Vol 5. No. 2, March, 2012 ISSN 2006-1781
African Journal of Computing & ICT

www.ajocict.net

Scheduling of Resources for a Grid Computing Application

[1]A.B. Sakpere
Department of Computer Science,
University of Ibadan, Ibadan, Nigeria
ronkeofe@yahoo.com

[2]J.F. Opadiji
[2] Department of Electrical Engineering,
University of Ilorin, Ilorin, Nigeria
jopadiji@unilorin.edu.ng

[3]J.O. Emuoyibofarhe
[3] Department of Computer Science and Engineering,
Ladoke Akintola University of Technology, Ogbomosho, Nigeria
eojustice@justice.com

ABSTRACT
This research work considers a job that consists of three tasks (face recognition, voice recognition and finger print) each that has been submitted for schedule over a set of resources (computer processor) with an objective to minimize tardiness. Genetic Algorithm was used to solve this problem. While obtaining an optimal solution is not always guaranteed, results show that a 'satisficing' solution can always be obtained. It was discovered that increase in iteration provides a higher chance for a better solution with tardiness minimized. Obtaining a quality solution, with increase in number of jobs, is proportional to the number of generations in the scheduling program.

Keywords*:* Scheduling, tardiness, grid application and genetic algorithm.

1. INTRODUCTION

Scheduling is the allocation of resources in order to complete tasks under certain time constraints with the aim to best fulfil specific task processing objectives. The goal of this research work is the determination of a schedule that specifies when and on which machine each task in given job sequences is to be executed with tardiness minimization as the processing objective. It is possible to cluster wide varieties of geographically distributed resources such as computers, storage systems, data sources, and special class of devices and use them as a single unified resource, thus forming what is popularly known as computational grids [4]. Resource sharing is the fundamental of grid computing [1].

African Journal of Computing & ICT Reference Format:

A.B. Sakpere, J.F. Opadiji and E.O. Emuoyibofarhe (2012). Scheduling of Resources for a Grid Computing Application . Afr J. of Comp & ICTs. Vol 5, No. 2. pp 51-. 56

Although various kinds of resources on the grid may be shared and used, they are usually accessed via an executing "application" or "job" [2]. A CPU-intensive grid application can be thought of as many smaller *subjobs* referred to as *tasks,* each executing on a different machine in the grid [2]; the results of all of the jobs must be collected and appropriately assembled to produce the ultimate answer for the application.

In this work Genetic algorithm (GA) is applied to perform the scheduling process. GA is a search technique based on the natural processes of biological evolution (survival-of-the-fittest principle); it searches for best possible solution in a solution space [3].

2. RELATED WORKS

In Abramson and Abela [1], the scheduling for exams was calculated through the use of a genetic algorithm. Because each student has a different schedule and different priorities, it would be very difficult to design a conventional search and constraint algorithm for a computer program. Instead, a genetic algorithm was used in which each chromosome is originally a randomly-created exam timetable.

The fitness function in the computer program simply tests each student's schedule into the timetable and rates the fitness of the timetable through several functions: how many conflicts there are, how many times a student has to take 2 consecutive exams, how many times a student has more than 3 exams in one day, and so on. This fitness function is used to test all the timetable chromosomes, and new generations are created, and eventually the optimized exam schedule is reached.

Also, Hai Zhu et al [7] came up with a comprehensive analysis of heterogeneous grid task scheduling, according to the features of the grid environment and the different needs of task scheduling, a security benefit function and the credibility dynamic evaluation model are constructed. Based on these, a new model of multi-objective constrained Grid task scheduling was set up. Through the subjection degree function, the multi-objective optimization model was transformed into a single-objective optimization model. By adopting the different weights, the different requirements of the users can be met. Through the design of new operators, a genetic algorithm called MUGA to solve the problem was proposed and the convergence of this algorithm was proved. Simulation results demonstrated that the proposed algorithm was better than the compared ones in terms of the length of the task scheduling, security efficiency value, reliability and scheduling costs.

3. PROBLEM FORMULATION

The mathematical model below represents the scheduling problem solved in this research work. Consider a set of n independent distributed computational jobs (in this case, biometric analysis)

$$J : \{j_1, j_2, ..., j_n\} \quad \text{..... (1)}$$

A job j_i consists of a set of r tasks represented by set

$$P : \{p_1, p_2, ..., p_r\} \quad \text{..... (2)}$$

Given a set of y available computational resources

$$M : \{m_1, m_2, ..., m_y\} \quad \text{..... (3)}$$

Each task, p_i belonging to j_i is to be scheduled on the y available computational resources.

The processing speed of the y machines is

$$S : \{s_1, s_2, ..., s_y\} \quad \text{..... (4)}$$

Processing load of each task is defined as

$$W : \{w_1, w_2, ..., w_r\} \quad \text{..... (5)}$$

Each job has represented in the set above is characterized by the following properties:

- Release time (synonymously arrival or ready time) rr_j: The point in time when a job j was submitted for processing.
- Processing time t_{ji}: The time it takes to process job j on machine i. This is calculated by dividing the task size by machine speed.
- Due time dd_j : The point in time at which job j should be completed.
- Tasks/operations p_j: Jobs are further subdivided into different operations/tasks p_1, p_2,…, p_r.

Given a set of N distributed biometric jobs where job j_i has r tasks and y available resources, the problem bothers on to allocate all the tasks in j_i on y resources in order to minimize tardiness in job. Before formulating the problem the following terms are defined:

N = Number of jobs
M = Number of available machines
C_i = Actual completion time of job j_i
C_i^e = Expected completion time or due time of j_i
L_i = Lateness of job j_i
τ_i = Tardiness of job j_i
T = sum of job tardiness J
t_k = Processing time of task p_k
w_k = Task size (load) of task p_k
s_i = processing speed of m_i

This scheduling problem is represented as

$$\min \sum_{i=0}^{n} \tau_i \quad \text{..... (6)}$$

Subject to

$$t_k \geq w_k / s_k \quad \forall\, m_k \in M \quad \text{and} \quad w_k = W \quad \text{..... (7)}$$

Where

$$\tau_i = \max(0, L_i) \quad \text{..... (8)}$$

$$L_i = C_i - C_i^e \quad \text{..... (9)}$$

Equation (6) is the objective function that minimizes the total tardiness of job.

Equation (7) is the machine capacity constraint, i.e. the time it takes a task to process is dependent on the speed of the machine and the size of the task.

4. METHODOLOGY

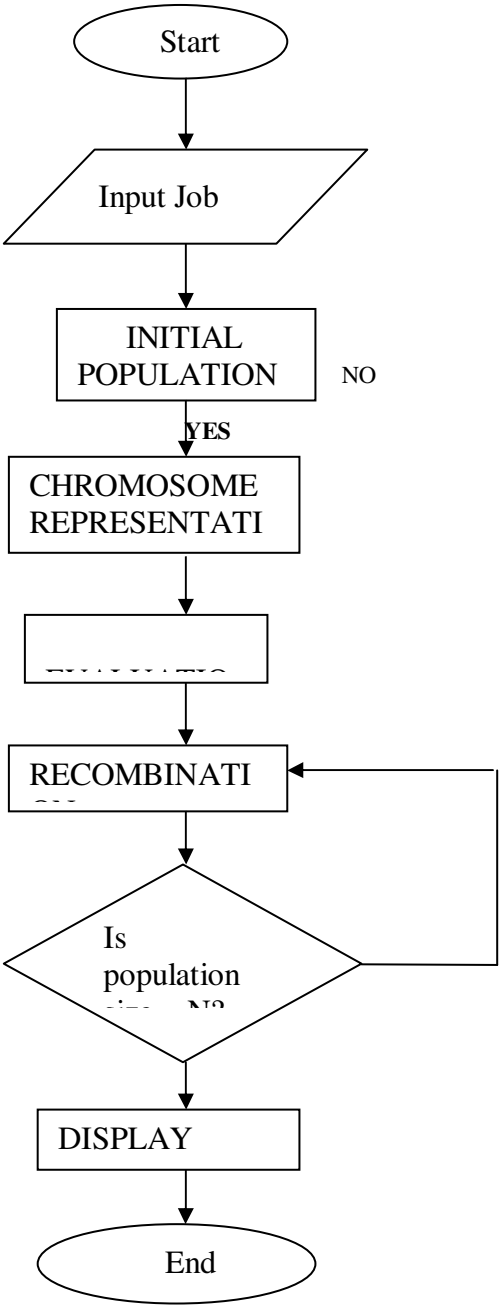


Fig 1: Flowchart representing the Genetic Algorithm used for this Research Work

Genetic Algorithm as described in Forrest [6], Fogel [5] and Haupt & Haupt [8] was applied to this work. Figure 1 shows the flowchart of the process followed in obtaining an acceptable solution to the optimization problem.

(1) **Initial population**: The first population which is our sample or initial feasible schedule was generated based on first come first served (FCFS) i.e. jobs were scheduled based on ascending order of their release date (arrival date). This forms the initial feasible solution.

(ii) **Chromosome representation**: The solut ion is coded into chromosomes. The job ids are used for the chromosome representation.

(iii) **Evaluation**: The tardiness of each chromosome (i.e. job) is calculated using the formula:

$$\max\{(et_{\max} - dd), 0\} \quad \ldots\ldots\ldots\ldots (10)$$

That is, the due date of a job is subtracted from its task maximum highest completion time (i.e. the task that has the highest completion time amongst all the task of a job). Any job that is not tardy has the value 0 as its tardiness otherwise its tardiness is a positive value greater than zero. Fitness function is a measure of quality. Since the objective function is to minimize tardiness, the fitness function is thus based on this and calculated using the formula below:

$$\text{Fitness function: } [\max\{(et_{\max} - dd), 0\}]^{-1} \quad \ldots\ldots (11)$$

That means the fitness value of a chromosome is obtained by finding the inverse of its tardiness. Thus, fitness value is inversely proportional to tardiness. This implies that the lower the tardiness, the higher the fitness value.

Where $et_{\max}$ represents the completion time of a job's task with maximum completion end time dd represents the due date or expected completion time of a job.

The fitness value is calculated as the inverse of tardiness.

(iv) **Recombination**: This leads to generation of subsequent population which evolves from the initial feasible schedule or solution. A subsequent generation is obtained by swapping chromosomes that have higher fitness value with their adjacent chromosomes provided the adjacent chromosome has a lower fitness value. This process is repeated till the termination criterion is met.

(v) **Termination criteria**: Once the termination criterion (defined or stated number of generations) is

reached, the algorithm stops. This was varied from 10 to 1000 generations depending on the number of jobs.

4.1 System Simulation

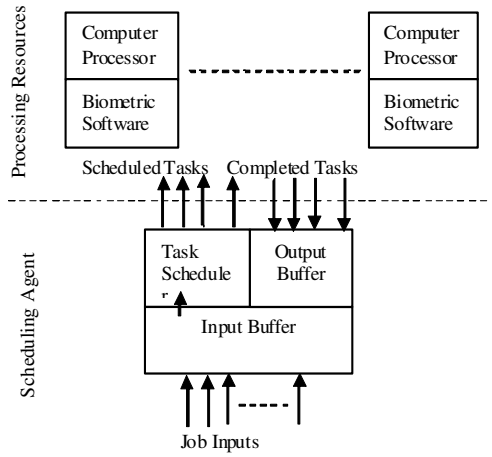


Fig 2: Target System showing scheduling agent and processing resources

Figure 2 shows a pictorial representation of the hypothetical target system used in this work. It is assumed that the operations to be performed by the computer processors in the system are analyses of biometric data for which they have already been equipped. The scheduling agent coordinates the allocation of jobs to machines available for task execution in the grid environment. It determines the machine to process a particular task. The scheduling agent is subdivided into:

- Input Buffer: The input buffer contains jobs that are submitted with information such as their release dates, due dates and the size of each of its task.
- Task Scheduler: This is controlled or directed by a scheduling algorithm based on the concept of genetic algorithm. The scheduling algorithm determines the order of task execution with an objective to minimize tardiness.
- Output Buffer: The output buffer contains the result of processed job.

The job, when submitted, is stored in the input buffer. The task scheduler picks up the job and distributes its task to the underlying machines based on the scheduling algorithm that was developed in this research work.
The machines return the completed task to the output buffer from where the results can be accessed. The two nodes represent the underlying machines that are available for the computation of the jobs. The machines contain biometric algorithm that process submitted task. In this research work, it is assumed that there are three available machines and each machine performs a designated task. It is also assumed that each job is divided into three tasks namely: Face recognition, voice recognition and finger print. Machine one is dedicated for task one (face recognition), machine two is dedicated for task two (voice recognition) and machine three for task three (finger print).

4.2 Hypothetical System

- There are n distributed network of machines where $n \geq 2$
- Machines have different processing speeds
- Each machine has a designated task it performs
- Application/job consists of a set of tasks to be carried out on different machines

4.3 System Implementation

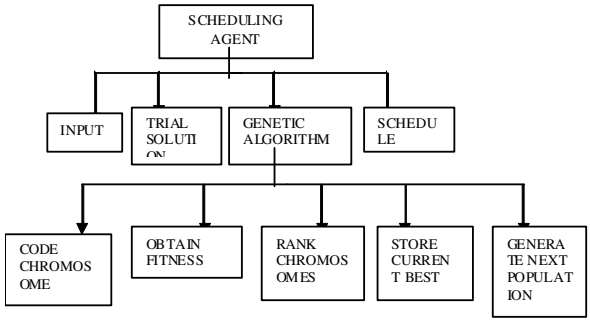
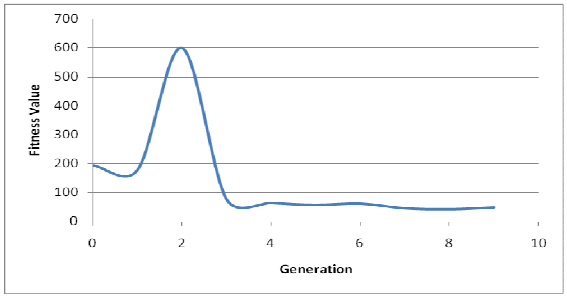


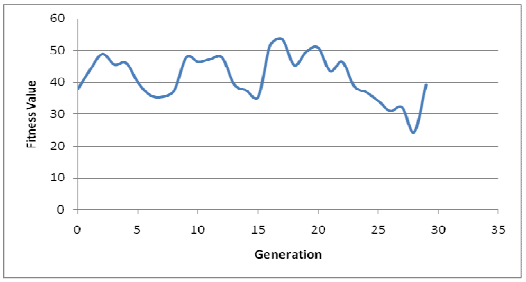
Fig 3: System Architecture

This project work was implemented using C++ programming language and compiled using turbo C++ version 4.5 compiler on a Microsoft Window XP professional version 2002 platform with an Intel centrino-duo processor. The work is divided into different modules in order to ease the implementation. The chart below shows each of these modules:

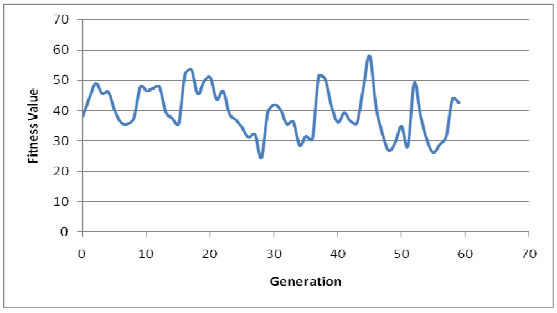
5. RESULT AND DISCUSSION



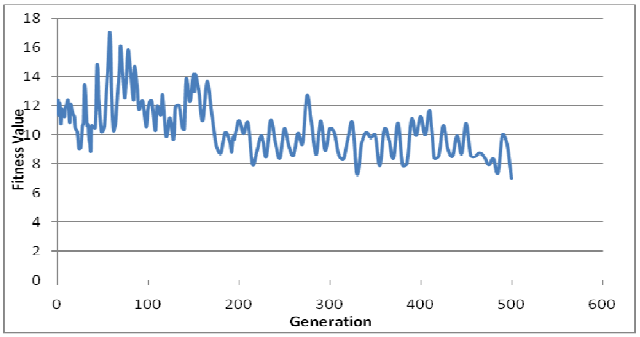
Graph 1: Showing the fitness value for schedule of 5 jobs for 10 generations



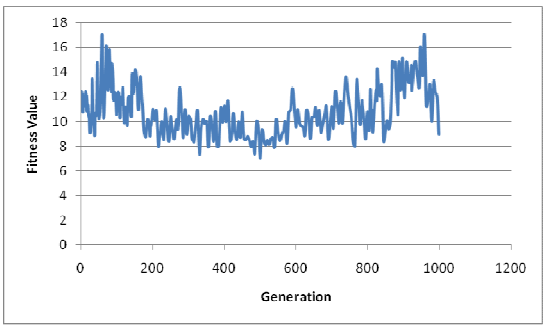
Graph 2: Showing the fitness value for schedule of 10 jobs for 30 generations



Graph 3: Showing the fitness value for schedule of 10 jobs for 60 generations



Graph 4: Showing the fitness value for schedule of 20 jobs for 500 iterations or generations



Graph 5: Showing the fitness value for schedule of 20 jobs for 1000 iterations or generations

Graph 1 represents the schedule for 5 jobs. It had its fitness value as 599.999 at the 3rd iteration out of a total of 10 iterations. Graph 2 and 3 depict the schedule of 10 jobs for 30 and 60 iterations or generations respectively. Graph 2 had its best fitness as 53.5714at the 18th iteration while graph 3 also had its best fitness as 58.1396 at the 46th iteration.

Graph 4 and graph 5 depict the schedule of 20 jobs for 500 and 1000 iterations or generations respectively. Graph 4 had its best fitness as 17.05514 at the 58th generation while graph 5 had its best fitness as 17.0764 at 959th generation. Thus, it was observed that an increase in iteration increases the space to be searched for solution, hence a higher chance of a better result.

In summary, increase in iteration provides a higher chance for a better and improved solution because more space is available for search. Also, it can be clearly seen that as the number of jobs increase, an increase in number of iterations is needed to obtain a quality solution. Therefore, an increase in the number of jobs requires an increase in the number of iterations to improve the quality of solution obtained.

6. CONCLUSION

In this research work, Genetic Algorithm concept was used to schedule jobs over resources with the aim or objective to minimize tardiness. We have presented a simple model in which the performance of machines in a grid setup can be optimized by scheduling different tasks contained in a job to multiple machines in the grid and later reassembling the tasks. While the arrival at optimal solution is not always guaranteed, the genetic algorithm solution method employed is able to provide a solution that both satisfies the constraints of the system and is sufficiently close enough to the objective function (a 'satisficing' solution)

REFERENCES

[1] Abramson D. and Abela J. (1991), A parallel Genetic Algorithm for solving the school timetabling problem, Technical report, Division of Information Technology, C.S.I.R.O.

[2] Alan B. (2002), Grid Computing, EDUCAUSE Center for Applied Research Bulletin, Volume 2002, Issue 17.

[3] Bart J., Michael B., Kentaro F., Nihar T. (2005), IBM Introduction to Grid Computing. (Available at www.ibm.com/redbooks)

[4] Barzan T.A. (2008): "Genetic Algorithm Techniques" (Available at http://www.devarticles.com/c/a/Development-cycles/Genetic-Algorithm- Techniques/, last accessed on 06/11/2008)

[5] Buyya R., Abramson D., and Giddy J. (2000) "Nimrod/G: An Architecture for a Resource Management and Scheduling System in a Global Computational Grid, Proceedings of 4th HPC ASIA'2000, China, IEEE CS Press, USA.

[6] Fogel, D. B. (1998), Evolutionary Computation: The Fossil Record, IEEE Press, New York.

[7] Forrest S. (1993) "Genetic algorithms: principles of natural selection applied to Computation." *Science,* vol.261, p.872-878.

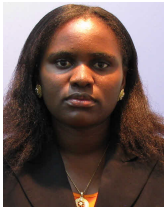
[8] Hai Z., Yuping W., Lei F. And Xiaoli W. (2010) "Grid Independent Task Scheduling Multi-Objective Optimization Model and Genetic Algorithm" Journal of Computers, Vol. 5, No. 12, December 2010

[9] Haupt R. and Haupt S. (1998) "*Practical Genetic Algorithms"*, John Wiley & Sons, 1998.

Authors' Briefs

Emuoyibofarhe O. Justice had his post-doctoral fellowship at the centre of excellence for mobile e-service, University of Zululand, South Africa in 2006. He is an associate professor in the Department of Computer Science and Engineering, Ladoke Akintola University of Technology Ogbomoso.

Dr. Jayeola Opadiji obtained his Doctor of Engineering in Computer and System Engineering from Kobe University, Japan. He is a lecturer in the Department of Electrical and Electronics Engineering, University of Ilorin, Nigeria.



Sakpere Aderonke Busayo is an Assistant Lecturer in the Department of Computer Science at University of Ibadan. She holds an MSc degree in Computer Science frm the University of Ilorin and she is a registered member of Nigeria Computer Society.