# Boosted System Performance based on Execution Time Using LiMca Scheduling Algorithm

**M. Sreenath[1], Dr. P. A. Vijaya[2]**
[1]Research Scholar, Department of ECE, BNMIT, Bengaluru under VTU, Belagavi & Asst. Professor, Department of ECE, ASCET, Gudur, Nellore, Andhra Pradesh, India
[2]Professor & Head, Department of ECE., BNMIT, Bengaluru, Karnataka, India

## Abstract

The scheduling algorithms have been examined by the process of task execution in a system to achieve maximum utilization of multiprocessors. Consequently, the research attempted to propose a new real time scheduling algorithm to support multiprocessor platform. The proposed scheduling algorithm, List Mcnaughton's amalgamation (LiMca) scheduling algorithm has been developed for an optimum solution with the features of List and Mcnaughton's scheduling algorithms to overcome the individual drawbacks (pre-emption and Precedence constraints). In LiMca, Workload has been distributed to the processors in a system by sorting the tasks in decreasing order with their precedence constraints including due dates, pre-emption, and context switching. In this arena, the LiMca scheduling algorithm has been developed on traditional avionic mission system and also simulated on real-time application. The extensive simulations were carried out on the time optimization of resources scheduling (TORSCHE) simulation toolbox. LiMca scheduling algorithm performed superior as compared to recently reported scheduling algorithms like list, hu's, mcnaughton's, brucker's, and Hodgson's in terms of computational performance.

*Keywords - Avionic system; Multiprocessor; List, Mcnaughton's; LiMca.*

## I Introduction

The Present technology has been changing day to day by providing more options to satisfy the user requirements with better performance in terms of executing number of tasks within a stipulated time [1]. A real-time environment incorporated with the closed-loop system followed quick response with high accuracy [2]. An amalgamation of firmware and hardware is defined as an embedded system to perform a pre-defined task [3]. Allocation of the tasks in a system is a crucial step at the execution of initial stage for an application. The system may leads to get a catastrophe atmosphere, if tasks are not properly allocated and executed [4]. Time constraint is one of the major observation in different types of scheduling tasks. Minimal computation task time is preferable for better performance in a system to speed up the operation [5]. Multiprocessor based system is authoritative to handle the intricate applications like avionics while considering the less execution time and low power consumption.

In multiprocessor based system, most scheduling algorithms are not satisfying the requirements to provide full optimal solution [6,7]. In this manuscript, information about implementation and development of LiMca scheduling algorithm on multi processors based system is present. On other side, this paper guarantees that an optimal solution rests realistic and proximate optimal solution when uncertain handling time changes. The rest of the manuscript is structured as follows: section 2, presents the Existing Scheduling Algorithms and Outcomes; section 3, describes the proposed methodology and implementation of LiMca;

section 4, explains the comparative simulation results on multi-processor based avionics system. Section 5, describes the conclusion and future scope.

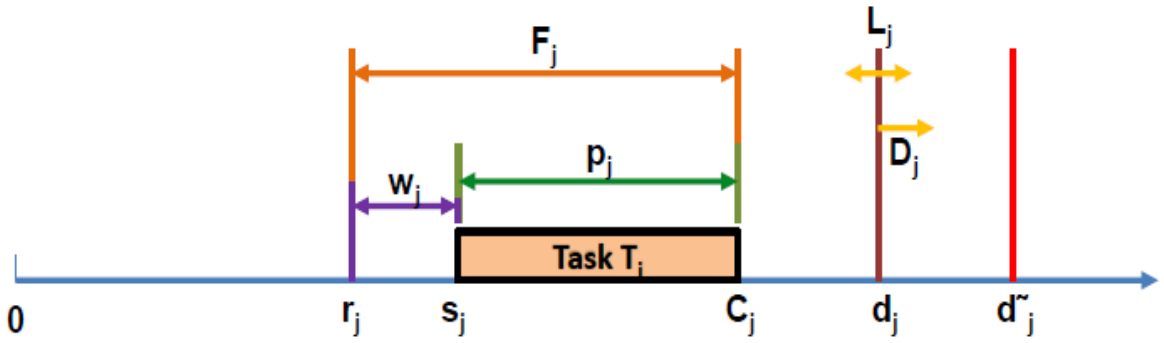



Fig 1.1: Task parameters representation

The Performance of a system has been determined by using the scheduling algorithm with task parameters like start time (sj), processing time ($p_j$), release time ($r_j$), deadline ($d_j$), due date ($d\tilde{}_j$), completion time ($C_j=s_j+p_j$), waiting time ($w_j=s_j-r_j$), flow time ($F_j=C_j-r_j$), lateness ($L_j=C_j+d_j$) and tardiness ($D_j=\max\{C_j-d_j,0\}$) as shown in Fig 1.1 [8]. Where 'j' represents the corresponding task-related information. The number of tasks that have been executed by the system per unit time defined as throughput for finding the efficiency of the system. Time taken by the processor, when it is in a ready state for execution of corresponding to the task called response time. For a given Taskset $T_1$ = [$task_1$ $task_2$ $task_3$], Precedence constraints are defined by the syntax: Taskset (T1,[0 1 1; 0 0 1; 0 0 0]). Adjacency matrix has been used to define a graph in which nodes correspond to the duties and edges represent precedence constraints between duties. If there's an aspect of $T_i$ to $T_j$ in the graph [i,j=1,2,3], it is mentioned that assignment Ti has been finished earlier than $T_j$ as proven in Fig 1.2. The objectives of the Scheduling algorithm have been considered for improvement of results in terms of efficiency in a system like maximum utilization of processor, throughput, and minimum turnaround time (flow time), waiting time, response time, and flow time. The properties of task set are likely task, name of the task, process time, release time, a deadline, and due date. Processing time ($p_j$) relates execution time of a task, release time ($r_j$) is the time at which task is ready for execution, deadline ($d_j$) is a timeline for completion, otherwise the process will fail, due date ($d\tilde{}_j$) specifies a timeline for completion, otherwise, have been charged a penalty [9].

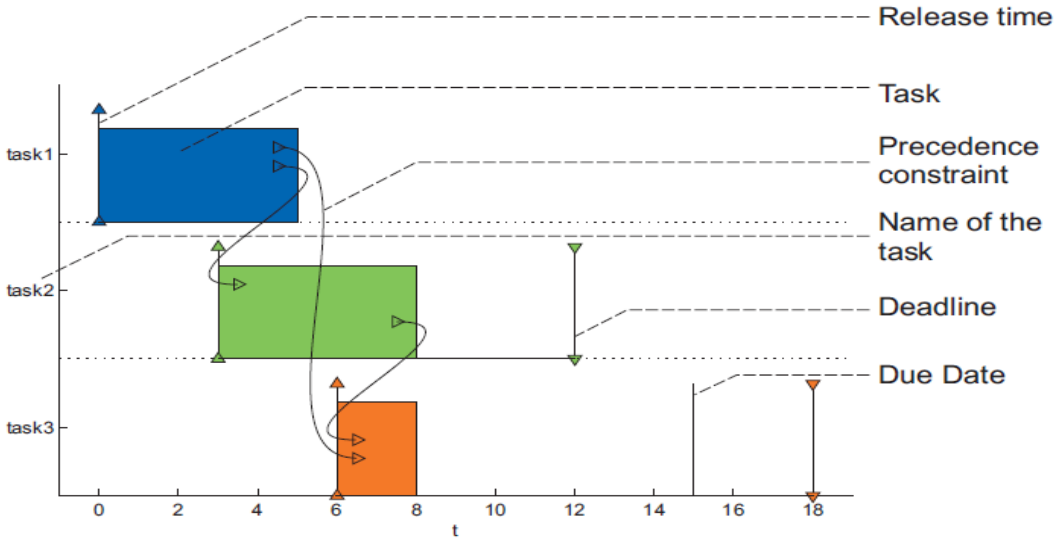



Fig 1.2: Gantt chart for a set of scheduled tasks

## II Existing Scheduling Algorithms and Outcomes

Avionic Mission System (AMS)

Real-time control is essential to meet the requirements for different applications like aircraft, manufacturing, information processing systems, etc. The aeronautics mission system is one of the most real-time embedded systems to sense the various parameters with multiple sensors [10]. The system's functionality will depend on the type and length of the tasks along with their dependencies. The selection of proper scheduling algorithms enhanced the performance by taking fewer clock cycles for the execution of tasks in terms of getting an optimum solution [11].

Table 2.1: Avionic Mission System (Taken from DOI: 10.1016/j.procs.2015.01.001)

| S.No | Name of the tasks | Nature of Tasks | Process time (sec) | Release time (sec) | Deadline time (sec) | Due date (sec) |
|---|---|---|---|---|---|---|
| 1 | Aircraft Flight Data | C | 8 | 0 | 12 | 10 |
| 2 | Steering | C | 6 | 8 | 18 | 16 |
| 3 | Radar Tracking | C | 2 | 14 | 20 | 18 |
| 4 | Poll RWR | C | 2 | 16 | 22 | 20 |
| 5 | Threat Response Display | C | 3 | 18 | 25 | 23 |
| 6 | HUD Display | C | 4 | 21 | 29 | 27 |
| 7 | MPD Display | NC | 2 | 25 | 31 | 29 |
| 8 | Target Tracking | NC | 6 | 27 | 37 | 35 |
| 9 | Target Sweetening | C | 8 | 33 | 45 | 43 |
| 10 | Auto/CCIP Toggle | NC | 1 | 41 | 46 | 44 |
| 11 | Weapon Selection | NC | 2 | 42 | 48 | 46 |
| 12 | Weapon Trajectory | C | 7 | 44 | 55 | 53 |
| 13 | Control Task | NC | 2 | 51 | 57 | 55 |
| 14 | HOTAS Bomb Button | C | 2 | 53 | 59 | 57 |
| 15 | Weapon Release | C | 1 | 55 | 59 | 58 |
| C: Critical Task, NC: Non Critical Task | | | | | | |

Basic tasks, control tasks, monitoring tasks and actuating tasks are involved in this application. The avionic mission system is taken as a reference application for verifying the behavior of the proposed scheduling algorithm with comparative analysis. Avionics mission system contains a 15 number of critical and non-critical tasks. A task set with attributes of an avionic mission system as shown in Table 2.1. Criticalities of tasks have supported the consequences in a system based on the needs. Thirty three seconds taken by using Enhanced Fault Tolerant Scheme (EDFS) on two processors as compared to the dual redundant system (DRS) [10]. In a List scheduling algorithm, tasks arranged based on different strategies like List, earliest completion time (ECT), earliest starting time (EST), shortest processing time (SPT), and longest processing time (LPT) patterns. Fifty six seconds taken by ECT & EST scheduling

strategy. The total tasks are arranged on two processors based on the list and LPT scheduling. Precedence constraints have been considered in precedence order after arranging the tasks on processors based on processing time [11]. Thirty three seconds have been taken by list, LPT, and SPT scheduling strategy [12]. Mcnaughton's schedule have been solved the disadvantages over a necessary to plan a group of freelance tasks on the same processors to reduce the schedule length. This rule provides task preemption with the schedule in order to get a good optimum solution. The Mcnaughton's algorithm have been scheduled the independent task set on identical processors for minimizing the length of the schedule [13]. Mcnaughton's algorithm gives improved results than the list scheduling strategy in terms of execution time (Makespan) (by taking 28 seconds). This algorithm provides improved results in terms of execution time and a utilization factor than the List, ECT, EST, LST, SPT scheduling algorithms. Utilization problems have been solved by minimizing the number of delayed tasks in a system with a single processor by applying EDD (Earliest Due Date First) rule on the task set T by Hodgson's scheduling algorithm [14]. Scheduled the unit length tasks based on precedence constraints in a system using Hu's scheduling algorithm [15]. The List scheduling algorithm has been used for sorting the tasks in decreasing order with their precedence constraints and due dates using Brucker's scheduling algorithm [16]. Explained the importance of scheduling algorithms in the manufacturing industry [17]. List, Hu's, Mcnaughton's, Brucker's, Hodgson's scheduling algorithms are emphasized the parameters like Completion time & precedence constraints, Completion time, Preemption & completion time, Latency & due date, and Utilization respectively.

## III Proposed Methodology

A multicore or processor-based system has difficulty reaching the parameters like lateness, makespan, pre-emption, etc. The amalgamation of scheduling algorithms used to minimize the execution time, lateness, waiting time, and etc. To satisfy the mentioned considerations, the predefined techniques viz., list, and Mcnaughton's scheduling strategies are used for complacent of a user by increasing the number of parameters covered. The Proposed scheduling algorithm i.e. List Mcnaughton's amalgamation (LiMca) scheduling algorithm has been developed for an optimum solution with the features of List and Mcnaughton's scheduling algorithms to overcome the individual drawbacks. In LiMca, Workload has been distributed to the available processors or cores in a system by sorting the tasks in decreasing order with their precedence constraints including due dates, pre-emption, context switching, and EDD.

### LiMca scheduling algorithm

Algorithm: LiMca Algorithm

Begin
Outline a collection of tasks;
Initialize (Taskset, problem, Processors);
Find the Completion time using process time and processor(s);
K:=1;
Define m(k);
Define n(l);

Due date: Non-decreasing order;
Run the programming rule based on precedence constraints;
For j: 1 to m(k) do
Begin
For i:=1 to n(l) do
Begin
If Process Time and precedence constraints less than or equal to C
Then processor schedule on the second task;
End;
Processor schedule on the first task;
If completion time is equal to process time
Until the condition is satisfied (i:=15);
Until criteria is met (j:=2);
End;
End;

The avionic mission system is considered for verifying the results of proposed algorithm. A classic list planning formula and its variations are planned and analyzed to unravel the portioning issues. A set of tasks is considered for partitioning, which is done using Mcnaughton's Algorithm.

## IV Simulation Results

LiMca Algorithm

Parameters have been optimized like worst-case completion time, the CPU performance, and scalability on multiprocessor based avionics system by list mcnaughton's amalgamation (LiMca) scheduling algorithm. LiMca has been implemented and verified the simulation result objectives in terms of lateness, flow time, ompletion time, Throughput, waiting time, Utilization factors by the amalgamation of the list, and Mcnaughton's features. The LiMca scheduling algorithm simulation results are deployed in Fig 4.1. The main importance of the proposed scheduling algorithm is for an optimal solution which corresponds to the minimum makespan, minimize lateness, minimum flow time, maximum utilization of processor, and minimum waiting time. The LiMca scheduling algorithm has improved the lateness of other algorithms. The comparative lateness analysis of the List, Mcnaughton's, and LiMca scheduling algorithms on avionics mission system shown in Fig 4.2 a. LiMca scheduling average lateness has considerably decreased when compared with List, ECT, EST, LST, SPT, and Mcnaughton's scheduling algorithms.

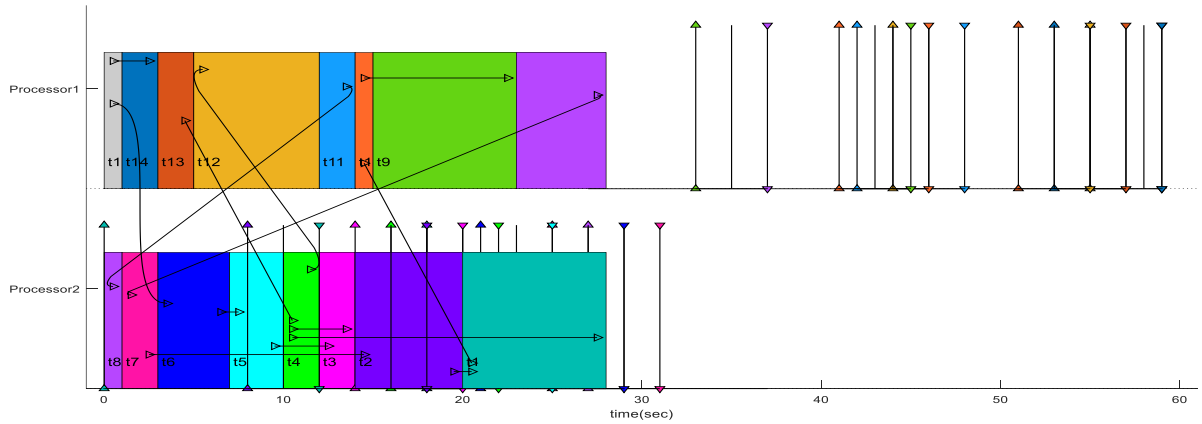
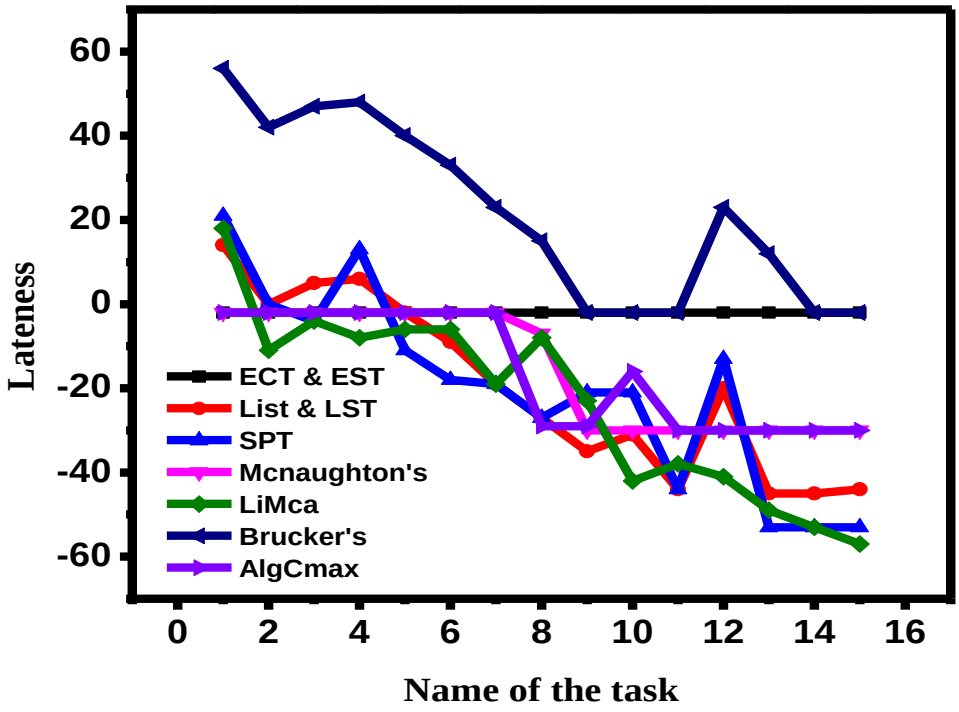
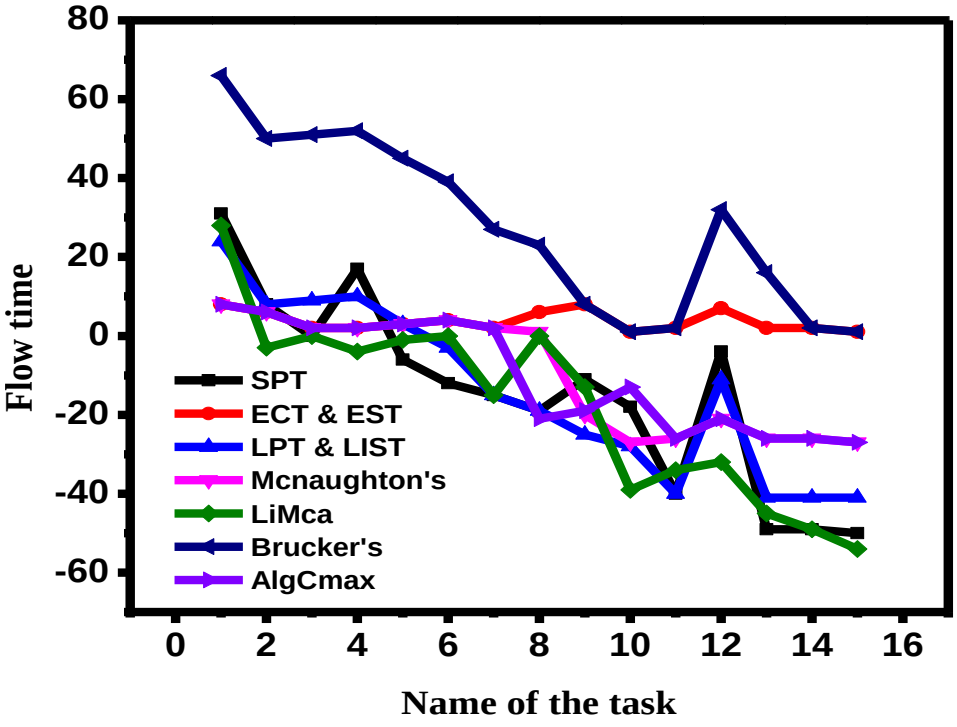



Fig 4.1: Gantt chart of LiMca algorithm

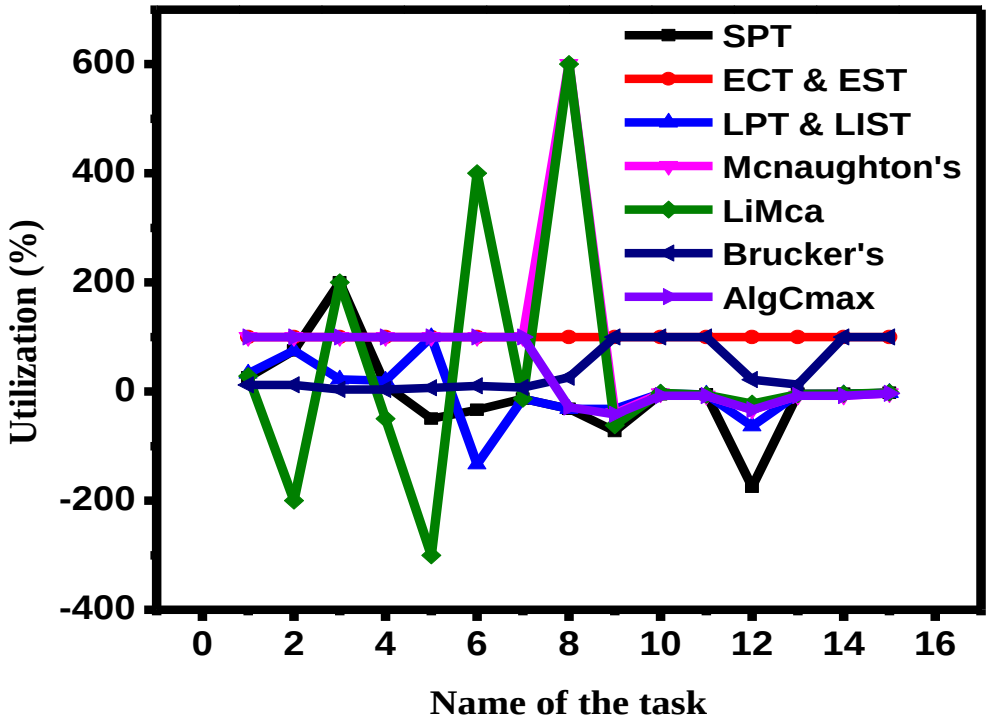
Average lateness has been evaluated effectively with LiMca for a solution that combined the features of Mcnaughton's and List as pre-emption, precedence constraints, due date, deadline, and release time. The resulting analysis has been completed with an experiment of the avionic mission system. The performance of the LiMca has improved results than other scheduling algorithms.




a) Lateness

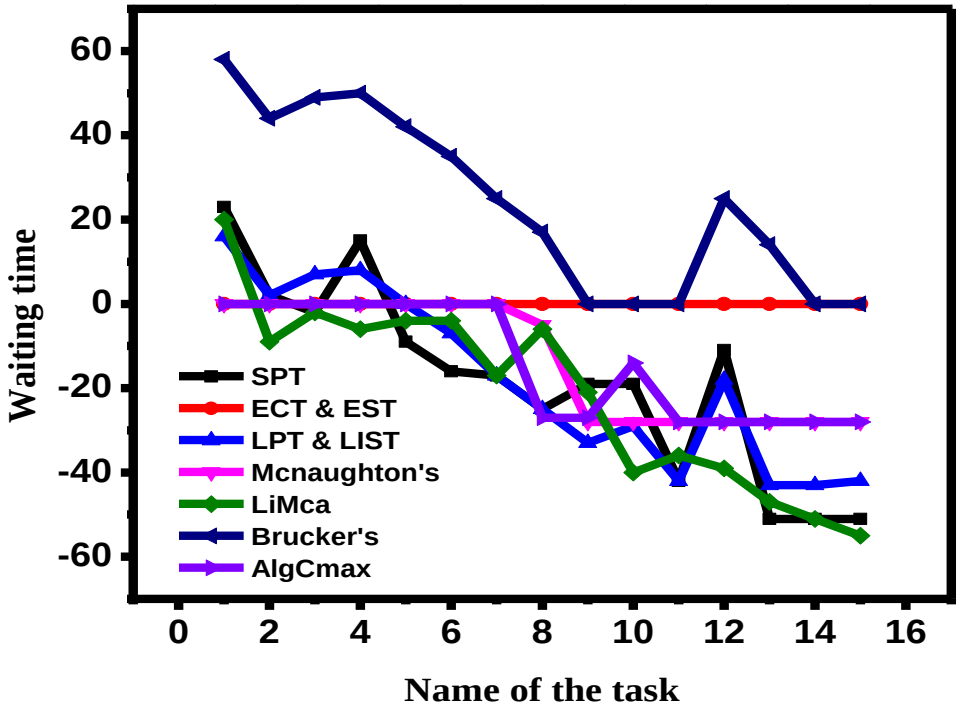



b) Flow time



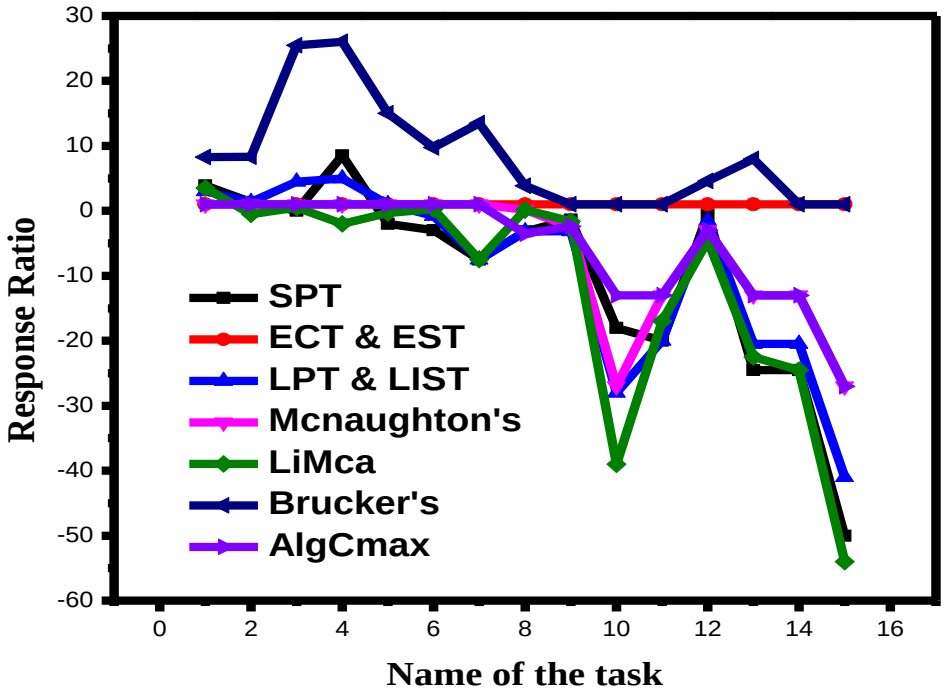

c) Utilization

The proposed simple approach can be useful on any multiprocessors based system for the reduction of the percentage of the worst delays in the process. Fig 4.2.b depicts the comparative analysis of flow time, which is considerably decreased in LiMca as compared to list, ECT, EST, LST, SPT, and mcnaughton's scheduling algorithms. Partition and allocation of tasks have been completed by mcnaughton's and list scheduling algorithm respectively to increase the run time speed and improves the performance.




d) Waiting time

Average flow time has been employed effectively for evaluation with LiMca using the features of Mcnaughton's and List as pre-emption, precedence constraints, due date, deadline, and release time. Fig 4.2 c depicts by considering LiMca for evaluating the utilization factor is considerably dynamically changed than List, ECT, EST, LST, SPT, and Mcnaughton's scheduling algorithms. The comparative analysis of utilization for List, Mcnaughton's, and LiMca scheduling algorithms on an avionics mission system as shown in Fig 4.2 c.




e) Response ratio

Fig 4.2: Comparative analysis of list, Mcnaughton's & LiMca algorithms

Average waiting time has been hired successfully for evaluating the performance. The comparative waiting time analysis of the list, mcnaughton's, and LiMca scheduling algorithms on an avionics mission system are shown in Fig 4.2 d. LiMca scheduling average waiting time is considerably decreased compared to List, ECT, EST, LST, SPT, and Mcnaughton's scheduling algorithms. Average waiting time has been evaluated with LiMca with the features of Mcnaughton's and List as pre-emption, precedence constraints, due date, deadline, and release time. The results analysis investigated through an experiment on Avionics Mission System using LiMca has been improved than other scheduling algorithms. The comparative response ratio analysis of the List, Mcnaughton's, and LiMca scheduling algorithms on the avionics mission system are shown in Fig 4.2.e. A response ratio of the LiMca scheduling algorithm is considerably decreased as compared to List, ECT, EST, LST, SPT, and Mcnaughton's scheduling algorithms. The results of LiMca have improved the performance than other scheduling algorithms. The comparative success ratio analysis of the List, Mcnaughton's, and LiMca scheduling algorithms on the avionics mission system are shown in Fig 4.2 f. The success ratio of the LiMca scheduling algorithm is considerably increased as compared to List, ECT, EST, LST, SPT, and Mcnaughton's scheduling algorithms. The average success ratio has been evaluated by LiMca with the features of Mcnaughton's and List in terms of preemption, precedence constraints, due date, deadline, and release time.

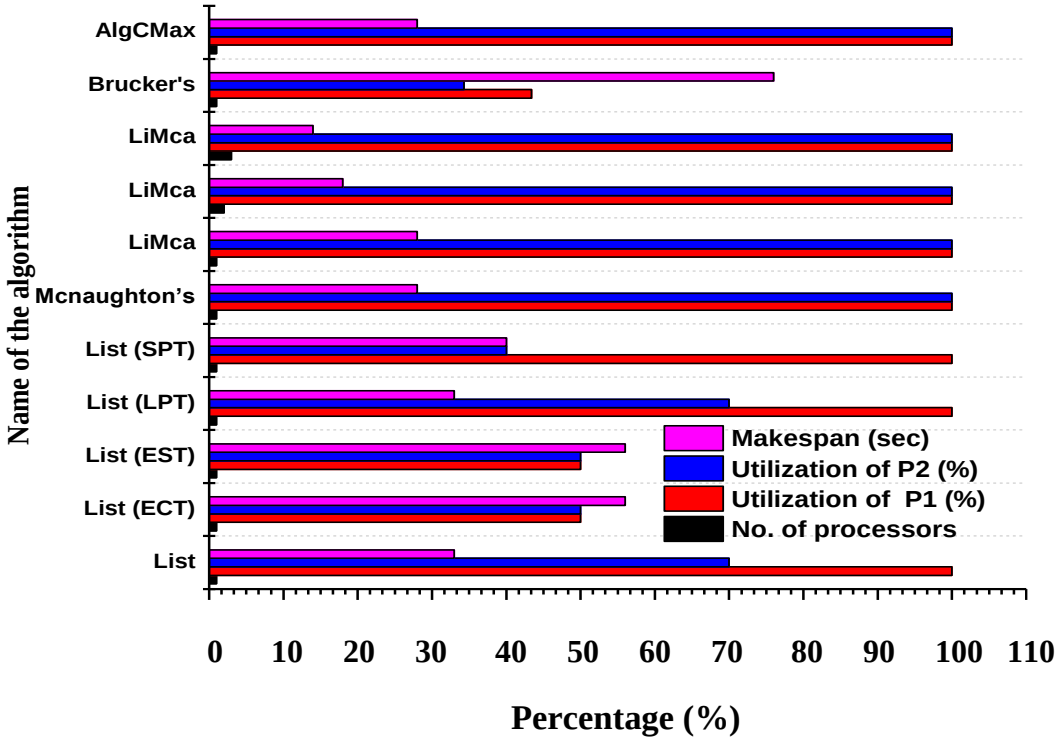



Fig 4.3: Comparative analysis of Throughput for list, Mcnaughton's & LiMca algorithms

Processors throughput has been hired successfully for evaluating the performance. Fig 4.3 shows the comparative analysis of processor throughput by the list, mcnaughton's, and LiMca scheduling algorithms on an avionics mission system. The processor throughput in the LiMca scheduling algorithm has been increased considerably in terms of less execution time compared to List, ECT, EST, LST, SPT scheduling algorithms. Mcnaughton's and List characteristics have been utilized for evaluating the processor throughput by the LiMca. The simulation results of LiMca have been improved throughput of processors than other scheduling algorithms.

Table 4.1: Comparative result analysis of different scheduling algorithms

| S.No | Name of the algorithm | No. of processors (m) | Utilization of processors (%) | | Preemption (Yes/No) | $C_{max}$ (sec) | Precedence constraints (Yes/No) | Due date (Yes/No) | Deadline (Yes/No) | Release time (Yes/No) |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | P1 | P2 | | | | | | |
| 1 | List [17] | 2 | 100 | 70 | No | 33 | Yes | No | No | No |
| | List (ECT) | 2 | 50 | 50 | No | 56 | No | No | No | Yes |
| | List (EST) | 2 | 50 | 50 | No | 56 | No | No | No | Yes |
| | List (LPT) | 2 | 100 | 70 | No | 33 | Yes | No | No | No |
| | List (SPT) | 2 | 100 | 40 | No | 40 | Yes | No | No | No |
| 2 | Mcnaughton's [18] | 2 | 100 | 100 | Yes | 28 | No | No | No | No |
| 3 | LiMca | 2 | 100 | 100 | Yes | 28 | Yes | Yes | Yes | Yes |
| | LiMca | 3 | 100 | 100 | Yes | 18 | Yes | Yes | Yes | Yes |
| | LiMca | 4 | 100 | 100 | Yes | 14 | Yes | Yes | Yes | Yes |

A set of tasks is considered for partitioning the tasks by Mcnaughton's and specified task allocation time using a List scheduling algorithm to improve throughput, utilization, and response time. LiMca algorithm shows the improved performance by taking 28 seconds in terms of makespan compared to list scheduling strategy and the schedulability with precedence constraints compared to mcnaughton's algorithm. Analysis of the list, ECT, EST, LST, SPT, Mcnaughton's, and LiMca scheduling algorithms by utilization of processors, preemption, completion time, precedence constraints, due date, deadline, and release time objectives in Table 4.1. The utilization of processors has been hired for evaluating the performance. The comparative utilization of processors analysis of the List, Mcnaughton's, and LiMca scheduling algorithms on the avionics mission system explained in Table 4.1. The utilization of processors with the LiMca scheduling algorithm is considerably improved compared to List, ECT, EST, LST, SPT, and Mcnaughton's scheduling algorithms.

Matlab R2019a is used to simulate the proposed algorithm. Time Optimization of Resources Scheduling (TORSCHE) simulation tool used along with Matlab 2019a for the simulation of the proposed algorithm. The proposed algorithm has been provided the improved results using a start time (sj), processing time (pj), Due date (d~j), Deadline (dj), Release time (rj), and Due date (d~j). LiMca algorithm has been Provided the results in terms of Response time (56.61% performance improved than Mcnaughton's algorithm), Completion time (Cj=sj+pj) (same as Mcnaughton's algorithm), waiting for time (wj=sj-rj) ( 63.4% performance improved than Mcnaughton's algorithm), flow time (Fj=Cj-rj) (55.51% performance improved than Mcnaughton's algorithm), lateness (Lj=Cj+dj) (66.57% performance improved than Mcnaughton's algorithm) and utilization of processors P1 & P2 is 100% (same as Mcnaughton's algorithm). LiMca scheduling algorithm Performance metrics superior to List, Hu's, Mcnaughton's, Brucker's, and Hodgson's scheduling algorithms in terms of computational performance.

## V Conclusions and Future Scope

The proposed algorithm has been implemented over an avionic mission system by 15 critical and non-critical tasks along with the process, release, deadline, and due date. Pre-emption, precedence constraints, process time, release time, and due date together are not supported by the algorithms reported in the literature. An attempt to consider Pre-emption, precedence constraints, process time, release time, and due date by the proposed scheduling algorithm.

The proposed algorithm executed over two processors. Improved performance of LiMca is observed even considering precedence constraints over a list and mcnaughton's in terms of completion time, utilization of processors, throughput, response ratio, waiting time, lateness, and flow time. In the real-time systems, the proposed algorithm executes the tasks faster and hence enhances the performance by assigning the time slots to the available processors in real-time scenarios. The proposed algorithm has changed to the new pattern with an addition of required features. Effectively employed this approach by extending the number of processors.

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

## Acknowledgement

I am thankful to my Research Supervisor Dr.P.A.Vijaya, Professor and Head in the Department of Electronics and Communication Engineering, BNMIT, Bengaluru. for her valuable guidance and suggestions in analyzing and testing throughout the period of doing this research work. I wish to express my sincere thanks to Management, BNMIT, Bengaluru. for providing ample facilities to do the research work.

## References

1. Yu XG, Sun JY, He B, Zhuang JJ, Dai ZC. Design and implementation of automatic invigilation functions using the embedded technology. *Procedia Comput Sci*. 2020;166:41-45. doi:10.1016/j.procs.2020.02.010
2. Raja Singh R, Yash SM, Shubham SC, et al. IoT embedded cloud-based intelligent power quality monitoring system for industrial drive application. *Futur Gener Comput Syst*. 2020;112:884-898. doi:10.1016/j.future.2020.06.032
3. Kursun O, Patooghy A. An Embedded System for Collection and Real-Time Classification of a Tactile Dataset. 2020;8. doi:10.1109/ACCESS.2020.2996576
4. Saadatmand FS, Rohbani N, Baharvand F, Farbeh H. TAMER: an adaptive task allocation method for aging reduction in multi-core embedded real-time systems. *J Supercomput*. 2020;(0123456789). doi:10.1007/s11227-020-03326-7
5. Chen J-J, Shi J, von der Brüggen G, Ueter N. Scheduling of Real-Time Tasks with Multiple Critical Sections in Multiprocessor Systems. Published online 2020:1-14. http://arxiv.org/abs/2007.08302
6. Anwar N, Deng H. A hybrid metaheuristic for multi-objective scientific workflow scheduling in a cloud environment. Applied Sciences. 2018;8:538.
7. Zhang Y, Sun J. Novel efficient particle swarm optimization algorithms for solving QoS-demanded bag-of-tasks scheduling problems with profit maximization on hybrid clouds. Concurrency Computat Pract Exper. 2017;29:e4249.
8. Kutil M, Sojka M. TORSCHE Scheduling Toolbox for Matlab. In: *2006 IEEE Conference on Computer Aided Control System Design,*. ; 2006:1181-1186. doi:10.1109/CACSD-CCA-ISIC.2006.4776810
9. Michal Kutil, P˘remysl ˘S˚ucha MS and ZH. *TORSCHE Scheduling Toolbox for Matlab User ' s Guide*.; 2007. http://rtime.felk.cvut.cz/scheduling-toolbox/%0AToolbox
10. Sreekumar A, Swetha K, Swetha A, Radhamani Pillay V. Enhanced performance capability in a dual redundant avionics platform-fault tolerant scheduling with comparative evaluation. *Procedia Comput Sci*. 2015;46(Icict 2014):921-932. doi:10.1016/j.procs.2015.01.001
11. Chen W, Xie G, Li R, Li K. Execution cost minimization scheduling algorithms for deadline-constrained parallel applications on heterogeneous clouds. *Cluster Comput*. 2020;0123456789. doi:10.1007/s10586-020-03151-w
12. Kushwaha S, Kumar S. An Investigation of List Heuristic Scheduling Algorithms for Multiprocessor System. Published online 2017:29.
13. Hossny A, Creighton D. Minimizing Impact of Bounded Uncertainty on McNaughton ' s Scheduling Algorithm via Interval Programming. Published online 2013:970-976. doi:10.1109/SMC.2013.171
14. Hossny AH, Creighton D, Nahavandi S, Member S. Reducing the Impact of Bounded Parametric Uncertainty on Hodgson' s Scheduling Algorithm Using Interval Programming. online 2015:1-11.

15. Series C. Cyclic scheduling for parallel processors with precedence constrains. Published online 2020. doi:10.1088/1742-6596/1658/1/012019

16. Brucker P, Jurisch B, Sievers B. A branch and bound algorithm for the job-shop scheduling problem. *Discret Appl Math*. 1994;49(1-3):107-127. doi:10.1016/0166-218X(94)90204-6

17. Sreenath M, PA Vijaya. Performance Evaluation of Embedded System Using Scheduling Algorithms. *IjsrcseitCom*. 2018;3(1):85-89. doi:10.32628/CSEIT183118