International Journal of New Technology and Research (IJNTR)
ISSN:2454-4116, Volume-4, Issue-11, November 2018 Pages 16-18

Performance Evaluation of Extended Latency Time Algorithm in different Linux based Operating Systems

Dipta Gomes, Aneem Al Ahsan Rupai, Mimun Barid, Abu Sufian

***Abstract* — Extended Latency Time (ELT) algorithm is an extension of the Latency time (LT) algorithm. Unlike LT, its extended version allows a system to assign tasks containing arbitrary time into the different processors. In doing so each task is assigned a time frame which decreases as each time unit passes. This report provides detailed information on the performance of ELT on different Linux based operating systems. The algorithm was implemented and the runtime was measured by providing graphs as input, in three different operation systems of Linux which are Ubuntu, Mint and Kali where average execution time in Kali Linux has been the highest which is close to 2.284 time units. From the three Ubuntu showed the most promising result which has shown an execution time of 2.198 time units. After some close observation it was found that the algorithm showed the best performance in Ubuntu.**

***Index Terms*— Extended Latency Time, Scheduling Algorithms, Algorithm Optimization**

I. INTRODUCTION

Shared memory is a way of making intercommunication between processes faster. Instead of communicating through the kernel the process share information in a shared space which is easily accessible by them [10]. As a result the computation time decreases. On the other hand, in a Distributed Shared Memory (DSM) there are multiple process with their own memory spaces. The processors are linked by an interconnecting network. If for any reasons one processor needs to access the other, requests (read/write) is made using the network which is similar to a data bus and responses are also given at the same way. DSM has the advantages of a decreased cost compared to other multiprocessor systems, provides a large virtual space, and it has better portability due to common programming interfaces [1]. Scheduling by means of which multiple processes are given access to memory for execution is very important. However it is very difficult to provide a proper allocation of the computer resources by scheduling [2].Over the past years there have been researches to solve this problem. Different types of scheduling algorithms have been proposed [9] [8]. One of them is ELT algorithm proposed in [3]. The main emphasis of the algorithm was to extend the latency time algorithm proposed in [4]. According to this algorithm any process will be broken down in smaller parts associating each with a time frame that can be updated. The synchronization time of the processes were also taken into account. However the ELT algorithm was not validated for optimization at different operating systems.

In designing a distributed system, a better choice for the intercommunication between the processes must be made. Shared memory can be thought of as a candidate for this matter as it provides a better understanding of the implementation procedure to the programmers. Michael Stumm and Sognian Zhon showed with implemented verifications that distributed shared memory has a competitive performance compared to data passing models which in some cases even out run the later one [5]. However scheduling different processes to run in such an environment is difficult. Martin, and Sanja proposed a framework that allows a decision maker to successively change the definitions of optimality criteria [6]. There have been huge amount of works done in order to address a scheduling algorithm that utilizes the computer resources properly. One such proposed algorithm is ELT and implemented by Irene Zuccar at. el and was verified to give a promising performance using DAG [3]. The algorithm was verified using heuristics, it was not implemented to demonstrate a practical performance.

There are different kinds of scheduling algorithm which operates in a similar way to ELT. One of them is 'List' that prioritize tasks, makes a list of the task and then assign them to available processors. Another one in Insertion Scheduling Heuristic that works like a list algorithm, but at first looks for empty time slots at the processor, and assigns tasks only if an empty slot is found [7]. The ELT algorithm on the other hand has the capability of assigning tasks whenever a processor is sitting idle regardless of whatever task is assigned to it [3]. As a result the computation speed increases.

This paper will provide information on the run time of the ELT algorithm on different operating systems. The algorithm will be implemented using C++ programming language. The reason for choosing C++ because it has a faster performance as the codes are typed checked before execution. In addition, it is a lower- level language, which enables the machine to convert the codes to machine language easily. Linux based

Dipta Gomes, Department of Computer Science, American International University-Bangladesh (AIUB), Dhaka, Bangladesh
Aneem Al Ahsan Rupai,, Department of Computer Science, American International University-Bangladesh (AIUB), Dhaka, Bangladesh
Abu Sufian,, Department of Computer Science, American International University-Bangladesh (AIUB), Dhaka, Bangladesh
Mimun Barid, Department of Computer Science, American International University-Bangladesh (AIUB), Dhaka, Bangladesh