How Hard Is Partitioning for the Sporadic Task Model?

Nathan Fisher
Department of Computer Science
Wayne State University
Detroit, MI USA
Email: fishern@cs.wayne.edu

***Abstract*—Partitioning n independent sporadic real-time tasks among m identical processors is known to be an NP-hard in the strong sense (by transformation from bin-packing). Therefore, current research on partitioning has focused on developing and analyzing various heuristics and approximation algorithms. However, currently only "loose" fundamental limits of approximation (trivially based on the known limits for bin packing and partitioning periodic tasks) are known for partitioning sporadic tasks . In this position paper, we briefly summarize known resource-augmentation approximation ratio results for sporadic task systems and argue for further theoretic investigation of approximation schemes and lower bounds for this problem. We believe the results of such an investigation will be invaluable (beyond their theoretic implications) to the real-time multicore system designer; such results will inform the system designer on the relative benefits and disadvantages of various task allocation strategies.**

***Index Terms*—partitioned scheduling; sporadic task systems; resource augmentation; approximation algorithms**

I. Introduction

Multicore processor architectures are quickly becoming the platform of choice in a variety of computational domains. The success of the multicore architecture is due, in part, to well-known benefits such as increased concurrency, decreased power-consumption, and better heat-dissipation. Recently, several chip manufacturers (e.g., Intel, Freescale, and P.A. Semi) have released symmetric multi-processing (SMP)-based multicore architectures for the embedded systems domain [22]. The widespread introduction and success of these multicore platforms in embedded systems (and other real-time system domain areas) has elevated the need for effective multiprocessor real-time scheduling algorithms and analysis. However, from a scheduling-theoretic perspective, *only* the multiprocessor scheduling of the simplest of task models or applications is well understood; for more complex task models or applications, numerous open questions remain. To completely and effectively utilize the benefits of multicore processors for complex real-time systems, it is evident that further investigation of multiprocessor scheduling of general task models is required. Furthermore, real-time multicore system designers are also interested in accounting for other real-world complexities (e.g., shared global resources) along with more general task models; hence, it behooves us to have a complete fundamental understanding of multiprocessor scheduling for general task models that may be used as a foundation for approaches that address more complex system issues.

§**Our Context.** A useful general model of real-time work is the *sporadic task model* [20]. A sporadic task, denoted τ_i, is characterized by a three tuple, (e_i, d_i, p_i) where e_i is the *worst-case execution requirement*, d_i is the *relative deadline*, and p_i is the *minimum inter-arrival separation parameter* (historically, called the "period"). A sporadic task τ_i may produce a (potentially infinite) sequence of jobs where each job has an execution requirement of e_i time units and must complete d_i time units after its arrival. The first job of τ_i may arrive at any time after system-start time; however, successive jobs of τ_i must arrive at least p_i time units apart. A sporadic task system $\tau \stackrel{\text{def}}{=} \{\tau_1, \tau_2, \ldots, \tau_n\}$ is a collection of n sporadic tasks. Two special subclasses of sporadic task systems are *constrained-deadline* and *arbitrary-deadline* sporadic task systems. A constrained-deadline sporadic task system have $d_i \leq p_i$ for each task $\tau_i \in \tau$. An arbitrary-deadline sporadic task system places no restriction on the relative value of a task's relative deadline or period parameter. The sporadic task model is a generalization of other simpler models such as the *Liu and Layland (LL) task model* [18] which implicitly required that the relative deadline of each task equal its period. LL task systems may be considered a subclass of sporadic task systems also called *implicit-deadline* sporadic task systems.

For multiprocessor scheduling, two common scheduling paradigms are *global* and *partitioned*. Global scheduling, alternatively, permits that a job may migrate freely between processors. In partitioned scheduling, each task is statically assigned to a processor (prior to system runtime) and all jobs generated by the task execute only on its assigned processor. For either multiprocessor scheduling paradigm, the scheduling algorithm must determine at every moment which jobs (among the set of active jobs) will execute on the processing platform. Priority-driven scheduling algorithms determine the order of execution among jobs by assigning each job a priority based on a particular rule. For example, the earliest-deadline-first (EDF) scheduling algorithm [18] prioritizes jobs according to their absolute deadline. The deadline-monotonic (DM) scheduling algorithm [17] prioritizes jobs proportional to the inverse of the relative deadline of its task. For this paper, we focus only on partitioned scheduling of sporadic task systems where either EDF or DM is used to schedule each individual processor.

A task system τ is said to be *feasible* on a multiprocessor platform, if for any legal sequence of job arrivals of τ there exists a schedule on the platform in which all deadlines are meet. A task system τ is A-*schedulable* if a given multiprocessor scheduling algorithm A meets all job deadlines for τ on the platform for all possible legal arrival sequences of τ. A multiprocessor scheduling algorithm A is *optimal*, if, given any task τ feasible on a given processing platform, τ is also A-schedulable. All known tractable solutions for the partitioning of sporadic task system are not optimal. Therefore, researchers have used concepts from approximation to measure the relative theoretic efficacy of various partitioning approaches. *Resource augmentation* [23] is one such technique for comparing the relative "goodness" of multiprocessor scheduling algorithm. Resource augmentation works by comparing a given scheduling algorithm A against the performance of a hypothetically optimal scheduling algorithm. The resource-augmentation metric is as follows: a scheduling algorithm A has a resource-augmentation speed-approximation ratio of $\rho \geq 1$, if, for any task system τ that is feasible upon a multiprocessor platform of m unit-speed processors, τ is guaranteed to be A-schedulable on a platform of m processors each of speed ρ.

§**Paper Objectives.** The goal of our paper is two-fold: (1) *briefly summarize known theoretical results concerning the partitioned scheduling of sporadic real-time tasks* and (2) *identify important open theoretic questions for future research*. To accomplish this, the remainder of the paper is organized as follows. Section II briefly describes the computational challenges of partitioning sporadic task systems. Section III summarizes the known research on partitioning. Section IV proposes some open questions whose solutions would (at least partially) address the difficulty of approximation for partitioning.

II. CHALLENGES

Three fundamental computational challenges are present in the development of partitioning algorithms for multiprocessor platforms:

1) **Bin-packing is NP-complete in the strong sense**: The bin-packing problem (e.g., see [16]) determines whether n one-dimensional (differently) sized items can be packed into m bins each of size B. Since the bin-packing problem is polynomially transformable to partitioning for LL tasks, partitioning LL tasks is also NP-complete in the strong sense. This intractability result extends to partitioning of sporadic tasks since general sporadic tasks are a generalization of LL tasks.
2) **Pseudo-polynomial or exponential time "packing criteria"**: For EDF-based partitioning scheduling of LL tasks, the condition that must be satisfied for a set of tasks $\Gamma \subseteq \tau$ to be safely "packed" together on the same processor (i.e., the tasks of Γ will always meet all deadlines on the assigned processor according to EDF) is that $\sum_{\tau_i \in \Gamma} e_i/p_i \leq 1$. Thus, it can be determined whether a task can be assigned to a processor in constant-time during the assignment phase of the partitioning algorithm. Furthermore, the above condition is an exact condition for EDF-scheduling of LL tasks. However, for DM-scheduling of LL tasks or both EDF and DM scheduling of sporadic tasks, all known exact packing conditions require pseudo-polynomial or exponential time in the worst-case.
3) **Multiple ordering criteria**: Most polynomial-time heuristics and approximation algorithms for bin-packing work by ordering the items according the size of the item. Then, items are packed into the bins according to this set order. This approach does not easily work for sporadic task systems. Specifically, what is the size of the item? Execution time? Utilization? There are numerous possible choices for ordering criteria. Recent research [11], [15] has ordered task in non-decreasing relative deadline order; however, whether this is the best choice is unclear.

III. KNOWN RESULTS

For LL task systems, multiprocessor scheduling is well understood for both the global [1], [9], [24] and partitioned [19], [21] scheduling paradigms. For global scheduling of LL tasks, there are optimal scheduling algorithms [9]. For both EDF and DM based partitioning, there are known algorithms with nearly "tight" resource-augmentation speed-approximation ratios equal to $2 - \frac{1}{m}$ [2], [19]. In a slightly different approximation setting, Eisenbrand and Rothvoß [14] show the following: if task system τ is feasible on m unit-speed processors, then for any fixed $\epsilon > 0$ there exists algorithm that can schedule τ (according to DM or any other static-priority uniprocessor algorithm) on at most $(1+\epsilon)\cdot m+1$ processors each of speed $(1+\epsilon)$. The preceding algorithm is known as a polynomial-time approximation scheme since its time complexity is polynomial in the number of tasks. Furthermore, they show that unless P=NP, an asymptotic fully polynomial-time approximation scheme (FPTAS)[1] cannot exist for DM-based partitioned scheduling of LL tasks.

Only recently have researchers begun to develop schedulability analysis for multiprocessor scheduling of sporadic task systems. For global scheduling of sporadic task systems, sufficient schedulability conditions have been developed for both global EDF [4], [7], [12] and DM [5], [10], [13]. Similarly, sufficient schedulability conditions have been developed for polynomial-time partitioned scheduling when each processor is scheduled according to EDF [11] or DM [15]. The current best-known resource-augmentation approximation ratios for partitioned or global scheduling of sporadic task systems are summarized in Table I. In each entry, we give the best-known

[1] An FPTAS is a polynomial-time approximation scheme (PTAS) that has an approximation ratio of $(1+\epsilon)$ time complexity that is polynomial in both the number of tasks and $1/\epsilon$.

Multiprocessor Paradigm:	Sporadic Tasks	
	Constrained Deadlines	**Arbitrary Deadlines**
Global	DM: $\approx 3.73 \parallel ?$ (source: [6], [10]) EDF: $2 - \frac{1}{m} \parallel 2 - \frac{2}{m}$ (source: [8], [23])	
Partitioned	Both EDF and DM: $3 - \frac{1}{m} \parallel 2 - \frac{2}{m}$ (source: [3], [11], [15])	Both EDF and DM: $4 - \frac{2}{m} \parallel 2 - \frac{2}{m}$ (from [3], [11], [15])

TABLE I
SUMMARY OF RESOURCE-AUGMENTATION APPROXIMATION RATIOS FOR EACH MULTIPROCESSOR SCHEDULING ALGORITHM

approximation ratio for EDF and DM in the first entry and the best-known lower-bound on the approximation ration in the second entry. A question mark is used if no result is known. The lower-bound on the approximation ratio for partitioned scheduling follows from the theorem below (due to Andersson and Tovar [3]):

Theorem 1 (from [3]): Any partitioned scheduling algorithm (both EDF and DM based) for sporadic task systems cannot have an speed-approximation ratio less than $2 - \frac{2}{m}$.

Proof Sketch: Consider the implicit-deadline task system $\tau \stackrel{\text{def}}{=} \{\tau_1 = (m-1, m, m), \tau_2 = (m-1, m, m), \ldots, \tau_m = (m-1, m, m), \tau_{m+1} = (m, m, m)$. Note that $\sum_{\tau_i \in \tau} e_i/p_i$ equals m; thus, by [9], τ is schedulable on m unit-speed processors by an optimal global scheduling algorithm. However, notice that in partitioning scheduling (regardless of uniprocessor scheduling algorithm), tasks $\tau_1, \ldots, \tau_m$ must each be on a distinct processor. The task τ_{m+1} does not fit on any the processors (after assigning the first m tasks) until the processors have been speedup by a factor equal to $2 - \frac{2}{m}$. The lower bound on the approximation ratio follows. ■

IV. Open Questions

From the previous section, the fact that there are still several important open fundamental questions on partitioned-scheduling of sporadic tasks is evident. In this section, we list a few important questions:

1) **Do polynomial-time partitioning algorithms with tight speed-approximation ratios exist?** From Table I, notice the gap between the best-known speed-approximation ratio and the lower bound from Theorem 1. In other words, we are interested in whether current polynomial-time approaches can be improved.
2) **Is comparing partitioned scheduling with global feasibility approaches fair?** The lower bound result of Theorem 1 compares the speed required for the example task system with the optimal global approach. However, it is worth noting that no partitioning algorithm (even one that used exhaustive enumeration) could schedule the example task system on any processing platform slower than $2 - \frac{2}{m}$. Thus, we may consider whether a redefinition of the speed-approximation ratio to be with respect to the optimal partitioning approach would be beneficial. Given this redefinition, it may be possible that both the upper and lower approximation ratios may be further reduced.
3) **Does an asymptotic PTAS or FPTAS exist for EDF-based partitioned scheduling of sporadic tasks?** Under the redefinition of speed-approximation ratio suggested by the previous question, an interesting theoretic question is whether we also may develop approximation schemes with a user-specified amount of error.

V. Conclusion

In this paper, we have only very briefly summarized a subset of the results on partitioned scheduling for sporadic tasks. Given these prior results, we have identified some fundamental questions on partitioning that remain open. It is our hope that future research efforts will be focused on answering these fundamental questions. We believe that many more practical design issues for developing real-time systems on multicore platforms require a fundamental understanding of which partitioning and resource-allocation algorithms have acceptable (and provably) performance guarantees. Using such approaches may well minimize the amount of computational resources that need to be allocated to a given real-time application on a multicore platform.

References

[1] B. Andersson, S. Baruah, and J. Jansson. Static-priority scheduling on multiprocessors. In *Proceedings of the IEEE Real-Time Systems Symposium*, pages 193–202. IEEE Computer Society Press, December 2001.

[2] B. Andersson and J. Jonsson. The utilization bounds of partitioned and pfair static-priority scheduling on multiprocessors are 50%. In *Proceedings of the EuroMicro Conference on Real-Time Systems*, pages 33–40, Porto, Portugal, July 2003. IEEE Computer Society Press.

[3] B. Andersson and E. Tovar. Competitive analysis of partitioned scheduling on uniform multiprocessors. In *Proceedings of the Workshop on Parallel and Distributed Real-Time Systems*, Long Beach, CA, March 2007.

[4] T. P. Baker. An analysis of EDF schedulability on a multiprocessor. *IEEE Transactions on Parallel and Distributed Systems*, 16(8):760–768, 2005.

[5] T. P. Baker. An analysis of fixed-priority schedulability on a multiprocessor. *Real-Time Systems: The International Journal of Time-Critical Computing*, 32(1–2):49–71, 2006.

[6] S. Baruah. Schedulability analysis of global deadline-monotonic scheduling. In submission, 2007.

[7] S. Baruah and T. Baker. Global EDF schedulability analysis of arbitrary sporadic task systems. In *Proceedings of the EuroMicro Conference on Real-Time Systems*, Prague, Czech Republic, July 2008. IEEE Computer Society Press.

[8] S. Baruah, V. Bonifaci, A. Marchetti-Spaccamela, and S. Stiller. Implementation of a speedup-optimal global EDF schedulability test. In *Proceedings of the EuroMicro Conference on Real-Time Systems*, Dublin, July. IEEE Computer Society Press.

[9] S. Baruah, N. Cohen, G. Plaxton, and D. Varvel. Proportionate progress: A notion of fairness in resource allocation. *Algorithmica*, 15(6):600–625, June 1996.

[10] S. Baruah and N. Fisher. Global deadline-monotonic scheduling of arbitrary-deadline sporadic task systems. In *Proceedings of the 11th International Conference on Principles of Distributed Systems*, Guadeloupe, French West Indies, December 2007. Springer-Verlag.

[11] S. Baruah and N. Fisher. The partitioned dynamic-priority scheduling of sporadic task systems. *Real-Time Systems: The International Journal of Time-Critical Computing*, 36(3):199–226, 2007.

[12] M. Bertogna, M. Cirinei, and G. Lipari. Improved schedulability analysis of EDF on multiprocessor platforms. In *Proceedings of the EuroMicro Conference on Real-Time Systems*, pages 209–218, Palma de Mallorca, Balearic Islands, Spain, July 2005. IEEE Computer Society Press.

[13] M. Bertogna, M. Cirinei, and G. Lipari. New schedulability tests for real-time tasks sets scheduled by deadline monotonic on multiprocessors. In *Proceedings of the 9th International Conference on Principles of Distributed Systems*, Pisa, Italy, December 2005. IEEE Computer Society Press.

[14] F. Eisenbrand and T. Rothvoß. A PTAS for static priority real-time scheduling with resource augmentation. Technical report, Institute of Mathematics, EPFL, 2008.

[15] N. Fisher, S. Baruah, and T. Baker. The partitioned scheduling of sporadic tasks according to static priorities. In *Proceedings of the EuroMicro Conference on Real-Time Systems*, Dresden, Germany, July 2006. IEEE Computer Society Press.

[16] D. S. Johnson. *Near-optimal Bin Packing Algorithms*. PhD thesis, Department of Mathematics, Massachusetts Institute of Technology, 1973.

[17] J. Leung and J. Whitehead. On the complexity of fixed-priority scheduling of periodic, real-time tasks. *Performance Evaluation*, 2:237–250, 1982.

[18] C. Liu and J. Layland. Scheduling algorithms for multiprogramming in a hard real-time environment. *Journal of the ACM*, 20(1):46–61, 1973.

[19] J. M. Lopez, J. L. Diaz, and D. F. Garcia. Utilization bounds for EDF scheduling on real-time multiprocessor systems. *Real-Time Systems: The International Journal of Time-Critical Computing*, 28(1):39–68, 2004.

[20] A. K. Mok. *Fundamental Design Problems of Distributed Systems for The Hard-Real-Time Environment*. PhD thesis, Laboratory for Computer Science, Massachusetts Institute of Technology, 1983. Available as Technical Report No. MIT/LCS/TR-297.

[21] D.-I. Oh and T. P. Baker. Utilization bounds for N-processor rate monotone scheduling with static processor assignment. *Real-Time Systems: The International Journal of Time-Critical Computing*, 15:183–192, 1998.

[22] F. Phelan. Rinding the next wave of embedded multicore processors. *Ecnmag.com*, August 2007. Available at URL http://www.ecnmag.com/Riding-the-Next-Wave-of-Embedded-Multicore-Processors.aspx .

[23] C. A. Phillips, C. Stein, E. Torng, and J. Wein. Optimal time-critical scheduling via resource augmentation. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 140–149, El Paso, Texas, 4–6 May 1997.

[24] A. Srinivasan and S. Baruah. Deadline-based scheduling of periodic task systems on multiprocessors. *Information Processing Letters*, 84(2):93–98, 2002.