Performance Evaluation and Comparison of Scheduling Algorithms for Heterogeneous Multiprocessor Systems

Bao Khoi Bui
Department of Computer Science
University of Houston
Houston, USA
bkbui@cougarnet.uh.edu

Abstract—**Efficient task scheduling is crucial in real-time heterogeneous multiprocessor systems to ensure optimal resource utilization and meet timing constraints. Traditional scheduling algorithms like Earliest Deadline First (EDF) face limitations in handling heterogeneity and overloaded conditions. This paper presents a simulation-based study to evaluate and compare the performance of three scheduling algorithms designed for heterogeneous multiprocessor systems: Dynamic Earliest Deadline First (D_EDF), Heterogeneous Earliest Finish Time (HEFT), and Critical Path on a Processor (CPOP). The study involves simulating diverse task sets with varying characteristics, such as execution times, deadlines, and dependencies, using Visual Studio Code and C++. The algorithms are evaluated and compared based on key performance metrics, including efficiency (the ratio of successfully scheduled tasks to total tasks), load balancing (the variance in processor utilization), and speedup (the ratio of sequential execution time to parallel execution time). The results of the simulations provide insights into the strengths and weaknesses of each algorithm under different task set characteristics. The findings highlight the trade-offs between the algorithms in terms of efficiency, load balancing, and speedup, and their suitability for specific scenarios. The study contributes to a comprehensive understanding of the performance characteristics of D_EDF, HEFT, and CPOP algorithms in heterogeneous multiprocessor environments. The insights gained from this research can guide the selection of appropriate scheduling algorithms for specific applications and inform the development of new or improved scheduling strategies for heterogeneous multiprocessor systems.**

Keywords—Real-time Scheduling, Heterogeneous multiprocessors, D_EDF, HEFT, CPOP, performance evaluation, simulations, efficiency, load balancing, speedup.

I. Introduction

In modern computing systems, real-time and multiprocessor environments are becoming increasingly prevalent, driven by the need for high-performance and efficient processing of complex tasks. Efficient task scheduling is crucial in these systems to ensure optimal resource utilization, meet timing constraints, and achieve desired performance levels. However, the introduction of heterogeneous multiprocessor systems, characterized by processors with varying capabilities and performance levels, poses significant challenges for task scheduling algorithms.

Heterogeneous multiprocessor systems are widely used in various domains, such as multimedia processing, scientific computing, and embedded systems, where tasks with diverse computational requirements need to be executed on processors with different architectures and performance characteristics. Traditional scheduling algorithms, like the Earliest Deadline First (EDF), while optimal for homogeneous systems, face limitations in heterogeneous environments, particularly under overloaded conditions.

To address these challenges, researchers have proposed several scheduling algorithms specifically designed for heterogeneous multiprocessor systems. Three notable algorithms are the Dynamic Earliest Deadline First (D_EDF) [2], the Heterogeneous Earliest Finish Time (HEFT) [1], and the Critical Path on a Processor (CPOP) [1]. These algorithms aim to improve scheduling efficiency, load balancing, and resource utilization by considering factors such as task dependencies, execution times, deadlines, and processor heterogeneity.

Evaluating and comparing the performance of these scheduling algorithms through simulations is crucial to gain a comprehensive understanding of their strengths, weaknesses, and suitability for different scenarios. By simulating diverse task sets with varying characteristics, such as execution times, deadlines, and dependencies, researchers can analyze the efficiency, speedup, and load balancing achieved by each algorithm under different conditions.

II. Related Work

Real-time scheduling on heterogeneous multiprocessor systems has been an active area of research, with numerous studies exploring various scheduling algorithms and their performance characteristics. In this section, we review some of the relevant literature and highlight the contributions of the three algorithms under consideration: D_EDF, HEFT, and CPOP.

The Dynamic Earliest Deadline First (D_EDF) algorithm, proposed by Devendra Thakor and Apurva Shah [2], is an extension of the classic EDF algorithm designed to address the limitations of EDF in overloaded conditions. D_EDF combines the EDF and Deadline Monotonic (DM) algorithms, switching between them based on the number of missed deadlines. This adaptive approach aims to improve the performance of real-time systems in both underloaded and overloaded scenarios.

The Heterogeneous Earliest Finish Time (HEFT) algorithm, talked about by Chandresh Suman and Gaurav Kumar [1], is a list-based scheduling algorithm specifically designed for heterogeneous multiprocessor systems. HEFT considers task dependencies and prioritizes tasks based on their upward and downward ranks, which represent the length of the critical path from the entry and exit tasks, respectively. By assigning tasks to processors that minimize their finish times, HEFT aims to improve overall schedule length and resource utilization.

The Critical Path on a Processor (CPOP) algorithm, also talked about by Suman and Kumar [1], is another list-based scheduling algorithm for heterogeneous multiprocessor systems. Like HEFT, CPOP considers task dependencies and prioritizes tasks based on their upward and downward ranks. However, CPOP also categorizes tasks as critical path or non-critical path tasks and assigns critical path tasks to processors that minimize the overall computation time, aiming to reduce the schedule length further.

Several studies have compared the performance of these algorithms under various scenarios. Suman and Kumar [1] provided a comprehensive analysis of process scheduling algorithms for multiprocessor systems, including a comparative study of scheduling algorithms based on performance metrics such as efficiency, scheduling length, load balancing, and speedup. They highlighted the strengths and limitations of various algorithms, such as HEFT and CPOP, in homogeneous and heterogeneous multiprocessor environments.

Singh and Auluck [3] presented a survey on real-time scheduling on heterogeneous multiprocessor systems, covering various task models, scheduling heuristics, and algorithms. They discussed the contributions of HEFT, CPOP, and other algorithms in addressing quality of service (QoS) parameters, such as guarantee ratio, fault tolerance, reliability, and security.

While these studies provide valuable insights into the performance of scheduling algorithms, there is a need for a comprehensive evaluation and comparison of D_EDF, HEFT, and CPOP algorithms through simulations involving diverse task sets and system configurations. Such an evaluation can contribute to a better understanding of the strengths and limitations of these algorithms, and potentially guide the development of new or improved scheduling strategies for heterogeneous multiprocessor systems.

III. METHODOLOGY

To evaluate and compare the performance of the D_EDF, HEFT, and CPOP scheduling algorithms in a heterogeneous multiprocessor environment, a simulation-based approach was implemented using C++ programming language and the Visual Studio Code (VSCode) integrated development environment.

A. *Task Model*

The task model adopted for the simulations was based on a set of independent, preemptive, and periodic tasks. Each task was characterized by its execution time, deadline, and dependencies on other tasks. To simulate diverse real-world scenarios, multiple task sets were generated with varying characteristics, including:

1) Task Set Sizes: Small (5-10 tasks), medium (10-20 tasks), and large (20-50 tasks).

2) Task Dependencies: Tasks with moderate dependencies.

B. *Simulation Implementation and Evaluation Metrics*

The simulations were implemented in C++ by developing separate functions for the D_EDF, HEFT, and CPOP scheduling algorithms. Task set generation functions were also implemented to create diverse task sets with the desired characteristics mentioned above.

The performance of the algorithms was evaluated based on the following metrics:

1) Efficiency: The ratio of successfully scheduled tasks to the total number of tasks, indicating the algorithm's ability to meet task deadlines and minimize missed deadlines.

2) Load Balancing: The variance in processor utilization across the system, reflecting the algorithm's ability to distribute tasks evenly among heterogeneous processors.

3) Speedup: The ratio of sequential execution time to the parallel execution time, indicating the algorithm's ability to leverage the parallelism of the heterogeneous multiprocessor system.

The results of the simulations were analyzed to identify the strengths and weaknesses of each algorithm under different task set characteristics and system configurations. Additionally, the trade-offs between efficiency, load balancing, and speedup were examined to determine the suitability of each algorithm for specific scenarios and requirements.

IV. RESULTS AND DISCUSSION

In this section, the results of the simulations are presented and discussed. The performance of the D_EDF, HEFT, and CPOP algorithms is evaluated based on the following metrics: efficiency, speedup, and load balancing. The results are represented through graphs, which highlight the impact of various task set characteristics on the performance of the algorithms.

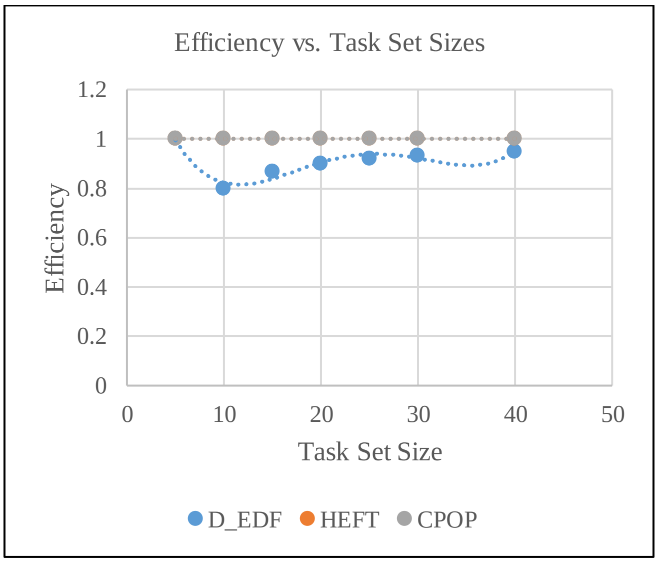

Fig. 1. Efficiency comparisons in increasing task sets with moderate dependencies

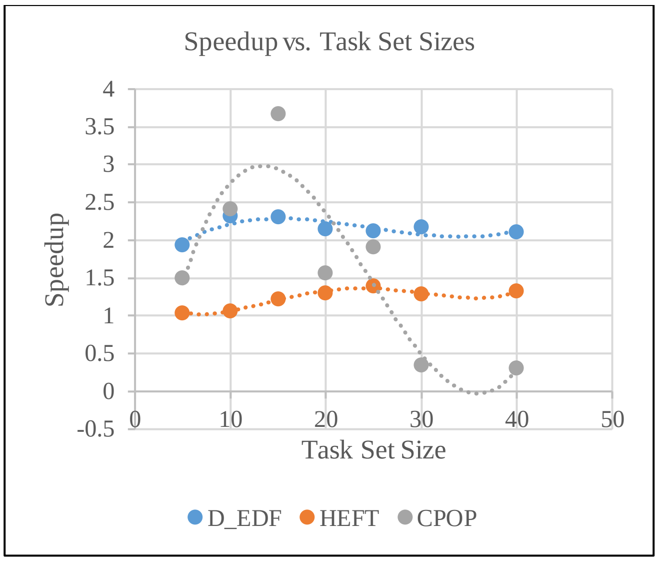

Fig. 2. Speedup comparisons in increasing task sets with moderate dependencies

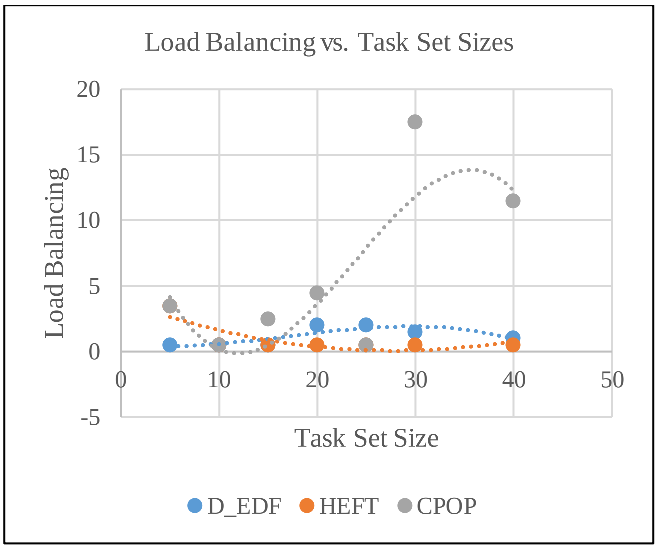

Fig. 3. Load Balancing comparisons in increasing task sets with moderate dependencies

The graphs illustrate the trade-offs and strengths of each algorithm concerning efficiency, speedup, and load balancing when dealing with task sets of increasing size and moderate dependencies.

Efficiency: In terms of efficiency, both HEFT and CPOP consistently maintain an efficiency of 1 across all task set sizes, indicating their ability to successfully schedule all tasks in the given scenarios. On the other hand, the efficiency of D_EDF slightly decreases as the task set size increases, suggesting that it may struggle with larger task sets, particularly under overloaded conditions. This observation aligns with the known limitation of the EDF algorithm in heterogeneous and overloaded environments, which the D_EDF algorithm aims to address by switching to the Deadline Monotonic (DM) scheduling policy when consecutive deadlines are missed.

Speedup: In terms of speedup, CPOP emerges as the clear winner, achieving the highest speedup values across most task set sizes. This can be attributed to CPOP's effective prioritization of critical path tasks, which helps minimize the overall schedule length and leverage the parallelism of the heterogeneous multiprocessor system. D_EDF follows closely behind CPOP in terms of speedup, likely due to its dynamic nature and ability to adapt to changing conditions. HEFT, on the other hand, exhibits the lowest speedup among the three algorithms, possibly because its focus is more on minimizing the overall schedule length rather than maximizing parallelism.

Load Balancing: Regarding load balancing, HEFT consistently achieves a load balancing value of 0, indicating perfect load balancing across most task set sizes. This is likely due to HEFT's inherent design, which aims to distribute tasks evenly among processors while minimizing the overall schedule length. In contrast, CPOP and D_EDF exhibit higher load balancing values, with CPOP showing a more pronounced increase in load imbalance for larger task sets. This suggests that CPOP's prioritization of critical path tasks may lead to an uneven distribution of tasks among processors, particularly as the task set size and dependencies increase.

Overall, the results suggest that CPOP excels in achieving high speedup, making it a suitable choice for scenarios where maximizing parallelism and leveraging the heterogeneous multiprocessor system is crucial. HEFT, on the other hand, shines in maintaining perfect load balancing while also ensuring high efficiency, making it a good option for scenarios where an even distribution of tasks among processors is important. D_EDF strikes a balance between efficiency, speedup, and load balancing, making it a versatile choice, particularly in scenarios where task set characteristics may vary dynamically.

V. Conclusion

Through extensive simulations, we thoroughly evaluated the performance of three popular scheduling algorithms - D_EDF, HEFT, and CPOP - designed for heterogeneous multiprocessor systems. These algorithms were put to the test with diverse task sets, varying in size and dependencies, to analyze their efficiency, speedup, and load balancing capabilities.

The results revealed the inherent strengths and trade-offs of each algorithm. CPOP excelled at maximizing parallelism, making it the go-to choice when you need to harness the full power of a heterogeneous multiprocessor system. On the other hand, HEFT consistently maintained a near perfect load balancing, evenly distributing tasks among processors while ensuring high efficiency. D_EDF proved to be the versatile all-rounder, striking a balanced performance across efficiency, speedup, and load balancing.

These insights shed light on the performance characteristics of these algorithms, helping developers and researchers make informed decisions when selecting the most suitable scheduling strategy for their specific applications and requirements. However, further exploration is still needed to understand the impact of additional factors, such as task execution time variations and varying levels of processor heterogeneity, on these algorithms' performance.

While this study lays a solid foundation, there's always room for improvement. Integrating advanced techniques like machine learning could potentially unlock new levels of adaptability and optimization for scheduling algorithms in heterogeneous multiprocessor environments.

References

[1] C. Suman and G. Kumar, "Analysis of Process Scheduling Algorithm for Multiprocessor System," 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO), Noida, India, 2018, pp. 564-569, doi: 10.1109/ICRITO.2018.8748657.

[2] D. Thakor and A. Shah, "D_EDF: An efficient scheduling algorithm for real-time multiprocessor system," 2011 World Congress on Information and Communication Technologies, Mumbai, India, 2011, pp. 1044-1049, doi: 10.1109/WICT.2011.6141392.

[3] J. Singh and N. Auluck, "Real time scheduling on heterogeneous multiprocessor systems — A survey," 2016 Fourth International Conference on Parallel, Distributed and Grid Computing (PDGC), Waknaghat, India, 2016, pp. 73-78, doi: 10.1109/PDGC.2016.7913118.