UNIVERSITÉ LIBRE DE BRUXELLES

INFO-F-404: Real-Time Operating Systems

Multiprocessor Scheduling

Herma ELEZI
Zexin ZHANG

Prof. Joël Goossens

PhD Student. Yannick Molinghen

2024

Contents

Introduction **1**

Dataset 1

Parallelization **2**

What Can Be Parallelized? 2

What cannot be parallelized 2

Parallelization Choices and Methodology 2

Experimental Results 2

Discussion 3

Schedulability Analysis **4**

Methodology for Determining Schedulability 4

Feasibility Interval 4

Visual Representation 5

Conclusion **6**

Introduction

This report discusses the implementation and analysis of a multiprocessor scheduling project, focusing on parallelization and schedulability analysis. The project includes the Partitioned EDF scheduling algorithm, and experiments were conducted on a dataset of task sets to evaluate the impact of parallelization.

Dataset

The taskset we use is from the original version of taskset "tasksets.zip".

Parallelization

What Can Be Parallelized?

In the context of scheduling task sets, the following aspects can be parallelized:

- Processing multiple task sets simultaneously: Since each task set can be processed independently, the scheduling algorithms can be applied to each task set in parallel.
- Simulation of schedulability over feasibility intervals: Independent simulations for different task sets can also be parallelized.

What cannot be parallelized

- Interdependent computations within a single task set: Some steps in the scheduling algorithm, such as task ordering and core assignments, depend on intermediate results and must be executed sequentially.

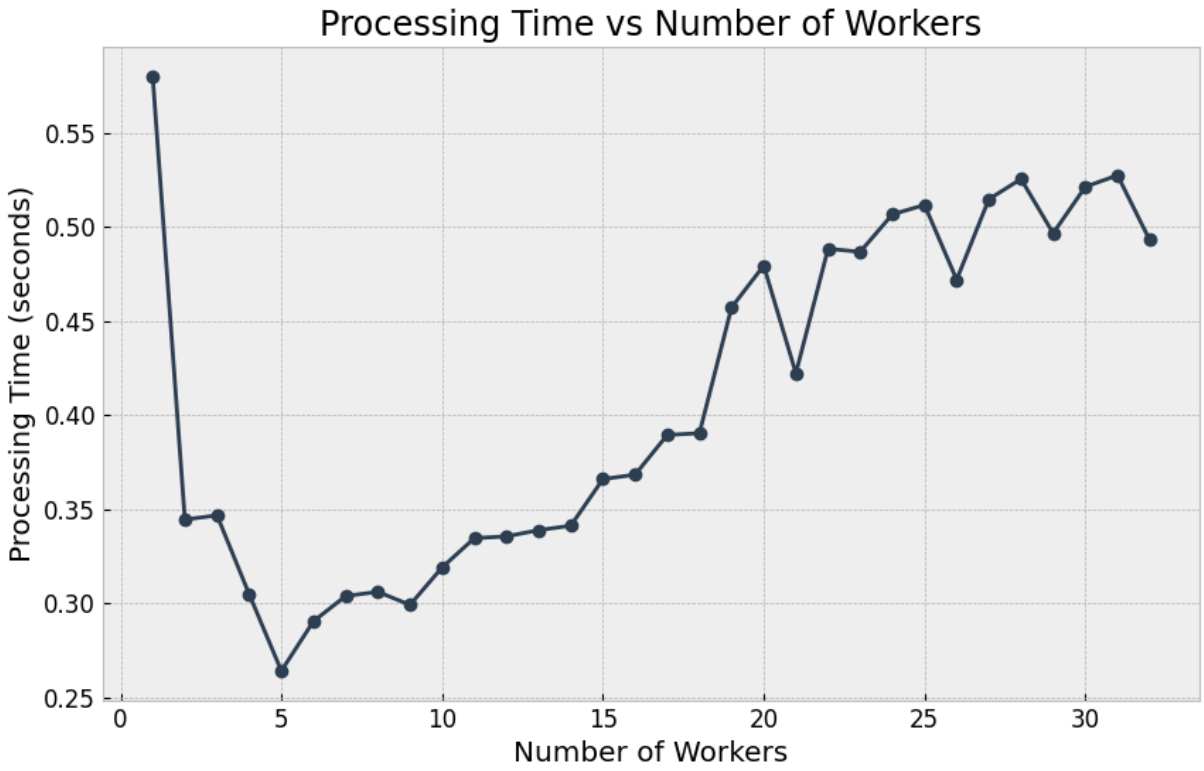
Parallelization Choices and Methodology

We chose to parallelize the processing of multiple task sets, leveraging Python's `multiprocessing` module. This decision was based on:

- The independence of task sets: No interdependencies exist between different task sets, making them ideal candidates for parallelization.
- The availability of multiple cores: By varying the number of workers (parallel processes), we aimed to optimize performance.

Experimental Results

The experiment measured the average execution time of the program with the Partitioned EDF algorithm, using BFDU heuristics and $m = 8$, while varying the number of workers. The dataset consisted of 1000 task sets provided for the project. Below are the results:



Discussion

The figure above demonstrate the following:

- **Speedup due to parallelism**: As the number of workers increases, the total processing time initially decreases significantly, showing effective parallelization.
- **Diminishing returns**: Beyond 5 workers, the processing time stabilizes and even slightly increases in some cases. When the number of worker is too many, it causes the significant increase of the processing time. This behavior aligns with the theory of parallelization, where increased overhead from additional workers eventually outweighs the benefits.

Methodology for Determining Schedulability

The process of determining the schedulability of a task set involves:

1. **Checking Necessary Conditions:** Total utilization $U \leq m$.
2. **Checking Sufficient Conditions:** Utilization bounds based on the chosen algorithm (e.g., Partitioned EDF).
3. **Simulation over Feasibility Interval:** If the conditions above are inconclusive, simulate the task set over its feasibility interval.

Feasibility Interval

For the Global EDF algorithm, the feasibility interval is given by $[O_{\max}, O_{\max} + 2P_{\max}]$, where:

- $O_{\max}$: Maximum offset in the task set.
- $P_{\max}$: Maximum period in the task set.

Visual Representation

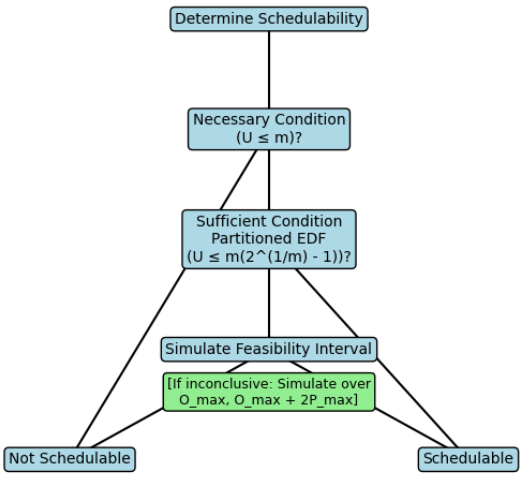


Figure 1: Detailed Schedulability Decision Tree.

Figure 1 illustrates the process of determining schedulability, starting from necessary conditions and proceeding to sufficient conditions and simulation.

Conclusion

The multiprocessor scheduling project demonstrated the effectiveness of parallelization in reducing execution time for processing task sets.

By leveraging independent task set processing and varying the number of workers, significant speedups were achieved. However, the results also showed diminishing returns for higher worker counts, highlighting the overhead of parallelization management. The schedulability analysis process, incorporating feasibility intervals and decision trees, ensured robust evaluation of task sets.

Future work includes exploring additional scheduling algorithms and optimizing parallelization for larger datasets.