Dynamic Scheduling Reduction Algorithm for Distributed Processing Resources through Effective Factors Using

Arash Ghorbannia Delavar † ,Mohsen Nejadkheirallah††, Mohsen Khajeh naeini ††† and Reza Sookhtsaraei ††††

† Computer Science Department, Payam Noor University, Tehran, Iran
†† Allameh Mohaddes Noori Institute of Higher Education Noor, Iran
††† Computer Science Department, Faculty Payam Noor University, Tehran, Iran
†††† Information and Communication Technology Engineering Department,Payame Noor University, 19395-4697 Tehran, I. R. of IRAN

Summary
By incorporating selected parent's chromosomes with the effective parameters, we could present a new dynamic scheduling reduction algorithm for distributed processing resources through effective factors using genetics algorithm. Combining the chromosomes, a competence function is formed which has optimal efficiency compared with the previous samples, for this purpose, first a list of dependent dynamic tasks with the complexity degree of O(nlogn) will be assessed. We need to integrate the fitness function and prioritization function to reduce the time waste of the processing resources while running the tasks list. With the help of proposed algorithm of the time parameters, cost, threshold and integration of chromosomes have been applied in addition to the previous affective parameters. This is done when in several simulation of the DSGR algorithm, efficiency and reliability is increased compared to the previous modes of the technical approach and has created this threshold methods to establish the chromosome integration through a more appropriate method. Finally, through a careful evaluation of the objective function, we managed to have going and returning time in a more favorable conditions than the previous algorithms, and to use less time to choose the Selected chromosomes.
Key words:
Genetic Algorithm; Scheduling; Grid Systems; DSGR.

1. Introduction

In the twentieth century, some case studies has been conducted on the scheduling algorithms whose concern is on the nodes aggregation as priority queue, for this purpose, we must consider the tasks associated with the scheduling algorithms in such a way that waiting queues in the processing resources could be reduced. To achieve the objectives of the distributed systems, we should evaluate the classification in two phases: in terms of software, the first phase is presented by the exact use of the previous algorithms of the new optimal algorithm which takes place with more reliability than the previous modes, the proposed algorithm must be in the similar environmental conditions in order for the obtained output to be optimum compared to the previous mode. But in the second phase, in terms of hardware, the network communication module and the speed of the processor which, considering the above conditions, can reduce the implementation time of the independent processing processes compared with the similar processes, and reduce the time allocated to the plan implementation process using the proposed algorithm and be able to optimize plan implementation faster by the above output algorithm [2].
In this paper, a new algorithm is offered for the optimal scheduling of the tasks on different sources based on the genetic algorithm divided based on the gender of the genes provided and managed to improve the makespan time and increase the grade environment efficiency by mentioning the indices and parameters effective on the previous algorithms on the fitness function and controlling the number of the created generations By the genetic algorithm. In this algorithm, roulette wheel schema is used to create the first generation of the chromosomes and Gridsim tool is used to perform simulations.

2. Literature review

We first reviewed the related tasks on dynamic tasks scheduling in the computing grid. Then, we introduce our unique approach in solving the reconstruction problem and discovering in scheduling the grid tasks.
As many researchers have different opinions in the dynamic scheduling tasks. The most common real function of the tasks scheduling problem is the makespan function. However, in the grid, makespan of a non-optimal scheduler might be much longer than the optimal makespan, because the computing power of the networks changes over time [10].
In the past, task scheduling was mainly proposed for supercomputers and real time and parallel computers in [7], [8], [9], [10].

Manuscript received March 5, 2011
Manuscript revised March 20, 2011

Hwang and Kesselman [10] point out that Grid environment is not inherently reliable by nature. They presented an error discover service and a flexible platform for error control as an error control mechanism on the grade

- In Grid environments, task performance failure for various reasons such as network failure, unexpected resource conditions, or lack of access to the required software components can happen. Therefore, the grid system should be able to identify and control failure and support the reliable performance in the presence of concurrency and failure.
- Application workflow scheduling on the Grid [3], [4], [6] is known as an active field of research. Workflow Scheduling is widely focused on the heuristic techniques using the executive models which chooses the resources and to whom tasks are assigns in order to run well [5], some limited algorithms provide the reliability for the Grid resources.
- Hwang et al. [9] offer error detection services along with flexible task framework to interact with grade errors.
- Limaye et al. Developed checkpoint / restart mechanism in which checkpoint location was determined based on the system reliability.
- Dongarra et al. used the outcome of the error ratio and the integrated interactions of the run-time to conduct scheduling of the independent tasks on the heterogeneous clusters.

3. The new DSGR algorithm

In this algorithm, we have provided a useful method by creating new methods in graphs grading which has been used exactly by tasks grading and the proposed algorithm, to be able to present the proposed algorithm, first, the created weighted grading graphs, which is subordinate to the majority, has been used that these graphs are implemented in three levels. First, each node in the graph is prioritized with the new algorithm by use of which a more optimal scheduling survey can be used than the previous methods.

DSGR index parameters in the 4 Level graphs: considering Figure 1, 18 nodes have been evaluated in the graph that node b have been first used as the first priority with respect to the scheduling priority of the DSGR algorithm, then the node a and b are used. Its weight factors have been correctly used in this figure, and it is a better approach to create the priority function to the DSGR algorithm.

Now by fully evaluating the proposed algorithm which has been created using index parameters, it is introduced as follows.

E_i : The number of the Input Ridges
E_o : The number of the output Ridges
L : level
R : Amount of the required processing power
W : Amount of the sent data between nodes (tasks)

1-1 : the number of the input edges of each node: In the DSGR algorithm, stagnation tasks are considered which considering the parameter , we can create a relationship between the upper layer node and the bottom layer node so that we could have an appropriate prioritization towards the tasks using this parameter and this parameter can establish an influence on the priority function.

1-2 : the number of the output edges of each node: another indicator parameter which has been used in the DSGR algorithm used is the number of output edges which has been studied by graph. This parameter specifies the release rate of the nodes whose tasks has been completed and now are as a selected task to whom is dedicated in the processing resource.

1-3 : L: node level: the number of the graph grading is the parameter of another considerable index. we usually use combined function to avoid occurring an optimum level so that its all states would be examined, if the level is L, 2L mode is created to the priority function so that by using this parameter the second node would not start working until the first level has been completed and also the priority would continue at the next levels.

1-4 : R: the amount of the required processing power: One of the other indicator parameters of the DSGR algorithm is R that the amount of the required processing power is used to run the processing resource. This processing amount has also been used in the competence function, because the processing amount has scheduling effects in the tasks implementation.

1-5 : W: the amount of the data sent between the nodes: the amount of the data sent between the nodes (tasks) is an indicator parameter which has been added in the new proposed algorithms to RQSG algorithm so that the amount of bandwidth between the processing elements could be taken into account. This way, the competence function will be revised through improving the output result and the final result of this function would get closer to the actual value.

Considering the above definitions, the following function is used.

In the DSGR algorithm, sending the primary data to the processing resource and returning the result to the scheduler is carried out through the message passing method and by considering the latency and bandwidth of the network. A data package is used to ensure the data arrival to processing resource and also the arrival of the derived result to data scheduler so that time scheduler would choose another source to run the task in case the data sent to the processing source would not be receipted in a specified time.

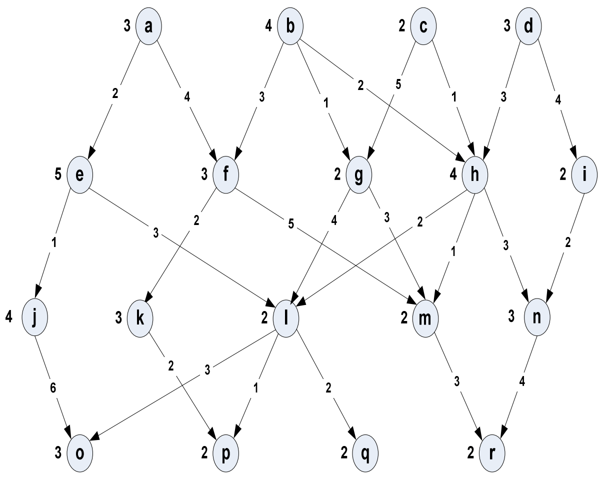


Figure1. Four level Graph with with the proposed DSGR algorithm

1) $$\alpha_j = \frac{E_o}{E_1 + 2^L} \quad j = 1, \dots, n \qquad P_j = R_j * \alpha_j \quad j = 1, \dots, n$$

$$\alpha_a = \frac{2}{0 + 2^1} \qquad P_a = 3 * 1 = 3$$

$$\alpha_b = \frac{3}{0 + 2^1} \qquad P_a = 4 * \frac{3}{2} = 6$$

$$\alpha_c = \frac{2}{0 + 2^1} \qquad P_a = 2 * 1 = 2$$

.
.
.

$$\alpha_r = \frac{0}{2 + 2^4} \qquad P_a = 2 * 0 = 0$$

Implementation order = b , a , d , c , e , h , f , g , l , j , i , k , n , m , o , p , q ,r

The Number of the available processors can be varied over time; tasks are independent from one another and can be processed by different processors in the distributed environment.

Transmission time, including transmission time of the task on the processing source and the returning time of the result to the Scheduler and Certificates Time, are also considered in this algorithm. The bandwidth of every processing resource grid which is inserted to the grade environment is calculated with respect to the scheduler so that the transmission time would be obtained according to this bandwidth. When tasks reach to the grade environment, they locate in the line of the unscheduled tasks. This line is processed as a batch and the tasks are allocated to the appropriate processing resource by the proposed algorithm.

4. Chromosome Presentation

Each chromosome of the population represents a possible solution to the scheduling. Chromosomes in this algorithm are shown as Table A with respect to some parameters affecting on the processing resources:

Table 1 - How to display chromosome in the proposed algorithm DSGR

Node ID	R_1	R_2	R_3	…	R_{n-2}	R_{n-1}	R_n
Number of Task	9	3	5	…	1	4	7
Expected Execution Time	15	20	17	…	32	16	29

This type of chromosome allows us to have the expected amount of the time required to implement any tasks allocated to a processing resource along with that. The first row of Table A represents the identity of the processing source to which the task has been assigned. Second row represents the number of the assigned tasks.

5. Fitness function

We use the fitness function to assess the suitability rate of the selected processing resource to run the desired task from the obtained number. The following parameters are required in order to define this function.

N :The number of the computational nodes in the grid
NTRj : The Number of the processed tasks by processing node with ID=j
CRj : processing capacity of the processor with ID=j
WTi :workload of task i
Wi :Total weight of output edges of node i
aj : calculated Priority in the previous step for node j
Rj :The amount of the required processing power for node j
The size of the primary source sent to the processing resource for task i : Bandwidth between the scheduler and the processing resource with ID = j
P : processing resources failure probability in the grade module

If the time required for data transmission from the scheduler to the processing node of the Dif destination and the bandwidth between them is RBj, it would be calculated by the following function in the proposed algorithm.

2) $$T_j = D_j$$

3) $WT_i = R_j * \alpha_j$

4) $TD = \alpha(P_M) + \beta(P_F) \quad , \quad P_M = 1 - P_F$

5) $EET_{i,j} = \left(\frac{w_i(R_j * \alpha_j)}{CB_j} + \frac{T_i}{RB_j} \right) TD$

6) $TR_j = \sum_{i=1}^{NTR_j} EET_{ij} * (1 + P)$

j=1, 2…, n

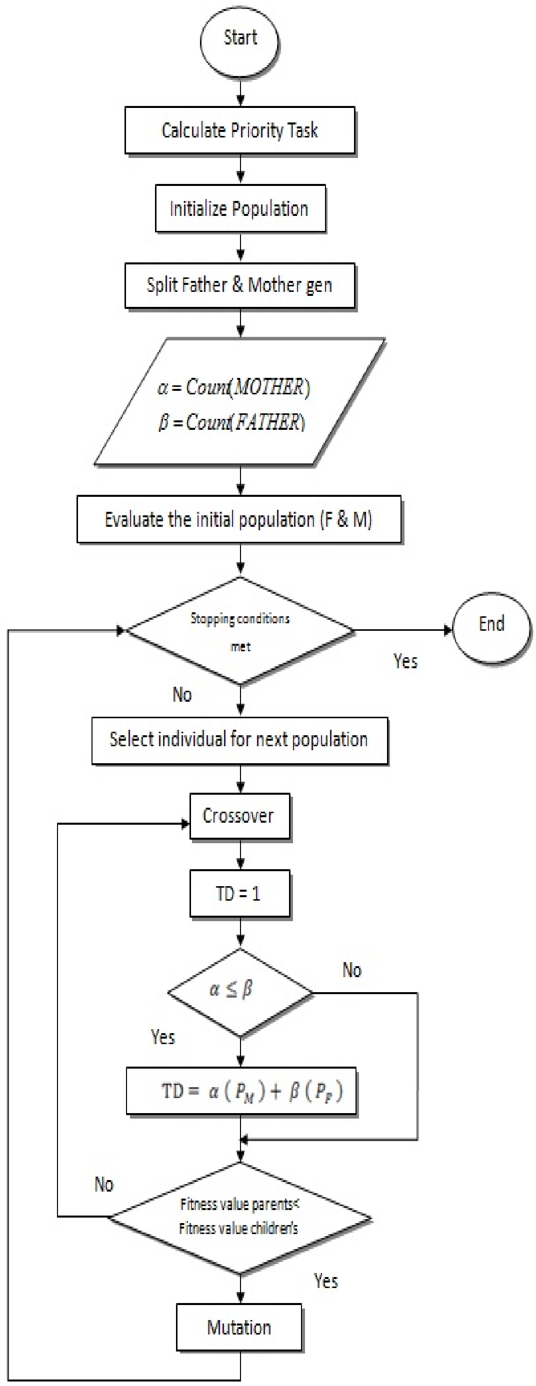
Total time of the tasks is which is dedicated to the processor with the j number and α is the number of maternal genes, and β the number of the Paternal genes, and PM can be the likelihood of the gene being mother and PF the likelihood of the gene being mother and TD is the threshold limit of the genes in the population.

Maximum of represents makespan and the fitness function is also defined as follows.

7) $fitnessValue = \left(\frac{1}{\max(TR_j)_{j \in m}} \right)$

The important point is that we considered a series of parameters which include communication costs (going and returning time of the tasks and the Certificates) and also the processing resources failure probability which is presented according to the DSGR algorithm and have significantly reduced the expected Conditions to the previous algorithms of the total time in running the tasks according to the above parameters.

Foreach (Task t1 in Graph)
calculate formula (1)
Initialize population base on solutions
Split population as MOTEHR and FATHER
α = Count(MOTH ER)
β = Count(FATH ER)
Calculate count and percent age for each gen type as count(type) & P
If (not stopping conditions)
begin
Choose individual for next population
repeat
begin
Calculate crossover on individual
TD=1
If (count(MOTHERS) <=count(FATHERS))
*TD = count(MOTHERS)*P $_M$ + count(FATHERS)*P $_F$*
end
Until (Fitness value parents < Fitness value children's)
Calculate mutation on individual
end



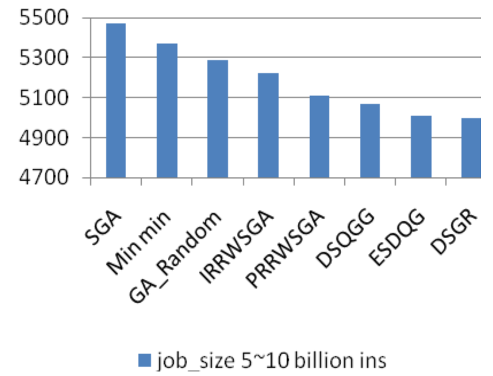
6. Fitness function

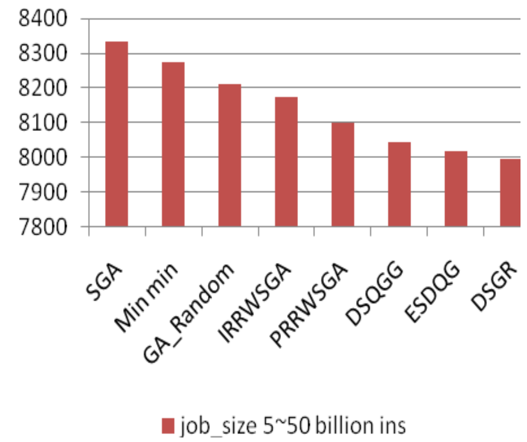
We first reviewed the related tasks on dynamic tasks scheduling in the computing grid. Then, we introduce our

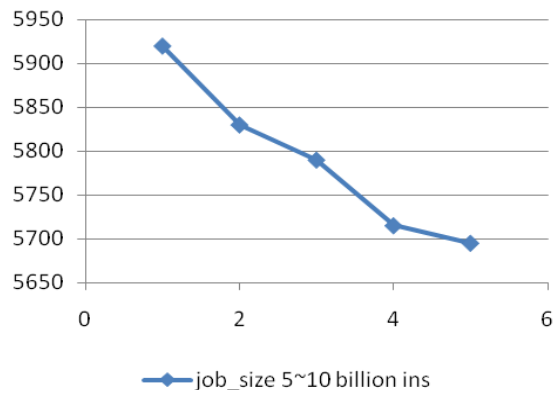
unique approach in solving the reconstruction problem and discovering in scheduling the grid tasks.
As many researchers have different opinions in the dynamic scheduling tasks. The most common real function of the tasks scheduling problem is the makespan function. However, in the grid, makespan of a non-optimal scheduler might be much longer than the optimal makespan, because the computing power of the networks changes over time [10].
In the past, task scheduling was mainly proposed for supercomputers and real time and parallel computers in [7], [8], [9], [10].
Hwang and Kesselman [10] point out that Grid environment is not inherently reliable by nature. They presented an error discover service and a flexible platform for error control as an error control mechanism on the grade
- In Grid environments, task performance failure for various reasons such as network failure, unexpected resource conditions, or lack of access to the required software components can happen. Therefore, the grid system should be able to identify and control failure and support the reliable performance in the presence of concurrency and failure.
- Application workflow scheduling on the Grid [3], [4], [6] is known as an active field of research. Workflow Scheduling is widely focused on the heuristic techniques using the executive models which chooses the resources and to whom tasks are assigns in order to run well [5], some limited algorithms provide the reliability for the Grid resources.
- Hwang et al. [9] offer error detection services along with flexible task framework to interact with grade errors.
- Limaye et al. Developed checkpoint / restart mechanism in which checkpoint location was determined based on the system reliability.
- Dongarra et al. used the outcome of the error ratio and the integrated interactions of the run-time to conduct scheduling of the independent tasks on the heterogeneous clusters.

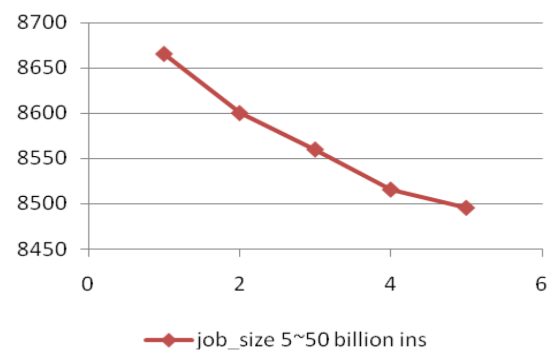
7. Simulation results

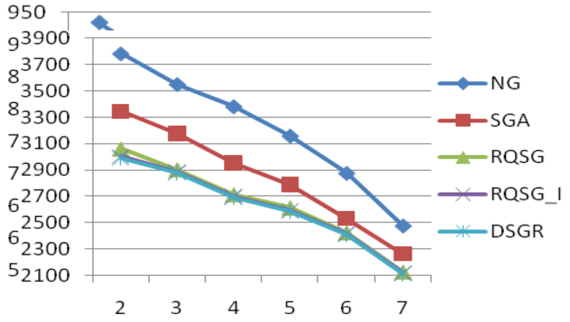
The used tool for simulations is Gridsim. In this paper is tried to select the simulation environment similar to previous studied algorithms as far as possible. were studied a multipurpose simulation with task number=25 & 100 using the index parameter.
The proposed algorithm has had the more optimized the more optimized makspan rather than other two algorithms. Using the index parameters and also priority of static independent tasks by the proposed algorithm has been decreased the scheduling survey rather than the studied algorithms.











8. Conclusion

We could create suitable changes in fitness function using various simulations rather than scheduling common algorithm and also it's optimized algorithm. Then we introduced a method with mixing of fitness function and scheduling queue for having more optimization than RQSG and RQSG_I algorithm. It has been created the better abilities rather than SGA and NG algorithm that will be decreased the going and returning tasks time and certifications according to results from various simulations. Finally, we introduced the tasks prioritizing with a technical method that has been increased 1% rather than RQSG_I algorithm in the worst case and also in the best case has been improved 4%. So the losses time has been an acceptable reduction than similar previous algorithms.

References

[1] V.Aghazarian and A.Ghorbannia Delavar and N.Ghazanfari Motlagh and M.Khajeh naeini, An Optimized Real time Scheduling Algorithm for Tasks Allocation in Grid Environments(ICCSM 2011)

[2] A.Ghorbannia Delavar and Mohsen Khajeh naeini, A new algorithm by index parameters at grid environment with static scheduling, manila, philippines(ICCSM 2010)

[3] A.Ghorbannia Delavar and V. Aghazarian and S.Sadighi, ERPSD: A New Model for Developing Distributed, Secure, and Dependable Organizational Softwares, CSIT'2009)

[4] A.Ghorbannia Delavar and M.Nejadkheirallah and M.Motalleb, A New Scheduling Algorithm for Dynamic Task and Fault Tolerant in Heterogeneous Grid Systems Using Genetic Algorithm, IEEE ICCSIT' 2010)

[5] A.Ghorbannia Delavar and M.Rahmany and M.Nejadkheirallah, GQSD: The new algorithm for optimizing the Quality of service heterogeneous possessing sources at distributed systems, ICSTE 2010)

[6] N. Fujimoto and K. Hagihara, "Near-Optimal Dynamic Task Scheduling of Independent Coarse-Grained Tasks onto a Computational Grid," Proc. Int'l Conf. Parallel Processing, 2003.

[7] K. Krauter, R. Buyya, and M. Maheswaran, "A Taxonomy of Grid Resource Management Systems for Distributed Computing," Software-Practice and Experience, vol. 32, no. 2, pp. 135-164, 2005.

[8] Y.K. Kwok and I. Ahmad, "Link-Constrained Scheduling and Mapping of Tasks and Messages to a Network of Heterogeneous Processors," Cluster Computing, vol. 3, no. 2, pp. 113-124, Sept. 2003

[9] M. Maheswaran, S. Ali, and H.J. Sigel, "Dynamic Mapping and Scheduling of Independent Tasks onto Heterogeneous Computing Systems," J. Parallel and Distributed Computing, pp. 107-131, 2001.

[10] S. Hwang and C. Kesselman, "A Flexible Framework for Fault Tolerance in the Grid," J. Grid Computing, vol. 1, no. 3, pp. 251-272, 2003.

Arash Ghorbannia Delavar received the MSc and Ph.D. degrees in computer engineering from Sciences and Research University, Tehran, IRAN, in 2002 and 2007. He obtained the top student award in Ph.D. course. He is currently an assistant professor in the Department of Computer Science, Payam Noor University, Tehran, IRAN. He is also the Director of Virtual University and Multimedia Training Department of Payam Noor University in IRAN. Dr. Arash Ghorbannia Delavar is currently editor of many computer science journals in IRAN. His research interests are in the areas of computer networks, microprocessors, data mining, Information Technology, and E-Learning.