ABSTRACT

Multicore processors have become the defacto standard today and this has led to renewed interest in multiprocessor scheduling. There are many factors that need to be considered while designing scheduling algorithms for multicore processors. These factors include reduction of overheads due to task migration / preemption, effective slack utilization and appropriate handling of task dependencies. This will lead to effective resource utilization and improved task schedulability. This thesis work has proposed four different scheduling algorithms that tackle these issues and aim at effective resource utilization.

Several researchers, as a means of avoiding wastage of resources, have looked at fairness strategies that grant resources to tasks based on their actual requirement. Proportionate fairness is one such strategy that grants resources to tasks based on the minimal resource requirement. However, the proportionate fairness approach is rigid as it does not allow subtasks to be scheduled if they miss their respective slots and keeps the processor idle when subtasks await execution. Further, the proportionate fairness approach also incurs overheads due to migration and preemption. The overheads incurred affect resource allocation and hence task schedulability. To make the proportionate fairness approach more flexible, this thesis work has modified it and has incorporated a migration window to accommodate all subtasks of a
particular task on the same core as far as possible. EDF constraints have been appended to ensure that all critical tasks are scheduled. The execution times of subtasks have been approximately estimated and a worst / best heuristic function has been appended to define the objective of the scheduler. The modified proportionate fairness approach is simulated using Cheddar, a real time scheduler and SESC, an architectural simulator. The simulated results show that resource utilization and task schedulability are improved, compared to the proportionate fairness approach.

The modified proportionate fairness strategy schedules tasks with a small quantum of resource, cumulating the unutilized resource as slack. The cumulated slack is used to schedule tasks that arrive at a later point of time with inadequate resources. The second approach proposed in this work uses a new slack distribution measure, non-uniform laxity, to handle slack effectively. The non-uniform laxity distributes slack based on the minimum execution time, worst case execution time, computation cost and laxity of a task. The simulated results indicate improved resource utilization and task schedulability, compared to existing laxity based approaches.

The third approach proposed in this work handles task dependencies. Dependent tasks are grouped as clusters by constructing a Directed Acyclic Graph (DAG). The clusters are partitioned as coarse and fine grain clusters based on granularity and are scheduled by computing a ranking parameter, which is based on granularity and non-uniform laxity. The fine grain clusters are scheduled followed by the coarse grain clusters. Parent tasks of clusters with high communication cost are scheduled by a selective
duplication strategy. The results show a reduction in communication cost and improvement in resource utilization and task schedulability.

The comparative evaluation carried out indicates that non-uniform laxity based approach outperforms the modified proportionate fairness and non-uniform laxity based ranking heuristic with selective duplication approaches when the independent tasks are more than the dependent tasks. The non-uniform laxity based ranking heuristic with selective duplication approach outperforms the modified proportionate fairness and non-uniform laxity based approaches when the number of dependent tasks is more than the independent tasks.

Further, in situations when the scheduling parameters are not precisely available, they can be specified as linguistic variables. This thesis work has incorporated a fuzzy inference engine that takes in the scheduling parameters as linguistic variables and produces a revised priority based on which a scheduling algorithm is developed. Thus, all the four proposed approaches work towards minimization of wastage of resources and improvement of task schedulability.