BGPS :A Scheduling Algorithm for Batch Pipeline Resources using GA

Arash Ghorbannia Delavar
Department of Computer, Payame Noor Universtiy,
PO BOX 19395-3697, Tehran,IRAN

Ayden Halakoui
Payame Noor Universtiy
Tehran,IRAN

ABSTRACT

In this paper, we will present a scheduling algorithm for batch pipeline resources using GA. BGPS algorithms for resource allocation with variable processing power to the jobs is offered. In previous algorithms, a source could select and run one job at a time. New techniques can batch jobs and perform them in parallel. so after the first job is completed, second job is started simultaneously.

The BGPS algorithm can reduce wasting time of sweeping jobs. And we can have the optimal use of existing resources. The new algorithm, in comparison to a similar algorithm with a bunch of jobs, can run them in a pipline. It will be able to request and confirmation of job after the end of previous job, and also allocate resources to perform jobs in parallel. The system is capable to accelerate of processing tasks with different parameters in function and increase system efficiency.

Keywords: Grid Computing, Genetic Algorithm, Distributed System, Scheduling, Batching, pipelining

1. INTRODUCTION

Current scientific problems is very large and complex and require high computational power and storage space [1] - The last methods is not suitable, such as parallel or distributed computing. The next generation of parallel and distributed computing is Grid computing that use Heterogeneous sources to solve various parallel tasks in the economy and engineering science. Grid computing is a hardware and software infrastructure that offers access to high-level computational capabilities to ensure consistent, pervasive and inexpensive [4]. From late 2000 human was thought the idea of parallel processing and integration jobs by resources. Workload distribution is important in distributed systems. Efficient allocation of tasks among processors to optimize the use of resources is essential [2]. The main goal of Grid computing, are favorable access to resources, maximizing their use and minimizing time to run tasks. The grid performance is highly depends on scheduling. A correct and efficient scheduling increase the efficiency of the grid. Dynamics of resource efficiency, due to heterogeneity, autonomy and sharable grid resources are. The goal of grid scheduling problem is the optimal assignment of tasks to resources. Being able to do a proper arrangement for processing, we have to have special circumstances. One of these possible conditions is batching jobs. A pipeline is as a replacement for the series. This is done when batching the jobs is a way to process them in parallelism, especially to raise the processing power sources in distributed systems. With regard to access to resources distributed batch processing systems, the related works are examined.

2. Related work

Scheduling in grid environment is an indeterminate problem. So we should use the inconsistent algorithms to improve the grid scheduling [8]. Discovery methods inspired by natural laws have been suggested. These techniques include genetic algorithms, taboo search, game theory and methods of their combination. The most useful techniques for scheduling problems is genetic algorithms. Here are some of the scheduling algorithms that use genetic algorithms: [7][9][10]

1) GEA algorithm use genetic algorithm search technique to find a near optimal scheduler in grid computing environment.

   In this method, the source number is used as the units of chromosomes and every index, shows the genetic information about the number of processing tasks. The initial population is created with using a random method. So the number of tasks that each source is given is random.

Task completion time depends on bandwidth, size of input and output and also depends on job length and capacity of the resource calculation.

In this algorithm, the chromosome that has the greatest value is chosen for the mating. The single-point mutation is used for moving tasks.

2) IRRWSGA algorithm [6] is a modified genetic algorithm to shorten the search time.

   Task completion time depends on job workload and capacity of the source. Initialization is done using an algorithm called the MCT that heuristically allocate each task to a resource that can quickly finish it. Evolutionary process is depends on applying mutation operators, crossover and roulette wheel selection based on the improved rating of a later generation.

3) DSQGG algorithms [5] is an improved scheduling algorithm that put the new entered job into unscheduled queue in grid environment. The batch queue is processed and jobs are allocated to appropriate resource by this algorithm.

   Each chromosome is comprised of three rows.

   First row, is the processors number, the second row is the number of tasks allocated to source and the last row is expected to perform jobs that assignment to its source.

   In this algorithm, the source number is used as chromosomes design unit.

Initial population is random. So the number of tasks allocated to each source is random. Crossover and mutation is used to modify algorithm. Task completion time depends on the size of input and output job and bandwidth and also to amount of workload of jobs and resource capacity.

By examining the previous work, in order to increase processing speed, and efficient use of resources and reduce processing time, new methods can be used to improve the previous algorithms. In previous work, the time scheduler had no filter in initialized population. Also there was not any policy to eliminate or reduce waste of resources when they were unemployment. The new algorithm has been tried to improve these points.

3. The new algorithm

New algorithms has been studied for distributed heterogeneous environments, all jobs are independent and are performed in completely independed sources. Each processing node that is inserted into the grid environment, the bandwidth of the scheduler with that node calculates until the transfer time of this bandwidth will be achieved and also its processing power is stored. The algorithm assumes that differences between tasks workload is high.

Table 1 Show the initial chromosome. Each chromosome represents a possible solution to schedule. It is displayed as follows:

Table1: showing the chromosomes in offered algorithm

Source node number	P1	P2	P3	…	Pm-2	Pm-1	Pm
Number of allocated task	1	(n-1)/(m-1)	(n-1)/(m-1)	…	(n-1)/(m-1)	(n-1)/(m-1)	(n-1)/(m-1)
expected run time	14	17	24		14	29	16

a) BGPS algorithm

1- Arrange all the resources according to their processing power in descending order and also arrange all of jobs according to their workload in descending order.

2- Check if the differences between the jobs workload is high, go to command 4, else go to command 3.

3- Performe DSQGG algorithm. [5] and then go to command 12.

4- Check if the difference between the first and second task workload is high, go to command 8, else go to command 5.

5- K can be calculated from the following formula:

(1) $(k = n/m)$

Where **m** and **n** respectively are, the number of sources and a number of jobs.

6- Allocate k jobs to m sources in following mode:

$s = s_1, s_2, \ldots s_m$

$j = j_1, j_2, \ldots j_n$

$S_1 \leftarrow j_1, j_n, j_{n-1}, \ldots j_{n-k+1}$

…

$S_m \leftarrow j_m, j_{n-mk+1}, j_{n-mk}, j_{n-mk-1}, \ldots j_{n-2mk+1}$

7- Go to command 12.

8 - Consider a threshold td. td is equivalent to work with the largest workload.

9 - K can be calculated from the following formula: (2) $(k = n-1/m-1)$

Where **m** and **n** respectively are, the number of sources and a number of jobs.

10 - The first work is given to the first source (the biggest source).

11 – Allocate k jobs to the rest sources.

$s = s_1, s_2, \ldots s_m$

$j = j_1, j_2, \ldots j_n$

$S_1 \leftarrow j_1$

$S_2 \leftarrow j_2, j_n, j_{n-1}, j_{n-2}, \ldots j_{n-k+2}$

$S_3 \leftarrow j_3, j_{n-k+1}, j_{n-k}, j_{n-k-1}, \ldots j_{n-2k+2}$

$S_m \leftarrow j_m, j_{n-mk+1}, j_{n-mk}, j_{n-mk-1}, \ldots j_{n-2mk+2}$

12 - Evaluate the fitness function.

13 - The crossover, mutation are performed.

14 - Termination conditions:

a. If no progress.

b. All chromosomes are similar.

c. End time.

Time expected to run the wj job on the sj source is calculated in the proposed following algorithm.

(3) $$\mathbf{Eet:}\ \frac{(w_{jm}+w_{jn-mk+1}+..w_{jn-2mk+2}/p_{sm}) + (s_{jm}+s_{jn-mk+1}+..s_{j\ n-2mk+2}/s_{sm})}{NTS_j}$$

(4) $$TS_j = \sum_{i=1} Eet_{ij}$$

TS_j is total time of WT_j work assigned to processor j.

Makespan is the maximum amount of TS_j function.

The competence is defined as follows:

(5) $$\mathbf{Fitnessvalue = (1/max\{TS_j\})}$$

W_j = workload of task j

P_s = processing power of source s

S_j = the size of job j

S_s = the bandwidth between resource s and scheduler

NTS = number of jobs processed by node j

And also we have the following abstract in flowchart:

n = number of jobs, m = number of sources.

j = job, s = source

$js = j_1 \ldots j_n$

$sc = s_1 \ldots s_m$

j_q , j_v = a sample of jobs that (q=v+1)

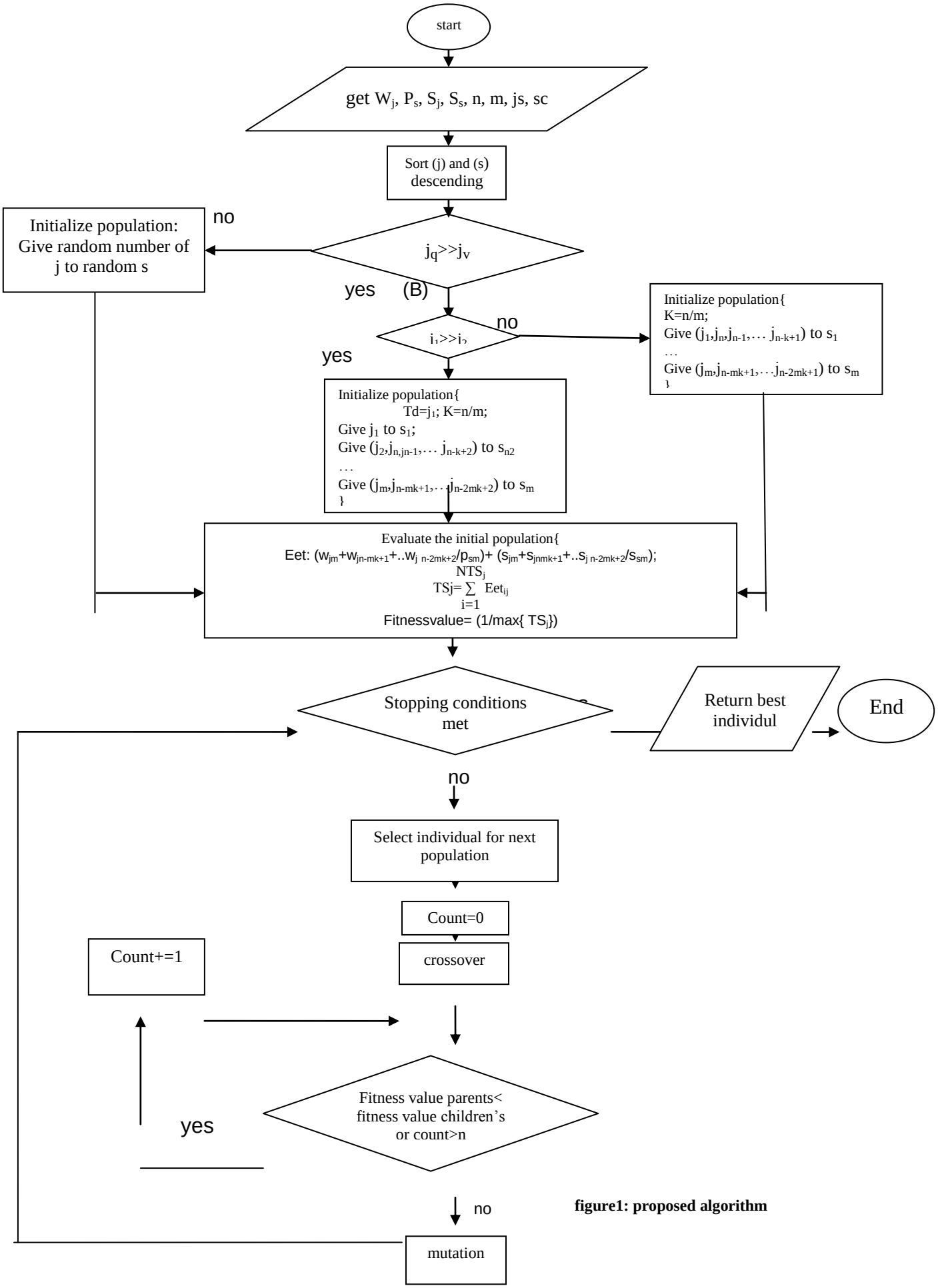


figure1: proposed algorithm

b) The results of the simulation algorithm with other algorithms

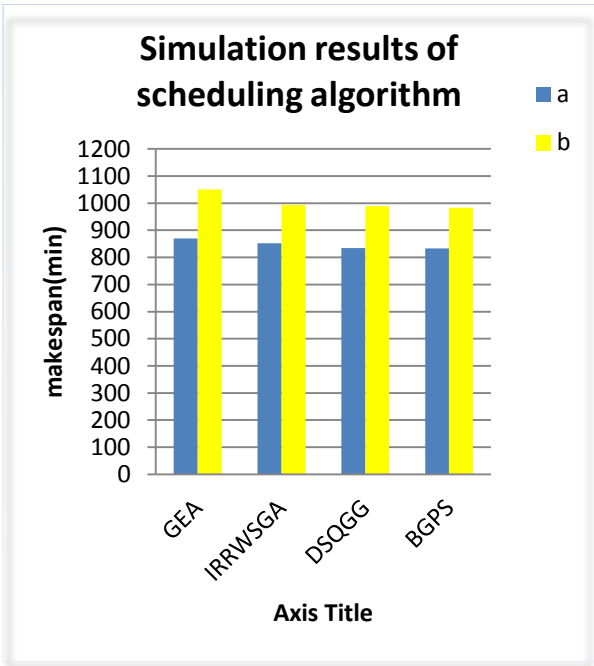
The first example) the parameters used in the simulation are shown in the table below.

Table2: the used parameters of simulating of proposed algorithm

Number of jobs	2000
Number of nodes	40
Job workload	5~15(billion instructions)
Node processing speed	40~100(billion instructions)
Max generation	200
Crossover rate	0.9
Mutation rate	0.01
Network bandwidth	2~8 Mb/sec

Simulation results of the new algorithm with other algorithms are shown in Figure 1:

Figure 1)



a) Billion ins 5~11job size

b) Billion ins 5~15 job size

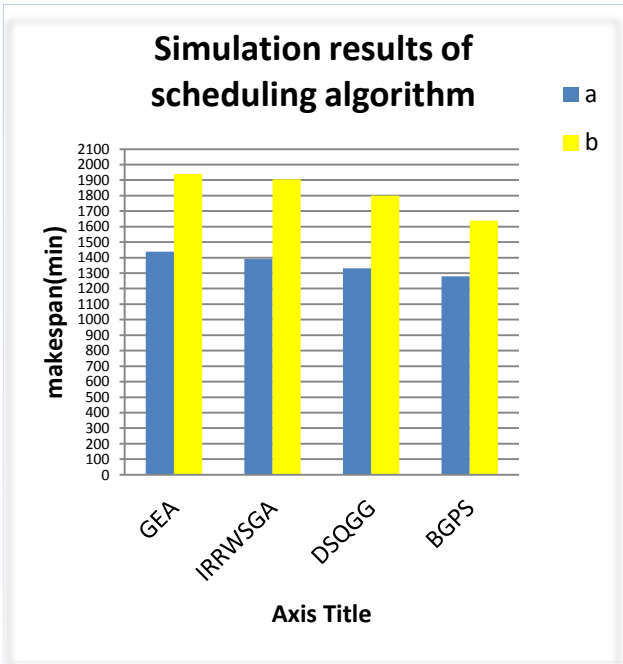
The second example) the parameters used in the simulation are shown in the table below. This time we increased workload of jobs.

Table3: the used parameters of simulating of proposed algorithm

Number of jobs	2000
Number of nodes	40
Job workload	5~30(billion instructions)
Node processing speed	40~100(billion instructions)
Max generation	200
Crossover rate	0.9
Mutation rate	0.01
Network bandwidth	2~8 Mb/sec

Simulation results of the new algorithm with other algorithms are shown in Figure 2:

Figure 2)



a) Billion ins 5~20job size

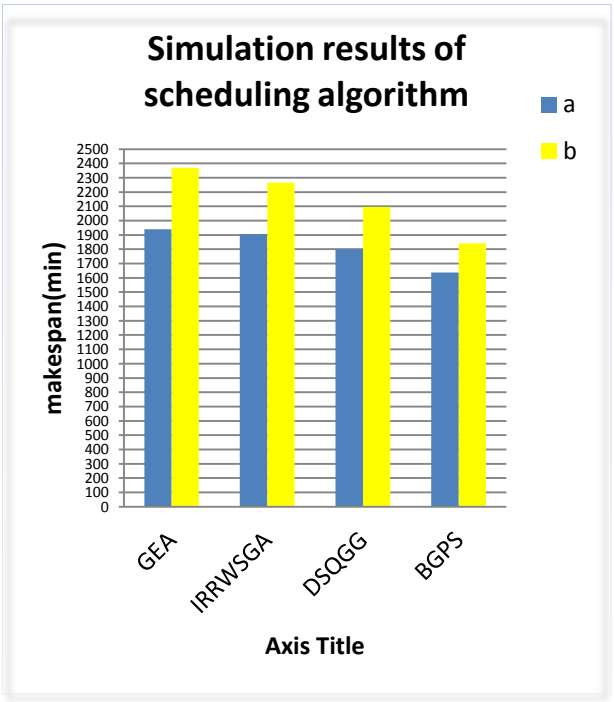
b) Billion ins 5~30 job size

The third example) the parameters used in the simulation are shown in the table below. This time we increased workload of jobs. And also the differences between their workload is high.

Table4: the used parameters of simulating of proposed algorithm

Number of jobs	2000
Number of nodes	40
Job workload	5~50(billion instructions)
Node processing speed	40~100(billion instructions)
Max generation	200
Crossover rate	0.9
Mutation rate	0.01
Network bandwidth	2~8 Mb/sec

Figure 3)



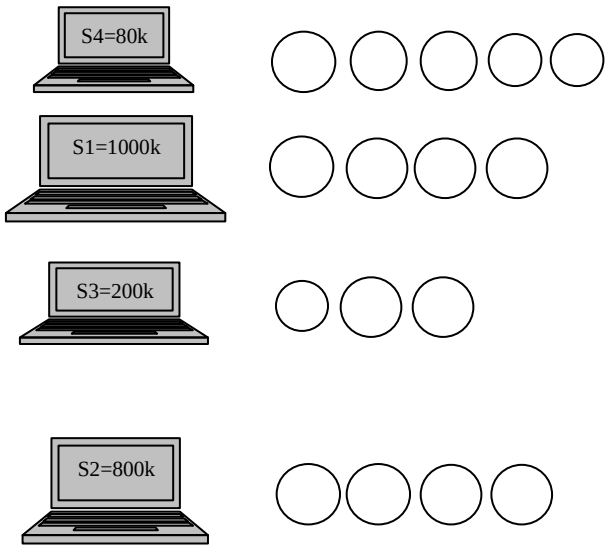
a) Billion ins 5~30job size

b) Billion ins 5~50 job size

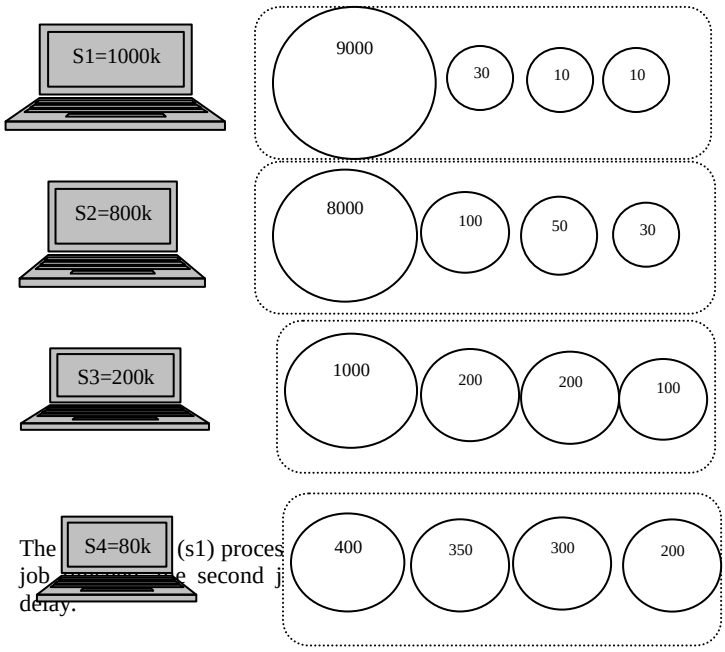
4. Example:

In this section we consider three different modes of the new algorithm. Also predicted result is given. we have 4 sources and 16 jobs. We showed sources in right column and jobs in left column. In each row, one source process one or more jobs. We consider 3 situations. In first example random jobs is allocated to random sources, because the power of resources is near and also workload of jobs is close too. In second and third example jobs allocated to sources in batch mode.

Example1) Jobs workload is close and also the processing power of resources is near. Then DSQGG algorithm is performed. Therefore the initial population is random and jobs are allocated to resources incidentally.

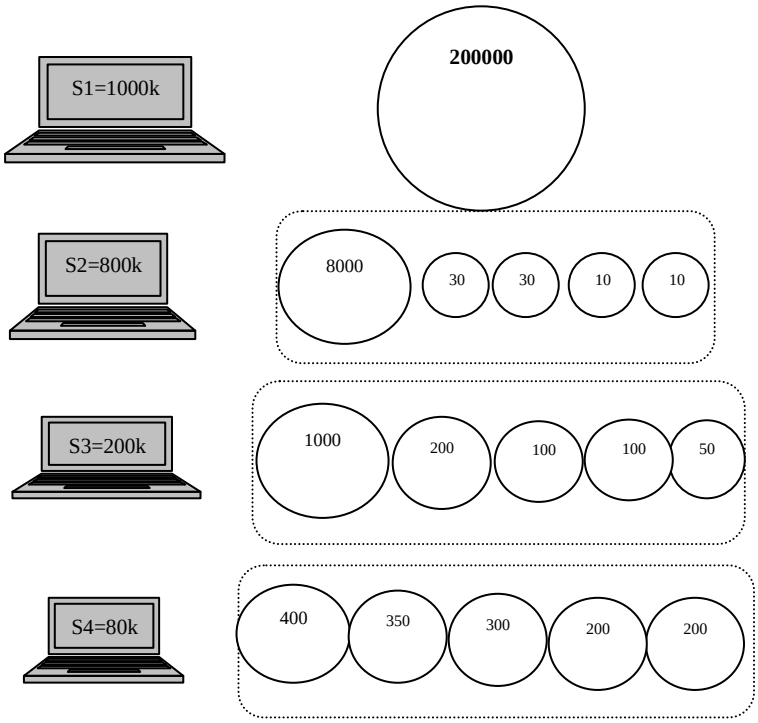


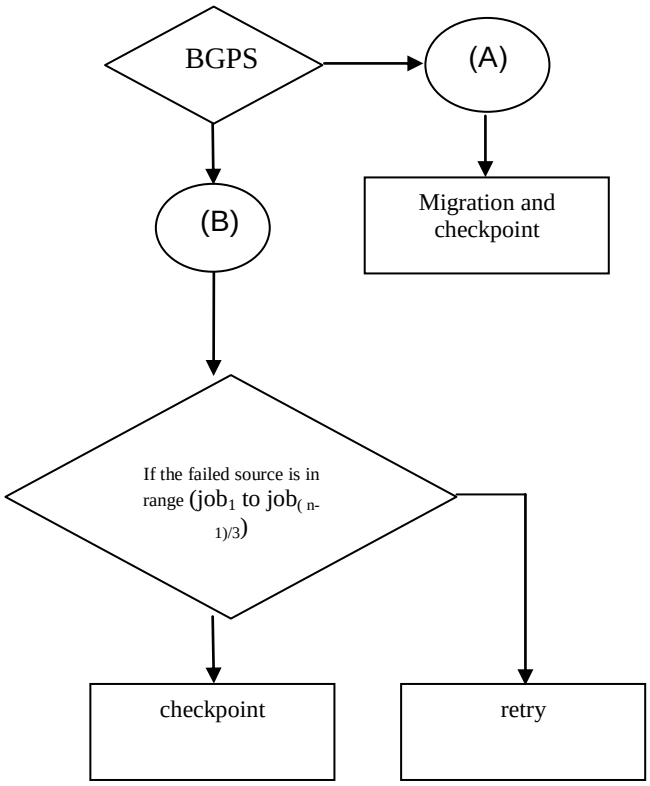
Example 2) Jobs workload and also the processing power of sources are more diverse than sample 1. But job1 is not much larger than job2. Thus, td is not considered. And all jobs are divided equally between the sources. The new method works much better than DSQGG algorithm.



The (s1) proces
job e second j
delay.

Example3) Jobs workload and also the processing power of sources are more diverse than sample 1. And job1 is much larger than job2. Thus, td is considered, and we allocate biggest source to the largest job and the rest of jobs are divided equally between the sources. In this case, in most cases the new method is much better than DSQGG algorithm.





Fault tolerance is not considered in above examples. In the next section, fault tolerant of the algorithm is studied.

5. Error tolerance:

According to the presented examples, fault tolerance in the algorithm will be as follows:

Example 1) In the first case, that samples are selected randomly, if the source is crashed, migration and checkpoint techniques are used in combination.

Examples 2 , 3) In samples 2 and 3, the new algorithm works. Two methods are used in case of failure. If a source with low processing power fails, retry techniques is used. In case of failure of the larger sources, checkpoint is used. The following flowchart shows the system performance in case of error:

6. Conclusions

In this paper we offered a new algorithm BGPS to improve jobs scheduling. The difference between this algorithm and similar algorithm is the initiative in initialized population based on the computation power of resources and workload of tasks that can greatly shorten the optimal final time.

Also, batching jobs and pipelining would reduce the sweep time of jobs. And resources would not stay idle and upon completion of the previous work, new work will begin without delay.

Using the new method, we were able to optimize the algorithm DSQGG up to 9%.

In future work, we will work more on the initial population.

7. REFERENCES

[1] An ant algorithm for balanced job scheduling in grids Ruay-Shiung Chang_, Jih-Sheng Chang, Po-Sheng Lin Department of Computer Science and Information Engineering, National Dong Hwa University, Shoufeng Hualien, 974 Taiwan, ROC

[2] New Self-Scheduling Schemes for Internet-Based Grids of Computers Javier Díaz, Sebastián Reyes, Alfonso Niño, and Camelia Muñoz-Caro

[3] A general model for the generation and scheduling of parameter sweep experiments in Computational Grid Environments- International Conference on Computational Science, ICCS 2010- Javier Díaza,∗, Sebastián Reyesa, Rosa M. Badiab, Alfonso Niñoa, Camelia Muñoz-Caroa

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

[5] DSQGG: An optimized genetic-based algorithm for scheduling in distributed grid Delavar, A.G.; Rahmany, M.; Halaakouie, A.; Sookhtsaraei, R.; Payam Noor Univ., Tehran, Iran 29 November 2010

[6] An improved rank-based genetic algorithm with limited iterations for grid scheduling Abdulal, W. CSE Dept., Osmania Univ., Hyderabad, India Al Jadaan, O. ; Jabas, A. ; Ramachandram, S. Industrial Electronics & Applications, 2009. ISIEA 2009. IEEE Symposium on

[7] On the Design of Fault-Tolerant Scheduling Strategies Using Primary-Backup Approach for Computational Grids with Low Replication Costs Qin Zheng Inst. of High Performance Comput., Agency for Sci., Singapore Veeravalli, B. ; Chen-Khong Tham March 2009

[8] A genetic algorithm for task scheduling in network computing environment Dongmei Liu State Key Lab. of Software Eng., Wuhan Univ., China Yuanxiang Li ; Mingzhao Yu 23-25 Oct. 2002

[9]GABased Job Scheduling Strategies for FaultTolerant Grid Systems Chao-Chin Wu; Kuan-Chou Lai; Ren-Yi Sun Asia-Pacific Services Computing Conference, 2008. APSCC '08. IEEE

[10] Arash Ghorbannia Delavar, Ali Reza Khalili Boroujeni and Javad Bayrampoor. Article: A Balanced Scheduling Algorithm with Fault Tolerant and Task Migration based on Primary Static Mapping (PSM) in Grid. International Journal of Computer Applications52(8):10-21, August 2012. Published by Foundation of Computer Science, New York, USA.