5th SASTech
Iran |Mashhad
May 12 - 17 | 2011
5th Symposium on Advances in Science & Technology

A new scheduling algorithm design for grid computing tasks

Amir M Bidgoli
Phd, Msc, Bsc, Manchester university,
Head of postgraduate Computer Science department at Islamic Azad University of Tehran North Branch
drbidgoli@gmail.com

Zahra Masoudi Nezad
MSc Student at Islamic Azad University
Science and Research Ahvaz Branch
z.masoudinezad@yahoo.com

Paper Reference Number: 0502-937
Name of the Presenter: Zahra masoudinezad

Abstract

Geographically distributed resources cooperate to solve big problems, is called grid computing. Grid computing, is distributed computing model that is provides easy access to heterogeneous resources that are geographically dispersed. Today, due to heterogeneous grid resources that belong to different organizations and locations with different access policies and terms of workload dynamics are inherent; the use of this type in grade sharing, selection and gathering resources computing has become popular. Scheduling in grid computing systems that are normally non-concentrated is important in military, mobile medical and laboratory control systems. Scheduling in grid computing is an inconclusive issue, so cannot used be certain of the algorithms to improve scheduling. In traditional scheduling approaches at grid computing, scheduling time to complete tasks is considered as the most important parameter, while the timing of the economic schedulers should also implement time jobs, cost of resource use is considered.

The algorithm proposed in this paper that called GCDM, considering the cost of data transfer between different tasks and dependencies between tasks, with the modeling as an acyclic directed graph (DAG), ultimately leads to minimize the final cost of implementation tasks.

Keywords: Grid computing, schedule, optimization, DAG, GCDM

1. Introduction

A grid computing infrastructure hardware and software that provided access to high level computational capabilities into a reliable, consistent, pervasive and inexpensive offers [1]. Grid is a shared environment that implemented with the establishment of lasting service and standards, these services support create and share resources to distribute. Computational grid environment are suitable for solving problems that are long and heavy computation. In this environment, resources are geographically distributed, but in terms of logical are seen as a source [2]. The main goal of grid is providing services with high reliability and lowest cost for large volumes of users and support group work and the most important issue in grid computing are resource management and control, reliability and security. Today increased efficiency of grid is an important issue. To increase the efficiency of grid a properly and useful scheduling is needed. Unfortunately, the dynamic nature of grid resources and the demands of different users are causing complexity in the scheduling grid [3]. Scheduling process in grid can be organized in three stages: resource discovery, resource selection and scheduling based on clear goals and the last stage is

request assignment to the most appropriate source. In this study, the scheduling of second stage is to be the focus.
Figure (1) is represented a model of grid scheduling system. The elements at work by two types of data streams are connected to each other: information flow of program or sources and work flow of orders.

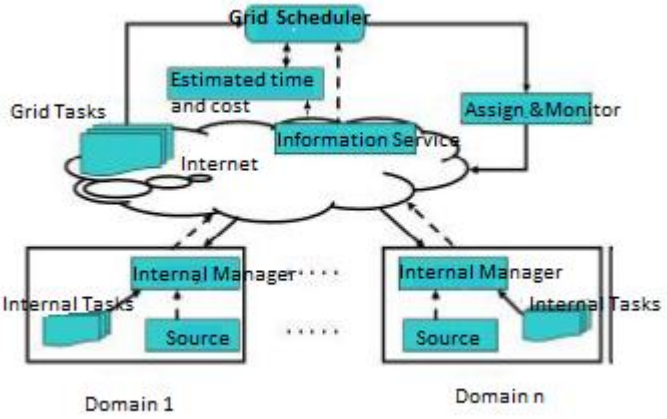

Fig 1: Scheduling tasks in grid environment [4]

Overall, a grid scheduler received requests from user's grid, then uses the information obtained from sources (Grid Information Service) modules appropriate selected the grid service information for the requests and ultimately based on objective functions and resource efficiency predicted, creates a mapping between requests and source.
At a high level, scheduling algorithms are divided to the two global and local. In local scheduling technique on how to allocate processes to a processor and will decide its implementation. In Global scheduling technique using information systems and is done processes allocated to multiple processors for optimize overall performance system [5]. Next level in the hierarchy of Scheduling is choosing between static and dynamic scheduling.
This selection is indicated when are taken scheduling decisions. In static algorithms, necessary information about all the grid resources is available at the moment of scheduling. Hence, can the actual implementations, bring action accurate estimates of the cost calculations. One of the main advantages of this approach is a simply programming point of view Scheduler. In this method, fixed assignment requests and is simple estimate costs requests. But cost estimates based on static information, is not to be adjusted such as in cases when to be broken one of the nodes selected to perform calculations. Or the high volume of load due to requests that answer Time higher than expected. In contrast, the main idea in dynamic scheduling, allocation requests during programs implementation. This method is used when is difficult the estimated cost of Programs or requests entered into the system dynamically. Request queue management in systems such as Condor and Legion is a good example of this method. Dynamically requests scheduling, has two main parts: [6] estimate the system state (in addition to estimating the cost of static methods), and Decision Making. System state estimate is includes gathering information from around the grid and then the estimate. Taken to according to estimates, is done the appointment decision of a request to selected source. Advantage of Load divided at dynamic approach to static scheduling is that the system is not needs to be aware of the behavior of running time request before the run. This point particularly, is beneficial to a system aimed at maximizing utilization of resources, not least by the time implementation of specific tasks.

In the proposed method in this paper, have been suggested a method based on (DAG) [7,8,9,10] that is graph structure tasks of the common methods for scheduling for Grid scheduling with an approach for optimization the final run time and cost. In this approach proposed algorithm with an approach toothily any time trying to find the best choice in terms of cost and implementation time. Next, expression the preliminary theory and concepts necessary for of DAG-based scheduling and then is described details the proposed algorithm.

2. Preliminaries and definitions scheduling based on DAG

DAG is a graph G (V, E) that V set of nodes for DAG such as V= {V_i; i=1…n}. Each node in the graph represents a task. E is series of graph edge and is display dependent on tasks and we E={(V_i , V_j); i , j=1…n&ə !Kink for V_i , V_j }. Each edge (V_i, V_j) in the graph means that in order to run V_j, V_i must be finished. In fact, this relationship is a partial order relation.

Each edge (V_i, V_j) in the DAG have weighing $D_{i,\ j}$, that is display the time required to transfer data that must to be transferred for run V_j from Vi And also noted that before the end of V_i this data will not be transferable to V_j. Each node in a DAG is an input nod or V_{entry} and an output node or V_{exit} that scheduling problem started with resource allocation to the input node and ends on completed resource allocation to the output node and run it. (This hypothesis does not entered fractured in whole issue, because if there is more than one input or output node is added to the DAG the hypothetical node an input or output with edges with zero weight). In given grid, T is number of tasks entered for the schedule grid. C_{T*T} matrix is data transmission cost matrix between the depending nodes so that C_{ij} is represents the cost of data transfer V_i to V_j. D_{T*T} matrix is dependency graph nodes matrix, in this case, D_{ij}, is 1 if V_i is dependent to V_j.

3. Related works

Different scheduling methods presented in the past, such as genetic algorithms, simulated cooling, Forbidden Search, game theory and methods of their combination. The most cost data transfer was ignored between tasks, such an algorithm is known that genetic algorithms, graph on the following tasks in a grid computing tasks has been allocated to processors as follows.

According to the Gantt charts is estimated the final run time at 22 units.

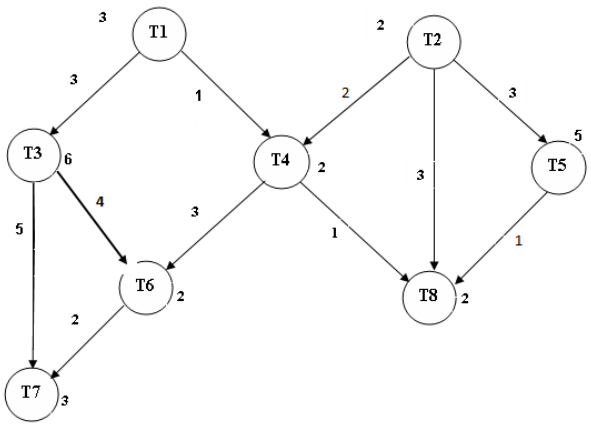

Fig 2: Example of a DAG

In this algorithm, the tasks T_8, T_5, T_3 and T_4 are allocated to the processor P_1 and the tasks T_2, T_1, T_6 and T_7 to the processor P_2, which in following the Gantt chart is observed:

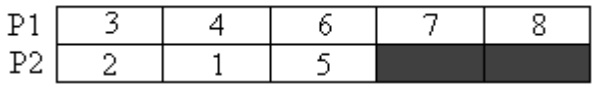

Fig 3: Assign tasks to processors in the traditional method

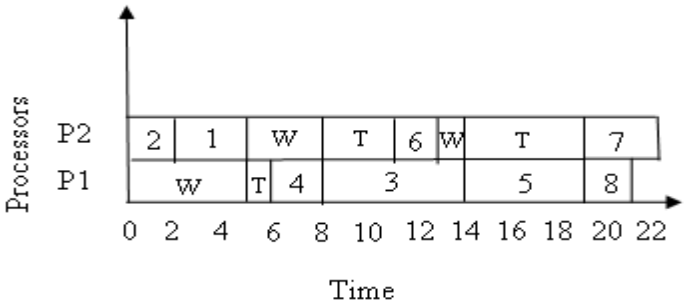

Fig 4: Gantt chart in the traditional method allocation of tasks

Here we have presented a new algorithm in which to consider the cost of data transfer between tasks, the final runtime is improved.

4. The proposed scheduling algorithm

In this way as a costs and dependence matrix for grid (GCDM), we first considering dependencies between tasks, data transfer time and execution time for tasks, we draw acyclic oriented graph (DAG). Then we created the C_{T*T}, D_{T*T} matrix. On Continue working according to the created matrices we explained the allocating tasks to the machines:

C_{8*8} and D_{8*8} matrices are as follows:

$$C_{8*8} = \begin{array}{c} \\ T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ T_7 \\ T_8 \end{array} \begin{array}{c} \begin{array}{cccccccc} T_1 & T_2 & T_3 & T_4 & T_5 & T_6 & T_7 & T_8 \end{array} \\ \begin{bmatrix} 0 & 0 & 3 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 3 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 4 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 3 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Cost data transmission matrix in the example desired graph

$$D_{8*8} = \begin{array}{c} \\ T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \\ T_7 \\ T_8 \end{array} \begin{array}{c} \begin{array}{cccccccc} T_1 & T_2 & T_3 & T_4 & T_5 & T_6 & T_7 & T_8 \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 4 & 5 & 0 \\ 1 & 1 & 0 & 0 & 0 & 3 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 2 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 2 & 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

Dependency between tasks matrix in the example desired graph

In this algorithm, the number of processors as the number of nodes is input, that the input nodes is determined according to the matrix D. (An input node is, that is zero the corresponding row in the matrix D). If we had only an input node, specifies the number of processors, via the first level of nodes dependent with the input nodes. In this example T_1 and T_2 are input nodes, that they assign the processors P_1 and P_2, respectively.

$P_1 = \{ T_1 \}$, $P_2 = \{ T_2 \}$

Then, according to the set of processors tasks and matrix C, we assign the next tasks to processors. Thus that considering line to set the desired tasks of the matrix C, occurs two cases on the following:

1- The Task, is not shared on the corresponding row between processors, so this Task is assigned to the processor that the tasks dependent on before it in set.
2- The Task, is shared on the corresponding row between processors, then Be dealt with one of the following two cases :
   - Be checked column's Task in the matrix D, and each of sets had the more tasks related to the task, take place the Task in set.
   - If the number of dependent tasks, it was equal in both sets, it add to set that are more transferring cost to its tasks.

In the above example, be checked T_1 lines in the matrix C that are including data transfer Cost to T_3 and T_4. Also, be checked a row of T_2 in the matrix C which is include the Cost data transfer T_4 and T_5. Because tasks T_3 and T_5 are not shared between the tasks set, so are being placed in the sets before its dependent tasks. So, at this stage task's set of P_1 and P_2 as came the following;

$P_1 = \{ T_1, T_3 \}$, $P_2 = \{ T_2, T_5 \}$

Because the task T_4 is common between affiliates of two sets, first, related column in the matrix D be checked, As a result, we see depend previous of T_4 is T_6, T_7 and T_8 and because none of these three tasks is not in the set P_1 and P_2 tasks. Therefore are more the transfer cost to each of the tasks set, is added to set which, according to the matrix C is determined that are more cost of data transmission T_2, so takes place in groups T_2.

$P_1 = \{ T_1, T_3 \}$, $P_2 = \{ T_2, T_5, T_4 \}$

This work will continue until the reach to the end node or end nodes, the nodes is achieved the check of matrix C. Thus end nodal that all elements in the matrix C is zero.

In continuing the above example, we realize of check the set tasks P_1 and P_2, that are affiliates T_1, T_3 tasks T_7, T_6 and affiliates T_2, T_5, T_4 tasks T_8, T_6. First, we assign T_7 to the p_1 and T_8 to the P_2, so at this stage task's set of P_1 and P_2 as came the following:

$P_1 = \{ T_1, T_3, T_7 \}$, $P_2 = \{ T_2, T_5, T_4, T_8 \}$

Because, the task T_6 is common between affiliates of two set, first, related column in the matrix D be checked As a result, we see depend previous of T_4 is to T_4, T_7 and T_3. And because set p_1 are more the number of dependent tasks, so tasks placed in the group of T_3, T_7. so at this stage task's set of P_1 and P_2 as came the following:

$P_1 = \{ T_1, T_3, T_7, T_6 \}$, $P_2 = \{ T_2, T_5, T_4, T_8 \}$
Will be checked at this stage that we have reached the final group or not? These, according to the matrix C, are tasks T_8, T_7. The take placed in P_1 and P_2 set tasks. Finally, we sorted ascending order tasks within the sets.
$P_1 = \{ T_1, T_3, T_6, T_7 \}$, $P_2 = \{ T_2, T_4, T_5, T_8 \}$
Gantt chart related to processes P_1 and P_2, which is as follows, compared to traditional methods of time to reduce the 13 units.

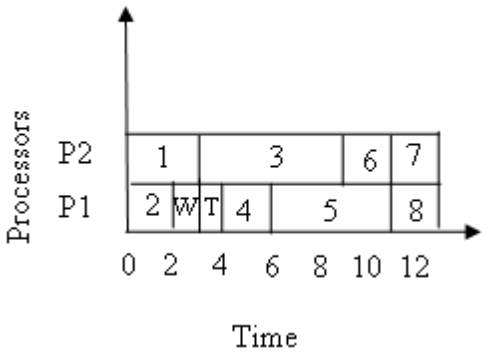

Fig 5: Gantt chart in the new schedule proposed method

5. Conclusions and future work

Optimization approach to scheduling tasks in grid environments, in addition to the time schedule parameters should is considered cost used in the resource scheduling process.
In most scheduling algorithms, the user should be to determine one limitations of time or cost for the scheduler. That this story, although the direction is good, but for the user is not able to provide a suitable area without the knowledge of market conditions, will create a problem.
In this paper, a scheduling algorithm is proposed considering the dependencies between tasks and data transfer cost between tasks in grid environments. The purpose of this algorithm is Coupled optimization of time and costs data transfer between tasks and are assigned select the best sources to the scheduler. To future work in related the proposed method, the issue of dynamic grid environment, such as errors in the allocated resources can add to the desired algorithm.

References

[1] Foster and C. Kesselman (editors), "*The Grid: Blueprint for a Future Computing Infrastructure*", Morgan Kaufmann Publishers, USA, 1999.

[2] Baker M., Buyya R., and Laforenza D., "*The Grid: International Efforts in Global Computing*", Proc. Of International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet, Rome, Italy, 2000.

[3] Ran Zheng, Hai Jin, "*An Integrated Management and Scheduling Scheme for Computational Grid*"

[4] Fangpeng Dong and Selim G. Akl, "*Scheduling Algorithms for Grid Computing: State of the Art and Open Problems*", Technical Report No. 2006-504, January 2006

[5] Tangpongprasit S., Katagiri T., Honda H., Yuba T., "*A Time-To-Live Based Reservation Algorithm on FullyDecentralized Resource Discovery in Grid Computing*"

[6] Rotithor H.G., "*Taxonomy of Dynamic Task Scheduling Schemes in Distributed Computing Systems*", in IEE Proc.on Computer and Digital Techniques, Vol.141, No.1, pp.1-10, January 1994.

[7] J. Blythe, S. Jain, E. Deelman, Y. Gil, K. Vahi, A. Mandal, and K. Kennedy. "*Resource Allocation Strategies for Workflows in Grids"* In IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2005).

[8] A. Mandal, K. Kennedy, C. Koelbel, G. Marin, J. Mellor- Crummey, B. Liu and L. Johnsson. "*Scheduling Strategies for Mapping Application Workflows onto the Grid*." In IEEE International Symposium on High Performance Distributed Computing (HPDC 2005), 2005.

[9] R. Sakellariou and H. Zhao. "*A low-cost rescheduling policy for efficient mapping of orkflows on grid systems".* In Scientific Programming, volume 12(4), pages 253–262, December 2004

[10] M. Wieczorek, R. Prodan and T. Fahringer. "*Scheduling of Scientific Workflows in the ASKALON Grid Environment".* In SIGMOD Record, volume 34(3), September 2005.