ABSTRACT

The term Grid computing originated in the early 1990s for making computer power as accessible as an electric power grid. Grid computing or the use of a computational grid applies the resources of many computers in a network to a single problem at the same time - usually to a scientific or technical problem that requires a great number of computer processing cycles or access to large amounts of data. It requires the use of software that can divide and form pieces of a program as several thousand computers. It can be taken as a distributed and large-scale cluster computing and as a form of network-distributed parallel processing.

The ideas of the grid are brought together by Ian Foster, the so called father of the grid. They created the Globus Toolkit incorporating not just CPU management examples such as cluster management and cycle scavenging but also storage management, security provisioning, data movement, monitoring and a toolkit for developing additional services based on the same infrastructure including agreement negotiation, notification mechanisms, trigger services and information aggregation. In short, the term grid has more implications. While Globus Toolkit remains the standard for building grid solutions, a number of other tools have been built which answer some subset of services needed to create an enterprise grid.
The task scheduling problem is an NP-complete problem. A critical issue for the performance of grid is task scheduling. In order to gain the shortest completion time of the application, task scheduling system assigns all tasks to corresponding resources on the bases of application requirements and resource information. It is one of the core functions in the middle layer of grid systems. Since, there is usually more than one grid resource that meets the needs of a task, the performance and the cost may vary between different grid resources. The time and cost of task scheduling may increase when an inappropriate assignment is chosen which will impact the overall performance. So, it is a NP hard problem to map the wide area distributed resources to user tasks. The task scheduling problem is the problem of assigning n tasks $T=\{t_1, t_2, t_3, \ldots, t_n\}$ to m resources $S=\{s_1, s_2, s_3, \ldots, s_m\}$ in a manner that will optimize the overall performance of the application, while assuring the correctness of the result. Many parallel applications consist of multiple computational components.

A vertex represents a task, its weight and the size of the task computation. An arc represents the communication among two tasks and its weight represents the communication cost. The directed edge shows the dependency between two tasks. One such example is shown in Figure 1.3. The primary goal of task scheduling is to schedule tasks on processors and minimize the make span of the schedule, i.e., the completion time of the last task relative to the start time of the first task. The output of the problem is an assignment of tasks to processors.
Job scheduling is known to be NP-complete, therefore the use of heuristics is the de-facto approach in order to cope in practice with its difficulty. Thus, the meta-heuristics computing research community has already started to examine this problem. The proposed method of scheduling called as residual fair sharing algorithm for scheduling of tasks in a heterogeneous grid environment and the results are compared with the conventional max-min scheduling algorithm. The experimental study revealed that the residual sharing algorithm performs better with respect to max min algorithm especially, there is a noticeable reduction in the time complexity between the two algorithms. The simulations have been conducted by thousands of tasks of varying size and workload variance, submitted to a multiprocessor computing system comprising of hundreds of processors of varying capacity. Experimental results and comparisons of the two scheduling schemes indicate that our proposed Residual Fair Sharing scheduling scheme is fairer and exploit the available multiprocessor Grid resources. The experiments also indicate that the Residual fair sharing algorithm is less sensitive to processor capacity variations than max-min sharing scheme. However, in all conditions, the proposed Residual algorithm is more effective and outperforms all the traditional scheduling algorithms. Studies are under way to apply Heuristic concepts to resolve interdependencies and further optimize the residual fair share scheduling.

The problem of scheduling of tasks to be executed on a multiprocessor system is one of the most challenging problems in parallel
computing. In this research work, DE-VNS with a modified mutation operator for the multiprocessor scheduling problem is used to get a reasonable finishing time or makespan of the schedule. The tasks have precedence relationship and are represented by using Directed Acyclic Graph. The individuals are represented in the form of vectors having a number of cells. The precedence relations of the tasks are considered. The initial population is generated randomly and fitness values are calculated for each individual. The crossover and mutation operations are performed until algorithm converges. The crossover of the tasks also guarantees that the new strings generated are legal. Furthermore, a tabu list is used for avoiding repetition of the parents whose children have been already searched in previous generations. The proposed algorithm will give a reduced finish time or makespan for the schedule of tasks represented by using the DAG with a reasonable execution time.

For scheduling problems, we consider several nature-inspired metaheuristics including evolutionary algorithms, simulated annealing, ant colony optimization and TLBO algorithms. We proposed a novel approach based on fuzzy TLBO. The representations of the position and velocity of the candidates in the conventional TLBO is extended from the real vectors to fuzzy matrices. The proposed approach is to dynamically generate an optimal schedule so as to complete the tasks within a minimum period of time as well as utilizing the resources in an efficient way.