Review and Determine Efficient Factors in Grid Computing

Mohammad Sazavar
Shabestar Azad University
sazavar@ymail.com
Paper Reference number: 1812-440

Abstract

In the grid computing collect distributed resources and optimum use of their, particular computational resources and storage resources like cpu and ram to use a specific application, in case of being transparent to applications, can act like a super computer. So for any organization is valuable to have access to a super computer and this is important. One of the most important parameters to achieve this goal is exploration resources. In these article important factors in optimizing a grid computing has been reviewed and determined. This algorithm explores resources and their management via compare network. Otherwise this algorithm gives calculation parameters and via compare network sort this input list and conversion to sorted output list. That algorithm can be decision to send which subworks to which resources. When system finished the subwork, the result should be submitted to the original system, and send information about cpu and ram of system with the prepared result. Original system sorted this information and information of the other systems via compare network and decides to sending subworks.

Keywords: Grid Computing, Optimization Performance, Distributed Systems, Grid Middleware, Compare Network

1. Introduction

Grid computing is a branch of the computations science in which it, computing system can be high operating [1]. The difference between high performance computing and high throughput computing has been illustrated by the Condor project [2]. Grid computing via transparency concept [3] provides resources that are distributed in geographic areas for large computing. Grid computing system is composed of hardware and software platform and divide a work to the several subworks for distribute that inside access within it, is reliable and stable. For instance projects in this field can be cited to datagrid [4], e-etoile [5], globus [6], teragrid [7]. Grid resources are heterogeneous and are common used [8]. So subworks are executed on heterogeneous resources that middleware manages this section [1].

1.1 The problem

Because resource are heterogeneous an efficient algorithm to assign tasks to resources is required [9], Therefore in this article is introduced algorithm that acquires resources

properties and based on those distributes subworks. If one calculates to get a job and represent with "W" then it should be divided to many of subworks and represent by

$$W1, W2, W3\ldots, Wn.$$

A problem that exists in the grid computing, is discover free resources that in this article for discover available resources in grid computing system compare network [10] method is used for optimization.

Where:
W= work of computation
W1, W2, W3… Wn= subworks from W
P= main system that W for distribution and complete is within
P1, P2, P3… Pm= another systems as recourse
CPUi= the remaining amount of processing for i-th CPU (KHz) and i= 1, 2, 3… m
RAMi= the remaining amount of memory for i-th RAM (KB) and i= 1, 2, 3… m
CPU-L= List of CPU values
CPU-LR= List of regular CPU values
RAM-L= List of RAM values
RAM-LR= List of regular RAM values
CN= compare network

To do this P system each time that to receive the result of the subwork and information of cpu and ram of system that has produced this result and now is free. So P has information of one system that complete own subwork and composed of two parts:

1. The remaining amount of processing that now is free
2. The remaining amount of memory that now is free

In each time that one subwork complete by j-th system, then information of process about j-th system are in P system.

1.2 The importance

As regards in grid computing, a work divided to several subworks and afterward are sent divided subworks to distributed systems, system throughput is related to distributed resources. So, obviously System for optimum performance must know the sources of free or busy and according to available information, send divided subworks to the relevant systems based on work load. In this system the following is achievable:

- Grid computing system will have processing power for any system and based on can calculate the whole processing power of grid computing system.
- Grid computing system will have storage power for any system and based on can calculate the whole storage power of grid computing system.

- Distribution of tasks will be based on their work load of subworks, so all the result of tiny subwork or heavy subwork complete in together. So can be implemented subworks that needed to complete to another subworks result.
- System works as coordinate and overall result will be ready soon.

1.3 The claim

A compare network always sort own inputs. Figure 1 shows sample of the compare network for 4 input. In this comparer network parallel lines are inputs and vertical lines show the comparator.

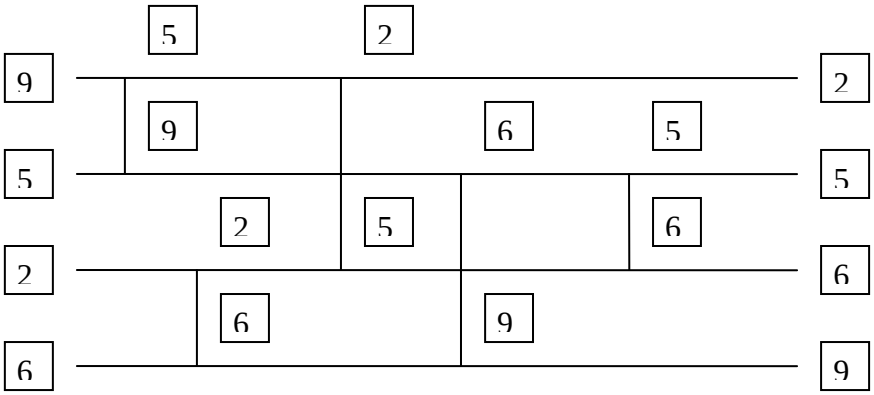
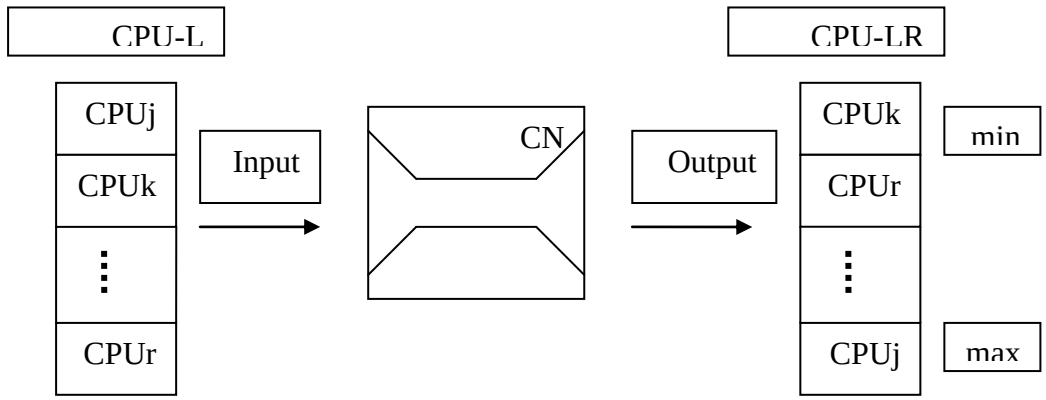


Fig 1: compare network for 4 input.

P system write information about cpu into the list that called CPU-L and write information about ram into the list that called RAM-L. These two lists as inputs are used CN. CPU-L or RAM-L is as input in the CN and CPU-LR or RAM-LR is as output (Fig. 2).



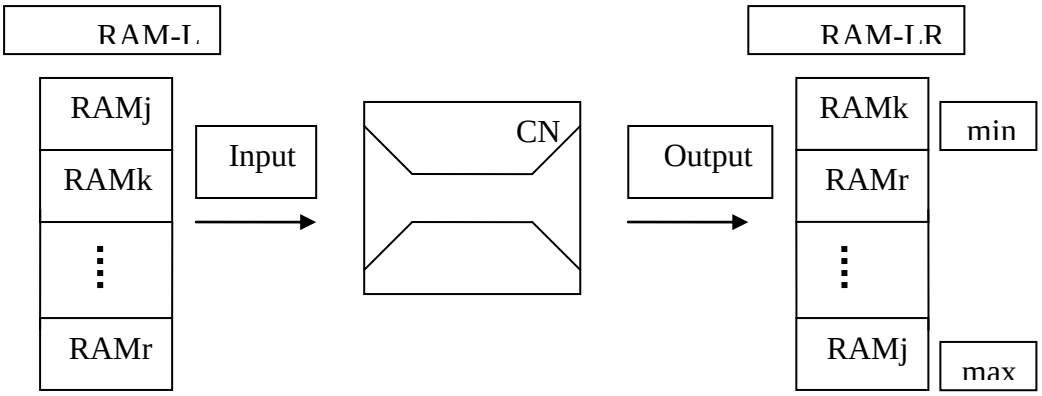


Fig 2: sort cpu and ram values by compare network.

Now algorithm can be divided subworks based on information is arranged in lists and started to send subworks to the relevant resources. So subwork that requires more memory, be sent to the relevant system that ram value is max or subwork that requires more process, be sent to the relevant system that cpu value is max and subworks that is very tiny, be sent to the relevant system that cpu value and ram value is min.
Algorithm in t1 time discovers resources and sent subworks in t1+t2 time. Otherwise t2 is time that be spent to sort cpu values and ram values and detection subworks to send. If n (n is list length) is the compare network inputs, "lg n" is the minimum compare network depth and number of compare is "n lg n".

1.4 Paper outline

The remainder of this article is organized as follows. In section 2 review an article briefly that is previous work in this field. Section 3 describes the new method that use compare network for sorting information. In this section express that how sending method in grid computing network, optimization by compare network. Finally section 4 express results of this method and future works.

2. Previous work

One of the good works done in this background about load balancing in a grid computing is in the "Implementation of Load Balancing Algorithm in a Grid Computing" 2006 Science Publications [9]. In published article, presented an algorithm that based on work be balanced in grid computing.

2.1 Mathematical developments

The mathematical developments are established in [11]:

$$W_i^{(t+1)} = W_i^{(t)} + \sum_{i,j=1..n} \alpha_{ij}(W_j^{(t)} - W_i^{(t)}) + \mu_i^{(t+1)} - k$$

Where:

$W_i^{(t)}$ is load with the node i at the time t, α_{ij} is parameter of exchange between the node I and j, $\mu_i^{(t)}$ is load supported by the node i at the moment t, k is represent the

load realized by a node at the end of iteration. This article tells, if $w_i^{(t)}$ is loaded work in at the time t, $w_i^{(t+1)}$ obtained via $w_i^{(t+1)} = M\, w_i^{(t)}$.

Divination 2.1 M is the matrix of distribution is calculated, while taking as a starting point the genetic algorithms: a node takes a half and diffuses the other on the whole of its neighbours.

$m_{ij} = \alpha_{ij} whre \alpha_{ij} = V/2$ if $i \neq j$ (V is number of neighbours of i **)**

$m_{ij} = 1/2$ if i=j

$m_{ij} = 0$ if i is not connected to j

2.2 Method

The distribution of the load is static. She is made after the system made the collection of the information of loads on all the nodes of the grid, to redistribute then the load. The centralization of the collection of the piece of information is justified by a certain number of advantages namely:

- It allows to avoid the problem of distribution all to all, what thus reduces considerably the traffic in the grid.
- The time of the collection of the piece of information is reduced, because the wait of the answer is dependent only on a node at the same moment.
- Any new node integrating the grid is easily considered in this strategy.

2.3 Experiment

The implementation is made on a cluster of 12 processors of Pentium type IV put rhythm by clocks of 2 Ghz, under Java [12]. This topology (Fig. 3) in star is identical to those used in the local networks. She has the advantage of:

- Mapping of quite the applications client / server.
- Offer an easy parallelism.
- Simplicity of realization with network equipments (hub, concentrator or multiplexer).

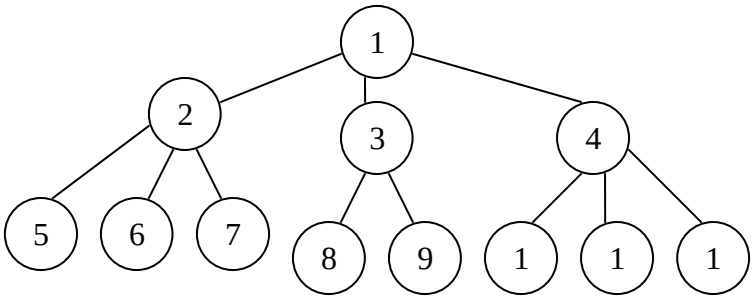


Fig 3: Grid of 12 nodes.

- Algorithm converges in a number limited by iterations that is 23 iterations.
- The convergence is more accelerated in the first 11 iterations than in the last ones.

- It is noticed that the nodes closest to to the node of scheduler are the first to reach balanced load.
- The ten percent residue added to the criterion of stop indeed assures the convergence of the algorithm for 90 % returns.

This algorithm would win more in reliability, by taking care of the weights of communication, so that knots served lastly can benefit from a lowering in charge of work.

3 The new method

The proposed algorithm in this section divides subworks based on available resources that have provided by CN and are present in the lists.

3.1 Explanation

In this method available lists update at the any time, so, algorithm is able to send subworks based on available lists. In this method subworks send to systems and after execute and prepare on the systems be referred to main system. When result be referred to main system, information about cpu and ram referred to main system, too. This information inter in two lists called CPU-L and RAM-L and sort via compare network and conversion to sorted lists called CPU-LR and RAM-LR. So grid system decide based on this and start to sending subworks.

3.2 Algorithm

Algorithm works in 5 phases:

1. First sent random subworks to the available resources. There is an empty list for ready subworks that is equal with list of subworks.

    - Available subwork list= Wa
    - Ready subwork list= Wr

2. In this stage two lists are created dynamically because resources are dynamic and maybe when systems out of network or enter the network. Because subworks are not completed with together, length of lists is determined lesser from total resources until resources that have been released earlier, used to take. Length of lists over form is 2^j that $j \in N$.

    - CPU-L [2^j], $j \in N$
    - RAM-L[2^j], $j \in N$

3. Ready subworks exposure in the ready subwork list. Cpu and ram information exposure in the CPU-L list and RAM-L list. Update subwork list and ready subwork list, if subwork list is empty and ready subwork list is full then execute stage 5. When CPU-L and RAM-L are full execute stage 4. If the results did not send submissions and required to results of other subworks then cpu and ram value did not send submissions to main system, because system not free for other subwork, so wanted results will be sent.

- Update available subwork list
- Update ready subwork list
- Update CPU-L [2^j], $j \in N$
- Update RAM-L[2^j], $j \in N$

4. In this stage CPU-L and RAM-L sorted by CN and determined which subwork corresponds with resource and sent. Continue from stage 2.

- CPU-L [2^j], $j \in N \rightarrow$ CPU-LR [2^j]
- RAM-L[2^j], $j \in N \rightarrow$ RAM-LR [2^j]

5. End of algorithm.

4 Conclusion

When there are sorted information about cpu and ram conditions in any system that is distributed, we can sending subworks based on available conditions in the grid computing network.

4.1 Increase computational bandwidth

The first the importance of this approach is that List of available resources there is in the main system and main system decides according to the available information that which subwork process by which recourse, then increase computational bandwidth. Also if subworks sent this method almost many of subworks complete together or complete in near time that this is important in divide and distribute a work, because most of the computational works, required to the other subworks results. Use from intelligent methods or other network compare methods can optimize this method.

4.2 Future work

Use from types of compare networks in the grid computing system is ascertainable in this field, such as use from bitonic compare network or merging compare network and or permutation compare network. Also we can research about special operating system for grid computing that include compare network.

References

1. Frederic Magoules, Jie Pan, Kiat-An Tan, Abhinit Kumar. (2009). Introduction to grid computing
2. Condor. (2007). Web published in http://www.cs.wisc.edu/condor
3. Tanenbaum, Andrew S. (2007). Distributed systems: principles and paradigms
4. Project datagrid. Web published in http://www.datagrid-international.com
5. Project e-toile. Web published in http://www.urec.cnrs.fr/etoile
6. Project globus. Web published in http://www.globus.org
7. Project teragrid. Web published in http://www.teragrid.org
8. Mansooreh Jalalyazdi, Mohsen Kahani. (2007). Resource Management in a Semantic Mobile e-Learning Grid
9. Abdallah Boukerram, Samira Ait Kaci Azzou. (2006). Journal of applied sciences 3 (4): 1810-1813
10. Cormen Thomas H. (2001). Introduction to algorithms
11. Boilat, J.E. (1990). Load balancing and poisson equation in a graph.

12. Kielman, T, P. Hatcher, L. Bouge and H. Bal. (2003). Enabling java for high performance computing: Exploiting distributed shared memory and remote