Process Scheduling in Heterogeneous Multiprocessor Systems Using Task Duplication

Pranay Chaudhuri, Jaypee University of Information Technology, India

Jeffrey Elcock, University of the West Indies, Barbados

ABSTRACT

Scheduling tasks in heterogeneous parallel and distributed computing environments continues to be a challenging problem. In this article, the authors investigate the Heterogeneous Earliest Finish Time (HEFT) algorithm, along with alternative scheduling policies for task prioritising phases and the Critical Path on a Processor (CPOP) for scheduling tasks on a heterogeneous multiprocessor system. The authors show that by combining the HEFT algorithm selection policy with the task duplication strategy, it is possible to further reduce the schedule length produced by both HEFT and CPOP. The process scheduling algorithm presented in this article compares favourably with other algorithms that use a similar strategy. The proposed algorithm has a time complexity of $O(|V|^2(p+d))$*, where* $|V|$ *represents the number of tasks, p represents the number of processors and d the maximum in-degree of tasks.*

Keywords: *Algorithm Complexity, Multiprocessor Environment, Process Scheduling, Schedule Length, Scheduling Algorithm, Task Graphs*

1. INTRODUCTION

Process scheduling in a multiprocessor environment refers to the assignment of tasks to different processors, satisfying a given set of constraints, so that the schedule length or overall task completion time is minimized. A process (or an application) consists a set of interdependent tasks and may be represented by a directed acyclic graph (DAG), $G = (V, E)$, where the set of nodes *V*, represents the set of tasks, and the set of edges *E*, represents the communication between tasks. If $(n_i, n_j) \in E$, then n_i is called the immediate predecessor of n_j, and n_j is called the immediate successor of n_i. Given a task n_a, its set of immediate predecessors, denoted by $ipred(n_a)$, is defined as $\{n_j | (n_j, n_a) \in E\}$. If n_a has two or more immediate predecessors then n_a is referred to as a join task. The immediate successors of n_a is denoted by $isucc(n_a)$ and is defined as $\{n_j | (n_a, n_j) \in E\}$. $D(n_a)$ is the set of all the descendants of task n_a, and is defined as $\bigcup_{n_j \in isucc(n_a)} (\{n_j\} \cup D(n_j))$.

DOI: 10.4018/jbdcn.2010010104

It is assumed that there is one entry task n_α and one exit task n_ω for the DAG.

In both homogeneous and heterogeneous environments the process scheduling problem is NP-complete (Garey & Johnson, 1979; Graham, Lawler, Lenstra, & Kan, 1979; Kasahara & Narita, 1984; Ullman, 1975). In homogeneous environments several heuristics were proposed using both list scheduling without task duplication (Ahmad & Kwok, 1996; Wu & Gajski, 1990) and list scheduling using task duplication (Ahmad & Kwok, 1998; Bansal, Kumar & Singh, 2003; Chaudhuri & Elcock, 2005; Park & Choe, 2002). In list scheduling, tasks are assigned a priority and based on that priority a task is scheduled. In task duplication, some tasks are deliberately scheduled on more than one processor to reduce inter-processor communication.

A collection of a diverse set of resources interconnected, in most cases, with a high speed network defines as a heterogeneous/grid environment. A similar list scheduling approach has also been taken in such an environment. The Heterogeneous Earliest Finish Time (HEFT) algorithm along with its alternative scheduling policies, and the Critical Path on a Processor (CPOP) algorithm use this approach (Topcuoglu, Hariri, & Wu, 2002). The HEFT algorithm and its variations, firstly assigns a weight to each node based on the average computational cost, and to each edge, based on the average communication cost. Secondly, they assign a rank, using a rank function which takes into account the assigned weights, to each node. Thirdly, the nodes are prioritized on the basis of their rank value. A task is then scheduled to a processor, which returns the earliest completion time. The CPOP algorithm, on the other hand, also assigns a weight to a task based on a rank function, but it first selects the critical path tasks, defined as those tasks with the highest, but equal, weights. The processor which minimizes the cumulative computation costs of the critical path tasks is then selected and identified as the critical-path processor. Only critical path tasks are scheduled on the critical-path processor while the other tasks are scheduled on any of the remaining processors which return the earliest completion time of the task.

The Dynamic Critical Path Duplication (DCPD) algorithm (Liu, Li, Lai, & Wu, 2006), is a duplication-based algorithm which uses a dynamic list-based approach for scheduling in a heterogeneous/grid environment. The DCPD algorithm determines the critical path of the DAG and dynamically selects the next node for scheduling. At each scheduling step the DCPD algorithm identifies the next task for scheduling as the dominant task (*D*tsk). To identify the *D*tsk, the DCPD algorithm first identifies the ready—tasks where their immediate predecessors have already been scheduled. Each task n_a in the ready list is then assigned a value based on three components. The first component is *b-level*(n_a)— the length of the longest path from task n_a to the exit node. The second component is *u-level*(n_a)—the shortest average length from the entry node to n_a; and thirdly, the average computational cost of task n_a. The dominant task, *D*tsk, is the task with the maximal value of *b-level*(n_a) added to the average computational cost of n_a minus *u-level*(n_a). The DCPD algorithm then identifies the dominant processor, *D*pro, as the one which allows the *D*tsk to complete the earliest, utilizing idle slots whenever possible during the identification process. The *D*tsk is then scheduled on the *D*pro.

In this article, we have combined the idea of task duplication with the ranking concept used in the HEFT algorithm and developed an algorithm which outperforms the HEFT algorithm, the six (6) alternative policies for the task prioritizing phase of the HEFT algorithm, the CPOP algorithm and the duplication-based DCPD algorithm. The rest of the article is organized as follows. Section 2 introduces the basis of the algorithm. The proposed algorithm and its complexity are provided in Section 3. In section 4, we have presented an illustrative example. Section 5 provides experimental results and compares these results with the performances of the DCPD algorithm and the

Precoder Design for Cognitive Multiuser Multi-Way Relay Systems Using MSE Criterion
Hua Mu and Jitendra K. Tugnait (2015). *Handbook of Research on Software-Defined and Cognitive Radio Technologies for Dynamic Spectrum Management (pp. 567-583).*
www.igi-global.com/chapter/precoder-design-for-cognitive-multiuser-multi-way-relay-systems-using-mse-criterion/123581?camid=4v1a

Performance and Complexity Evaluation of OTR-UWB Receiver
Hossein Gharaee, Abdolreza Nabavi and Jalil ("Joe") Etminan (2009). *International Journal of Interdisciplinary Telecommunications and Networking (pp. 62-77).*
www.igi-global.com/article/performance-complexity-evaluation-otr-uwb/37205?camid=4v1a