LIST BASED SCHEDULING ALGORITHM FOR HETEROGENEOUS SYSTEM

C. Subramanian¹, N. Rajkumar², S. Karthikeyan³, Vinothkumar⁴

¹Assoc. Professor, Department of Computer Applications, Dr. MGR Educational and Research Institute University, Maduravoyal, Chennai
², ³Asst. Professor, Department of Computer Applications, Dr. MGR Educational and Research Institute University, Maduravoyal, Chennai
⁴MPhil Student, Department of Computer Applications, Dr. MGR Educational and Research Institute University, Maduravoyal, Chennai

ABSTRACT

In this paper, we present a novel list-based scheduling algorithm called Predict Earliest Finish Time (PEFT) for heterogeneous computing systems. The algorithm has the same time complexity as the state-of-the-art algorithm for the same purpose, that is, $O(v^2 \cdot p)$ for $v$ tasks and $p$ processors, but offers significant improvements by introducing a look-ahead feature without increasing the time complexity associated with computation of an optimistic cost table (OCT). The calculated value is an optimistic cost because processor availability is not considered in the computation. Our algorithm is only based on an OCT that is used to rank tasks and for processor selection. The analysis and experiments based on randomly generated graphs with various characteristics and graphs of real-world applications show that the PEFT algorithm outperforms the state-of-the-art list-based algorithms for heterogeneous systems in terms of schedule length ratio, efficiency, and frequency of best results.

1. INTRODUCTION

A heterogeneous system can be defined as a range of different system resources, which can be local or geographically distributed that are utilized to execute computationally intensive applications. The analysis and experiments based on randomly generated graphs with various characteristics and graphs of real-world applications show that the PEFT algorithm outperforms the state-of-the-art list-based algorithms for heterogeneous systems in terms of schedule length ratio, efficiency, and frequency of best results.

We proposed a new list scheduling algorithm with quadratic complexity for heterogeneous systems called PEFT. PEFT is the first algorithm to outperform HEFT while maintaining the same time complexity of $O(v^2 \cdot p)$. The simulations performed for real-world applications also verified that PEFT performed better than the remaining quadratic algorithms. PEFT exhibits the best performance for the static scheduling of DAGs in heterogeneous platforms with quadratic time complexity and the lowest quadratic time complexity.

2. LITERATURE REVIEW

During the past few years, we have witnessed a spectacular growth of parallel computing hardware platforms. This is
because a variety of architectures have emerged exploiting advancements in processors technology, low overhead switches, fast communication channels, and rich interconnection network topologies. As the hardware of parallel processing systems evolves towards achieving the goal of a teraflop performance, the software designers of these systems face increasingly difficult challenges. These include designing new algorithms, programming models, languages, automated programming aids and performance assessment tools. Perhaps, the most crucial component of efficient parallel processing software is the scheduling and allocation of the modules of a parallel program to the processors. This is because the modules of the parallel program must be properly arranged in time and space in order to optimize performance. Given a parallel program represented by a task graph, in which the nodes represent the tasks and the edges represent the communication costs and precedence constraints among the tasks, a scheduling algorithm determines the execution order of tasks and a mapping algorithm determines the allocation of these tasks to processors. The prime objective of scheduling and mapping is to minimize the execution time. This is equivalent to maximization.

This paper addresses the problem of evaluating the schedules produced by list based scheduling algorithms, with metaheuristic algorithms. Task scheduling in heterogeneous systems is a NP-problem, therefore several heuristic approaches were proposed to solve it. These heuristics are categorized into several classes, such as list based, clustering and task duplication scheduling. Here we consider the list scheduling approach. In this paper, I will have an overview on six well known list based scheduling algorithms (HEFT, CPOP, HCPT, HPS, PETS and look ahead) and compare the results of them. A heterogeneous system can be defined as a range of different system resources, which can be local or geographically distributed, utilized to executing computationally intensive application. The efficiency of executing parallel applications on heterogeneous systems critically depends on the methods used to schedule the tasks of a parallel application.

3. METHODOLOGY

EXISTING SYSTEM:
Static DAG scheduling has focused on finding suboptimal solutions to obtain a good solution in an acceptably short time. List scheduling heuristics usually generate...
high-quality schedules at a reasonable cost. In comparison with clustering algorithms, they have lower time complexity, and in comparison with task duplication strategies, their solutions represent a lower processor workload. Many list scheduling algorithms have been developed by researchers. This type of scheduling algorithm has two phases: the prioritizing phase for giving a priority to each task and a processor selection phase for selecting a suitable processor that minimizes the heuristic cost function. If two or more tasks have equal priority, then the tie is resolved by selecting a task randomly.

**DRAWBACKS IN EXISTING SYSTEM:**

- Higher time complexity.
- More processor power used.
- The duplication heuristics produce the shortest make spans.
- The authors compared 20 scheduling heuristics and concluded that on average, for random graphs, HEFT is the best one in terms of robustness and schedule length.

**PROPOSED SYSTEM:**

In proposed we introduce a new list-based scheduling algorithm for a bounded number of heterogeneous processors, called PEFT. This algorithm has two major phases: a task prioritizing phase for computing task priorities, and a processor selection phase for selecting the best processor for executing the current task. We present a new list scheduling algorithm for a bounded number of fully connected heterogeneous processors called Predict Earliest Finish Time (PEFT) that outperforms state-of-the-art algorithms such as HEFT in terms of make span and Efficiency. The time complexity is \(O(n^2):P\), as in HEFT. To our knowledge, this is the first algorithm to outperform HEFT while having the same time complexity. We also introduce one innovation, a look ahead feature, without increasing the time complexity. Other algorithms such as LDCP and Look ahead have this feature but with a cubic and quartic time complexity respectively.

**ADVANTAGES IN PROPOSED SYSTEM:**

- Quadratic time complexity.
- Less processor power used.
- The objective is to minimize the overall completion time or make span.
- The execution of tasks and communications with other processors can be achieved for each processor simultaneously and without contention.

The PEFT has two phases, task prioritizing phase for computing task priorities, and a processor selection phase for selecting the best processor for executing the current task. We observed that the best metaheuristic schedules could not be achieved if we followed the common strategy of selecting processors based only on current task execution time, because the best schedules consider not only the immediate gain in processing time but also the gain in a sequence of tasks.

This project consists of five modules for the task prioritization and processor selection.

- **User Interface Design:**

  The User Interface Design plays an important role for the user to move login the Application. This module has created for the security purpose. In this login page we have to enter user name and password, it will check username and password, if valid means directly go to home page, invalid username or password means show the error message and redirect to registration page. So we are preventing from unauthorized user entering into the
login page to user page. It will provide a good security for our project.

**File Upload:**
In this module Store User data’s like Username, password and ip address and port number and Uploading file Size and finally While sending time how many packet splitted for particular file. This type of data’s are store in the database. and also monitor audit each and every thing for network. Sender how many data’s send through server to particular User.

- **Task Prioritization:**
  This is the module for tasks are categorized in levels such that in each level, the tasks are independent. In the task prioritization phase, priority is computed and assigned to each task using the attributes Average Computation Cost (ACC), Data Transfer Cost (DTC), and the Rank of Predecessor Task (RPT). At each level, the task with the highest rank value receives the highest priority, followed by the task with next highest rank value and so on. A tie is broken by selecting the task with the lower ACC value. As in some of the other task scheduling algorithms, in the processor selection phase, this algorithm selects the processor that gives the minimum EFT value for executing the task. It also uses an insertion-based policy for scheduling a task in an idle slot between two previously scheduled tasks on a given processor.

- **Processor Performance:**
  In this module to select a processor for a task, we compute the Optimistic EFT (OEFT), which sums to EFT the computation time of the longest path to the exit node. In this way, we are looking forward (forecasting) in the processor selection; perhaps we are not selecting the processor that achieves the Earliest Finish Time for the current task, but we expect to achieve a shorter finish time for the tasks in the next steps. The aim is to guarantee that the tasks ahead will finish earlier, which is the purpose of the OCT table.

- **Server monitoring:**
  The server only responsible for collecting user request and send back to the response. Server monitoring the entire network who are all in network and what they are doing. The server maintaining the information about who communicate with whom. The server having the information about sender ip address, port no, source domain, destination ip, port no, destination domain and what data to be shared (file name, file path).

4. **SYSTEM ALGORITHMS:**
**Algorithm: Predict Earliest Finish Time (PEFT)**
1: Compute OCT table and rankoct for all tasks
2: Create Empty list ready-list and put nentry as initial task
3: while ready-list is NOT Empty do
4: ni the task with highest rankoct from ready-list
5: for all processor pj in the processor-set P do
6: Compute EFTðni;iÞ value using insertion-based scheduling policy
7: OEFT ðni;iÞ; pjðØ EFTðni;iÞ; pjð³ OCTðni;iÞ; pjð³
8: end for
9: Assign task \( n_i \) to the processor \( p_j \) that minimize
OEFT of task \( n_i \)
10: Update ready-list
11: end while

5. CONCLUSIONS
We proposed a new list scheduling algorithm with quadratic complexity for heterogeneous systems called PEFT. This algorithm improves the scheduling provided by state-of-the-art quadratic algorithms such as HEFT. To our knowledge, PEFT is the first algorithm to outperform HEFT while maintaining the same time complexity of \( O(n^2 p) \). The algorithm is based on an optimistic cost table that is computed before scheduling. This cost table represents for each pair (task, processor) the minimum processing time of the longest path from the current task to the exit node by assigning the best processors to each of those tasks. The table is optimistic because it does not consider processor availability at a given time. The values stored in the cost table are used in the processor selection phase. Rather than considering only the Earliest Finish Time for the task that is being scheduled, PEFT adds to EFT the processing time stored in the table for the pair (task, processor). All processors are tested, and the one that gives the minimum value is selected. Thus, we introduce the look ahead feature while maintaining quadratic complexity. This feature has been proposed in other algorithms, but all such algorithm increase the complexity to cubic or higher orders.

REFERENCES