Energy Efficient Scheduling of Independent Tasks on Multicore Processors with Software Controlled Dynamic Voltage Scaling

Satya Prakash, Ashish Kumar Maurya, Anil Kumar Tripathi
Department of Computer Science & Engineering
Indian Institute of Technology (BHU)
Varanasi, India
{satya.prakash.cse13, akmaurya.rs.cse14, aktripathi.cse}@iitbhu.ac.in

***Abstract*— Energy-efficient task scheduling for multicore processors is a vital design issue in parallel and distributed computing. Nowadays multicore processors are designed by using software controlled Dynamic Voltage Scaling (DVS) to reduce the energy consumption of cores. We propose a modified EETSA algorithm to schedule the independent tasks on multicore processors that have to be completed within a given common deadline. The proposed algorithm works in two phases. In the first phase, tasks are scheduled on different cores of multicore processors and in the second phase, energy consumption is optimized. The complexity of proposed algorithm is same as the EETSA. We perform the experiments on a system with a single multi-core processor with software controlled DVS having a limited set of core speeds. The results of proposed algorithm are compared with EETSA and OPT algorithms. The results show that proposed algorithm outperforms the compared algorithms regarding scheduling time and energy consumption.**

Keywords—Dynamic Voltage Scaling, Energy efficient scheduling, Multi-core processor, Task allocation.

I. INTRODUCTION

The multi-core processor architecture helps to cope up with the high-performance computing and high power consumption problems. A multi-core processor is an integrated circuit that consists of two or more independent processing units. It helps in enhancing performance, reducing power consumption and simultaneous processing of multiple tasks efficiently [1]. Multi-core processors are widely used in digital signal processing, graphics, embedded computing, network processing etc.

In computer architecture, DVS [2] is a power management technique, where the supply voltage is varied to vary the speed of core. The voltage supplied to a core can be increased or decreased depending upon the circumstances. If the computational requirement for a core is low, then voltage supply is reduced called undervolting. In a case of high computational requirement, the voltage supply is increased to increase the speed of the core called overvolting [2].

For proper utilization of multiple cores, task scheduling problem comes into the picture. In task scheduling on a multicore processor, the objective is to find the schedule of different tasks on different cores and schedule of speed/voltage of different cores to minimize the energy consumption and complete the given set of independent tasks under given deadline [3].

We have considered a system with single multi-core processor with software controlled DVS which has finite core speeds. In this paper, we have addressed energy efficient task scheduling problem. Here, the problem is to schedule a set of independent tasks on different cores considering the optimized energy consumption. We have attempted to introduce a meaningful modification in an existing algorithm for the purpose. For solving the problem, EETSA [4] algorithm is modified, and the obtained results are compared with the known EETSA and optimal (OPT) algorithms.

The rest of paper is organized as follows. In section 2, we have discussed the literature overview. Section 3 introduces about modified EETSA. In section 4, we compare the proposed algorithm with EETSA algorithm and OPT algorithm for dual-core processors and quad-core processors. Section 5 concludes the work.

II. RELATED WORK

Ample of energy efficient task scheduling algorithms are proposed in the literature for different system models.

Euiseong et al. [5] considered the problem of task scheduling in mobile real-time systems and suggested the algorithm that dynamically balances the tasks loads of different cores for optimal power consumption. To reduce leakage of power consumption, the authors have proposed Dynamic Core Scaling algorithm, which reduces the number of active cores.

Fanxin et al. [6] addressed the problem of energy-efficient scheduling of real-time tasks on cluster based multicores. They have developed an algorithm for task scheduling having timing and frequency operation as constraints. Proposed algorithm is a polynomial algorithm that calculates minimum energy consumption for a given task partition.

In [7] Manojit et al. solved the online scheduling of aperiodic real time task on a large multi-threaded multiprocessor system.

Zhang et al. [8] proposed and solved the problem of energy-efficient scheduling of real time dependent task on a

given number of variable voltage processors in two phases. In the first phase, a task is scheduled and ordered on different cores. In the second phase, voltage is properly selected to optimize energy consumption. In the proposed algorithm, voltage selection problem can be solved in polynomial time.

Jing Liu et al. [9] proposed the problem of task scheduling in heterogeneous multicore embedded systems. Although the problem is NP-hard, several heuristic techniques like ISGG and RLDG are proposed to solve the problem efficiently.

Houssam et al. [10] considered the problem of execution real-time data processing applications on heterogeneous computing platforms. The main aim of the author is to optimize the energy consumption. In the paper, they have focused on DVFS, parallelization and real-time scheduling and task allocation.

Yang et al. [3] tackled the problem of energy efficient scheduling for a chip-multiprocessor. The processor consists of DVS that can use continuously varying processor speeds with no upper bound. They solved the voltage scheduling problem for a set of independent tasks having a common deadline.

Suhaimi et al. [11] considered the problem of optimizing the total processor energy consumption of a set of nonpreemptible tasks. Tasks are executed on MPSoC without shared memory and have individual deadlines. They have proposed nonlinear programming to calculate execution speed for each task considering the optimal energy consumption.

Alexei et al. [12] solved the problem of scheduling of real-time software components onto heterogeneous cores for optimal energy consumption. They have proposed a heuristic approach that approximates the load distribution. We have attempted to achieve enhanced performance with lower energy consumption through a simple modification in the existing EETSA algorithm proposed by Mishra et al. [4] as detailed in section 3.

III. Modified Energy Efficient Task Scheduling Algorithm

In [4], Mishra et al. proposed an energy efficient task scheduling algorithm (EETSA) to solve real time task scheduling problem on multicore processors. The algorithm solves the task scheduling problem in two phases. In first phase, it schedules the task on different cores (sorted by its computational requirement) using worst fit strategy. In second phase, it optimises the energy consumption. In the proposed Modified EETSA (MEETSA) algorithm, we modify the first phase and use best fit strategy to allocate tasks on multicore processors. The second phase that optimizes energy consumption is same as EETSA algorithm.

Like EETSA, inputs to modified EETSA are p, q, m, n, t C, Q, and D. Where p is number of cores, q is possible number of discrete speed of a core, m is the number of attempts to schedule task on different cores before declaring no possible solution, n is the number of attempts to optimize the energy consumption on getting feasible solution and t is the number of independent tasks in the task set. C is an array of size t which represents the computation required by each task. Q is an array that represents the different possible speed of a core and D is the common deadline for all tasks in the task set. Modified EETSA is described in Algorithm 1 as follows.

Algorithm 1 MEETSA

```
01 Begin
02 Sort(C)
03 Assign A_1Xp, F_1Xq and H_pXq, with 0 matrix
04 Assign speed profile as D for maximum speed
05 for i = 1 to m do
06   flag = true
07   for each task j = 1 to t do
08     Assign task to different core using best fit
09     Prepare binary partition matrix for each task
10     Unable to assign task, set flag = false and break
11   end for
12   if flag = false, then
13     A ← 0_1Xp
14     random-swap(1, j)
15   else
16     break
17   end if
18 end for
19 if flag = false, then
20   return NO SOLUTION
21 else
22   Optimize energy same as EETSA
23 end if
24 end
```

In the above algorithm, computational requirements of different tasks are sorted in increasing order. After that, all tasks are scheduled on different cores using best fit and prepare a binary partition matrix, unlike in EETSA which uses worst fit strategy to schedule tasks on different cores. Binary partition matrix represents which task is allocated on which core. If a task is not scheduled on any core, two tasks are selected randomly and swapped regarding their position. Again all the steps of the proposed algorithm are repeated to schedule the tasks. This way maximum m attempts are made to schedule tasks. Even after m iterations, there are tasks that could not be scheduled on different cores, return NO SOLUTION. If we get a feasible solution, then that optimization of energy is taken up. The complexity of the MEETSA algorithm works out to be $O(t(mp + q + \log(t)) + p(t + q)(D^{pq} + n))$ that is same as for the EETSA algorithm.

IV. Experimental Results

In this section, we perform the experiments of task scheduling algorithm for symmetrical and asymmetrical task sets for dual-core processors as well as quad-core processors.

A. *Experimental results for dual-core processors*

For dual-core processor, we take p = q = 2 and for MEETSA algorithm, we take m = n = 10.

(i) Results for symmetrical sets

In the given below Table I, column t represents the number of tasks in the task set, column c represents the computation requirement for each task in the task set, and Column D gives the deadlines of the tasks in the set. Column E_{EETSA} gives energy computation of EETSA algorithm. Column T_{EETSA} shows the running time of EETSA algorithm. Column E_{OPT} gives energy computation of OPT algorithm, and column T_{OPT} gives the running time of OPT algorithm. Column E_{MEETSA} gives the energy consumption of modified EETSA using best fit strategy, and T_{MEETSA} gives the running time of the algorithm using best fit strategy. Table I shows the comparison for energy consumption and running time of EETSA, OPT and MEETSA algorithm for dual core processor. Here, it can be observed that the energy consumption is always minimum in case of OPT algorithm but the running time for OPT algorithm is very high. It can also be observed that the running time for MEETSA is also less than EETSA which is because of less comparison involved while allocating the task to different cores using best-fit strategy as compared to worst-fit strategy.

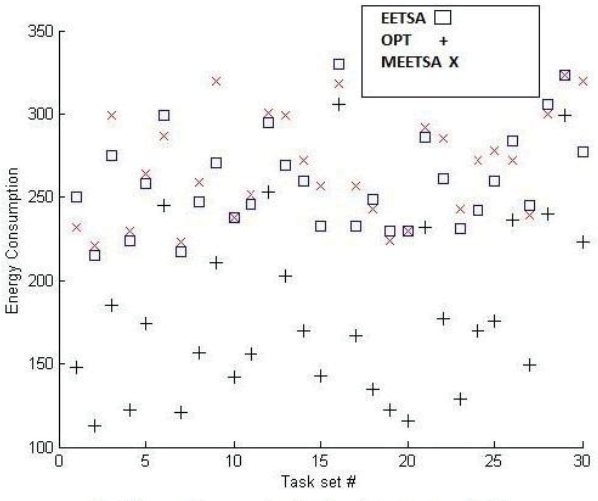

Fig. 1. Energy consumption for 5 node set task graph on dual-core

(ii) Results for asymmetrical sets

For asymmetrical task set, D = 30 is considered as a deadline. 60 tasks set are randomly generated with the number of tasks t as 5 and 10 each having 30 task sets. For the task set having a number of tasks as 5, we randomly generate tasks having computational requirement between 10 and 20. These task sets are having low variation in their computational requirement. For the tasks set having the number of tasks as 10, we randomly generate task having computational requirement between 1 and 20. These task sets are having high variation in computational requirement.

Fig. 1 and 2 shows the average comparison of average energy consumption among EETSA algorithm, OPT algorithm, MEETSA using best fit for task set having 5 and 10 tasks respectively. When the number of nodes in the task set is low then, energy consumption in case of MEETSA is more as compared to EETSA. But if we increase the number of nodes in the tasks set then the energy consumption of MEETSA approaches EETSA, and sometimes it is better than EETSA.

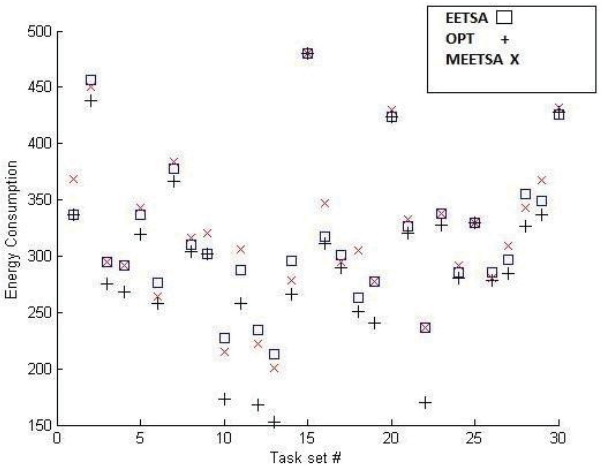

Fig. 2. Energy consumption for 10 node set task graph on dual-core

OPT algorithm takes exponential time to schedule task and energy optimization. So, the running time of OPT algorithm is very high as compared to EETSA and MEETSA. There is a very marginal difference between the running time of EETSA and MEETSA. But still, MEETSA takes less time as compared to EETSA because of less computation is required in task allocation.

B. *Experimental results for quad-core processors*

For quad-core processor, we take p = 4 and q= 2, and for MEETSA algorithm, we take m = 10 and n = 10.

(i) Results for symmetrical sets

In Table II, column t represents the number of the task in the task set, c represents the computational requirement for each task and D represents a common deadline for all tasks. Column E_{EETSA} gives energy consumption of EETSA algorithm. Column T_{EETSA} gives running time for EETSA algorithm. Column E_{OPT} shows energy consumption using OPT algorithm and column T_{OPT} shows running time of OPT algorithm. Column E_{MEETSA} and T_{MEETSA} gives energy consumption and running time for modified EETSA using best fit strategy respectively.

Table II shows the comparison of energy and running time of EETSA algorithm, OPT algorithm and MEETSA algorithm for the symmetrical task set for quad core processor. From the table, it can be observed that energy consumption of EETSA and MEETSA are same but running time of MEETSA is less than EETSA because of less computation performed to schedule task on different cores. Running time for OPT algorithm is very high because it takes exponential time to schedule task on different cores and optimizes energy.

(ii) Results for asymmetrical sets

For the asymmetrical task, we generated 60 tasks sets randomly with the number of tasks t as 4 and 6 respectively,

each having 30 tasks set. The computational requirement for these task set ranges from 1 to 10. For the task set with the number of tasks as 4, we consider the deadline as 5, and for the task set with the number of tasks as 6, we take the deadline as 6.

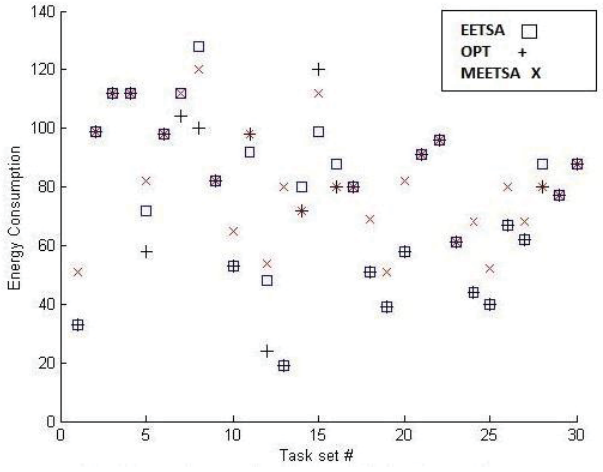

Fig. 3. Energy consumption for 4 node set task graph on quad-core

Fig 3-4 shows the comparison of energy consumption between the EETSA, OPT algorithm, MEETSA using best-fit strategy for the task set having 4 and 6 tasks respectively. From the graph of energy consumption we can see that as we increase the number of nodes in the task set, the energy consumption value, in the case of MEETSA approaches EETSA and OPT algorithm.

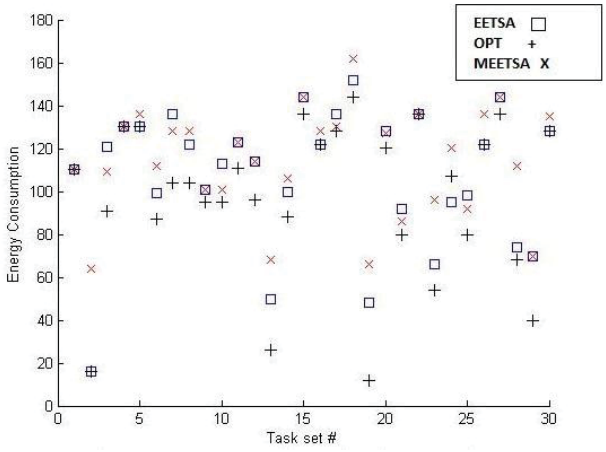

Fig. 4. Energy consumption for 6 node set task graph on quad-core

From the graph, it can be observed that the energy consumption for OPT is less as compared to EETSA and MEETSA because it is the most optimized energy and for which it takes exponential time. There is not much difference between energy consumption value of EETSA and MEETSA.

V. Conclusion

A system with a single multi-core processor with software controlled DVS that have a finite set of core speeds was considered here in this work. The problem was to find an energy efficient task scheduling, given a set of independent tasks having a common deadline. We proposed a modified energy efficient task scheduling algorithm (MEETSA) to schedule independent tasks on multicore processors. The time complexity of the proposed algorithm is $O(t(mp + q + \log(t)) + p(t + q)(D^{pq} + n))$. We have compared the energy consumption value and running time of proposed algorithm with EETSA and OPT algorithm. The proposed MEETSA is faster as compared to EETSA and OPT algorithm because of less computation performed while scheduling the task. Energy consumption value of MEETSA is more than EETSA when the number of nodes in the task set is less because in that case, it might be that all the tasks will get scheduled onto the same core. Thus, the core will have to complete the tasks at maximum speed thus consume more energy. On increasing the number of nodes in the task set all the tasks gets distributed on different cores. Hence energy consumption value approaches to that of EETSA.

References

[1] J. Fruehe, "Multicore processor technology," Reprinted from Dell Power Solutions www. dell. com/powersolutions (Obtained from the Internet on Mar. 23, 2012), pp. 67-72, 2005.

[2] P. Pillai and K. G. Shin, "Real-time dynamic voltage scaling for low-power embedded operating systems," in ACM SIGOPS Operating Systems Review, vol. 35, no. 5. ACM, 2001, pp. 89-102.

[3] C.-Y. Yang, J.-J. Chen, and T.-W. Kuo, "An approximation algorithm for energy effcient scheduling on a chip multiprocessor," in Proceedings of the conference on Design, Automation and Test in Europe-Volume 1. IEEE Computer Society, 2005, pp. 468-473.

[4] A. Mishra and A. K. Tripathi, "A monte carlo algorithm for real time task scheduling on multi-core processors with software controlled dynamic voltage scaling," Applied Mathematical Modelling, vol. 38, no. 7, pp. 1929-1947, 2014.

[5] E. Seo, J. Jeong, S. Park, and J. Lee, "Energy effcient scheduling of real-time tasks on multicore processors," IEEE transactions on parallel and distributed systems, vol. 19, no. 11, pp. 1540-1552, 2008.

[6] F. Kong, W. Yi, and Q. Deng, "Energy effcient scheduling of real-time tasks on cluster-based multicores," in Design, Automation & Test in Europe Conference & Exhibition (DATE), 2011. IEEE, 2011, pp. 1-6.

[7] M. Ghose, A. Sahu, and S. Karmakar, Energy effcient online scheduling of aperiodic real time task on large multi-threaded multiprocessor systems," in India Conference (INDICON), 2016 IEEE Annual. IEEE, 2016, pp. 1-6.

[8] Y. Zhang, X. S. Hu, and D. Z. Chen, "Task scheduling and voltage selection for energy minimization," in Proceedings of the 39th annual Design Automation Conference. ACM, 2002, pp. 183-188.

[9] J. Liu, K. Li, D. Zhu, J. Han, and K. Li, "Minimizing cost of scheduling tasks on heterogeneous multicore embedded systems," ACM Transactions on Embedded Computing Systems (TECS), vol. 16, no. 2, p. 36, 2016.

[10] H.-E. Zahaf, A. E. H. Benyamina, R. Olejnik, and G. Lipari, "Energy-effcient scheduling for moldable real-time tasks on heterogeneous computing platforms," Journal of Systems Architecture, 2017.

[11] S. A. Ishak and H. Wu, "Energy-aware task scheduling with precedence and deadline constraints on mpsocs," in High Performance Computing and Communications; IEEE 14th International Conference on Smart City; IEEE 2nd International Conference on Data Science and Systems (HPCC/SmartCity/DSS), 2016 IEEE 18th International Conference on. IEEE, 2016, pp. 1163-1172.

[12] A. Colin, A. Kandhalu, and R. Rajkumar, "Energy-effcient allocation of real-time applications onto heterogeneous processors," in Embedded and Real-Time Computing Systems and Applications (RTCSA), 2014 IEEE 20th International Conference on. IEEE, 2014, pp. 1-10.

TABLE I. ENERGY CONSUMPTION AND RUNNING TIME FOR DUAL-CORE PROCESSOR FOR SYMMETRICAL TASKS

	t	C	D	E_{EETSA}	T_{EETSA}	E_{OPT}	T_{OPT}	E_{MEETSA}	T_{MEETSA}
1	5	15	30	255	0.000443	213	0.507	304	0.000054
2	5	20	30	400	0.000016	400	0.474	400	0.000082
3	6	20	30	480	0.000023	480	0.821	480	0.000049
4	6	25	40	570	0.001160	570	3.042	570	0.000441
5	7	25	50	704	0.000043	704	13.783	704	0.000067
6	7	30	60	840	0.000048	840	32.926	840	0.000101
7	8	10	30	272	0.000456	200	1.946	320	0.000052
8	8	15	30	480	0.000016	480	2.199	480	0.000051
9	9	15	40	513	0.000761	513	10.618	513	0.000472
10	9	20	50	720	0.000047	720	29.001	720	0.000134
11	10	15	40	570	0.000483	570	13.557	570	0.000450
12	10	20	50	800	0.000037	800	38.764	800	0.000084

TABLE II. ENERGY CONSUMPTION AND RUNNING TIME FOR QUAD-CORE PROCESSOR FOR SYMMETRICAL TASKS

	t	C	D	E_{EETSA}	T_{EETSA}	E_{OPT}	T_{OPT}	E_{MEETSA}	T_{MEETSA}
1	5	15	30	72	0.000012	72	0.0044	72	0.000006
2	5	20	30	18	0.000125	18	0.2604	72	0.000042
3	6	20	30	16	0.000385	16	13.524	64	0.000106
4	6	25	40	20	0.000411	20	9.827	74	0.001006
5	7	25	50	61	0.021703	37	34.163	97	0.001034
6	7	30	60	120	0.000164	120	4.801	120	0.000046
7	8	10	30	144	0.000069	144	22.500	144	0.000041
8	8	15	30	150	0.008387	150	162.514	150	0.003070
9	9	15	40	200	0.000087	200	304.821	200	0.000067
10	9	20	50	224	0.000179	224	761.811	224	0.000105