International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878, Volume-8 Issue-3, September 2019

Formulation of Control Strategies for IoT Task Scheduling

Prasantha Rao A, G. Sekhar Reddy, CH N Santhosh Kumar, K.S. Reddy

Abstract*: The various Internets of Things (IoT) application tasks are difficult to schedule due to heterogeneity properties of IoT. So an efficient algorithm is required that forms < task, processor> pair appropriately. This paper presents a more sensible model for varying execution times of tasks and deviation in task parameters for building a schedule is allowed. The system provides an adaptive learning mechanism called Expected Time Matrix ETM (i, j). When the environment of the system changes dynamically, the system learns and adapts itself to the new changes automatically, since the learning mechanism has been incorporated in the system. ETM (i, j) concepts allows the system to learn from past instances as well. The work is supported by simulations that highlight the viability of concepts proposed. The key objective of this paper is to present the developed scheduling algorithm that is self-configurable and dynamic

Keywords*: Load Balancer, Task Model, Task Cluster, Self-Configurable, Cluster, Control Strategy, Dynamic Scheduler, Heterogeneity.

I. INTRODUCTION

The admiration of Internet of Things (IoT) requires an extremely massive number of electronic components to be integrated to the existing IoT. Controlling IoT devices becomes critical in a large distributed environment without having the best design that leads to major performance degradation. The main attributes of these IoT devices are heterogeneity, scheduling and dynamicity. Heterogeneity refers to the communication technology, processing capability, functional capabilities, communication sequence and security mechanisms [12]. The key objective of scheduling is to collect the data from an embedded device in a much better way. The data that is required to be collected from the devices should be temporarily synchronized to accomplish the task.

Manuscript published on 30 September 2019
* Correspondence Author
Prasantha Rao A*, Professor IT Dept., Anurag Group of Institutions, Hyderabad, India. Email: adirajuapppy@yahoo.com
G. Sekhar Reddy, Asst. Professor IT Dept., Anurag Group of Institutions, Hyderabad, India, Email: sudheercse@gmail.com
CH.N.Santhosh Kumar, CSE Department, Anurag Engg. College, Kodada, India,. Email santhosh.ph10@gmail.com
K. S. Reddy, Hyderabad, India.

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an open access article under the CC-BY-NC-ND license http://creativecommons.org/licenses/by-nc-nd/4.0/

Dynamicity is the frequency of altering positions or appearing, disappearing of embedded devices at a given location. In today's world, the accurate processing of task is very important with a proper defined deadline. The cognitive or intelligent model of task scheduling algorithm for a heterogeneous parallel processor environment is to ensure that tasks are allocated to a specific processor according to the given schedule.

The Shopping Mall application and the application of Traffic monitoring system monitor the customer activities using a handheld device. The moment when the customer with a handheld device enters the shopping mall, communication system that is equipped in the mall instantaneously senses the presence of the handheld device it send a request to the customer for authorization in order to add the device to the network. When the customer accepts, the handheld device will be part of the network and it starts regulate the customer's activities viz., procurement, payments, etc. There are other systems that exist similar to the above said systems can be employed for traffic control, hot spot identification, and implement effective crowded control mechanisms. All such type of systems requires hand held devices to perform analysis of real time sensed data to upsurge the computation power of devices. The measure of processing of an IoT device may vary due to heterogeneity of such devices. So, it is perceived that the intelligent schedule must be required to handle heterogeneity and dynamic environment. Hence, there is need to a scheduling algorithm that is more suitable for unpredictable nature of task execution and dynamically changing environment.

This paper is organized to have few sections, an introduction is presented in first section, and the second section describes objectives. Review and Definitions presents in sections 3.Section 4 demonstrate IoT components and Section 5 presents IoT Task model. Section 6 elaborates results and discussions. The conclusion remarks are presented in the section 7.

II. OBJECTIVES

Objective of this paper is to develop a scheduling algorithm that is self-configurable and dynamic. The features of scheduling algorithms mentioned in the [4], [9] are taken into this algorithm to achieve increased efficiency.

The algorithm proposed is dynamic and it takes care of task variances. The system automatically adds/deletes resources based on the task at hand. The proposed algorithm is tuned in such way that it can hold out without any variation in task execution. Fig 4.1 demonstrates the proposed architecture.

Further, this paper explores the learning capabilities of RT tasks and its nature that help us to develop an intelligent scheduler. The present study is to develop adaptive control strategies for task scheduling in Real Time heterogonous computing environment that will be useful to perform higher level tasks on a cloud of devices that have been configured and are made available.

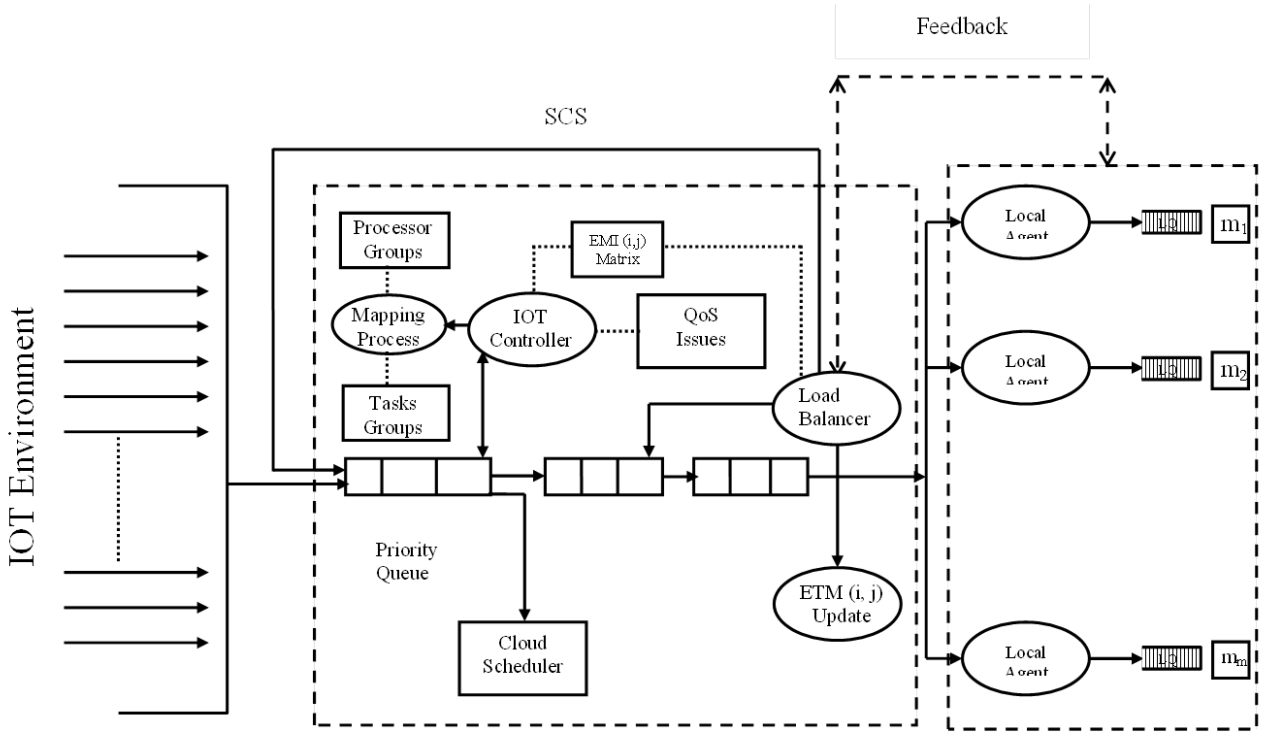

Fig. 1.IoT Schedule Architecture Diagram

III. LITERATURE REVIEW

There is a great possibility in future that, communications would be more or less is based on IoT applications. These applications are diverse that ranges from ordinary voice recognition to critical space programs. A lot of efforts have been put in by the researchers across to design an Operating System for IoT devices as there is less possibility of running IoT applications on traditional UniX or Windows operating systems, and further, the existing real time operating systems are unable to meet the demands of heterogeneity of IOT applications.

There are numerous scheduling algorithms that exist and must be adaptable under diverse computing environments [3], [11], [12], [13] but are suitable for dedicated systems. Several clustering and scheduling algorithms were established for identifying execution environment and <task, processor> pairs for efficiently [1], [2], [3], [4], [5]. The existing clustering algorithms generated k schedules where each schedule having its own property.

This literature provides a detailed comparison of the Operating systems designed for IoT devices on the basis of their architecture, scheduling methods, networking technologies, programming models, power and memory management methods, together with other features required for IoT applications [14], [15], [16], [18], [19], [20].

IV. IOT COMPONENTS

The main focus of this paper is to develop an appropriate <processor, task>pair which is dynamically configurable based on incoming tasks and these tasks are generated from different IoT devices. The main characteristics of <processor, task>pair were discussed [4], [9] to achieve better efficiency. The different IoT devices are configured with the system and their events are captured properly for given task. All such types of <processor, task>pair are scheduled in efficient manner and system automatically adds/deletes resources based on the demand of incoming tasks. The architecture shown in Fig 4.1 demonstrates with all feasible components. This paper explores learning capabilities of IoT tasks and its nature which help us to develop an intelligent <task, processor> pair.

The main objective of the proposed model is to balance the load among all feasible components which minimize the total execution time units allocated any processing element in an unrelated heterogeneous computing system.

The proposed scheduler identifies the best computing resource available to run this job and assigns the job to that processors job queue. The present scheduler also keeps track of the tasks run-time behavior compared to planned completion times and records that in ETM(i,j). The proposed scheduler uses the learned ETM (i, j) entries for future scheduling of the same/similar job and assigns the jobs.

The advantage of proposed scheduler is, it can create a cloud of resources and run the tasks as needed. The major components in the scheduler model are given in the following section.

Task Queues – A Specialized priority queuing data structures designed to monitor incoming tasks and generates the status report.

IoT Controller - is employed to accept or reject incoming tasks which in turn generates an appropriate <task, processor>pair. If proper resources are unavailable for a rejected task; it is transferred to a cloud of resource queue where task can be rescheduled.

Load Balancer (LB) - balances the load among different processing elements.

The proposed load balancing algorithm is given in 3.1:

```
Begin
1. Estimate Load on each processing element
        for (j=0; j<=n; j++)
   Queue Lenth_j = T_ij
2. If (New Task) then BC computes of all PE's
   Queue Length_j = T_ij + worst case execution time
elseif (Queue Lenght_j<T_ij) then
        Add to LIST of j
   else
        Task allocated to mapped processor
3. Repeats step 2 for j= 1 to m_j
end
```

Algorithm - Load Balancer

The executing tasks may take longer than the usual because of the a) Unpredictable tasks b) Unpredictable task behavior c) I/O delay and d) Switch time. The reasons are for indicative purpose only but are not limited to the list.

The key function of a LB is to minimize finish times of all PE's.

Let d_{ij} be the deadline of task i on PE j and this task deadline will be setting as FC parameter. TE_{ij} is total execution units allocated to processor j. Ld be the load balancing parameter which is given by

$$Ld = \sqrt{\sum_{j=1}^{m} \left(TE_j - d_{lj}\right)^2 / m} \quad (1)$$

a. ETM (i, j) Matrix:

The ETM (i, j) is the main component of this work. The current work implemented a highly adaptive, efficient and scalable version of ETM (i, j) that constantly monitors the tasks on all the registered PE's and provides constant feedback to the scheduler to facilitate better scheduling decisions. The proposed ETM (i, j) gets constant updates from the agents and its task characteristics. Based on the initial task definition and observed task behavior, skew boundaries are derived by the present research. The novelty of the proposed approach is it ignores the minor deviations from earlier observations to minimize constant updates due to environmental factors. The major advantage of this is, more intelligence can be added to ETM (i, j) for better analysis of the PE/task behavior which is based on past data of same or similar task/processors. The proposed work also assume realistic default values in a fully-automated mode as it gathers/learns more information about the task/PE's in the long run. The ETM (i, j) will be stored and it will retain its state data across multiple runs.

Local Agent: The proposed local agent (LA) runs on all the registered PE's to which the scheduler is configured to assign the tasks. The local agent allocates the tasks to the processor as defined in section 3.1. The advantage of a local agent is to notify to the scheduler whenever predefined execution time of task changes. This feature provides agent to learn the task behavior to take proactive decision.

b. Mapping Process:

The proposed mapping process (MP) maps the <*task, processor*> pair. The ETM (i,j) generates the processor cluster and the task cluster generated from the COBWEB algorithm. These two clusters are mapped using the proposed [1], [8].

Cloud Scheduler: The proposed cloud scheduler (CS) is accountable for adding/deleting resources to the existing system whenever available resources are insufficient.

c. Structure of <task, processor>

The structure of <task, processor>can be defined from the processor cluster and the task cluster composition as shown in Algorithm 5.1.The global scheduler picks up one of the processor from the group and it also tries to adjust the load among PE's.

Step 1: Submit new task with input parameters
Step 2: Compute weight of each task using COBWEB algorithm [5]
Step 3: Invoke task cluster [5]
Step 4: Repeat Step 1 to Step 3 for all incoming tasks
Step 5: Invoke processor cluster [5]
Step 6: Generate <Task, Processor > pair by comparing task cluster and processor cluster.

d. IoT Task Model

The IoT applications environment generates multiple tasks from network-connected devices, embedded devices in the physical environment gathers information from different sensors. The execution results of all tasks are recorded in ETM (i, j) matrix that forms the processor cluster [5] and the task cluster generated using COBWEB algorithms [5, 8].

V. SIMULATION, RESULT AND DISCUSSION

The Expected Execution Time ETM (i, j) is a type of inconsistent matrix. The ETM (i, j) matrix generates between 25 and 95 using the random function (Math. Random ()* 60) + 30).The different types of tasks such as periodic, aperiodic and non-real time tasks are in the ratios of 50:30:20 respectively. The researchers have performed simulations by employing MATLAB 6.0 in Assistive Technology Lab, located at Hyderabad. The simulation results are in form of tables and graphs.

The ETM (i, j) matrix generated using random function (Math. Random ()* 60) + 30) and its values are shown in the table 6.1. Invoke task cluster algorithm for generating different clusters [5] and processor cluster [8]

from ETM (i, j). The <task, processor> pairs can be generated by comparing task cluster and processor cluster using feasible conditions as shown in the figure 2.

a. Updations of ETM (i,j)

Initially, the task is allocated to random processor when time ETM (i, j) matrix generated and its values dynamically updated by local agent.

The local agent constantly monitors the execution times of tasks and updates any deviation in task or arrival of new task and its observations are list in table below.

Table 5.1 Initially ETM (i, j) matrix at time t1

Task / Processor	m0	m1	m2	m3	m4	m5
τ_0	90	59	8	84	59	58
τ_1	90	63	72	69	54	59
τ_2	90	66	63	60	4	45
τ_3	90	8	11	61	18	93
τ_4	90	68	5	19	24	91
τ_5	90	44	82	62	18	36
τ_6	90	29	85	55	52	21
τ_7	90	45	86	45	57	72
τ_8	90	59	62	60	53	45

The task cluster algorithm can be invoked with an arrival of the new task that computes weight of the new incoming task. The balancer controller allocates one the processor by comparing processor cluster and task cluster. Later ETM (i, j) updates new execution values on all processing elements that are shown in table 5.2.

Table 5.2 Initially ETM (i, j) matrix at time t2

Task / Processor	m_0	m1	m2	m3	m4	m5
τ_0	90	59	10	84	59	58
τ_1	90	63	72	69	54	59
τ_2	90	66	63	60	4	45
τ_3	90	8	11	61	18	93
τ_4	90	68	5	19	24	91
τ_5	90	44	82	62	18	36
τ_6	90	29	85	55	52	21
τ_7	90	45	86	45	57	72
τ_8	90	59	62	60	53	45
τ_9	90	6	10	57	46	51

b. Deviation in Task Execution

Initially, ETM (i, j) is shown in table 5.1. The local agent continuously monitors deviation in task execution will be noted and its value is updated in the matrix. The task execution for a given pair < τ0, m2> constantly monitors by its local agent and it notices that there is change in task execution value. The ETM (i, j) matrix can be updated by new value if there is consistent deviation across multiple runs as shown in table 5.3**.**

Table 5.3 Initially ETM (i, j) matrix at time t2

Task / Processor	m0	m1	m2	m3	m4	m5
τ_0	90	59	10	84	59	58
τ_1	90	63	72	69	54	59
τ_2	90	66	63	60	4	45
τ_3	90	8	11	61	18	93
τ_4	90	68	5	19	24	91
τ_5	90	44	82	62	18	36
τ_6	90	29	85	55	52	21
τ_7	90	45	86	45	57	72
τ_8	90	59	62	60	53	45

c. Adding new Processor(s) to the existing System

All new tasks and recurring tasks are scheduled to measure <task, processor > performance which is recorded in ETM (i,j) with multiple runs for future allocation purpose. Using this technique processor affinity can be analyzed and matrix can be updated properly as shown in table 5.4.

Table 5.4 Updated ETM (i,j) matrix at time t2

Task / Processor	m_0	m_1	m_2	m_3	m_4	m_5	m_6
τ0	90	59	2	84	59	50	58
τ_1	90	63	12	69	23	59	59
τ_2	90	66	63	60	4	45	27
τ_3	90	8	11	61	18	93	
τ_4	90	68	5	19	24	91	
τ_5	90	44	82	62	18	36	53
τ_6	90	29	85	55	52	21	44
τ_7	90	45	86	45	51	12	
τ_8	90	59	62	60	53	45	73

d. Cluster Structure

The scheduling probability defines that the probability of allocation a task to appropriate processor and executing a given task in that processor is known to be processor affinity. The graph between processor index on x-axis and scheduling probability on y-axis is generates from ETM (i,j) matrix as shown in the Figure 2.

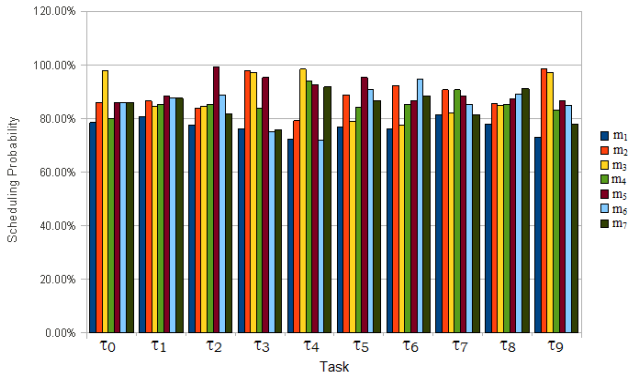

Fig. 2.Simulation Results –Scheduling task probability

Task nature can be predicted based on the weights of the incoming tasks from the simulations.

SUMMARY

This paper presents dedicated schedule, balance the load among different processing elements and maps <task, processor> pairs appropriately. This means that a dedicated schedules (Image editing, DSP processor > pair forms to increase processing speed which shows that a dedicated task allocated to particular domain processor. The system slowly learns about the incoming tasks to allocate appropriate processor. The effective control strategies are formulated to form <task, processor> pairs when computing environment changes dynamically should be

a) Dynamic version of ETM (i,j)
b) Provide Learning capability to ETM (i, j) where i is task index and j processor index.
c) Scheduler maps the <task, processor> pair appropriately.

REFERENCES

1. A. Prasanth Rao, SwathiAgarwal, Resource allocation for RT tasks using its nature, international Journal of Computer Engineering & Technology (IJCET), Volume 8, Issue 3, May-June 2017,pp.56-67,Article ID:IJCET_08_03-006, IAEME Publication, ISBN 0976-6375.
2. BabakHamidzadeh and David J.Lilja, Self-Adjusting Scheduling: An On-Line Optimization Technique for Locality Management and Load Balancing, Proceedings of the International Conference on Parallel Processing (ICPP'94),IEEE,1994; pp39-46.
3. Chin-Fu Kuo and Ying-Chi Hai, Real-Time Task Scheduling on Heterogeneous Two-Processor Systems.-H.Hsu et al. (Edu.): ICA3PP 2010, Part II LNCS 6082, 2010 (c) Springer- Verlag Berlin Heidelberg 2010, pp 68-78
4. Cho. H., Ravindran. B. and Jensen. E.D., An Optimal Real Time Scheduling Algorithm for Multiprocessors. In Proceedings of the 27th IEEE Real Time Systems Symposium, pages 101-110, 2006.
5. A. Prasanth Rao, Adaptive Control Strategies For Task Scheduling in Real Time Systems,PhD Thesis, JNTU Hyderabad, 2014.
6. Zhu XM, Lu PZ. Multi-dimensional scheduling for real-time tasks on heterogeneous clusters. Journal of Computer Science and Technology 24(3): May 2009, pp 434 – 446
7. Braun, T.D., Siegel, H.J., Beck, N., ¨ol¨ oni, L.B., Maheswaran, M., Reuther., A. I., Robertson, J. P., Theys, M. D., Yao, B., Freund, R. F., and Hensgen, D., "A comparison study of static mapping heuristics for a classify meta-tasks on heterogeneous computing systems," 8th IEEE Heterogeneous Computing Workshop (HCW'99), pp.15–29,(Apr. 1999).
8. A. Prasanth Rao, Swathi Agarwal, K.Srinivas, B. Kavitha Rani, Learning Mechanism for RT Task Scheduling, 2015 IEEE International Conference On Computational Intelligence And Computing Research 2015 Dec 10th to 12th IEEE ISBN 978-1-4799-7848-9
9. A.Prashanth Rao, A.Govardhan, Self-Configurable Real time Scheduling Algorithm for Heterogeneous Computing Systems, HPAGC 2011,CCIS 169, © Springer –Verlag Berlin Heidelberg 2011, pp. 306-315.
10. Nilsson N.J, Principles of Artificial Intelligence, Tiogo,Palo Alto, CA 1980.
11. C.Perera, A, Zaslavsky, P. Christen, A. Salehi, and D. Georgakopoulos, Sensing as a service model for smart cities supported by internet of things, Transactions on Emerging Telecommunications Technologies, pp.(99):1-13, 2013.
12. Rajeev Piyare, Internet of Things: Ubiquitous Home Control and Monitoring System using Android based Smart Phone, International Journal of Internet of Things 2013, 2(1): 5-11.
13. Melanie Swam, Sensor Mania! The Internet of Things Wearablr Computing, Objective Metrices and the Quantified Self 2.0, Journal of Sensor and Actuator Networks, 2012, 1, 217-253.
14. ChalloufSabri, LobnaKriaa, Saidane Leila Azzouz, "Comparison of IoT Constrained Devices Operating Systems: A Survey", Computer Systems and Applications (AICCSA) 2017 IEEE/ACS 14th International Conference on, pp. 369-375, 2017.
15. FarhanaJaved, Muhamamd Khalil Afzal, Muhammad Sharif, Byung-Seo Kim, "Internet of Things (IoT) Operating Systems Support Networking Technologies Applications and Challenges: A Comparative Review", Communications Surveys & Tutorials IEEE, vol. 20, no. 3, pp. 2062-2100, 2018.
16. Mahmoud H. Qutqut, Aya Al-Sakran, FadiAlmasalha, Hossam S. Hassanein, "Comprehensive survey of the IoT open-source OSs", Wireless Sensor Systems IET, vol. 8, no. 6, pp. 323-339, 2018.
17. CNS Kumar, VS Ramulu, KS Reddy, S Kotha, CM Kumar, "Spatial data mining using cluster analysis", International Journal of Computer Science & Information Technology 4 (4), pp.71-75.
18. KS Reddy, GPS Varma, MK Reddy, "An Effective Preprocessing Method for Web Usage Mining", International Journal of Computer Theory and Engineering 6 (5), pp. 412-415.
19. Santhosh Kumar, C.N., Pavan Kumar, V., Reddy, K.S., "Similarity matching of pairs of text using CACT algorithm", International Journal of Engineering and Advanced Technology 8(6), pp. 2296-2298.
20. Naga Santhosh Kumar, C.H., Reddy, K.S., "Effective data analytics on opinion mining", International Journal of Innovative Technology and Exploring Engineering 8(10), pp. 2073-2078.

AUTHORS PROFILE

Dr. A. Prasantha Rao is working as a Professor, Information Technology, Anurag Group of Institutions, Hyderabad since, 2012. He has published several articles in International Journals of repute. He has several projects in his hand.

G. Sekhar Reddy is working as a Asst. Professor, in IT Dept., at Anurag Group of Institutions Hyderabad. He has over 16 years of experience. He is currently pursuing his Ph.D from Acharya Nagarjuna University, Andhra Pradesh, India. He has guided over 50 B. Tech Projects.

Dr. Ch N Santhosh Kumar is working as a Professor in Anurag Engineering College, has over 20 years of experience in academia and 3 years in Research. He has about 15 publications in renowned international journals and conferences of repute.