulation

Resource	N_{Rmax}	T_{ST}	cost	σ
Grid	100	6.0	1.5	1.8
Supercomputer	80	4.0	2.0	0.1
Cloud1	40	5.5	1.8	0.6
Cloud2	20	4.8	1.9	0.4

The results are shown as percentage of trade-off production time and cost: when *Target Time* is indicated it means that the average result in terms of computation time to complete the whole production in the simulations is compared to the trade-off time decided before starting to schedule tasks, with the Optimizer trying to minimize the cost at the same time. Similarly when *Target Cost* is displayed: a cost is fixed before starting the simulated productions and the system tries to optimize allocation of resources in order to fulfill the cost requirement minimizing the time in the meanwhile. A result lesser than or equal to 100% means that the goal has been reached completing the production within a shorter time or with a cost lower than the wished ones. On the contrary a result greater than 100% shows a missed goal since the time or the cost has been longer or larger than the fixed ones.

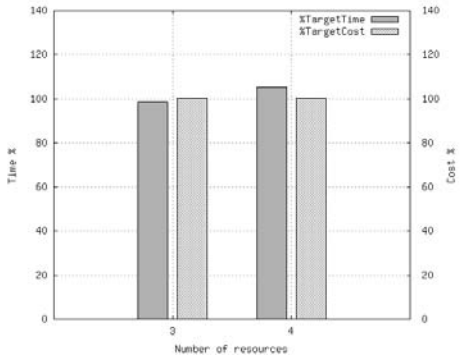

Fig. 1: Results of first simulation

Results corresponding to resources described by Table 1are shown in Fig. 1: in case of three resources the first ones listed in the table have been used. This is the case of *stable* resources and the Optimizer shows good performance especially about the fulfillment of cost targets.

Table 2: Resources parameters of second simulation

Resource	N_{Rmax}	T_{ST}	cost	σ
Grid	100	7.0	1.2	2.8
Supercomputer	80	4.0	2.0	1.1
Cloud1	40	5.6	1.6	1.6
Cloud2	20	5.0	1.8	1.4

In Fig. 2 results from simulated resources of Table 2 are illustrated showing the same good behavior of the Optimizer about cost targets but with some difficulties encountered in order to accomplish time goals. This is probably due to the method used to find the best configuration of resources' allocation in the case of a fixed time requirement trying to simultaneously minimize the cost.

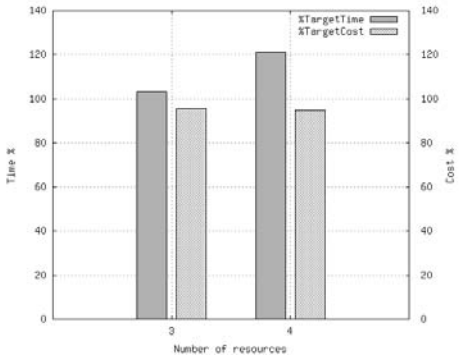

Fig. 2: Results of second simulation

5. CONCLUSION AND FUTURE WORKS

The proposed resource allocation system lets the user choose a trade-off in term of a deadline to be accomplished or a total real cost to be paid to complete a large production
of independent tasks: once one of the constraints has been fixed the other one is minimized by the system during the execution of tasks according to an adaptive behavior. Performed simulations show fairly good results when dealing with stable resources, exhibiting instead some issues with more unstable resources. As future improvements the optimization methods in the case of a fixed target time should be enhanced, and also the variability of N^i_{Rmax} could be consideredas a variable parameter describing resources. Modelscloser to real world resources should be derived withobservations of practical tests, in order to be able to applythe allocation approach to real productions.

8. REFERENCES

[1] Alvin Auyoung, Brent N. Chun, Alex C. Snoeren, and Amin Vahdat. "Resource allocation in federated distributed computing infrastructures", 2004.

[2] Pengcheng Xiong and Yushun Fan. "Cost-aware grid workflow resource allocation". *Semantics, Knowledge and Grid, International Conference*, pp. 422–425, 2007.

[3] Nicolas Dube and Marc Parizeau. "Utility computing and market-based scheduling: Shortcomings for grid resources sharing and the next steps". *High Performance Computing Systems and Applications, Annual International Symposium*, pp. 59–68, 2008.

[4] Ruben Van den Bossche, Kurt Vanmechelen, and Jan Broeckhove."Cost-optimal scheduling in hybrid iaas clouds for deadline constrained workloads". *Cloud Computing, IEEE International Conference*, pp. 228–235, 2010.

[5] Xavier Grehant and Isabelle Demeure."Symmetric mapping: An architectural pattern for resource supply in grids and clouds". *Parallel and Distributed Processing*

Symposium, International, pp. 1–8, 2009.
[6] http://simpy.sourceforge.net/. Accessed on April 2011.

Gianni M. Ricciardi
He received his B.S. degree in electrical engineering from the University of Rome La Sapienza. He then joined the Italian National Institute of Nuclear Physics (INFN) where he has been involved in several projects related to Grid computing. He is currently a Master's degree student at University of Science and Technology in Korea, where he could join the Korea Institute of Science and Technology Information (KISTI).His current research interests include HTC, HPC, multi-resource computing and parallel programming.

Dr. Soonwook Hwang
He is a team leader at Korea Institute of Science and Technology Information (KISTI), responsible for the research and development of enabling technologies such as Grid and Cloud computing for the realization of e-Science. He received the B.S in Mathematics and the M.S in Computer Science from Seoul National University, Korea and got the Ph.D in Computer Science from University of Southern California, USA.