A Comparison of Global and Partitioned EDF Schedulability Tests for Multiprocessors
TR-051101

Theodore P. Baker
Florida State University
Dept. of Computer Science
Tallahassee, FL 32306 USA
baker@cs.fsu.edu

Abstract

This paper compares the performance of several variations on EDF-based global and partitioned multiprocessor scheduling algorithms, together with their associated feasibility tests, on a variety of pseudo-randomly chosen sets of sporadic tasks. A new hybrid EDF-based scheme is shown to perform better than previously studied priority-based global scheduling schemes, though not as well as EDF-based first-fit partitioned scheduling.

1 Introduction

Recent trends in microprocessor design have drawn interest to multi-core and multiprocessor designs for high performance embedded real-time systems. The predominant approach to scheduling multiprocessor hard-real-time systems has been partitioned, in which each task is assigned statically (more or less) to one processor. Partitioned scheduling has the virtue of permitting schedulability to be verified using well-understood single-processor analysis techniques.

The alternative to partitioned scheduling is global scheduling, in which there is a single job queue, from which jobs are dispatched to any available processor according to a global priority scheme. Until recently, it was believed that global scheduling policies with fixed job priorities, such as earliest-deadline-first (EDF), could not even guarantee schedulability for systems of hard-deadline tasks whose total processor demand exceeded the capacity of a single processor. However, there have been several recent improvements in the worst-case analysis of global hard-deadline multiprocessor scheduling[1, 6, 13, 14, 20, 2, 10, 11, 12, 5]. Among other developments, the EDF-US[1/2] scheduling policy, in which a few high-utilization tasks are scheduled at top priority and other tasks are scheduled according to deadlines, has been shown to guarantee worst-case schedulability up to the same processor utilization level as partitioned EDF scheduling.

Global scheduling remains controversial. There are individuals who believe strongly that the overhead of synchronizing schedulers between processors and lost performance due to translation look-aside buffer and memory cache misses following the migration of a task between processors will inevitably negate any possible improvement in scheduling efficiency. On the other hand, the concept of global scheduling is appealing, especially in systems where average as well as worst-case response time is important. It is a well-known result of queueing theory that single-queue scheduling produces better average response times than queue-per-processor scheduling[16].

This paper attempts to compare the present state of the art for global EDF scheduling against the state of the art for partitioned EDF scheduling. Because the worst-case performance of both approaches has been shown to be the same, at least for the case where deadline equals period, the comparison is of empirical performance. That is, what are the odds that a randomly chosen set of periodic or sporadic tasks can be guaranteed schedulable by a given combination of scheduling policy and and feasibility test? As a further contribution, the paper introduces a previously unpublished improved global EDF feasibility test, and a previously unpublished hybrid global scheduling algorithm.

2 Prior Work

As mentioned above, the theoretical worst-case achievable processor utilizations of the global and partitioned scheduling approaches have been shown to be very similar, for sporadic or aperiodic task sets with deadline equal to period.

Andersson, Baruah, and Jonsson[1] showed that the utilization guarantee for EDF or any other static-priority multiprocessor scheduling algorithm – partitioned or global – cannot be higher than $(m+1)/2$ for an m-processor platform. Doing such partitioning optimally is reducible to the bin packing and integer partition problems, which are known to be NP complete. Therefore, research has focused on the analysis of heuristic algorithms for the assignment of tasks to processors, and on bounding how badly they can do compared to an optimal algorithm. Some of this research has looked at average-case performance. For example, in [17], Lee *et al.* studied a general multi-resource allocation problem, and empirically compared the average solution quality and execution time of a search-bounding heuristic algorithm against those of dynamic programming and integer linear programming algorithms. Other research has attempted to find tight bounds on the worst-case performance of heuristic partitioning algorithms.

2.1 Partitioned Scheduling

Lopez, Diaz, Garcia and Garcia[18] showed that it is possible to schedule on m processors any system of n independent periodic tasks with maximum individual utilization $u_{\max}$ and total utilization $U < \frac{m\beta_{EDF}+1}{\beta_{EDF}+1}$ where $\beta_{EDF} = \lfloor 1/u_{\max} \rfloor$. For the unrestricted case, where $u_{\max} = 1$ and $\beta_{EDF} = 1$, this says the guaranteed utilization bound is $(m+1)/2$. It follows from Andersson, Baruah, and Jonsson[1] that this result is tight.

Baruah and Fisher[5] recently studied a partitioning algorithm that assigns tasks to processors in deadline-monotonic first-fit order. The single-processor test for fit is based analysis of a demand-bound function, as follows:

Theorem 1 (GF) *A set of sporadic tasks $\tau_1,\ldots,\tau_N$ is EDF schedulable on one processor if both of the following hold for each task τ_i:*

$$d_i - \sum_{j=1}^{N} DBF^*(j, d_i) \geq c_i \tag{1}$$

$$1 - \sum_{j=1}^{N} u_i \geq u_i \tag{2}$$

where $u_i = c_i/T_i$ and

$$DBF^*(i,t) = \begin{cases} 0, & \text{if } t < d_i \\ c_i + (t - d_i)u_i, & \text{otherwise} \end{cases}$$

2.2 Global Scheduling

Goossens, Funk, and Baruah [14] showed that a system of independent periodic tasks can be scheduled successfully on m processors by EDF scheduling if the total utilization is at most $m(1-u_{\max})+u_{\max}$, where $u_{\max}$ is the maximum utilization of any individual task. They also showed that this utilization bound is tight, in the sense that there is no utilization

bound $\hat{U} > m(1 - u_{\max}) + u_{\max} + \epsilon$, where $\epsilon > 0$, for which $U \leq \hat{U}$ guarantees EDF schedulability. Srinivasan and Baruah[20] also examined the global EDF scheduling of periodic tasks on multiprocessors, and showed that any system of independent periodic tasks for which the utilization of every individual task is at most $m/(2m-1)$ can be scheduled successfully on m processors if the total utilization is at most $m^2/(2m-1)$.

In 2002, Srinivasan and Baruah[20] proposed a method for dealing with a few heavy tasks, using a *hybrid* scheduling policy. Their idea is to give highest (fixed) priority to to tasks of utilization greater than some constant ζ, and schedule the other tasks according to the basic EDF algorithm. This algorithm is called EDF-US[ζ]. Algorithm EDF-US[$m/(2m-2)$] was shown to correctly schedule on m processors any periodic task system with total utilization $U \leq m^2/(2m-2)$, and RM-US[$m/(2m-2)$] was shown to correctly schedule on m processors any periodic task system with total utilization $U \leq m^2/(3m-2)$.

Baker[2, 3] derived several sufficient feasibility tests for m-processor preemptive EDF scheduling of sets of periodic and sporadic tasks with arbitrary deadlines, and showed that the optimal value of ζ in EDF-US[ζ] with respect to maximizing the worst-case guaranteed schedulable utilization is $\zeta = 1/2$, for which the utilization bound is $(m+1)/2$. It follows from the argument in [1] that this bound is tight, and it is identical to the worst-case utilization bound for EDF-based first-fit-decreasing (FFD) partitioned scheduling.

Bertogna, Cirinei and Lipari[10] made further improvements in global EDF schedulability tests. First, they observed that the proof of the utilization bound test of [14] extends naturally to cover pre-period deadlines if the utilization u_i is replaced by c_i/D_i. As observed by Sanjoy Baruah[1], the same proof extends to the case of post-period deadlines if c_i/D_i is replaced by $\lambda_i = c_i/\min\{D_i, T_i\}$.

Theorem 2 (GFB) *A set of sporadic tasks τ_1,...,τ_N is EDF schedulable on m identical processors if*

$$\sum_{i=1}^{N} \lambda_i \leq m - \lambda_{\max}(m-1)$$

where $\lambda_{\max} = \max\{\lambda_i | i = 1, \ldots, N\}$.

Bertogna *et al.* also developed the following new schedulability test.

Theorem 3 (BCL) *A set of sporadic tasks τ_1,...,τ_N (with constraint $d_i \leq T_i$) is EDF schedulable on m identical processors if for each task τ_k one of the following is true:*

$$\sum_{i \neq k} \min\{\beta_i, 1 - \lambda_k\} < m(1 - \lambda_k) \tag{3}$$

$$\begin{array}{l} \sum_{i \neq k} \min\{\beta_i, 1 - \lambda_k\} = m(1 - \lambda_k) \quad \textit{and} \\ \exists i \neq k : 0 < \beta_i \leq 1 - \lambda_k \end{array} \tag{4}$$

where

$$\beta_i = \frac{N_i c_i + \min\{c_i, \max\{0, d_k - N_i T_i\}\}}{d_k}$$

and

$$N_i = \left\lfloor \frac{d_k - d_i}{T_i} \right\rfloor + 1$$

Bertogna *et al.* demonstrated that the BCL, GFB, and Baker[2, 3] tests are generally incomparable, but observed that the BCL test seemed to do better than the rest on task sets with a few "heavy" (high utilization) tasks. They reported simulations

[1] personal communication

on collections of pseudo-randomly generated tasks sets with a few heavy tasks, for which the BCL was able to discover significantly more schedulable task sets than either of the other two tests. However, they did not compare these results against the EDF-US[ζ] hybrid method of handling heavy tasks proposed in 2002 by Srinivasan and Baruah[20].

A key observation of BCL is that if a task τ_k misses a deadline that is due to *interference* from other tasks, and the maximum fraction of the workload of any task that can contribute to the interference is $1 - \lambda_k$. Baker[4] has incorporated this observation into the analysis of [3], to obtain the following previously unpublished schedulability test:

Theorem 4 (BAK2) *A set of sporadic tasks τ_1,...,τ_N is EDF schedulable on m identical processors if for each task τ_k there exists $\lambda > \frac{c_k}{T_k}$ such that one or more of the the following is true:*

$$\sum_{i=1}^{N} \min\{\beta_k^\lambda(i), 1-\lambda_k\} < m(1-\lambda_k) \tag{5}$$

$$\begin{array}{l} \sum_{i=1}^{N} \min\{1-\lambda_k, \beta_k^\lambda(i)\} = m(1-\lambda_k) \text{ and} \\ \exists i \;\; 0 < \beta_k^\lambda(i) < 1-\lambda_k \end{array} \tag{6}$$

$$\sum_{i=1}^{N} \min\{1, \beta_k^\lambda(i)\} \leq m(1-\lambda_k) + \lambda_k \tag{7}$$

where $\lambda_k = \lambda \max\{1, \frac{T_k}{d_k}\}$ *and*

$$\beta_k^\lambda(i) = \begin{cases} \max\{u_i, u_i(1-\frac{d_i}{d_k}) + \frac{c_i}{d_k}\} & \text{if } \frac{c_i}{T_i} \leq \lambda \\ \frac{c_k}{T_k} & \text{if } \frac{c_i}{T_i} > \lambda \text{ and } \lambda \geq \frac{c_i}{d_i} \\ \frac{c_i}{T_i} + \frac{c_i - \lambda d_i}{d_k} & \text{if } \frac{c_i}{T_i} > \lambda \text{ and } \frac{c_i}{d_i} > \lambda \end{cases}$$

The proof is similar the analysis in [3], except that the notion of busy interval is refined. The definition of busy interval in [3] is based on the a total processor demand over a given time interval that is necessary to permit a missed deadline. Bertogna *et al.*[10] pointed out that it is not processor demand but *interference* from other tasks that causes missed deadlines, where the interference $I_k(t-\Delta, t)$ of a task τ_k over a time interval $[t-\Delta, t)$ is defined to be the sum of the lengths of all the subintervals during which τ_k is backlogged but unable to execute due to preemption. It is tempting to believe one can redo the analysis of [3] with processor demand replaced by the ratio of interference to interval length, but the proof does not appear to go through in that form. As a compromise, one can look at the ratio $B(t-\Delta, t)/\Delta$, where $B(t-\Delta, t)$ is the *block busy time* of the interval $[t-\Delta, t)$, defined as the sum of the lengths of all the subintervals during which all m processors are executing. If one then uses the following definition of τ_k^λ-busyto define the busy interval, the above theorem can be proven.

Definition 1 *An interval $[t-\Delta, t)$ is τ_k^λ-*busy *for a given constant $\lambda \geq c_k/T_k$ if*

$$B(t-\Delta, t) > \Delta - \lambda(\Delta + T_k - d_k) \tag{8}$$

where $B(t-\Delta, t)$ is the block busy time *of the interval.*

The above theorem can be used as a schedulability test by attempting tp verify the three conditions for each value of k. The test is of complexity $O(N^3)$ since the only values of λ that need be considered are the minimum and the points where $\beta_k^\lambda(i)$ is discontinuous, i.e.,

- $\lambda = u_i$, $i = 1, \ldots, N$
- $\lambda = c_i/d_i$, if $d_i > T_i$

3 Empirical Comparisons

To evaluate the efficacy of the new BAK2 test in comparison to previous global EDF feasibility tests, and to compare the efficacy of global *versus* partitioned scheduling, a series of experiments were conducted on pseudo-randomly generated sets of periodic tasks.

3.1 Methodology

A variety of EDF-based scheduling policies and schedulability tests were tested on several datasets. Each dataset contained 1,000,000 sets of tasks.

The task periods were generated pseudo-randomly with a uniform distribution between 1 and 1000.

The processor utilizations (and, implicitly, the compute times) were chosen according to the following distributions, truncated to bound the utilization between 0.001 and 0.999, including:

1. uniform distribution, between 1/period and 1
2. bimodal distribution: heavy tasks uniform between 0.5 and 1; light tasks uniform between 1/period and 0.5; probability of being heavy = 1/3[2]
3. exponential distribution with mean 0.25
4. exponential distribution with mean 0.50

The deadlines were chosen in two different ways:

A constrained: deadlines uniformly distributed between the execution time and the period

B unconstrained: deadlines uniformly distributed between the execution time and 4 times the period

Datasets were generated for 2, 4, and 8 processor systems, as follows. An initial set of $m + 1$ tasks was generated, and tested. Then another task was generated and added to the previous set, and all the schedulability tests were run on the new set. This process of adding tasks was repeated until the total processor utilization exceeded m. The whole procedure was then repeated, starting with a new initial set of $m + 1$ tasks.

All experiments were run on these 24 datasets. Only the results of a few of the experiments are reported here, due to space limitations and because the trends across the experiments were quite similar.

3.2 Representative of Global EDF

To choose a representative for global EDF scheduling, simulations were run to compare the performance of the various schedulability tests, for both pure EDF scheduling and several hybrids.

Experiments were performed to compared the following sufficient tests for feasibility under global EDF scheduling:

BAK Baker's test from [2, 3]
GFB Goosens, Funk and Baruah's test, extended to arbitrary deadlines by Bertogna, Cirinei and Lipari, as stated Theorem 2 above.
BCL Bertogna, Cirinei, and Lipari's test, as stated in Theorem 3 above.

[2]Intended to bias toward cases with a few heavy tasks, similar to the experiments of Bertogna, Cirinei, and Lipari[10].

BAK2 Baker's revised test, as stated in Theorem 4 above.

Figures 1-3 show the performance of these tests on three of the datasets. Like the other graphs in this paper, each is a histogram of 100 buckets, each bucket corresponding to a range of 1 percent of the full range of total utilizations possible for a given number of processors. So, with four processors the buckets each represent a total processor utilization range of four percent. The three lower lines in each graph show how many task sets were verifiably feasible according to each of the three tests. The upper line shows the total number of task sets in each bucket, including both feasible and infeasible task sets.

The infeasible task sets are included in the count because the only necessary and sufficient test for feasibility of global EDF scheduling of N tasks on m processors known to this author has worst-case execution time of the order $O(mN \cdot \Pi_{i=1}^{N} T_i c_i)$. The author implemented and tested that algorithm, but running it on datasets of the size considered here was not practical. Reporting the relative performance of the efficient sufficient tests of feasibility on large numbers of tasks sets seemed more important than comparing them against perfection on a much smaller number of task sets, with smaller periods.

Note that in [10] it is claimed that "simulation of the schedule up to the hyper-period checking for missed deadlines" is a necessary and sufficient test for schedulability. However, this author is not aware of any proof that such a simulation is a sufficient test for feasibility of sporadic task sets, or even of periodic tasks sets with arbitrary initial release time offsets. Even under the assumption of strictly periodic tasks and simultaneous start times, if periods can exceed deadlines simulation to the hyperperiod is not sufficient.

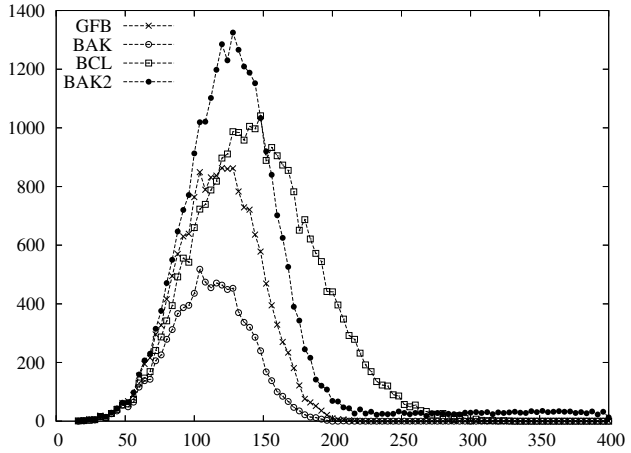

Figure 1. Constrained deadline, bimodal utilization distribution, 4 CPUs

Figure 1 is for one of the datasets were the BCL test was more effective than the other tests over some range of the utilization distribution. The dataset shown is for constrained deadlines and bimodal task utilizations on four processors.

Figure 2 is for one of the datasets where the GFB test was most accurate over some range of the utilization distribution. The dataset shown is for unconstrained deadlines and exponentially distributed task utilizations with mean 0.25 on two processors.

Figure 3 is for a 4-cpu dataset with unconstrained deadlines and exponentially distributed task utilizations with mean 0.25.

Because the three tests each have some cases where they are more accurate in determining feasibility, it makes sense to apply them in combination, starting with the computationally least expensive, that is:

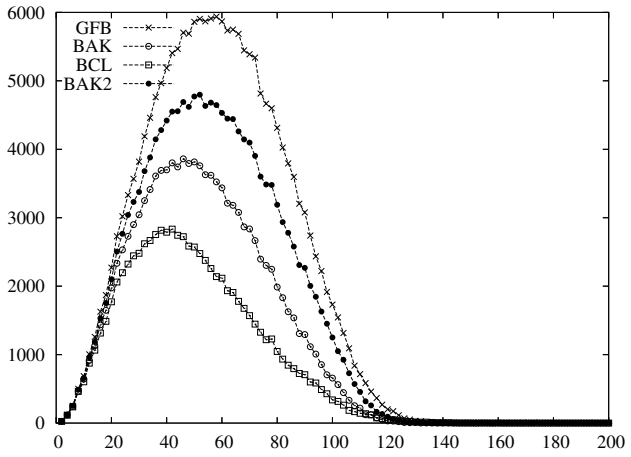

Figure 2. Constrained deadline, exponential utilization w/mean 0.25, 2 CPUs

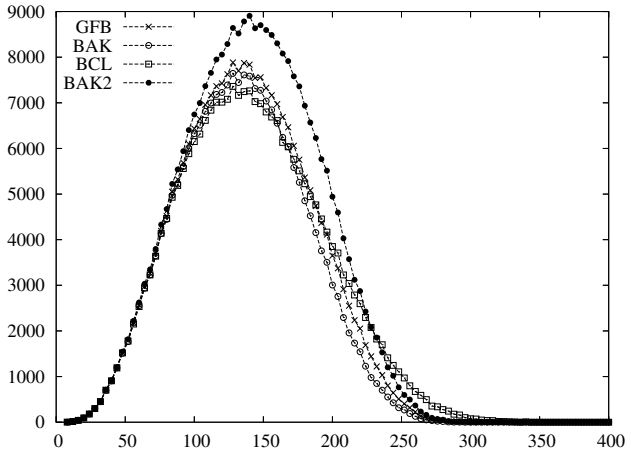

Figure 3. Unconstrained deadline, exponential utilization w/mean 0.25, 4 CPUs

1. apply the GFB test
2. if the GFB test fails and the deadlines do not exceed the periods, apply the BCL test
3. if the other tests fail, apply the BAK2 test

Figures 4-5 show the results of applying such a three-stage test on the same datasets reported in Figures 1 and 3. These are typical of the results on all of the 24 datasets.

It is clear that the combination of all the applicable tests is the winner among the feasibility tests for pure EDF scheduling. This test is called “GBB” for short, from this point on.

3.3 Hybrid Global Schemes

In addition to basic EDF scheduling, the performance of the following hybrids of EDF and highest-utilization-first scheduling was tested:

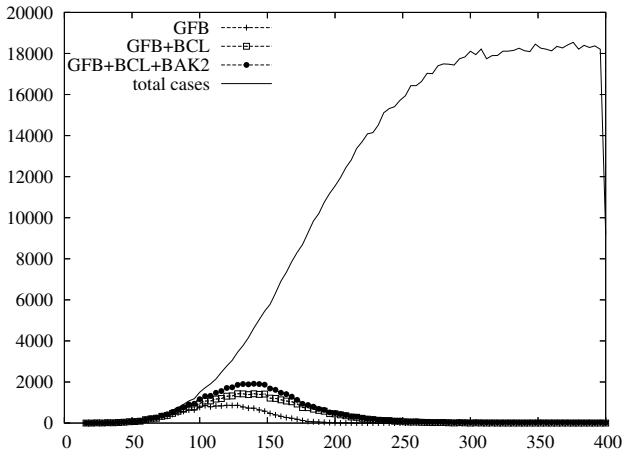

Figure 4. Constrained deadline, bimodal utilizations, 4 CPUs

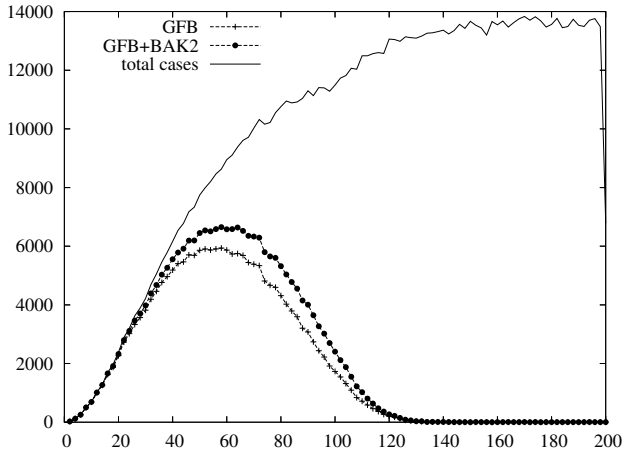

Figure 5. Constrained deadline, exponential utilization w/mean 0.25, 2 CPUs

1. EDF-US[1/2]: give special priority to the tasks of utilization greater than 1/2, which is the cut-off value that guarantees the highest worst-case utilization when deadline=period;
2. EDF-UM: give special priority to the k tasks with highest utilization, where k is the smallest value between 0 and m for which the system can be verified as schedulable according to the GBB Test.
3. EDF-LM: give special priority to the k tasks with highest value of $c_i / \max\{T_i, d_i\}$, where k is the smallest value between 0 and m for which the system can be verified as schedulable according to the GBB Test.

Figures 6-7 show the results of applying these three hybrid EDF scheduling policies with the GBB test on the same datasets reported in Figures 1 and 3. The GBB test was applied to the remaining $N - k$ tasks on $m - k$ processors, after choosing k special tasks to receive top priority. These result are typical of what was observed on all of the 24 datasets, *i.e.*, tThe EDF-LM hybrid scheme clearly finds the highest number of verifiably schedulable task sets at every total utilization level.

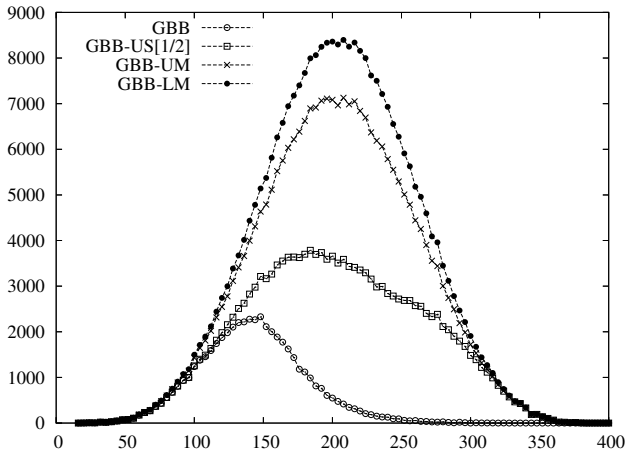

Figure 6. Constrained deadline, bimodal utilization, 4 CPUs

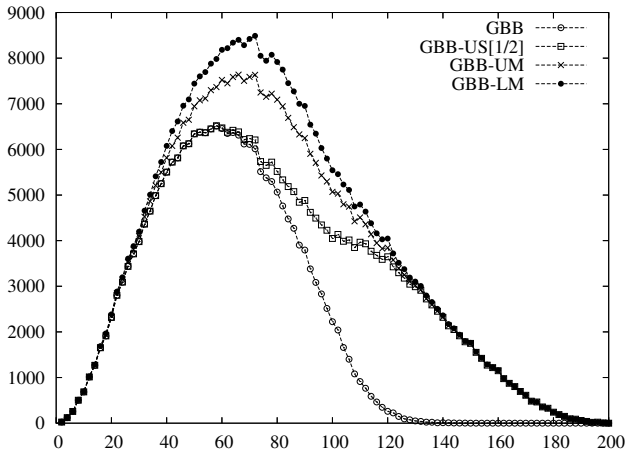

Figure 7. Constrained deadline, exponential utilization w/mean 0.25, 2 CPUs

3.4 Representative of Partitioned EDF

To select a representative for partitioned scheduling, several EDF-based partitioning schemes were evaluated. In each case the tasks were assigned to processors according to a first-fit heuristic, in order of some metric, such as relative deadline (d_i). Two tests for fit were evaluated: (GF) the sufficient test of [5]; (BHF) the necessary and sufficient test of Baruah *et al.*[9]. The strength of the GF test is its simplicity. In contrast, the worst-case upper bound on the complexity of the BHR test is the LCM of the task periods.

To avoid long running times when possible, the BHR test was only applied to task systems that could not be resolved using the following two well-known simpler tests:

1. A task set is feasible if $\sum_{i=1}^{N} \frac{c_i}{\min\{d_i,T_i\}} < 1$
2. A task set is not feasible if $\sum_{i=1}^{N} u_i > 1$

With the above two screening tests the BHR test converged very quickly for all of the task sets tested. However, in

principle, it appears that the BHR test may not always be practicable, either because of integer overflow or long compute times. The GF test has much lower worst-case computational complexity. Surprisingly, even though the BHR test is more exact it does not always produce the best FFD partition (FFD is only a heuristic).

Figures 8-9 show the success rates of the two methods, using three different order heuristics: (FFD-U) decreasing u_i; (FFD-L) decreasing $c_i / \min\{d_i, T_i\}$; (FFD-D) increasing d_i. For these and all other datasets BF-FFD-U and BF-FFD-L gave the best results, with each having a slight advantage on different datasets. The GF FFD-L scheme was chosen for comparison against global scheduling.

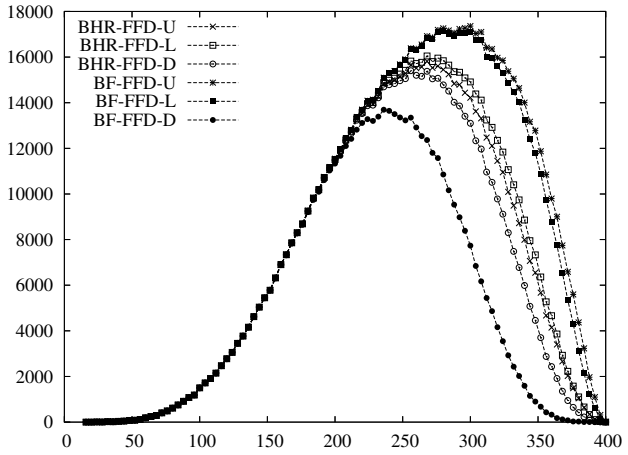

Figure 8. Constrained deadline, bimodal utilization, 4 CPUs

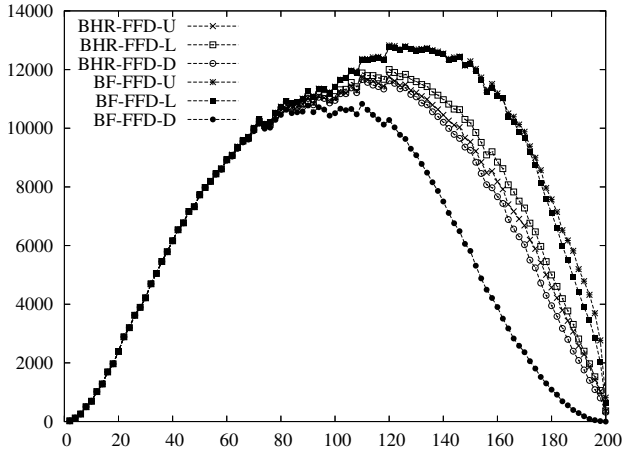

Figure 9. Constrained deadline, exponential utilization w/mean 0.25, 2 CPUs

3.5 Partitioned *versus* Global

Figures 11-16 show the performances of the GBB and GBB-LM hybrid schemes against the GF partitioning scheme on four datasets. The pattern exhibited in these examples persisted over all 24 of the datasets tested. In all cases the hybrid global scheduling scheme improved the success rate significantly over pure EDF, but it still fell short of the success rate with partitioned scheduling.

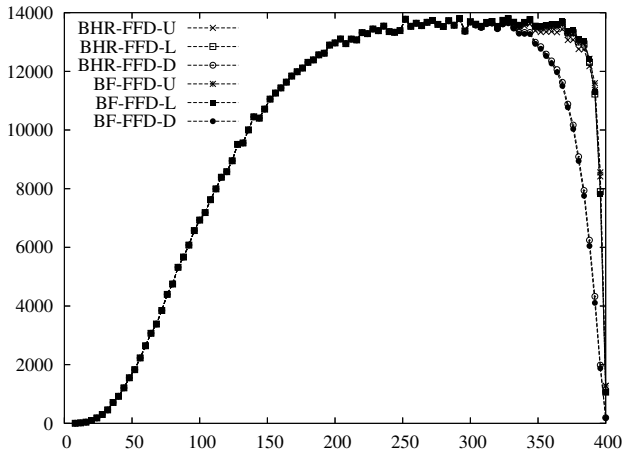

Figure 10. Unconstrained deadline, exponential utilization w/mean 0.25, 4 CPUs

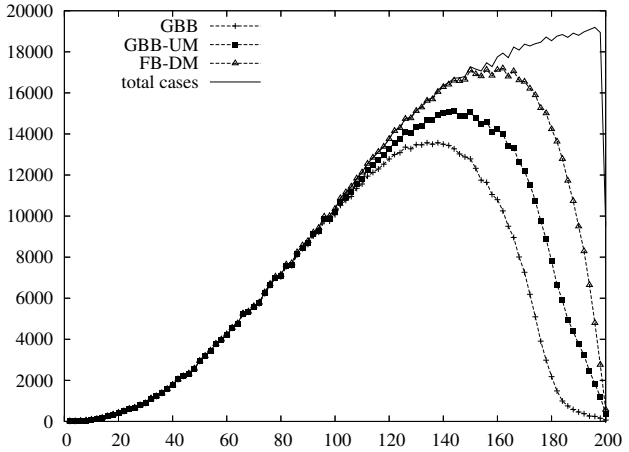

Figure 11. Unconstrained deadline, bimodal utilization, 2 CPUs

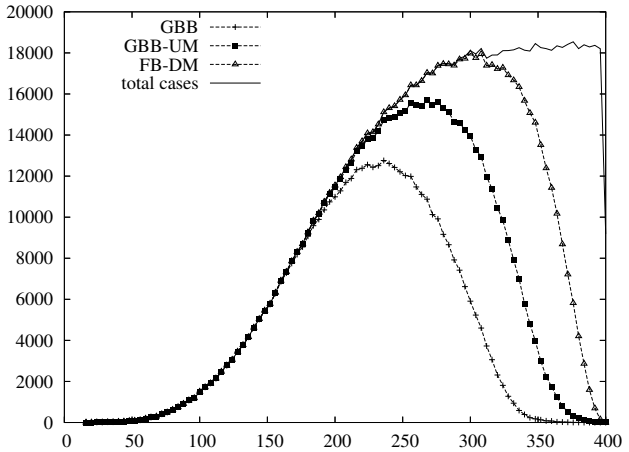

Figure 12. Unconstrained deadline, bimodal utilization, 4 CPUs

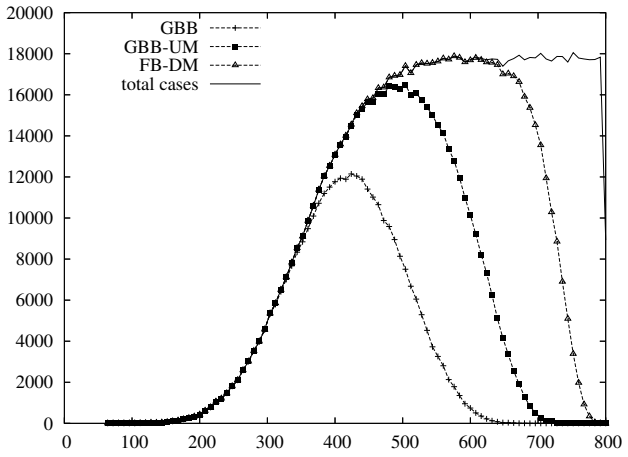

Figure 13. Unconstrained deadline, bimodal utilization, 8 CPUs

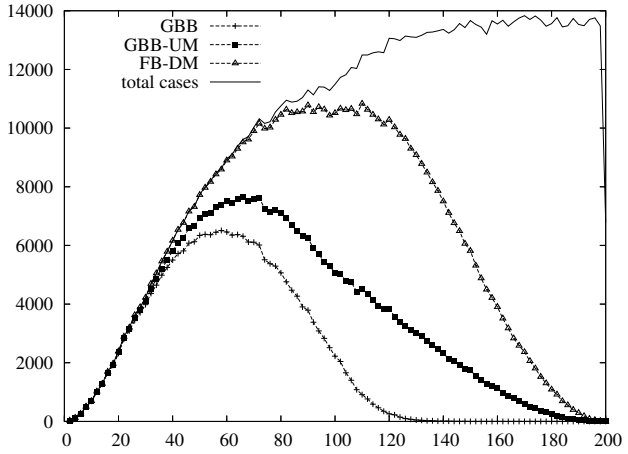

Figure 14. Constrained deadline, exponential utilization w/mean 0.25, 2 CPUs

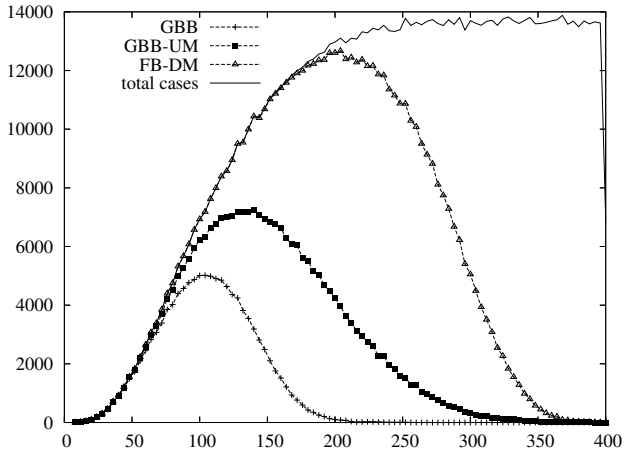

Figure 15. Unconstrained deadline, exponential utilization w/mean 0.25, 4 CPUs

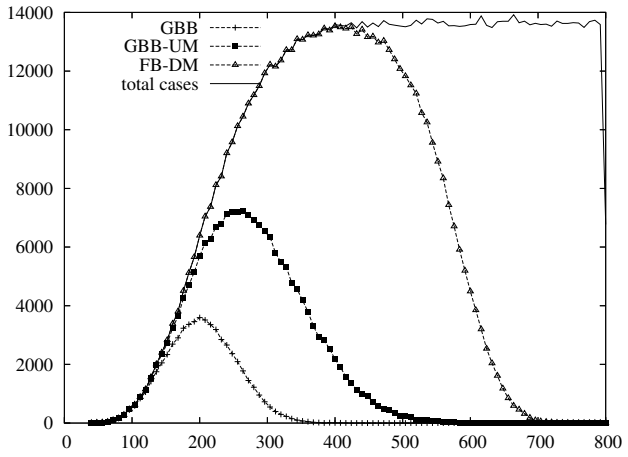

Figure 16. Unconstrained deadline, exponential utilization w/mean 0.25, 8 CPUs

4 Conclusions and Future Work

The experiments reported here indicate that the available schedulability tests for global EDF scheduling have improved significantly. However, the global approach has not yet pulled ahead. Partitioned scheduling still appears to have an advantage over the best feasibility tests for global scheduling, with respect to the statistical chance of being able to schedule an arbitrary hard-deadline task set. Add to this the fact that static task assignment has lower runtime overhead, and partitioned scheduling looks even stronger.

This is not be the end of global vs. partitioned scheduling question. Further progress in the analysis of global EDF scheduling appear possible. Even if global EDF does not ultimately prove to be competitive with partitioned EDF scheduling, there are other global scheduling schemes to be considered. Some of these can gurantee even worst-case schedulability at higher processor utilization levels than the $(m + 1)/2$ worst-case bound for job-static priority scheduling.

There are several variants of the PFAIR concept. Baruah, Cohen, Plaxton and Varvel[7, 8] showed that PFAIR scheduling is optimal for scheduling periodic tasks on a multiprocessor, has a linear-time necessary and sufficient schedulability test, and for sufficiently small quantum size can guarantee schedulability at processor utilization levels arbitrarily close to m. Srinivasan and Anderson showed that the PFAIR approach is also optimal for multiprocessor scheduling of sporadic and rate-based tasks[19], and there have been many more variations and extensions to the PFAIR theory made since that. The main problem with PFAIR scheduling is the need to slice time into small quanta, and the consequently high implementation overhead. In this regard, the fixed-job-priority algorithms, like those considered in this paper have an advantage, whether applied globally or partitioned.

Is there another algorithm that can break the $(m + 1)/2$ bound but does not require such frequent time slicing as the PFAIR approach? One possibility is throw-forward, shown by Johnson and Maddison[15] to be optimal for scheduling independent jobs on a multiprocessor system. It will be interesting to see whether their analysis of throw-forward scheduling can be extended to provide a sufficient test for schedulability of periodic and sporadic task systems.

Of course there are also some remaining questions about the comparative implementation overhead of the global vs. partitioned approaches. Global scheduling can have higher overhead in at least two respects: the contention delay and the synchronization overhead for a single dispatching queue is higher than for per-processor queues; the cost of resuming a task may be higher if it is on a different processor (due to interprocessor interrupt handling and cache reloading) than on the processor where it last executed. The latter cost can be quite variable, since it depends on the actual portion of a task's memory that remains in cache when the task resumes execution, and how much of that remnant will be referenced again before it is overwritten. These issues are discussed at some length by Srinivasan *et al.* in [21], which includes

some simulation results comparing the overhead of global EDF and PD^2 scheduling, a PFAIR variant. It seems that only experimentation with actual implementations can make a conclusive case as to how serious are these overheads, and how they balance against any advantages global scheduling may have for on-time completion of tasks in real applications.

References

[1] B. Andersson, S. Baruah, and J. Jonsson. Static-priority scheduling on multiprocessors. In *Proc. 22nd IEEE Real-Time Systems Symposium*, pages 193–202, London, UK, Dec. 2001.

[2] T. P. Baker. Multiprocessor EDF and deadline monotonic schedulability analysis. In *Proc. 24th IEEE Real-Time Systems Symposium*, pages 120–129, 2003.

[3] T. P. Baker. An analysis of EDF scheduling on a multiprocessor. *IEEE Trans. on Parallel and Distributed Systems*, 15(8):760–768, Aug. 2005.

[4] T. P. Baker. Further improved schedulability analysis of EDF on multiprocessor platforms. Technical Report TR-051001, Florida State University Department of Computer Science, Tallahassee, FL, Nov. 2005.

[5] S. Baruah and N. Fisher. Partitioned multiprocessor scheduling of sporadic task systems. In *Proc. of the 26th IEEE Real-Time Systems Symposium*, Miami, Florida, Dec. 2005.

[6] S. Baruah and J. Goossens. Rate-monotonic scheduling on uniform multiprocessors. *IEEE Trans. Computers*, 52(7):966–970, July 2003.

[7] S. K. Baruah, N. Cohen, C. G. Plaxton, and D. Varvel. Proportionate progress: a notion of fairness in resource allocation. In *Proc. ACM Symposium on the Theory of Computing*, pages 345–354, May 1993.

[8] S. K. Baruah, N. Cohen, C. G. Plaxton, and D. Varvel. Proportionate progress: a notion of fairness in resource allocation. *Algorithmica*, 15:600–625, 1996.

[9] S. K. Baruah, R. R. Howell, and L. E. Rosier. Algorithms and complexity concerning the preemptive scheduling of periodic real-time tasks on one processor. *Real-Time Systems*, 2, 1990.

[10] M. Bertogna, M. Cirinei, and G. Lipari. Improved schedulability analysis of EDF on multiprocessor platforms. In *Proc. of the 17th Euromicro Conference on Real-Time Systems*, Palma de Mallorca, Spain, July 2005.

[11] M. Bertogna, M. Cirinei, and G. Lipari. New schedulability tests for real-time tasks sets scheduled by deadline monotonic on multiprocessors. In *Proc. of the 9th International Conf. on Principles of Distributed Systems*, Pisa, Italy, Dec. 2005.

[12] N. Fisher and S. Baruah. The partitioned, static-priority scheduling of sporadic real-time tasks with constrained deadlines on multiprocessor platforms. In *Proc. of the 9th International Conf. on Principles of Distributed Systems*, Pisa, Italy, Dec. 2005.

[13] S. Funk, J. Goossens, and S. Baruah. On-line scheduling on uniform multiprocessors. In *Proc. 22nd IEEE Real-Time Systems Symposium*, pages 183–192, London, UK, Dec. 2001. IEEE Computer Society.

[14] J. Goossens and R. Devillers. Feasibility intervals for the deadline driven scheduler with arbitrary deadlines. In *Proc. 6th International Conf. Real-Time Computing Systems and Applications (RTCSA'99)*, 1999.

[15] H. H. Johnson and M. S. Maddison. Deadline scheduling for a real-time multiprocessor. In *Proc. Eurocomp Conference*, pages 139–153, 1974.

[16] L. Kleinrock. *Queueing Systems - Volume 2: Computer Applications*. Wiley Interscience, 1976.

[17] C. Lee, J. Lehoczky, D. Siewiorek, R. Rajkumar, and J. Hansen. A scalable solution to the multi-resource QoS problem. In *Proc. of IEEE Real-Time Systems Symposium*, Phoenix, AZ, USA, Dec. 1999.

[18] J. M. Lopez, J. L. Diaz, M. Garcia, and D. F. Garcia. Worst-case utilization bound for EDF scheduling on real-time multiprocessor systems. In *Proc. 12th Euromicro Conf. Real-Time Systems*, pages 25–33, 2000.

[19] A. Srinivasan and J. Anderson. Optimal rate-based scheduling on multiprocessors. In *Proc. 34th ACM Symposium on Theory of Computing*, pages 189–198. ACM, May 2002.

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

[21] A. Srinivasan, P. Holman, J. H. Anderson, and S. Baruah. The case for fair multiprocessor scheduling. In *Proc. 11th International Workshop on Parallel and Distributed Real-time Systems*, Apr. 2003.