ENERGY EFFICIENT TASK SCHEDULING OF SEND-RECEIVE TASK GRAPHS ON DISTRIBUTED MULTI-CORE PROCESSORS WITH SOFTWARE CONTROLLED DYNAMIC VOLTAGE SCALING

Abhishek Mishra and Anil Kumar Tripathi

Department of Computer Engineering, Institute of Technology, Banaras Hindu University, Varanasi, India 221 005

{abhishek.rs, aktripathi}.cse@itbhu.ac.in

ABSTRACT

In this paper we propose a model of distributed multi-core processors with software controlled dynamic voltage scaling. We consider the problem of energy efficient task scheduling with a given deadline on this model. We consider send-receive task graphs in which the initial task sends data to multiple intermediate tasks, and the final task collects the data from these intermediate tasks with the restriction that the initial and final tasks should be assigned on the same core.

KEYWORDS

Distributed System, Dynamic Voltage Scaling, Energy Efficient Scheduling, Multi-Core Processors

1. Introduction

Multi-core processors are designed to meet the growing challenges of high performance computing applications (Fruehe [11], Schauer [20]). A multi-core processor has more than one core on a single chip. The result is that this greatly reduces the hardware size without compromising on the performance. The communication delay between cores is negligible as compared to multiprocessors. This results in improvement of computational performance with minimal hardware.

By using dynamic voltage scaling (DVS) we can vary the supply voltage that results in changing the speed of cores (Pillai and Shin [19]). By using DVS we can reduce the energy consumption when the computational load is low by reducing the speed of cores. On the other hand when the computational requirement is high we can increase the speed of cores at the cost of increased energy consumption. This gives rise to the voltage scheduling problem (Ishihara and Yasuura [17]) for multi-core processors when a deadline is given.

Our motivation for this work comes from systems that use more than one multi-core processor. For example the *SPARC ENTERPRISE T5440* servers (Fujitsu [12]) use four *UltraSPARC T2 Plus* (Fujitsu [12]) processors that have up to *8* cores. This gives rise to newer problems for energy efficient task scheduling.

We propose a model of distributed multi-core processors with software controlled DVS that has a finite set of discretely available core speeds. We consider the problem of energy efficient task scheduling with a given deadline on this model. We consider send-receive task graphs in which the initial task sends data to multiple intermediate tasks, and the final task collects the data from these intermediate tasks with the restriction that the initial and final tasks should be assigned on the same core.

DOI : 10.5121/ijcsit.2011.3215

The rest of the paper is organized as follows: section 2 presents existing work from the literature about DVS enabled scheduling. In section 3 we propose a model for distributed multi-core processors with software controlled DVS that has a finite set of discretely available core speeds. In section 4 we consider the properties of the send-receive task graph that has to be allocated on the distributed multi-core processor model. In section 5 we consider the *Energy Efficient Task Scheduling on Distributed Multi-Core Processors Problem (EETSD)* on the proposed processor model. We conclude in section 6.

2. Existing Work

There are some uniprocessor energy efficient scheduling algorithms proposed in the literature. For example: Aydin et al. [4], Chen et al. [8, 9], Irani et al. [16], Ishihara and Yasuura [17], Alvarez et al. [3], Yao et al. [24], Yun and Kim [25], Yang et al. [23].

Ishihara and Yasuura [17] solved the problem of voltage scheduling on a uniprocessor with DVS that can use only a small number of discretely variable voltages. Chen, Kuo, and Yang [8] solved the problem of profit-driven uniprocessor scheduling with energy and timing constraints. Yao, Demers, and Shankar [24] solved the problem of minimum energy scheduling of independent jobs with arrival times, deadlines, and a given amount of computation on a uniprocessor with variable speeds under the assumption that the power function is a convex function of the processor speed. Irani, Shukla, and Gupta [16] extended the previous problem (Yao et al. [24]) to include the case in which a processor can go into a sleep state. Chen, Kuo, and Lu [9] extended the problem of Yao et al. [24] for the case of jobs with precedence constraints. They considered the case of weakly dynamic voltage scheduling in which speed change is not allowed in the middle of processing a job.

There are also some multiprocessor energy efficient scheduling algorithms proposed in the literature. For example: Anderson and Baruah [2], Chen et al. [7], Gruian [13], Gruian and Kuchcinski [14], Mishra et al. [18], Zhang et al. [26], Zhu et al. [27], Yang et al. [22].

Yang, Chen, and Kuo [22] solved the problem of energy consumption minimization for a chip-multiprocessor with DVS that can use continuously varying processor speeds with no upper bound. Zhang, Hu, and Chen [26] solved the problem of energy efficient scheduling of real time dependent tasks on a given number of variable voltage processors.

3. System Model

3.1. Multi-Core Processors with Software Controlled DVS

Enhanced Intel(R) SpeedStep(R) Technology [15], and *AMD PowerNow!(TM) Technology* [1] are some examples of software controlled DVS. Our model is suitable for multi-core processors that are having a small number of cores with software controlled DVS and also having a small number of discretely available core speeds. Some examples of this kind of processors are: *Enhanced Intel(R) SpeedStep(R) Technology* [15] for the Intel(R) Pentium(R) M processor supports processor speeds of 600 MHz to 1.6 GHz with a step of 200 MHz. *AMD PowerNow!(TM) Technology* [1] supports the complete frequency operating range of the processor in use allowing steps of *33* or *50* MHz from an absolute low of *133* or *200* MHz.

3.2. Notation

Let N denote the set of natural numbers. Let R denote the set of real numbers. Let $Z_{<} : R \times R \rightarrow \{0, 1\}$ be a function such that $Z_{<}(x, y)$ is 1 if $x < y$, otherwise it is 0. Let $Z_{\leq} : R \times R \rightarrow \{0, 1\}$ be a function such that $Z_{\leq}(x, y)$ is 1 if $x \leq y$, otherwise it is 0. Let $Z_{=} : N \times N \rightarrow \{0, 1\}$ be a function such that $Z_{=}(m, n)$ is 1 if $m = n$, otherwise it is 0. Let $Z_{v} : \{0, 1\} \times \{0, 1\} \rightarrow \{0, 1\}$ be a function such that $Z_{v}(m, n)$ is 0 for $m = n = 0$, otherwise it is 1. *max()* is the function used to return the maximum of the input parameters.

3.3. Distributed Multi-Core Processor Model

We take the power consumption function of a multi-core processor same as in Chandrakasan et al. [6], Weiser et al. [21], and Yang et al. [22]:

$$P(s) = \alpha s^3, \quad (1)$$

where α is a constant.

The energy consumed by a core running at a speed of s during time t is assumed to be given by $P(s)t$. The overheads in changing the supply voltages are assumed to be negligible. It is assumed that any core can be taken into a sleep mode with $s = 0$, but all of non-sleeping cores must run at the same speed (Yang et al. [22]). The computational work done c in cycles of a core running at a speed of s during time t is assumed to be given by:

$$c = st. \quad (2)$$

Our distributed multi-core processor model has software controlled DVS. Frequent speed/voltage switching may cause unnecessary overhead on the system. Fine-grained control over power management can be achieved by setting periodic checkpoints of time period δt at which the DVS software checks whether to switch the speed/voltage or not. As an example, the *Crusoe™ LongRun™ Power Management* [10] for the Crusoe™ processor supports upto 200 speed/voltage changes per second. Azevedo et al. [5] proposed a profile-based dynamic voltage scheduling heuristic using program checkpoints. Program checkpoints indicate places in the code where the core speed/voltage should be recalculated and they are generated at compile time. One of the advantages of using periodic checkpoints over program checkpoints is that the periodic checkpoints can be set at the run time.

The DVS software is assumed to be implemented as a periodic process that wakes up periodically to check if there is a need to change the voltage, and otherwise it sleeps. Therefore after finishing its computation, a core cannot go into the sleep mode abruptly. The core has to wait for the next periodic invocation of the DVS software. It will remain idle wasting the energy for the remaining period of time.

In our distributed multi-core processor model there are r multi-core processors. Processors are numbered from *1* to r. Each multi-core processor has p homogeneous cores. On each processor, the cores are numbered from *1* to p. The DVS software is assumed to be a periodic process so that the supply voltage can change only in steps of a certain amount of time δt. Without loss of generality we take this time step as our unit of time ($\delta t = 1$). There are q possible core speeds (non-zero) that are given by the set Q:

$$Q = \{s_i \mid (i \in [1, q] \cap N) \wedge (s_i \in N)\}. \quad (3)$$

4. Send-Receive Task Graphs

Let there be $t+2$ tasks given by the set T:

$$T = \{T_i \mid i \in [0, t+1] \cap N\}, \quad (4)$$

where T_0 is the initial task, T_{t+1} is the final task, and the remaining tasks are the intermediate tasks. For $i \in [0, t+1] \cap N$, let c_i cycles of a core ($c_i \in N$) be the computational requirement of task T_i, and let h_i time units ($h_i \in N$) be the migration overhead of task T_i. By migration overhead of a task we mean the time taken to migrate the task from one processor to another processor (inter-processor migration). Intra-processor (between different cores on the same processor) migration overhead of tasks is assumed to be negligible. Let:

$$C = \{c_i \mid (i \in [0, t+1] \cap N) \wedge (c_i \in N)\}, \quad (5)$$

$$H = \{h_i \mid (i \in [0, t+1] \cap N \wedge (h_i \in N)\}. \quad (6)$$

For $i \in [1, t] \cap N$, there is a communication requirement of a_i time units ($a_i \in N$) from T_0 to T_i (for inter-processor communications), and a communication requirement of b_i time units ($b_i \in N$) from T_i to T_{t+1} (for inter-processor communications). Intra-processor communication overhead is assumed to be negligible. Let:

$$A = \{a_i \mid (i \in [1, t] \cap N \wedge (a_i \in N)\}, \quad (7)$$

$$B = \{b_i \mid (i \in [1, t] \cap N \wedge (b_i \in N)\}. \quad (8)$$

5. The Energy Efficient Task Scheduling on Distributed Multi-Core Processors Problem (EETSD)

We assume non-preemptive scheduling. We also assume that a task starts as soon as possible whenever it finds idle time on the (proc, core) to which it is allocated. The send-receive task set T is given. The tasks are initially allocated on (proc *1*, core *1*). A deadline of D time units ($D \in N$) is given. The *Energy Efficient Task Scheduling on Distributed Multi-Core Processors Problem (EETSD)* is to find an allocation of tasks to (proc, core) and speed scheduling of processors so as to minimize the energy consumed with all tasks finishing their computations within the deadline and with the restriction that T_0 and T_{t+1} cannot be migrated.

For $i \in [0, t+1] \cap N$, let (n_i, m_i) be the (proc, core) on which the task T_i is allocated. For the initial and final tasks we have:

$$n_0 = 1, \quad (9)$$

`

$$m_0 = 1, \quad (10)$$

$$n_{t+1} = 1, \quad (11)$$

$$m_{t+1} = 1. \quad (12)$$

For $i \in [1, t] \cap N$ (intermediate tasks), we have:

$$1 \leq n_i \leq r, \quad (13)$$

$$1 \leq m_i \leq p. \quad (14)$$

Let:

$$N = \{n_i \mid i \in [0, t+1] \cap N\}, \quad (15)$$

$$M = \{m_i \mid i \in [0, t+1] \cap N\}. \quad (16)$$

For $i \in [0, t+1] \cap N$, let g_i be the start time, and let f_i be the finish time of task T_i. We assume that the initial task T_0 starts at time *0*:

$$g_0 = 0. \quad (17)$$

Let:

$$G = \{g_i \mid i \in [0, t+1] \cap N\}. \quad (18)$$

Let $S :([1, r] \cap N) \times [0, D] \rightarrow Q \cup \{0\}$ be the speed profile of processors. For $i \in [0, t+1] \cap N$, $S(n_i, t)$ gives the speed of processor n_i at time t.

For $i \in [0, t+1] \cap N$, we have the following work constraints and the deadline constraints for the task T_i:

$$\int_{gi}^{fi} S(n_i, t)\, dt = c_i, \quad (19)$$

$$fi \leq D. \quad (20)$$

For $i \in [1, t] \cap N$, the intermediate tasks T_i can start their execution only after receiving the communication from the initial task T_0:

$$\max (h_i Z_{<}(1, n_i), f_0 + a_i Z_{<}(1, n_i)) \leq g_i. \quad (21)$$

For $i \in [1, t] \cap N$, the final task T_{t+1} can start its execution only after receiving the communications from the intermediate tasks T_i:

$$f_i + b_i Z_<(1, n_i) \leq g_{t+1}. \quad (22)$$

For $(i, j) \in ([0, t+1] \cap N) X ([0, t+1] \cap N)$, if the tasks T_i and T_j are allocated on the same (proc, core) with the task T_i starting earlier, then it must also finish before the task T_j can start its execution:

$$f_i Z_=(n_i, n_j) Z_=(m_i, m_j) Z_<(g_i, g_j) \leq g_j Z_=(n_i, n_j) Z_=(m_i, m_j) Z_<(g_i, g_j). \quad (23)$$

Let $X_{r X p X D}$ be the 3-dimensional binary matrix for busy time slots. For $(u, v, w) \in ([1, r] \cap N) X ([1, p] \cap N) X ([1, D] \cap N)$, x_{uvw} is *1* if the core v on processor u is running in the w^{th} time slot $([w-1, w])$, otherwise it is *0*. The energy consumed E is given by:

$$E = \alpha \sum^{D}_{w=1} \sum^{r}_{u=1} (\sum^{p}_{v=1} x_{uvw}) S(u, w-1)^3. \quad (24)$$

If no task is running in the time slot $[w-1, w)$ on (proc u, core v), then that core should be in the sleep mode. Let:

$$y_{uvw} = \sum^{t+1}_{i=0} (1 - Z_v(Z_<(f_i, w-1), Z_\leq(w, g_i))) Z_=(n_i, u) Z_=(m_i, v), \quad (25)$$

then we have:

$$x_{uvw} = Z_<(0, y_{uvw}). \quad (26)$$

Definition 1. Given the input (A, B, C, D, H), the Energy Efficient Task Scheduling on Distributed Multi-Core Processors Problem (EETSD) is to find (N, M, G, S) such that the constraints (9), (10), (11), (12), (13), (14), (17), (19), (20), (21), (22), (23), (25), and (26) are satisfied while also minimizing the energy consumed (24).

6. Conclusion

We proposed a model of distributed multi-core processors with software controlled DVS that has a finite set of discretely available core speeds. We considered the problem of energy efficient task scheduling with a given deadline on this model. We considered send-receive task graphs in which the initial task sends data to multiple intermediate tasks, and the final task collects the data from these intermediate tasks with the restriction that the initial and final tasks should be assigned on the same core. We formulated the *Energy Efficient Task Scheduling on Distributed Multi-Core Processors Problem (EETSD)*.

There are many problems for future work. The first problem to consider is to find the complexity of the *EETSD* problem. We have to find whether the *EETSD* problem is NP-Complete or NP-Hard. If the *EETSD* problem is NP-Complete or NP-Hard, then the second problem to consider is to look for some heuristics for solving the problem. The third problem to consider is to look for approximation algorithms for the *EETSD* problem. If we are not able to find the approximation algorithm, then the fourth problem to consider is to find whether the *EETSD* problem is inapproximable or not.

Acknowledgements

The authors would like to thank the anonymous referee for providing helpful suggestions for improving the quality of the paper.

References

[1] Advanced Micro Devices, Inc., (2000) AMD PowerNow!™ Technology, Informational White Paper.

[2] J. H. Anderson, & S. K. Baruah, (2004) "Energy-efficient synthesis of periodic task systems upon identical multiprocessor platforms", *In Proceedings of the 24th International Conference on Distributed Computing Systems*, pp. 428-435.

[3] P. Mejia-Alvarez, E. Levner, & D. Mosse, (2004) "Adaptive scheduling server for power-aware real-time tasks", *ACM Transactions on Embedded Computing Systems,* Vol. 3, No. 2, pp. 284-306.

[4] H. Aydin, R. Melhem, D. Mosse, & P. Mejia-Alvarez, (2001) "Dynamic and aggressive scheduling techniques for power-aware real-time systems", *In Proceedings of the 22nd IEEE Real-Time Systems Symposium*, pp. 95-105.

[5] A. Azevedo, I. Issenin, R. Cornea, R. Gupta, N. Dutt, A. Veidenbaum, & A. Nicolau, (2002) "Profile-based dynamic voltage scheduling using program checkpoints", *In Proceedings of Design, Automation and Test in Europe Conference and Exhibition*, pp. 168-175.

[6] A. Chandrakasan, S. Sheng, & R. Broderson, (1992) "Lower-power CMOS digital design", *IEEE Journal of of Solid-State Circuit,* Vol. 27, No. 4, pp. 473-484.

[7] J.-J. Chen, H.-R. Hsu, K.-H. Chuang, C.-L. Yang, A.-C. Pang, & T.-W. Kuo, (2004) "Multiprocessor energy-efficient scheduling with task migration considerations", *In Proceedings of the 16th Euromicro Conference on Real-Time Systems*, pp. 101-108.

[8] J.-J. Chen, T.-W. Kuo, & C.-L. Yang, (2004) "Profit-driven uniprocessor scheduling with energy and timing constraints", *In ACM Symposium on Applied Computing, ACM Press*, pp. 834-840.

[9] J.-J. Chen, T.-W. Kuo, & H.-I. Lu, (2005) "Power-Saving Scheduling for Weakly Dynamic Voltage Scaling Devices", *In Algorithms and Data Structures, Lecture Notes in Computer Science, Volume 3608/2005, Springer*, pp. 338-349.

[10] M. Fleischmann, (2001) Crusoe™ LongRun™ Power Management: Dynamic Power Management for Crusoe™ Processors.

[11] J. Fruehe, (2005) "Planning Considerations for Multicore Processor Technology", *Dell Power Solutions,* May 2005, pp. 67-72.

[12] Fujitsu, (2009) SPARC ENTERPRISE T5440 SERVER ARCHITECTURE, White Paper.

[13] F. Gruian, (2001) "System-Level Design Methods for Low-Energy Architectures Containing Variable Voltage Processors", *In Power-Aware Computing Systems, Lecture Notes in Computer Science, Volume 2008/2001, Springer*, pp. 1-12.

[14] F. Gruian, & K. Kuchcinski, (2001) "Lenes: Task scheduling for low energy systems using variable supply voltage processors", *In Proc. Asia South Pacific Design Automation Conference*, pp. 449-455.

[15] Intel(R) Corporation, (2004) Enhanced Intel(R) SpeedStep(R) Technology for the Intel(R) Pentium(R) M Processor, White Paper.

[16] S. Irani, S. Shukla, & R. Gupta, (2003) "Algorithms for power savings", In Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, pp. 37-46.

[17] T. Ishihara & H. Yasuura, (1998) "Voltage scheduling problems for dynamically variable voltage processors", *In Proceedings of the 1998 international symposium on low power electronics and design*, pp. 197-202.

[18] R. Mishra, N. Rastogi, D. Zhu, D. Mosse, & R. Melhem, (2003) "Energy aware scheduling for distributed real-time systems", *In International Parallel and Distributed Processing Symposium*, pp. 21.2.

[19] P. Pillai, & K. G. Shin, (2001) "Real-time dynamic voltage scaling for low-power embedded operating systems", *In ACM Symposium on Operating Systems Principles*, pp. 89-102.

[20] B. Schauer, (2008) "Multicore Processors - A Necessity", *ProQuest Discovery Guides,* September 2008, pp. 1-14.

[21] M. Weiser, B. Welch, A. Demers, & S. Shenker, (1994) "Scheduling for reduced CPU energy", *In Proceedings of Symposium on Operating Systems Design and Implementation*, pp. 13-23.

[22] C.-Y. Yang, J.-J. Chen, & T.-W. Kuo, (2005) "An approximation Algorithm for Energy-Efficient Scheduling on A Chip Multiprocessor", *In the Eighth ACM/IEEE Conference of Design, Automation, and Test in Europe (DATE)*, pp. 468-473.

[23] C.-Y. Yang, J.-J. Chen, L. Thiele, & T.-W. Kuo, (2010) "Energy-Efficient Real-Time Task Scheduling with Temperature-Dependent Leakage", *In the ACM/IEEE Conference of Design, Automation, and Test in Europe (DATE)*, pp. 9-14.

[24] F. Yao, A. Demers, & S. Shankar, (1995) "A scheduling model for reduced CPU energy", *In Proceedings of the 36th Annual Symposium on Foundations of Computer Science, IEEE*, pp. 374-382.

[25] H.-S. Yun, & J. Kim, (2003) "On energy-optimal voltage scheduling for fixed-priority hard real-time systems", *ACM Transactions on Embedded Computing Systems,* Vol. 2, No. 3, pp. 393-430.

[26] Y. Zhang, X. Hu, & D. Z. Chen, (2002) "Task scheduling and voltage selection for energy minimization", *In Annual ACM/IEEE Design Automation Conference*, pp. 183-188.

[27] D. Zhu, R. Melhem, & B. Childers, (2001) "Scheduling with dynamic voltage/speed adjustment using slack reclamation in multi-processor realtime systems", *In Proceedings of IEEE 22nd Real-Time System Symposium*, pp. 84-94.

Authors

Abhishek Mishra is pursuing his Ph.D. degree in Computer Engineering at Institute of Technology, Banaras Hindu University. He received his Bachelor degree in Computer Engineering from the same institute in 2003. His current research interests include approximation algorithms and combinatorial optimization.

Anil Kumar Tripathi is Professor of Computer Engineering at Institute of Technology, Banaras Hindu University. He received his Ph.D. degree in Computer Engineering from the same institute; and M.Sc. Engg. (Computer) degree from Odessa National Polytechnic University, Ukraine. His research interests include parallel and distributing computing, and software engineering.