Evaluation and Comparison of Load Balancing in RTDS using Information Theoretic Entropy

Rashmi Sharma
Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, Solan-173234, Himachal Pradesh, India
rashmi.nov30@gmail.com

Nitin
Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, Solan-173234, Himachal Pradesh, India
delnitin@juit.ac.in

***Abstract*— Load Balancing is very important requirement of any Distributed System. Migration of task is one of the eminent methodologies for the same purpose. Till now utilization is the only key based on which source and destination processor of victim task are being selected. This paper evaluates information theoretic based entropy factor that works similar to utilization factor and its better performance in task migration methodology. We have computed the performance of our system by calculating the Average scheduling latency, Deadline missing rate, Migration rate and finally the Execution ratio of total tasks for 5 and 10 numbers of processors of RTDS.**

Keywords- Load Balancing; Real Time Distributed System (RTDS); Task Migration; CPU Utilization; Maximum Entropy Model.

I. INTRODUCTION AND MOTIVATION

Arrangements of more than one processor compose a Distributed System. If contributor processors are Real time processors than resultant distributed system becomes Real Time Distributed System (RTDS) [1]. Main benefit and requirement of any distributed system is load balancing that is achieved by using task migration or task duplication methodology. The task migration technique can be used for independent as well as dependent tasks but duplication of tasks specially implement on dependent tasks in order to lessen the execution cost of an entire Directed Acyclic Graph (DAG) by reducing the communication costs in-between tasks [2, 3]. This paper is all about the load balancing using migration of tasks from one processor to another processor. Most illustrious factor based on which scheduler take a decision on victim task, source and destination processors is utilization (Reported till now). Processor having utilization value greater than 1 becomes the source processor and on the other side processor with minimum utilization values will be the destination of victim task. Here, victim task is the task due to which source becomes overloaded ($U > 1$) and therefore that task has to migrate towards target processor. Hence, we can say utilization is the only factor due to which complete distributed system administer dynamics. Earliest Deadline First (EDF), Rate Monotonic (RM) are few renowned real time scheduling algorithms that used utilization factor for task execution [4-7]. In our research paper [8,9] we have visualized RTDS through entropy instead of utilization and also explain some of its additional advantages over utilization.

Entropy is the measure of uncertainty associated with random variables (information theory) [10,11]. In computer science, the arrival and execution of any data is information and entropy is used to compute the amount of improbability in that information. Correspondingly, the arrival and execution of real time tasks are used to collect the magnitude of information present in the entire system. Similarly, the main four tuples of any real time task *i.e.* $\tau_{arrival}$, τ_{wcet}, $\tau_{deadline}$ and τ_{period} are used to compute the amount of information present in the given task. These parameters help in assessing the presence of information and additionally, entropy values are computed to review the amount of uncertainty (ambiguity) present in the system. Since, the dimension of entropy is in bits [12]. These bits also tell the space volume in the memory of the system; hence, by using entropy the space available in the system will be computed. In migration scheme, processors having less available memory become the source processor and processors of maximum available memory will be the destination of victim task. How all these manipulations are done is explained in further sections of this paper.

The programmed structure of given paper is: 2nd section explains the ground level information of entropy with a little bit explanation of utilization. Next section, explains the proposed strategy of load balancing in RTDS using entropy. Further results and discussion with simulation will be explained followed by conclusion and references. Following is the nomenclature of all symbols that are used in this paper:

TABLE I. NOMENCLATURE

Symbol	Definition
τ	Task
$\tau_{arrival}$	Task Arrival time
τ_{wcet}	Task Worst case execution time
$\tau_{deadline}$	Task Deadline
τ_{period}	Task Period

978-1-4799-2572-8/14/$31.00 ©2014 IEEE

A_{SL}	Average Scheduling Latency
D_{MR}	Deadline Missing Rate
$MigR$	Migration Rate
U	CPU utilization
E	CPU entropy
Pr (τ_{miss})	Deadline missing probability of task
Pr (τ_{meet})	Deadline meeting probability of task
$I(\tau_{meet})$	Amount of information of task meeting deadline
$I(\tau_{miss})$	Amount of information of task missing deadline
$\tau_{miss_{entropy}}$	Entropy of task missing deadline
$\tau_{meet_{entropy}}$	Entropy of task meeting deadline
$\tau_{entropy}$	Total Entropy of given task
τ_{maxe}	Task Maximum entropy value
cpu_{maxe}	Maximum Entropy value of CPU
$cpu_{available\ entropy}$	CPU available entropy

II. Background and Preliminaries

This paper exploits Information theoretic entropy concept for load balancing in loosely coupled distributed system. Till now CPU utilization is the only parameter that plays the vital role in load balancing. Many years back some researchers state that CPU load is efficient for load balancing as compared to CPU utilization [13]. The cause behind CPU load did better is possible because when a host is heavily loaded, its CPU utilization is expected to be nearly 100% and it is unable to reveal the exact load level of the utilization. In contrast, CPU queue lengths can directly reflect the amount of load on a CPU.

By the passage of time, techniques for load balancing have been improved. Now, researchers start working on CPU utilization instead of load. Mostly scheduling algorithms use CPU utilization for the same purpose of loosely coupled RTDS. But, our paper follows the older concept of CPU load with some extra zest *i.e.* entropy. Simply dissimilarity between the CPU load and entropy is that the later one is computed by gathering the information from previous one. Further, computed values are used to determine the entropy values that tell about the presence of uncertainty (improbability) in retrieved information. Before turning over the calculation methods of Entropy, let us talk about load balancing with the help of CPU utilization and CPU Entropy

A. Load Balancing

Load balancing is the advantage of any distributed system or we can say that without load balancing distributed system is worthless. Load balancing itself is a self descriptive term that regularly distributes the load among nodes/processors by the migration of tasks of heavily loaded or utilized processors (Source) to lightly loaded or less utilized processors (Destination/Target). The tasks that migrated in-between the nodes are recognized as victim tasks. Load among the processors is balanced by using task migration or task duplication techniques [2, 14-18]. There are few parameters that help in taking the decision for selection of target processor for migration or duplication of victim tasks. These parameters are CPU load and CPU utilization but in this paper we are using a parameter CPU Entropy instead of load and utilization. Reference [8, 9] explains the reason behind using entropy in place of utilization but here we are using entropy for balancing the load of processors.

B. Load Balancing and Utilization

As we know that CPU utilization is the time taken by CPU to execute the given task. If processor's utilization is near about 100% then it will be a source processor and processor having least utilization becomes the target processor. Based on such judgment of utilization, researchers design many tools and scheduling algorithms that execute real time tasks of distributed system. EDF-fm, RealtssMP tool etc. [19-27] has been used CPU utilization for the scheduling as well as migration of tasks.

C. Load Balancing and Entropy

Although, the conception of entropy initially a thermodynamic construct, it has been customized in the other spheres of study together with information theory, Natural Language Processing (NLP), Image Processing, thermo-economics/ ecological-economics and evolution. All these are few interdisciplinary applications of entropy [28-30]. This paper uses information theoretic based entropy for managing the load of participant processors of loosely coupled distributed system.

The arrival and execution of periodic tasks generate arbitrariness in the system. The term entropy is used to calculate the amount of uncertainty of a particular processor. Less uncertain processor becomes the destination and most uncertain will be a source of victim task. With the help of maximum entropy model ($maxe$) [31, 32] scheduler decides the threshold limit of entropy ($E <= maxe$) that behaves like the maximum limit of utilization *i.e.* ($U <= 100$). The computation of maximum entropy value is explained in next section.

III. Proposed Entropy Based Load Balancing Scheduling Algorithm

This paper explains the load balancing with the help of Information theoretic entropy concept. This entropy perception follows the CPU load concept. It uses the information of task to compute entropy values of task and processor as well. Basic difference between Entropy and Utilization parameter is that the entropy deals with space and time but utilization deals with time only [8]. Load Balancing is done by computing the available entropy of participant processors of RTDS.

A. Entropy Computation

Entropy computes the amount of uncertainty present in the given information. Amount of information is figured by using the occurrence probability of particular events. Any real time task has two events missing and meeting of deadlines. The presence of amount of information about the

meeting and missing a deadline is computed by evaluating the probability of occurrence of these two events.

$$\Pr(\tau_{meet}) = \frac{\tau_{arrival}+\tau_{wcet}}{\tau_{deadline}} \quad (1)$$

$$\Pr(\tau_{miss}) = 1 - \Pr(\tau_{meet}) \quad (2)$$

With the help of equation (1) and (2), amount of information about meeting and missing of deadline will be computed in equation (3) and (4).

$$I(\tau_{meet}) = \log_2 \frac{1}{\Pr(\tau_{meet})} \quad (3)$$

$$I(\tau_{miss}) = \log_2 \frac{1}{\Pr(\tau_{miss})} \quad (4)$$

Equations (5) and (6) are evaluating the amount of uncertainty present in retrieved information from (3) and (4).

$$\tau_{meet_{entropy}} = \Pr(\tau_{meet}) \times I(\tau_{meet}) \quad (5)$$

$$\tau_{miss_{entropy}} = \Pr(\tau_{miss}) \times I(\tau_{miss}) \quad (6)$$

Further equation (7) is used to evaluate the total entropy of entire task.

$$\tau_{entropy} = \Pr(\tau_{meet}) \times I(\tau_{meet}) + \Pr(\tau_{miss}) \times I(\tau_{miss}) \quad (7)$$

Now, equation (8) computes the entropy of particular processor by the accumulation of entropy values of all available tasks on a given processor.

$$E = \sum_{i=1}^{n} \tau_{i_{entropy}} \quad (8)$$

B. Maximum Entropy Computation

In order to compute the maximum entropy of system, we should choose the probability that gives higher entropy from available entropies of all events [31]. Hence, equation (5) and (6) computes the entropy of all events and equation (9) and (10) returns the maximum entropy valued event. These computed maximum entropy is used to decide the maximum entropy value of entire system.

From equation (5) and (6)

If $(\tau_{meet_{entropy}} > \tau_{miss_{entropy}})$

$$\tau_{maxe} = \tau_{meet_{entropy}} \quad (9)$$

Else

$$\tau_{maxe} = \tau_{miss_{entropy}} \quad (10)$$

Equation (11) is used to determine the maximum entropy value of a particular processor. For deciding the maximum entropy value of any processor we have taken some fundamental definitions of parameters of given task [8]. Consider the threshold limit of given processor is 200 tasks. For 200 tasks maximum entropy will be

$$cpu_{maxe} = \sum_{i=1}^{200} \tau_{i_{maxe}} = 19.931 \quad (11)$$

Following table is showing the values of maximum entropy on different threshold limit.

TABLE II. MAXIMUM ENTROPY VALUES AND CPU MAXIMUM UTILIZATION WITH RESPECT TO THRESHOLD LIMIT

CPU Threshold Limit	CPU Maximum Entropy
200	13.287
400	39.863
600	79.726
800	132.877
1000	199.315

C. Available Entropy Computation

The measurement unit of entropy is in bits [12] which are used to evaluate the volume of space present in the CPU memory. Every task requires some space for allocation and then time for execution. Therefore, available entropy gives scheduler the existing space in the memory of a given processor. Equation (12) is used to evaluate the available space in the CPU.

$$cpu_{available\ entropy} = cpu_{maxe} - E \quad (12)$$

This computed available entropy plays main role in balancing the load in-between processors of the RTDS. Following is the algorithm that is used to balance the load by using entropy values.

Fig. 1. Entropy Based Load Balancing Algorithm

Input: Periodic generation of independent tasks with its $\tau_{arrival}$,τ_{wcet},$\tau_{deadline}$ and τ_{period}

Output: Execution of tasks with A_{SL},D_{MR},$MigR$ and Execution Ratio of total tasks

BEGIN

1. $maxe = 19.931$ // **For 200 independent tasks on each processor**
2. $CPU\ Current\ Entropy = \sum_{i=1}^{n} Task\ Entropy_i$
3. $Available\ CPU\ Entropy = maxe - CPU\ Current\ Entropy$
4. ***If*** $(CPU\ Current\ Entropy \leq maxe)$
5. $Task\ is\ Executable\ on\ source\ processor$
6. ***Else***
7. $Check\ (Available\ CPU\ Entropy\ of\ all\ Processors)$
8. $Migration\ of\ victim\ task\ on\ Processor\ of\ maximum\ available\ CPU\ Entropy$

END

Above stated algorithm 3.1 and complete simulation is implemented in Eclipse Java EE IDE environment running with Ubuntu Version 11.10. Java threads and synchronization functions are implemented here for the generation, execution or synchronization between periodic independent real time tasks. Following are some functions that are used to execute the entire simulation.

TABLE III. FUNCTIONS USED IN SIMULATION AND THEIR RESPONSIBILITIES

Functions	Responsibility
rand.nextInt(10)+1	Random generation of τ_{wcet},$\tau_{deadline}$.
c.get(GregorianCalendar.MILLI SECOND)	Generate attributes of task in millisecond
(Math.log(1/P1)/Math.log(2))	Log function uses to compute the amount of information as well as entropy value
Thread.sleep(10)	Generate tasks in every 10 milliseconds
starttime = System.nanoTime()	Compute the start time of destination searching
estimatedTime = System.nanoTime() – starttime	Compute total time of destination searching
list.addLast(task)	Add tasks in a local task queue of a given processor
Thread.currentThread().join(10)	This function is used to maintain the synchronization between tasks
Thread.currentThread().interrupt ()	Function is used to stop the execution of tasks

IV. COMPARISON AND DISCUSSION OF PROPOSED WORK

Earliest Deadline First (EDF) and Rate Monotonic (RM) [24, 26] are aged and matured scheduling algorithms that are used for the execution of real time tasks. In current scenario, load balancing is done by using utilization parameter only. We consider here a loosely coupled RTDS, and use EDF, RM and proposed entropy based scheduling algorithms for the execution of tasks with load balancing. Further, the performance of existing and proposed scheduling algorithms will be evaluated on the basis of few parameters followed by discussion.

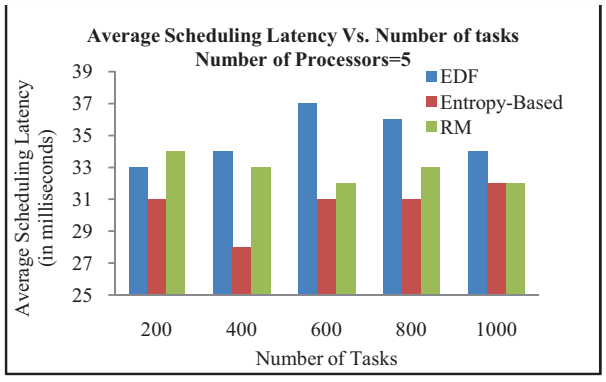
A. Performance Evaluation

We asymptotically ran up to 5,000 independent real time tasks 30 times for 5 and 10 processors and took readings to compute the scheduling latency, deadline missing rate, migration rate and execution ratio of total tasks.

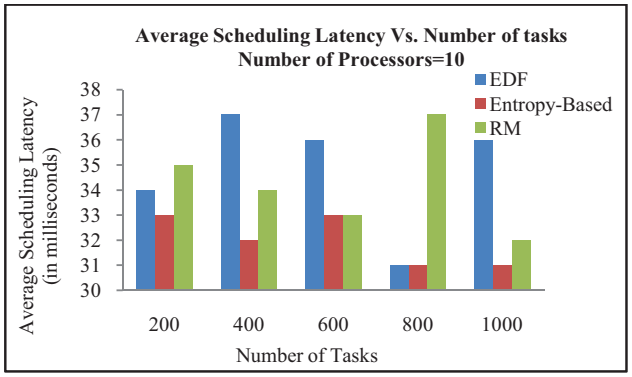
1) Scheduling Latency(A_{SL})

Scheduling latency is the time when the system is unproductive because of scheduling tasks. It is a system latency incurred because it has to spend time scheduling.

$$A_{SL} = \frac{Tasks\ with\ scheduling\ latency}{Total\ number\ of\ tasks} \quad (13)$$



(a) A_{SL} for 5 Processors



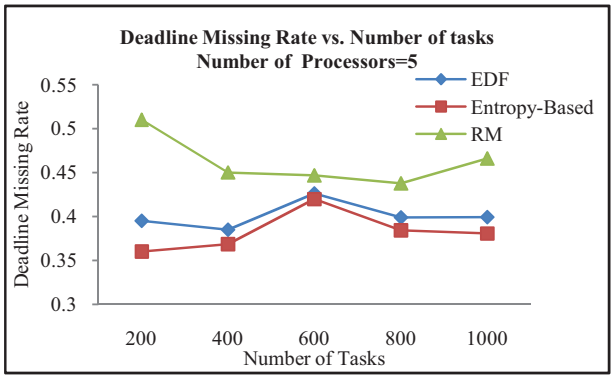
(b) A_{SL} for 10 Processors

Fig. 2. Performance of EDF, RM and Proposed Algorithm on A_{SL} for (a) 5 and (b) 10 processors

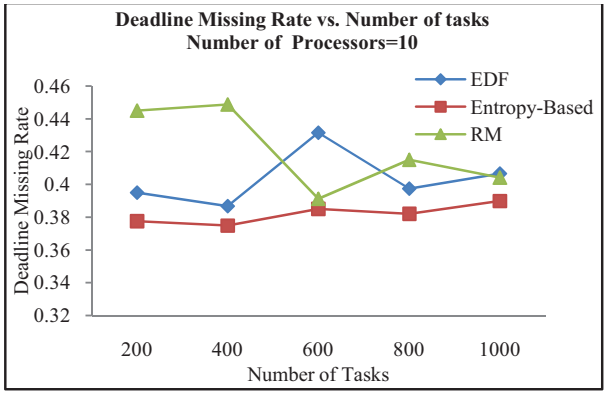
2) Deadline missing rate (D_{MR})

This parameter calculates the number of tasks missing deadline per total number of tasks generated at that period.

$$D_{MR} = \frac{Tasks\ missing\ Deadline}{Total\ Number\ of\ tasks} \quad (14)$$



(a) D_{MR} for 5 Processors



(b) D_{MR} for 10 Processors

Fig. 3. Performance of EDF, RM and Proposed Algorithm on D_{MR} for (a) 5 and (b) 10 processors

3) Migration rate (MigR)

Number of tasks migrates per total number of tasks generated has been computed in this parameter.

$$MigR = \frac{Number\ of\ tasks\ Migrate}{Total\ Number\ of\ tasks} \quad (15)$$

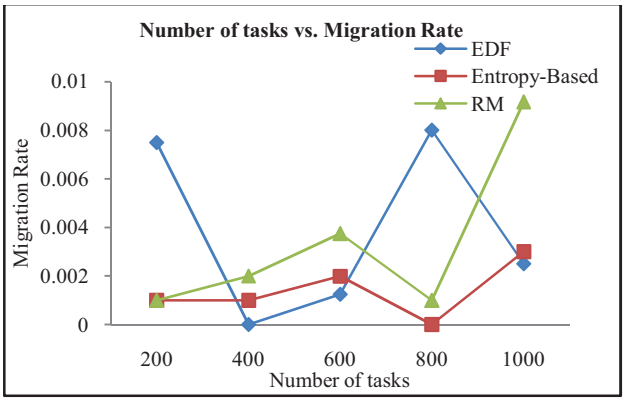
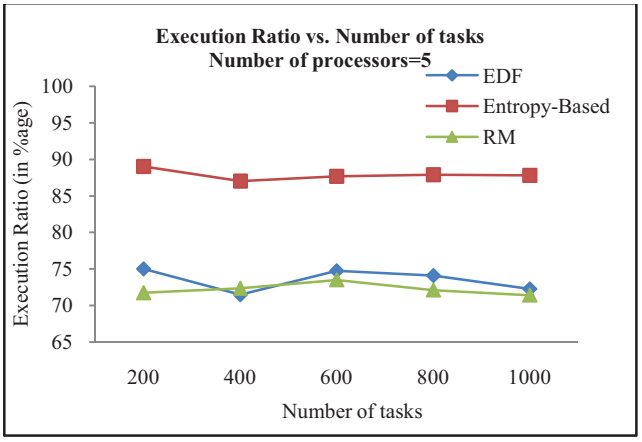


Fig. 4. Performance of EDF, RM and Proposed Algorithm based on $MigR$

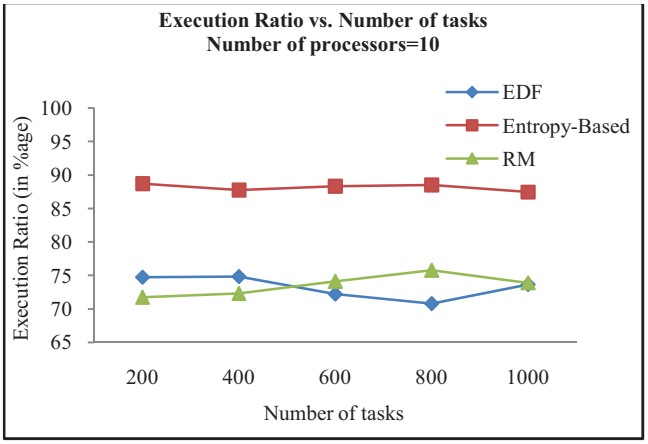
4) Execution Ratio

Under the proposed schedulability test, only those tasks are schedulable whose $\tau_{entropy} \leq cpu_{maxe}$, in EDF $\frac{\tau_{wcet}}{\tau_{period}} \leq 1$ and in RM $\frac{\tau_{wcet}}{\tau_{period}} \leq n(2^{\frac{1}{n}} - 1)$. After scheduling of tasks, these tasks will be ready for the execution. Hence,

$$Execution\ Ratio = \frac{Number\ of\ executable\ tasks}{Total\ Number\ of\ tasks} \quad (16)$$



(a) Execution Ratio for 5 Processors



(b) Execution Ratio for 10 Processors

Fig. 5. Performance of EDF, RM and Proposed Algorithm based on the Execution Ratio for (a) 5 and (b) 10 Processors

B. Discussion

In order to evaluate the performance of proposed parameter we run 5,000 independent tasks up to 30 times for 5 and 10 processors. After putting entropy in place of utilization we get better results. Our performance matrix contains scheduling latency, deadline missing rate, migration rate and execution ratio of total tasks arrived in the system. Graphs of Fig 1 show that the computed scheduling latency of entropy based algorithm is comparatively lower than the EDF and RM as well. In order to execute tasks on or before deadline Real Time environment has been used. Therefore, the Deadline missing rate is computed in Fig 2 in which we can see that entropy based algorithm gives healthier results than existing algorithms. Moreover, Fig 3 is explaining the Migration rate that is used for balancing the load in-between processors. Entropy based algorithm gives alike or better migration rate comparatively. Our last parameter is execution ratio that gives excellent results as compared to existing scheduling algorithms. Reason behind these results is that entropy decides the acceptance of task on the basis of space and time both.

V. Conclusion and Future Work

In this paper, we have generated independent real time tasks in loosely coupled distributed system. In order to perform load balancing among the processors, scheduling and execution of tasks; we replace utilization factor with entropy factor. Further, simulation of existing algorithms EDF, RM and proposed algorithm has been done on the same data. After the evaluation of resultant parameters we have got comparatively good results of the proposed algorithm. In future, we will implement it on dependent tasks in which task duplication will be used for load balancing and reduction of execution time of entire real time DAG.

REFERENCES

[1]. Singh, Y.J., Mehrotra, S.C., An Analysis of Real Time Distributed System under different priority policies, World Academy of Science, Engineering and Technology (2009).

[2]. Sharma, R., Nitin, Duplication with Task Assignment in Mesh Distributed System, IEEE World Congress on Information and Communication Technologies (WICT), (2011) 672-676.

[3]. Bestavros, A., Load Profiling in Distributed Real-Time Systems (1996).

[4]. Sharma, R., Nitin, Optimal Method for Migration of Tasks with Duplication, 14th International Conference on Modelling and Simulation, (2012) 510-515.

[5]. Dhakal, S., Hayat, Majeed M., Pezoa, Jorge E., Dynamic Load Balancing in Distributed Systems in the Presence of Delays: A Regeneration-Theory Approach, IEEE Transactions on Parallel and Distributed Systems, Volume. 18, Issue. 4, (2007) 485-497.

[6]. Darbha,S., Agrawal,D.P., A task duplication based scalable scheduling algorithm for distributed memory systems, Journal of parallel and Distributed Computing, Vol. 46, No. 1, (1997) 15-27.

[7]. Suen, T.T.Y., Wong, J.S.K., 1992, Efficient Task Migration Algorithm for Distributed Systems, IEEE Transactions on Parallel and Distributed Systems, Volume 3, Issue 4, 488-499.

[8]. Sharma, R., Nitin, Sharma, Rashmi, Visualization of Information Theoretic Maximum Entropy Model in Real Time Distributed System, 3rd IEEE International Conference on Advances in Computing and Communications (ICACC-2013), 282-286.

[9]. Sharma, R., Nitin, Entropy, a new dynamics governing parameter in real time distributed system: a simulation study, International Journal of Parallel, Emergent and Distributed Systems, Taylor and Francis (ahead-of-print), 1-25.

[10]. Dong, Yu., Deng, Li., Acero, Alex., Using continuous features in the maximum entropy model, Pattern Recognition Letters 30, (2009) 1295–1300.

[11]. Shannon, C.E., A Mathematical Theory of Communication, Bell Systems Technical Journal, (1949) 1-54.

[12]. Feldman, D., A Brief Introduction to: Information Theory, Excess Entropy and Computational Mechanics (October 2002).

[13]. Ferrai, D., Zhou, S., An Empirical Investigation of load indices for load balancing applications (1987).

[14]. Sharma, R., Nitin, Task Migration with EDF-RM Scheduling Algorithms in Distributed System, 2nd IEEE International Conference on Advances in Computing and Communications, (ICACC-2012), 182-185.

[15]. Fisher, N.W., The Multiprocessor Real-Time Scheduling of General Task Systems, Ph.D. dissertation. Department of Computer Science, The University of North Carolina at Chapel Hill, NC (2007).

[16]. Wang, X., Jia, D., Lu, C., Koutsoukos, X., Decentralized Utilization Control in Distributed Real-Time Systems, 26th IEEE International Real-Time Systems Symposium (RTSS-2005).

[17]. Wang, X., Jia, D., Lu, C., Koutsoukos, X., DEUCON: Decentralized End-to-End Utilization Control for Distributed Real-Time Systems, IEEE Transactions on Parallel and Distributed Systems, Vol. 18(7), (2007) 996-1009.

[18]. Srinivasan, A., Anderson, J.H., Efficient scheduling of soft real-time applications on multiprocessors, In Proceedings of the 15th Euromicro Conference on Real-Time Systems, (2003) 51–59.

[19]. Arnoldo Díaz-Ramírez, Dulce K. O., Pedro Mejía-Alvarez, A Multiprocessor Real-Time Scheduling Simulation Tool, 22nd International Conference on Electrical Communications and Computers (CONIELECOMP), (2012) 157-161.

[20]. Singh, J., Singh, S.P., Schedulability Test for Soft Real-Time Systems under Multiprocessor Environment by using an Earliest Deadline First Scheduling Algorithm (2012).

[21]. Bertogna, M.,Cirinei, M.,Lipari,G., Schedulability Analysis of Global Scheduling Algorithms on Multiprocessor Platforms, IEEE Transactions on Parallel and Distributed Systems, Vol.20(4), (2009) 553-566.

[22]. Anderson, J., Bud, V., Devi, U.C., An EDF-based Restricted-Migration Scheduling Algorithm for Multiprocessor Soft Real-Time Systems, The International Journal of Time-Critical Computing (2008).

[23]. Emberson, P., Bate, I., Minimizing Task Migration And Priority Changes In Mode Transitions, 13th IEEE Real Time and Embedded Technology and Applications Symposium (RTAS' 2007)158-167.

[24]. Anderson, J., Bud, V., Devi, U.C., An EDF-based scheduling algorithm for multiprocessor soft real-time systems, In IEEE ECRTS, (2005) 199-208.

[25]. Goossens, J., Baruah, S., Funk, S., Real-time scheduling on multiprocessor, 10th International Conference on Real-Time System (2002).

[26]. Akgün, B.T., BAG Distributed Real-Time Operating System and Task Migration, Turkish Journal Electrical Engineering, Vol. 9, (2001)123-136.

[27]. Cheriton, D.R., The V Distributed System, Comm. of the ACM, Vol. 31(3), (1988) 314-333.

[28]. Lu, C., Wang, X., Koutsoukos, X., End-to-End utilization control in distributed real-time systems, International Conference on Distributed Computing Systems (ICDCS), Tokyo, Japan (March 2004).

[29]. Malouf, R., A comparison of algorithms for maximum entropy parameter estimation, In Proceedings of the Sixth Conference on Natural Language Learning (CoNLL), (2002) 49–55.

[30]. Zurek, W.H., Thermodynamic cost of computation, algorithmic complexity and the information metric, Nature, 347, (1989b) 119–124.

[31]. Penfield, P., Chapter-10, Principle of Maximum Entropy, Version 1.0.2, www.mtl.mit.edu/Courses/6.050/2003/notes/chapter10.pdf (2003)

[32]. Gull, S.F., Skilling, J., Maximum entropy method in image processing, IEEE Proceedings vol. 131(6), (1984) 646-659.