Noname manuscript No.
(will be inserted by the editor)

On the Ineffectiveness of 1/m-based Interference Bounds in the Analysis of Global EDF and FIFO Scheduling

Alessandro Biondi · Youcheng Sun

Received: date / Accepted: date

Abstract Enormous efforts have been spent in the derivation of sufficient schedulability tests for popular global schedulers such as *global fixed-priority* (G-FP) and *global earliest-deadline first* (G-EDF). Among all the proposals, response-time analysis techniques are established as the most popular ones and have been widely adopted by several authors in subsequent researches. Such tests are based on interference bounds that have been derived by exploiting the work-conserving property of such schedulers, and are typically expressed with a $1/m$ multiplier in the interference equation (with m being the number of available processors).

This paper shows that the popular response-time analysis for G-EDF is ineffective with respect to *partitioned EDF* (P-EDF) scheduling. That is, it is proven that every task set that is deemed schedulable by such an analysis is also schedulable under P-EDF by adopting a trivial partitioning algorithm. Furthermore, an analogous result is proven for *global first-in first-out* (G-FIFO) scheduling when analyzed with a $1/m$-based interference bound. Finally, experimental results are presented to compare sufficient tests for G-EDF and G-FIFO with the corresponding partitioned schemes under a vast collection of partitioning heuristics. The results show that the considered tests for global scheduling are always outperformed by those for partitioned scheduling exhibiting a significant performance gap.

Alessandro Biondi
Scuola Superiore Sant'Anna, Pisa, Italy
E-mail: alessandro.biondi@santannapisa.it

Youcheng Sun
University of Oxford
E-mail: youcheng.sun@cs.ox.ac.uk

1 Introduction

Multicore platforms are, de-facto, the today's standard for modern processing systems. During the last two decades, with the pervasive adoption of multicores, multiprocessor scheduling received enormous attention by the real-time research community, which proposed a plethora of results to manage real-time applications on such platforms.

Two major approaches have been identified for scheduling real-time tasks on multiprocessors: *partitioned scheduling*, where each task is statically allocated to a given processor, and *global scheduling*, where each task can execute on any processor. Both the approaches have pros and cons [22]. Most relevant to this paper, it is worth mentioning that the timing properties of real-time tasks under uniprocessor (and hence also partitioned) scheduling are analytically well-understood, and efficient techniques are known to implement *exact* schedulability tests under both *fixed-priority* (FP) and *earliest-deadline first* (EDF) scheduling. Conversely, few of the results derived for uniprocessor scheduling generalize to multiprocessor global scheduling, which instead has been shown to originate very challenging problems when worst-case timing analysis is concerned. In particular, even relatively simple global schedulers such as *global fixed-priority* (G-FP) and *global earliest-deadline first* (G-EDF) are difficult to analyze. Also, their corresponding schedulability problems have been proven to be PSPACE-complete by Geeraerts et al. [27]. To date, all the techniques that allow implementing exact schedulability tests for G-FP [44, 45] and G-EDF [3,27,14] suffer from severe scalability issues, and many of them can produce a result in a reasonable amount of time only when the task parameters belong to fairly limited ranges (e.g., see the experimental results in [17]).

Due to the hardness in solving the schedulability problem under G-FP and G-EDF, several authors spent enormous effort in deriving sufficient schedulability tests for such schedulers. The interested reader can refer to the survey of Davis and Burns [22] for more detailed information on such results. Among all the proposals, *response-time analysis* techniques [10] established as very popular and influential, which likely happened because of their "conceptual compatibility" with the pre-existing analysis methods for uniprocessor systems, their higher (empirical) schedulability performance with respect to other sufficient tests (e.g., see [9,43]), and because they have been found easy for being extended. In fact, such techniques have been widely adopted by several authors as a fundamental building block to analyze more complex systems, including the support for global locking protocols [47], hierarchical scheduling [48], mixed-criticality scheduling [39], processor affinities [31] and parallel tasks [20].

The key concept of the response-time analysis for G-FP and G-EDF consists in exploiting the *work-conserving* property of such schedulers, which is leveraged in the derivation of upper bounds on the maximum interference that a task can suffer in a given time window. The resulting expressions for the interference bound comprise a $1/m$ multiplier (where m is the number of available processors), which plays a crucial role in overcoming most of the difficulties that would arise in the derivation of the exact interference. Although enabling a tractable analysis, this approach carries considerable pessimism, which is mostly related to the fact that it conservatively assumes that a sequential task can occupy more than one processor at the same time (thus generating more interference than the one that is actually possible).

Very recently, Sun and Di Natale [42] demonstrated that the pessimism introduced by the response-time analysis for G-FP is so large that such an analysis technique is dominated by pure *partitioned* FP (P-FP) scheduling. In particular, the authors proved that, as long as a task set is schedulable according to the G-FP test in [29], a heuristic that keeps the same G-FP priority assignment and allocates the tasks by following a decreasing priority order still guarantees the schedulability. Unfortunately, due to the fundamental differences with respect to G-FP, their result does not apply to G-EDF scheduling.

Besides G-FP and G-EDF, another popular scheduling policy that is adopted in real-time systems is *first-in-first-out* (FIFO). As representative examples, FIFO has been used to design predictable locking protocols, to ensure predictable delays when managing hardware accelerators, and in the arbitration of communication buses and memory transactions. Whenever multiple processors (or in general, resources) are available, the concept of global scheduling also applies to FIFO. For instance, *global* FIFO (G-FIFO) scheduling has been adopted in locking protocols for replicated shared resources [15], to manage hardware accelerators deployed into field programmable gate arrays (FPGAs) [12], and is even currently available in Linux to schedule processes (see the `SCHED_FIFO` scheduling class)—thus being exposed to billion of devices around the world. To the best of our knowledge, no efficient and exact analysis for G-FIFO is available. Due to this fact, similarly to G-FP and G-EDF, some authors derived a sufficient analysis by leveraging a $1/m$-based interference bound, e.g., as done in [38] and [12].

1.1 Contribution

This paper demonstrates that, if a task set composed of periodic/sporadic real-time tasks is guaranteed to be schedulable by means of the popular response-time analysis for G-EDF [10], then it is also schedulable under P-EDF. Furthermore, it is proven that the analysis of G-FIFO by means of a $1/m$-based interference bound is dominated by partitioned FIFO scheduling. In addition to the identified theoretical dominances, results from a large-scale experimental study are also presented to compare schedulability tests for global schedulers against partitioned scheduling with a vast collection of simple placement heuristics.

These results establish the ineffectiveness of such widespread and influential analysis techniques with respect to far simpler, predictable, and lightweight partitioned schedulers.

1.2 Paper structure

The remainder of this paper is organized as follows. Section 2 presents the adopted system model. Section 3 addresses the case of G-EDF scheduling, while Section 4 focuses on G-FIFO scheduling. Section 5 presents the experimental results. Section 6 reviews the related work. Finally, Section 7 discusses the presented results and concludes the paper.

2 System model

This paper considers the problem of scheduling a set $\Gamma = \{\tau_1, \dots, \tau_n\}$ of n sporadic real-time tasks upon m identical processors. Each task τ_i is characterized by a *worst-case execution time* (WCET) C_i, a *minimum inter-arrival time* T_i, and a relative *deadline* $D_i \le T_i$. The *utilization* of task τ_i is defined as $U_i = C_i/T_i$, while its *density* is defined as $\delta_i = C_i/D_i$. Tasks are assumed to be independent and do not self-suspend their execution. Each task τ_i generates an infinite sequence of *jobs* $J_i^1, J_i^2, \dots$, where job J_i^k is released at time r_i^k and finishes at time f_i^k. A job J_i^k is said to be *pending* during $[r_i^k, f_i^k)$. The (exact) response time R_i of a task τ_i is defined as the least upper-bound on the lengths of the time intervals in which one of its jobs is pending, i.e., $R_i = \sup_k\{f_i^k - r_i^k\}$.

This work considers both partitioned and global schedulers. Under partitioned scheduling, each task is *statically* assigned to a given processor, i.e., its jobs can only be executed on one processor. The scheduler maintains an ordered ready queue of jobs for each processor and, at any time t, executes the jobs at the head of the queues on the corresponding processors. Conversely, under global scheduling, tasks can execute on any of the available processors. The scheduler maintains a single ready queue of jobs and, at any time t, executes the first m jobs (if they exist) in the queue on the m processors.

The order of the above-mentioned ready queues depends on the implemented scheduling policy. This paper focuses on EDF and FIFO scheduling. Under EDF scheduling, each job J_i^k is assigned an absolute deadline $d_i^k = r_i^k + D_i$; the ready queues are then ordered by increasing absolute deadlines. Under FIFO scheduling, the ready queues are ordered by increasing release times r_i^k of the jobs.

A scheduling policy is said to be *work-conserving* if it leaves a processor idle only when there are not enough pending tasks to execute. Note that both EDF and FIFO are work-conserving scheduling policies. A task τ_i is said to be *interfered* by another task τ_j when τ_i is prevented to execute because of the execution of τ_j. System overheads (e.g., task preemption costs, migration costs) are not considered.

3 On the analysis of G-EDF

This section demonstrates that the well-established response-time analysis for G-EDF by Bertogna and Cirenei [10] is dominated by P-EDF. Before presenting such a result, it is first necessary to briefly recall the Bertogna and Cirenei's analysis and another important result related to uniprocessor EDF scheduling. To not unnecessarily complicate the presentation, prior work is recalled without providing the actual expression for all the terms in the involved formulas, while concentrating just on those that are relevant for the purpose of this work. This simplification does not affect the presented results and the interested reader can refer to the specific papers (properly referenced in the following) for further details.

3.1 Preliminaries

Theorem 1 (From [10]) *A task set Γ is schedulable under G-EDF on a platform with m identical processors if, for all tasks $\tau_i \in \Gamma$,*

$$\exists R_i^{ub} \in [0, D_i] \mid R_i^{ub} \geq C_i + \frac{1}{m} \sum_{\tau_j \in \Gamma \setminus \{\tau_i\}} I_{j,i}(R_i^{ub}), \tag{1}$$

where $I_{j,i}(t)$ denotes a bound on the maximum interference generated by τ_j on τ_i within any time window of length t.

In [10], the check for the existence (and the corresponding computation) of a value of R_i^{ub} that satisfies Equation (1) was performed by means of a fixed-point iteration. The bound $I_{j,i}(t)$ was derived by looking at particular worst-case execution patterns of tasks that maximize the amount of workload they can execute in an arbitrary time window of length t, *independently of the number of available processors*. Hence, $I_{j,i}(t)$ also expresses a conservative bound on the interfering workload under uniprocessor scheduling. This reasoning leads to the following observation.

Observation 1 *Note that Theorem 1 is also valid as a sufficient schedulability test for uniprocessor EDF scheduling, i.e., when applied with $m = 1$.*

As a confirmation, this fact has also been formally proven with the COQ proof assistant in the context of the PROSA project [19], where the results of [10] have been validated for $m \in \mathbb{N}_{\geq 1}$.

Concerning uniprocessor EDF scheduling, it is also necessary to recall another important result presented by Guan and Yi [30].

Theorem 2 (From [30]) *Consider a task set Γ scheduled by uniprocessor EDF. Let R_i be the (exact) response time of each task $\tau_i \in \Gamma$. For any two tasks τ_i and τ_j in Γ, if $D_i \leq D_j$, then $D_i - R_i \leq D_j - R_j$.*

From Theorem 2, it is possible to derive a simple, but very useful corollary.

Corollary 1 *Consider a task set Γ scheduled by uniprocessor EDF and let τ_i be the task (or one of the tasks) with the shortest relative deadline in Γ. If Γ is not schedulable, then τ_i is not schedulable.*

Proof By contradiction. Suppose that τ_i is schedulable but Γ is *not* schedulable, i.e., there exists another task $\tau_j \in \Gamma$ that misses its deadline. Since τ_i is schedulable, it must be that $R_i \leq D_i$, which gives $D_i - R_i \geq 0$. Similarly, since τ_j is *not* schedulable, it must be that $R_j > D_j$, which gives $D_j - R_j < 0$. By hypothesis, $D_i \leq D_j$. Hence, by Theorem 2 it follows that $D_i - R_i \leq D_j - R_j$. Contradiction. □

3.2 Main result

The following proof is based on a simple task partitioning algorithm named *First-Fit with Decreasing Deadlines* (FFDD).

Definition 1 (FFDD Partitioning) Given a task set Γ to be scheduled under P-EDF, sort all the tasks in Γ by *decreasing* relative deadlines. Then, following the resulting order, assign the tasks to the available processor according to the first-fit policy. The well-known exact feasibility test for EDF [7] is used to verify whether a task fits in a processor. Whenever a task cannot be allocated to any processor, the partitioning algorithm fails. Otherwise, if all the tasks are allocated, then the algorithm succeeds.

With the definition of the FFDD partitioning in place, it is finally possible to present the main result of this section.

Theorem 3 *If a task set Γ is schedulable under G-EDF according to Theorem 1, then it is also schedulable under P-EDF by applying the FFDD partitioning.*

Proof The proof is by contradiction. Suppose that the FFDD partitioning fails but Γ is schedulable under G-EDF according to Theorem 1.

Let τ_i be the task that the FFDD partitioning fails in allocating and let $\Gamma(P_k)$ be the set of tasks allocated to processor P_k (with $k = 1, \ldots, m$) before the algorithm fails. First note that, by the order with which the FFDD partitioning allocates tasks, all the tasks in the sets $\Gamma(P_1), \ldots, \Gamma(P_m)$ cannot have a relative deadline shorter than τ_i, that is

$$\forall k = 1, \ldots, m, \;\; \forall \tau_j \in \Gamma(P_k), \;\; D_i \leq D_j.$$

Also, since the FFDD partitioning fails, note that the task sets $\Gamma(P_1) \cup \{\tau_i\}, \ldots, \Gamma(P_m) \cup \{\tau_i\}$ are *all* not schedulable under uniprocessor EDF. By combining these two observations with Corollary 1, it follows that τ_i is *not* schedulable under uniprocessor EDF within each task set $\Gamma(P_1) \cup \{\tau_i\}, \ldots,$ $\Gamma(P_m) \cup \{\tau_i\}$.

Hence, by Observation 1, this also means that the following holds

$$\forall k = 1, \ldots, m, \quad \forall R_i^{ub} \in [0, D_i], \quad R_i^{ub} - C_i < \sum_{\tau_j \in \Gamma(P_k)} I_{j,i}(R_i^{ub}). \tag{2}$$

Now, consider a given $R_i^{ub} \in [0, D_i]$, then rename the LHS and the RHS of Equation (2) as A and B_k, respectively, for $k = 1, \ldots, m$. In this way, Equation (2) can be expressed as m inequalities $A < B_1, A < B_2, \ldots, A < B_m$, from which it is possible to derive a single inequality $mA < B_1 + B_2 + \ldots + B_m = \sum_{k=1}^{m} B_k$. By exploiting such a result, it follows that Equation (2) implies

$$\forall R_i^{ub} \in [0, D_i], \quad m(R_i^{ub} - C_i) < \sum_{k=1}^{m} \sum_{\tau_j \in \Gamma(P_k)} I_{j,i}(R_i^{ub}).$$

Let $\Gamma^* = \bigcup_{k=1}^{m} \Gamma(P_k)$. Then, the latter equation can be rewritten as

$$\forall R_i^{ub} \in [0, D_i], \quad R_i^{ub} - C_i < \frac{1}{m} \sum_{\tau_j \in \Gamma^*} I_{j,i}(R_i^{ub}).$$

Since, by construction, Γ^* is a subset of Γ that does not include τ_i, and $I_{j,i}(t)$ cannot be negative, it follows that also the following holds:

$$\forall R_i^{ub} \in [0, D_i], \quad R_i^{ub} - C_i < \frac{1}{m} \sum_{\tau_j \in \Gamma \setminus \{\tau_i\}} I_{j,i}(R_i^{ub}).$$

Hence, Γ is *not* schedulable under G-EDF according to Theorem 1, thus reaching a contradiction. The theorem follows. □

Theorem 3 has been based on the FFDD partitioning for the sake of simplicity. However, note that the same result can also be achieved with a partitioning algorithm based on a policy different from first-fit as long as tasks are ordered by decreasing deadlines.

3.3 Open problem

In 2015, Sun and Lipari [43] combined the G-EDF response-time analysis of [10] with the analysis technique proposed by Baruah in [5], which also targeted G-EDF. The resulting schedulability test was proved to dominate both the tests of [10] and [5]. Unfortunately, both the schedulability tests proposed in [43] and [5] make use of a different analysis window with respect to the one adopted in Theorem 1, hence making the results presented in this section not applicable. Whether a dominance of P-EDF over the test in [43] can be proven is therefore an open problem. Nevertheless, the experimental results reported in Section 5 shows that P-EDF significantly outperforms the test in [43] from an empirical perspective.

4 On the analysis of G-FIFO

This section focuses on FIFO scheduling and presents an analogous result to the one proposed in the previous section for EDF scheduling. FIFO is an extremely simple and predictable scheduling policy that guarantees the absence of starvation and a fair access to the contended shared resources. On the other hand, it generally forbids task preemption and typically leads to lower schedulability performance when compared to FP and EDF scheduling. As representative examples, FIFO scheduling is widely used in locking protocols, mechanisms to manage hardware accelerators, and in the arbitration of communication buses and memory transactions.

For the sake of adopting a homogeneous system model across the paper, in the following it is targeted FIFO scheduling for a set of sporadic real-time tasks modeled as presented in Section 2. However, the results also extend in a seamless manner to the case of a set of requests (with bounded duration) to be served by a resource (e.g., the case of critical sections or the use of hardware accelerators).

The following theorem, which is recalled from [1], expresses a condition for ensuring the schedulability of a set of tasks under uniprocessor FIFO scheduling.

Theorem 4 (From [1]) *A task set Γ is schedulable under FIFO scheduling on a single processor if, for all tasks $\tau_i \in \Gamma$,*

$$C_i + \sum_{\tau_j \in \Gamma \setminus \{\tau_i\}} C_j \leq D_i. \tag{3}$$

The above theorem can also be expressed in a compact form as follows.

Corollary 2 *A task set Γ is schedulable under FIFO scheduling on a single processor if*

$$\sum_{\tau_i \in \Gamma} C_i \leq \min_{\tau_i \in \Gamma}\{D_i\}.$$

Proof The corollary following by noting that the LHS of Equation (3) is the same for all tasks $\tau_i \in \Gamma$ and is equal to $\sum_{\tau_i \in \Gamma} C_i$.

For the case of multiple processors, or more in general when multiple identical resources are managed according to FIFO, two natural extensions can be devised: **(i)** P-FIFO scheduling, where each task is statically assigned to a processor; and **(ii)** G-FIFO scheduling, where pure global scheduling is applied following a FIFO-ordered ready queue.

The schedulability under P-FIFO can be verified by applying Theorem 4 to each task partition. Conversely, to the best of our knowledge, the schedulability problem under G-FIFO scheduling cannot be solved in an efficient way. However, a simple sufficient schedulability test for G-FIFO can be derived by leveraging a $1/m$-based interference bound as adopted in Theorem 1.

Theorem 5 *A task set Γ is schedulable under G-FIFO scheduling on a platform with m identical processors if, for all tasks $\tau_i \in \Gamma$,*

$$C_i + \frac{1}{m} \sum_{\tau_j \in \Gamma \setminus \{\tau_i\}} C_j \leq D_i.$$

Proof Consider a task τ_i. Being G-FIFO scheduling work conserving, if τ_i is delayed, then it means that *all* the m processors are busy serving other tasks. As a consequence, if τ_i incurred in interference for I_i time units, then the platform processed mI_i time units of the interfering workload. Analogously to the uniprocessor case (Theorem 4), the maximum workload that can interfere with τ_i is given by $X = \sum_{\tau_j \in \Gamma \setminus \{\tau_i\}} C_j$. Hence, it follows that $mI_i \leq X$, which gives $I_i \leq X/m$. This allows concluding that the response time of τ_i is upper-bounded by $R_i^{ub} = C_i + X/m$. The task is schedulable if its response time is not greater than its relative deadline. The theorem follows.

For instance, an analogous analysis approach for G-FIFO has been adopted in [38] and [12]. With the above results in place, it is finally possible to present the main theorem of this section. Similarly as done for G-EDF in the previous section, a simple partitioning algorithm is defined.

Definition 2 (FFDD-FIFO Partitioning) Same as FFDD partitioning defined in Definition 1, but Theorem 4 is used in place of the EDF feasibility test to verify whether a task fits in a processor.

The following theorem states that the FFDD-FIFO partitioning dominates global FIFO scheduling when its schedulability is checked with a $1/m$-based interference bound.

Theorem 6 *If a task set Γ is schedulable under G-FIFO according to Theorem 5, then it is also schedulable under P-FIFO by applying the FFDD-FIFO partitioning.*

Proof The proof is by contradiction. Suppose that the FFDD-FIFO partitioning fails but Γ is schedulable under G-FIFO according to Theorem 5.

Let τ_i be the task that the FFDD-FIFO partitioning fails in allocating and let $\Gamma(P_k)$ be the set of tasks allocated to processor P_k (with $k = 1, \ldots, m$) before the algorithm fails. By the order with which the FFDD-FIFO partitioning allocates tasks, all the task sets $\Gamma(P_1), \ldots, \Gamma(P_m)$ include tasks with relative deadlines that are no shorter than the one of τ_i, that is

$$\forall k = 1, \ldots, m, \quad \forall \tau_j \in \Gamma(P_k), \ D_i \leq D_j,$$

which implies

$$\forall k = 1, \ldots, m, \quad \min_{\tau_j \in \Gamma(P_k) \cup \{\tau_i\}} \{D_j\} = D_i.$$

Also, since the FFDD-FIFO partitioning fails, the task sets $\Gamma(P_1) \cup \{\tau_i\}, \ldots, \Gamma(P_m) \cup \{\tau_i\}$ are *all* not schedulable according to Theorem 4.

Hence, by Corollary 2, it follows that

$$\forall k = 1, \ldots, m, \quad \sum_{\tau_j \in \Gamma(P_k)} C_j > \min_{\tau_j \in \Gamma(P_k) \cup \{\tau_i\}} \{D_j\} - C_i = D_i - C_i. \tag{4}$$

Similarly as argued in the proof of Theorem 3, Equation (4) implies that

$$\sum_{k=1}^{m} \sum_{\tau_j \in \Gamma(P_k)} C_j > m(D_i - C_i).$$

Let $\Gamma^* = \bigcup_{k=1}^{m} \Gamma(P_k)$. Then, the latter equation can be rewritten as

$$\frac{1}{m} \sum_{\tau_j \in \Gamma^*} C_j > D_i - C_i.$$

Since, by construction, Γ^* is a subset of Γ that does not include τ_i, it follows that also the following holds

$$C_i + \frac{1}{m} \sum_{\tau_j \in \Gamma \setminus \{\tau_i\}} C_j > D_i.$$

Hence, Γ is *not* schedulable under G-FIFO according to Theorem 5, thus reaching a contradiction. The theorem follows. □

5 Experimental results

Although the previous sections showed a theoretical dominance of partitioned EDF and FIFO scheduling over the corresponding $1/m$-based schedulability tests for global scheduling, an experimental study has been conducted to assess the empirical schedulability performance of the considered analysis approaches. In particular, the experimental study focused on comparing the two schedulability tests for global scheduling discussed in the previous sections against *thirty* schemes for partitioned scheduling, which resulted from the combination of:

- three popular *placement heuristics*, namely First-fit (FF), Worst-fit (WF), and Best-fit (BF); and
- ten *task orderings*, which are Increasing Deadline (ID), Decreasing Deadline (DD), Increasing WCET (IW), Decreasing WCET (DW), Increasing Period (IP), Decreasing Period (DP), Increasing Density (IDen), Decreasing Density (DDen), Increasing Utilization (IU), and Decreasing Utilization (DU).

In the following, the combination of a placement heuristic with a task ordering is denoted with the two corresponding acronyms separated by an hyphen. The complete data set is publicly available [13].

5.1 Task set generation

Given a target utilization U and a number of tasks n, task sets have been generated by means of the Emberson et al.'s [25] generator, which has been configured for generating random task periods in the range $[1, 1000]$ ms with log-uniform distribution. For each task τ_i, the relative deadline D_i has been randomly generated in the range $[C_i+\beta(T_i-C_i), T_i]$ with uniform distribution, where β is a generation parameter.

5.2 Experiments for EDF scheduling

A first experiment has been performed to compare the schedulability test for G-EDF provided by Theorem 1 (denoted as G-EDF-BC [10]) and the test presented by Sun and Lipari in [43] (denoted as G-EDF-SL) against the thirty partitioning schemes for P-EDF. By following the theoretical results presented in [43] and the experimental results reported in [9], the G-EDF-SL test results the one with the highest empirical performance among those available in the published literature to date: for this reason, it has been included in the comparison. The well-established processor-demand criterion [7] for uniprocessor EDF scheduling has been used when checking the system schedulability under P-EDF. The schedulability tests have been compared with a multidimensional exploration of the parameters that control the task set generation, which have been varied as follows:

- the number of processors $m \in \{4, 8, 16\}$;
- the number of tasks $n \in \{m + 2, 2m, 3m, 4m\}$;
- $\beta \in \{0.5, 0.75, 1\}$;
- the utilization $U \in [0.1m, 0.975m]$ with step $0.025m$.

For each configuration, 1000 randomly-generated task sets have been tested. By looking at all the obtained results, the following conclusions emerged:

(i) *All* the combinations of placement heuristics and task orderings for P-EDF significantly outperform *both* the schedulability tests for G-EDF in all the tested scenarios.
(ii) In terms of schedulability performance, the choice of the placement heuristic has a minimal impact. Conversely, the choice of the task ordering has a high impact.
(iii) The DU task ordering generally shows the highest schedulability performance, while the IU ordering tends to exhibit the lowest performance.
(iv) As one might easily expect, the performance of P-EDF tends to increase as the number of tasks increases.

Figure 1 reports the results for some representative configurations as a function of the task set utilization U. The graphs also report the cumulative schedulability performance obtained by testing all the thirty strategies (i.e., the schedulability test of all the strategies merged in logic OR condition). In all the tested scenarios, the performance gap between the tests for G-EDF and those for P-EDF is evident and wide. As it can be observed from the graphs, there are several configurations in which the tests for G-EDF cannot schedule task sets while P-EDF is able to schedule the 100% of the generated task sets with all the tested heuristics (e.g., see $U = 5.5$ in Figure 1(b)). The performance gap between the two tests for G-EDF is always marginal. Furthermore, no counter-examples have been found in which the tests for G-EDF deem a task set schedulable while it is not under P-EDF: hence, there may be room for proving that also the test from [43] is analytically dominated by P-EDF. Nevertheless, note that there may be some P-EDF schemes (e.g., under some of the tested partitioning strategies) that do not dominate the tests for G-EDF: counter-examples have been found during the experimentation.

Figure 2 reports the results for other configurations in which there is a low number of tasks ($n = m + 2$), hence being unfavorable for P-EDF (for a fixed system utilization U, the existence of a suitable task partitioning for P-EDF is notably less likely in the presence of a few tasks as long as $n > m$). As it can be noted from the plots, similar conclusions to the ones discussed above can also be drawn in such scenarios.

5.3 Experiments for FIFO scheduling

A second experiment has been performed to compare the sufficient schedulability test for G-FIFO provided by Theorem 5 (denoted as G-FIFO-1m) against the thirty partitioning schemes for P-FIFO. For the sake of simplicity, the same experimental setting used for EDF has also been adopted in this case. By looking at the complete set of experimental results, three major trends emerged:

(i) The G-FIFO-1m test shows very poor schedulability performance even for low values of utilization U, and is outperformed by all the considered P-FIFO schemes in all the tested scenarios.
(ii) Similarly to the case of EDF scheduling, the choice of the placement heuristic has a limited impact in terms of schedulability performance. Conversely, the choice of the task ordering has a high impact.
(iii) The combination of all the partitioning schemes exhibits a significant performance improvement with respect to each individual heuristic.

Figure 3 reports the results for some representative configurations as a function of the task set utilization U. In all the three graphs, the G-FIFO-1m test is almost ineffective, while the DD, DP and DW task orderings show the highest performance. Note that the results are a bit noisy because the tests for P-FIFO are significantly affected by smallest deadline in the considered task set (see Theorem 4), which is randomly generated and not explicitly controlled by varying the utilization U.

For the same reason, a variant of this experiment has been performed by changing the minimum task period from 1 ms to 10 ms in the configuration of the task set generator: the results showed an improvement of the schedulability performance for all the schedulability tests as the minimum period increases. As a representative case, Figure 4 reports the experimental results for the same configuration of Figure 3(a) but with task periods generated in the range [10,1000] milliseconds.

6 Related Work

For the case of uniprocessor systems, both FP and EDF scheduling have been demonstrated to benefit from the critical instant theorem [35,40], which allows deriving sufficient and necessary conditions that guarantee the schedulability of a set of sporadic tasks by looking at a particular release pattern. Furthermore, tasks released as soon as possible represents the worst-case scenario for such a task model. Unfortunately, under G-FP and G-EDF, the uniprocessor critical instant theorem does not necessarily allow identifying the worst-case release pattern [2], and variable (and non-trivial) inter-arrival times of sporadic tasks may lead to the maximum interference.

Consequently, all the attempts to achieve an exact (or very precise) schedulability analysis of G-FP and G-EDF had to take into account a large number of task activations and interleaving patterns (or even all possible). The first of such proposals is due to Baker and Cirenei [3], which adopted a brute-force approach for both G-FP and G-EDF. Later, Geeraerts et al. [27] improved the approach of [3] by adopting techniques developed for formal verification. In 2012, Bonifaci and Marchetti-Spaccamela [14] presented an algorithm for checking the feasibility of a set of sporadic tasks under multiprocessor global scheduling. Still based on the same approach, the authors also provided algorithms for checking the schedulability of G-FP and G-EDF. Building upon the method presented in [14], Burmyakov et al. [17] proposed a faster (w.r.t. the state-of-the-art) and exact test for G-FP by relying on accurate state-space pruning. Sun and Lipari [44,45] proposed an exact analysis for G-FP by modeling the system as a linear hybrid automaton. The authors identified a pre-order relationship over the symbolic states of the model to mitigate the problem of state space explosion. Other methods have been proposed by Guan et al. [28] and Cucu-Grosjean and Goossens [21] for G-FP: the authors limited the analysis to the case of strictly periodic tasks, thus allowing to reduce the number of scenarios to be considered. Notably, all such analysis techniques often suffer *severe* scalability problems.

On the other hand, the major efforts in the literature of multiprocessor global schedulability analysis concern approximate results, among which Baker's work [2] served as the basis for most of the prominent results. As the Liu and Layland's critical instant theorem fails under multiprocessor global scheduling, Baker [2] considered that the interfering tasks may bring carry-in

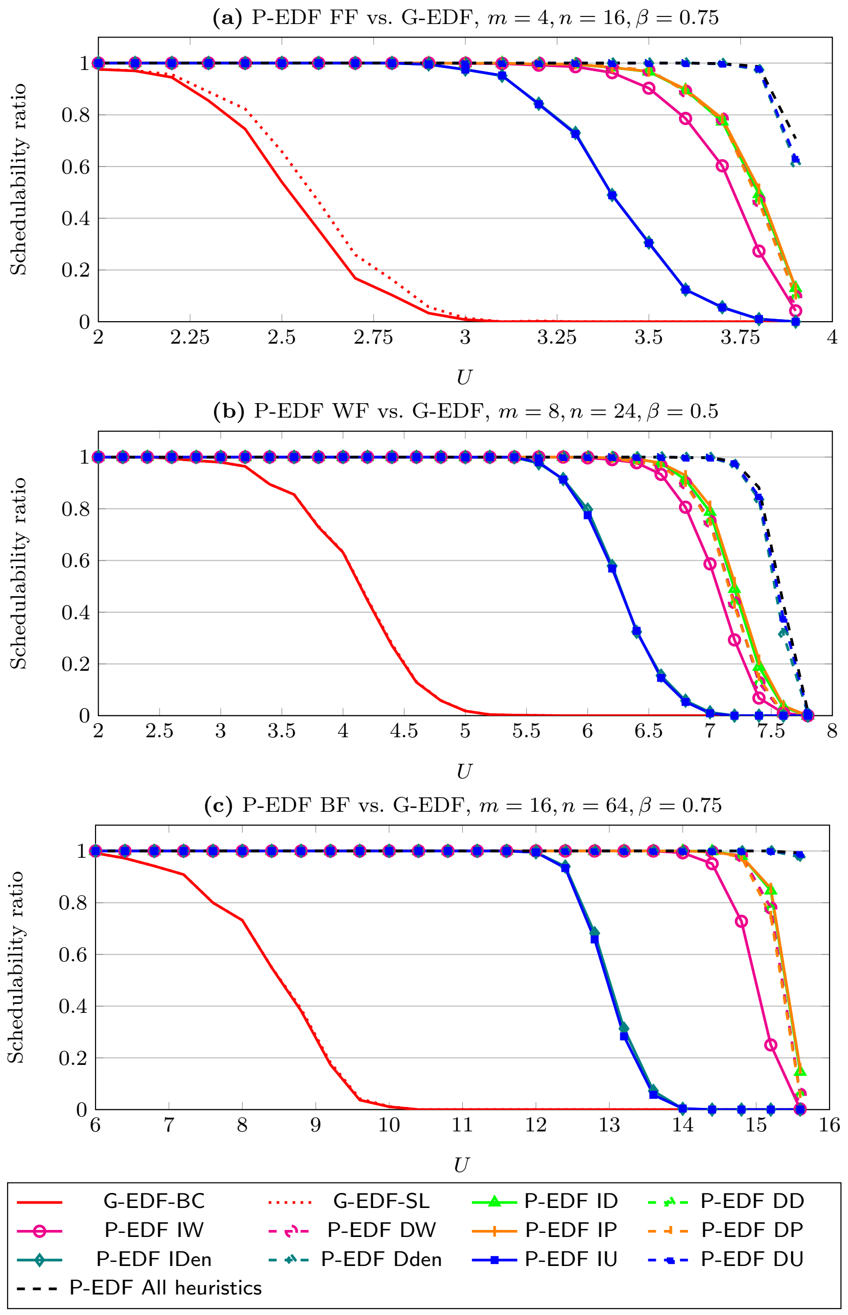

Fig. 1 Experimental results comparing G-EDF with P-EDF under three representative configurations. The parameters of each configuration are reported in the captions above the graphs.

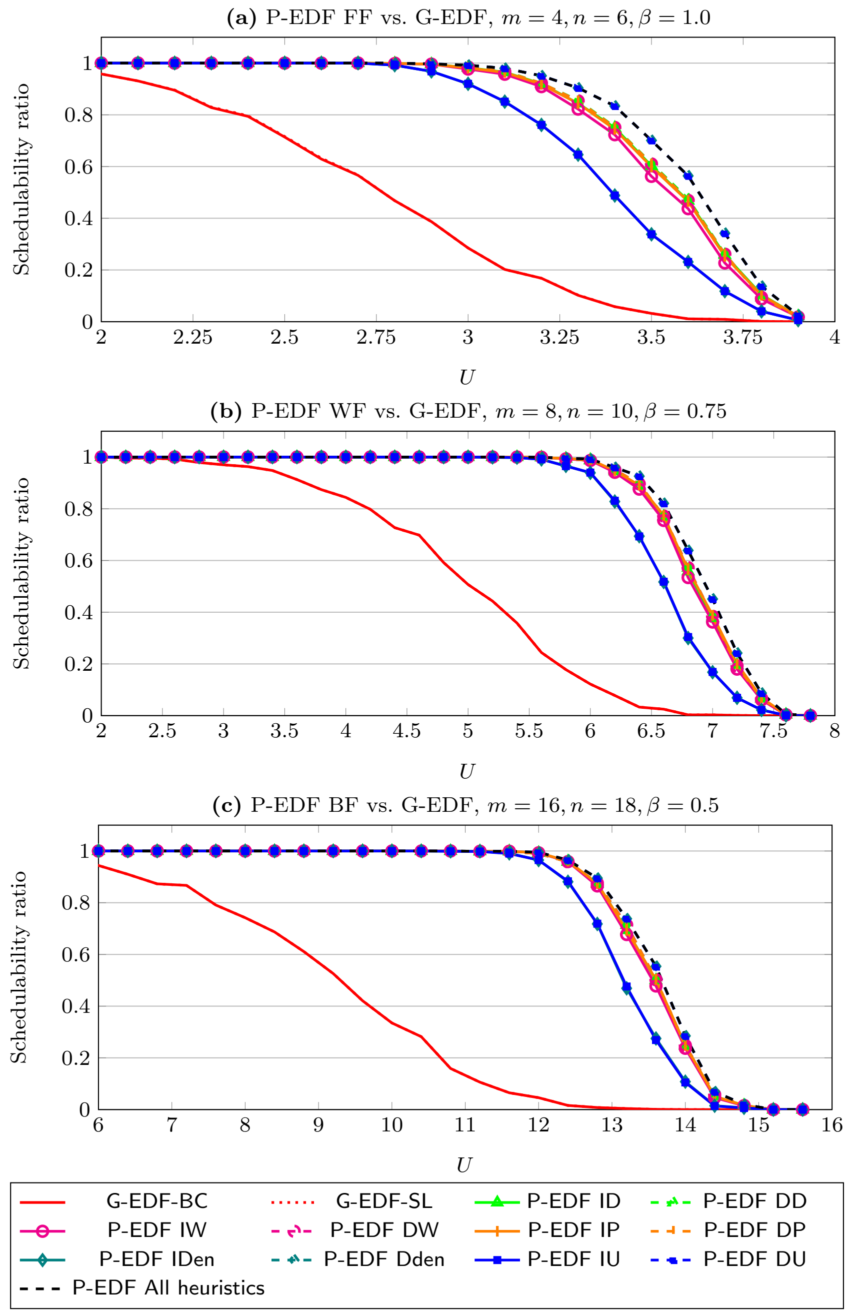

Fig. 2 Experimental results comparing G-EDF with P-EDF under three representative configurations with a low number of tasks $n = m + 2$. The parameters of each configuration are reported in the captions above the graphs.

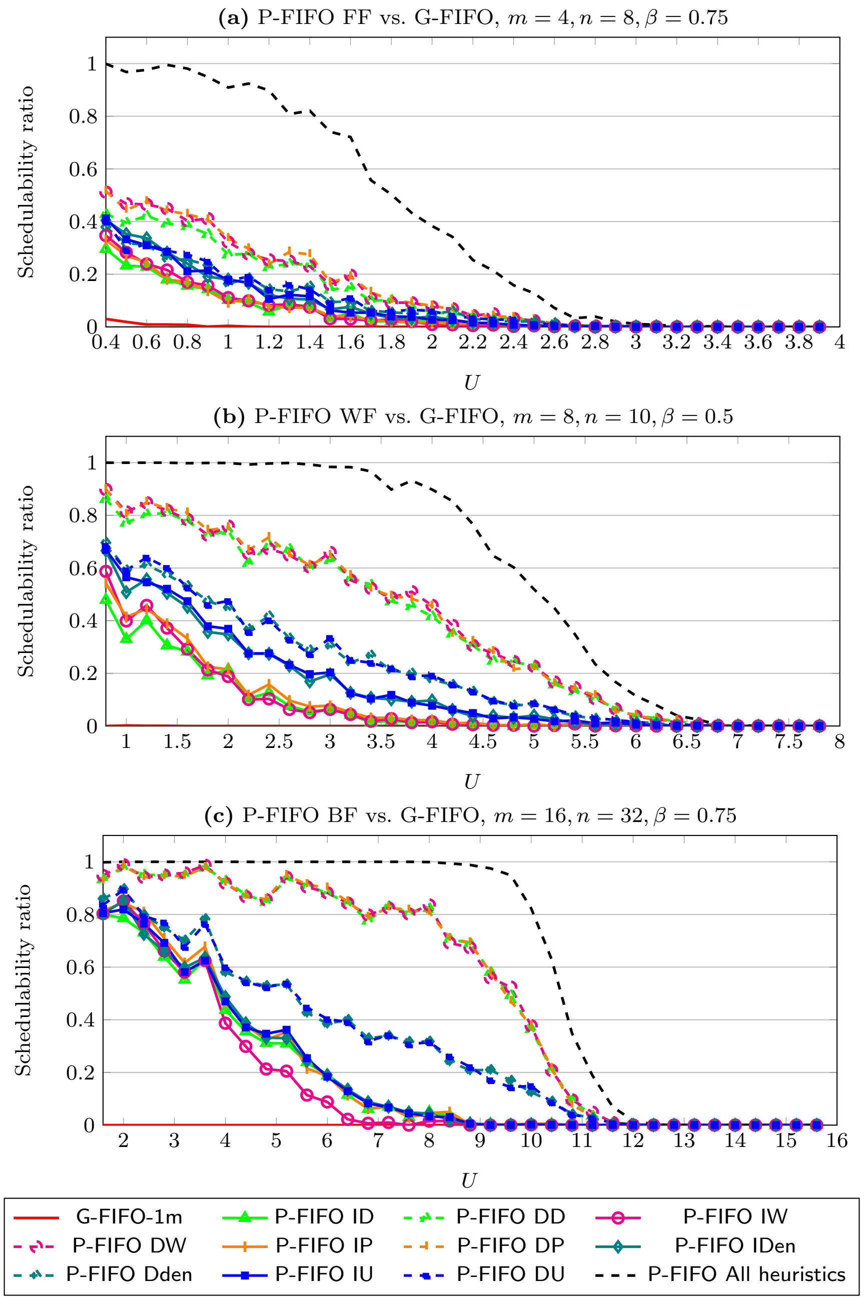

Fig. 3 Experimental results comparing G-FIFO with P-FIFO under three representative configurations. The parameters of each configuration are reported in the captions above the graphs.

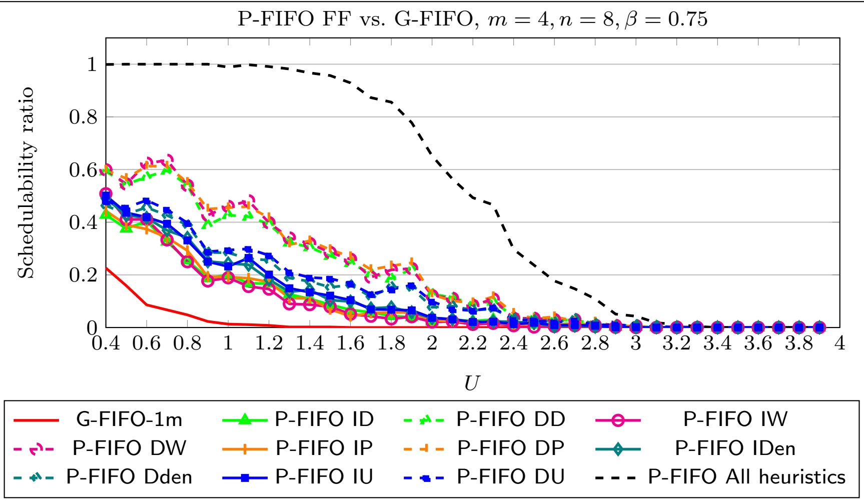

Fig. 4 Experimental results for the configuration reported in Figure 3(a) but with task periods generated in the range [10,1000] milliseconds.

(CI) interference into the analysis window of a given task.[1] The concept of CI received considerable attention in the works that targeted the analysis of G-FP and G-EDF. In the same paper, Baker identified a safe bound to the CI workload that any task can cause. Later, Bertogna et al. [11] improved the results of [2] by relying on a more precise characterization of the interference. In 2007, Bertogna and Cirenei [10] presented the response-time analysis for both G-FP and G-EDF (that has been objective of Section 3), which improved the performance of the approaches in [2] and [11]. Still in 2007, Baruah [5] proposed the *limited-CI* technique, which relies on the fact that it is sufficient to consider at most $m - 1$ tasks in the identification of the CI interference. This result significantly reduced the analysis pessimism. Guan et al. [29] and Sun and Lipari [43] integrated the limited-CI technique with the response-time analysis presented in [10] for G-FP and G-EDF, respectively.

Concerning G-FP, since the publication of [29], most following works are based on the results presented in [29]: e.g., the ones by Davis and Burns [23] on priority assignment for G-FP, Liu and Anderson [34] on self-suspending tasks, and Sun et al. [46] and Huang and Chen [32] on improved analysis techniques and models. However, it bears repeating that the analysis presented in [29] has been found to be dominated by P-FP in [42].

Another result presented by Sun [41] (Chapter 9) warns the use of multiprocessor global schedulability tests under the assumption of discrete time. In particular, it is demonstrated that for G-FP scheduling the schedulability result is sensitive to the selected time granularity.

[1] An interfering job is called the carry-in job if it is released before the start of the problem window under analysis and it still has unfinished execution when entering the window. A task may generate larger interference if it brings carry-in.

Finally, from a practical perspective, it is worth mentioning interesting research [8,33] that studied with quantitative measurements the overheads introduced by both G-EDF and P-EDF, where the former has been found to suffer from scalability issues as the number of cores increases.

6.1 Different partitioning schemes

In this paper, the FFDD partitioning scheme has been adopted to prove the dominance of P-EDF over two schedulability tests for G-EDF and G-FIFO. However, the authors do not have any intention to give readers the impression that FFDD partitioning is more effective than other partitioning heuristics. In fact, the experimental results presented in Section 5 witness that higher empirical performance can be obtained with partitioning schemes different from FFDD (e.g., by adopting the DU task ordering).

The partitioning problem of real-time tasks for multiprocessor systems is in principle a bin-packing problem. Due to the complexity of its exact formulation [4], in practice, many heuristic algorithms have been developed in previous work. Such algorithms often share the A-B-C pattern: A) the fitting strategy, B) the direction of the task ordering (decreasing or increasing), and C) the parameter with respect to which the tasks are ordered (e.g, deadline, density, etc.). López et al. [37,36] investigated a variety of these heuristics for the partitioned EDF scheduling of implicit-deadline tasks. Baruah and Fisher [6] studied the use of non-decreasing deadline ordering to partition sporadic tasks on multiprocessors, where EDF is considered as local scheduling policy. A similar idea has also been applied to partition tasks when the local scheduler in a processor is fixed-priority [26]. In both the cases, different fitting strategies can be employed. As a matter of fact, the decreasing deadline is not typically regarded as an effective method for task partitioning. For example, in the context of fixed-priority scheduling, Davis et al. [24] showed that by using first-fit fitting with decreasing density the schedulability performance results were significantly better than when using FFDD. Similar conclusions have been obtained in our experimental study (see Section 5).

7 Discussion and Conclusion

This paper proved that a popular and influential analysis technique for G-EDF, in particular the one based on response-time analysis for sporadic real-time tasks, is dominated by P-EDF with a trivial partitioning scheme. In fact, every task set that is deemed schedulable under G-EDF with such an analysis, is demonstrated to be also schedulable by statically assigning tasks to the available processors with a first-fit heuristic (specifically, selecting tasks by decreasing relative deadlines). This result establishes the ineffectiveness of such analysis techniques with respect to partitioned scheduling, which is arguably far simpler to implement and more predictable with respect to global scheduling, as well as known for introducing less run-time overhead. An analogous result has been proved for G-FIFO scheduling, which found application in relevant domains such as locking protocols for replicated shared resources and the management of hardware accelerators. The identified pessimism is attributed to the approach used to bound the interference suffered by tasks, which is based on exploiting the work-conserving property of such schedulers and typically includes a $1/m$ multiplier in the bound expression. A large-scale experimental study has also been conducted to compare $1/m$-based sufficient tests for G-EDF and G-FIFO with the corresponding partitioned schemes under thirty partitioning heuristics. The results showed that the considered tests for global scheduling are always outperformed by *all* the thirty heuristics for partitioned scheduling. Furthermore, the performance gap between the tests for global and partitioned scheduling resulted very large (even up to 100%).

The analysis techniques for global scheduling targeted by this paper have been adopted by a wide number of authors in subsequent researches, which used them as a building block to analyze more complex systems (e.g., locking protocols, hierarchical scheduling) or enriched task models (e.g., self-suspending tasks, parallel tasks). In the light of the findings of this paper, it may be possible that the conclusions drawn in some of those works are biased by the identified pessimism.

Furthermore, the results of this paper provide new food for thought concerning the need for global scheduling in managing real-time workload, which has recently been questioned in recent works [16,18] by showing that simple semi-partitioned scheduling can achieve very high schedulability performance. The considered tests for global scheduling also found a questionable application in the context of dynamic workload: since the partitioning scheme that dominates them is very simple, the same worst-case performance may be guaranteed by employing efficient algorithms for on-line task re-partitioning, with a resulting reduction of run-time overhead in steady-state conditions.

Nevertheless, if it is intended to pursue the study of global scheduling, new improved (and efficient) analysis techniques are needed. Note that, with a proper empirical evaluation of schedulability tests for global scheduling against tests for partitioned scheduling, the results presented in this work may have been discovered many years ago. Therefore, the authors argue that future work on the analysis of global schedulers should not only compare with state-of-the-art analysis techniques but also include a comparison with partitioned schedulers under multiple simple placement heuristics.

Finally, future work should verify whether similar results also hold for other analysis techniques for global scheduling, and for more expressive task models such as parallel tasks or tasks with self-suspensions.

References

1. S. Altmeyer, S. M. Sundharam, and N. Navet. The case for FIFO real-time scheduling. In *Technical report, University of Luxembourg*, 2016. `http://hdl.handle.net/10993/`

24935.

2. T. P. Baker. Multiprocessor EDF and deadline monotonic schedulability analysis. In *Proceedings of the 24th IEEE Real-Time Systems Symposium (RTSS), December 3-5, 2003, Cancun, Mexico*, pages 120–129.
3. T. P. Baker and M. Cirinei. Brute-force determination of multiprocessor schedulability for sets of sporadic hard-deadline tasks. In *Principles of Distributed Systems, 11th International Conference, OPODIS, December 17-20, 2007, Guadeloupe, French West Indies*, pages 62–75.
4. S. Baruah and E. Bini. Partitioned scheduling of sporadic task systems: an ILP-based approach. Nov. 2008.
5. S. K. Baruah. Techniques for multiprocessor global schedulability analysis. In *Proceedings of the 28th IEEE Real-Time Systems Symposium (RTSS), December 3-6, 2007, Tucson, Arizona, USA*, pages 119–128.
6. S. K. Baruah and N. Fisher. The partitioned multiprocessor scheduling of deadline-constrained sporadic task systems. *IEEE Trans. Computers*, 55(7):918–923, 2006.
7. S. K. Baruah, L. E. Rosier, and R. R. Howell. Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor. *Real-Time Systems*, 2(4):301–324, 1990.
8. A. Bastoni, B. B. Brandenburg, and J. H. Anderson. An empirical comparison of global, partitioned, and clustered multiprocessor edf schedulers. In *31st IEEE Real-Time Systems Symposium (RTSS)*, Nov. 2010.
9. M. Bertogna and S. Baruah. Tests for global EDF schedulability analysis. *Journal of Systems Architecture*, 57(5):487–497, 2011. Special Issue on Multiprocessor Real-time Scheduling.
10. M. Bertogna and M. Cirinei. Response-time analysis for globally scheduled symmetric multiprocessor platforms. In *Proceedings of the 28th IEEE Real-Time Systems Symposium (RTSS), December 3-6, 2007, Tucson, Arizona, USA*, pages 149–160.
11. M. Bertogna, M. Cirinei, and G. Lipari. Improved schedulability analysis of EDF on multiprocessor platforms. In *Proceedings of the 17th Euromicro Conference on Real-Time Systems (ECRTS), July 6-8, 2005, Palma de Mallorca, Spain*, pages 209–218.
12. A. Biondi, A. Balsini, M. Pagani, E. Rossi, M. Marinoni, and G. C. Buttazzo. A framework for supporting real-time applications on dynamic reconfigurable FPGAs. In *Proceedings of the 37th IEEE Real-Time Systems Symposium, (RTSS), November 29 - December 2, 2016, Porto, Portugal*, pages 1–12.
13. A. Biondi and Y. Sun. On the ineffectiveness of 1/m-based interference bounds in the analysis of global edf and fifo scheduling: complete data set of experimental results. `http://retis.sssup.it/~a.biondi/1mexps/`.
14. V. Bonifaci and A. Marchetti-Spaccamela. Feasibility analysis of sporadic real-time multiprocessor task systems. *Algorithmica*, 63(4):763–780, 2012.
15. B. B. Brandenburg. *Scheduling and locking in multiprocessor real-time operating systems.* PhD thesis, The University of North Carolina at Chapel Hill, 2011. `http://www.cs.unc.edu/~bbb/diss/brandenburg-diss.pdf`.
16. B. B. Brandenburg and M. Gul. Global scheduling not required: Simple, near-optimal multiprocessor real-time scheduling with semi-partitioned reservations. In *Proceedings of the 37th IEEE Real-Time Systems Symposium (RTSS), November 29 - December 2, 2016, Porto, Portugal*, pages 99–110.
17. A. Burmyakov, E. Bini, and E. Tovar. An exact schedulability test for global FP using state space pruning. In *Proceedings of the 23rd International Conference on Real Time Networks and Systems (RTNS), November 4-6, 2015, Lille, France*, pages 225–234.
18. D. Casini, A. Biondi, and G. C. Buttazzo. Semi-partitioned scheduling of dynamic real-time workload: A practical approach based on analysis-driven load balancing. In *Proceedings of the 29th Euromicro Conference on Real-Time Systems (ECRTS), June 27-30, 2017, Dubrovnik, Croatia*, pages 13–23.
19. F. Cerqueira, F. Stutz, and B. B. Brandenburg. PROSA: A case for readable mechanized schedulability analysis. In *Proceedings of the 28th Euromicro Conference on Real-Time Systems (ECRTS), July 5-8, 2016, Toulouse, France*, pages 273–284.
20. H. S. Chwa, J. Lee, K. Phan, A. Easwaran, and I. Shin. Global EDF schedulability analysis for synchronous parallel tasks on multicore platforms. In *Proceedings of the*

25th Euromicro Conference on Real-Time Systems (ECRTS), July 9-12, 2013, Paris, France.

21. L. Cucu-Grosjean and J. Goossens. Exact schedulability tests for real-time scheduling of periodic tasks on unrelated multiprocessor platforms. *Journal of systems architecture*, 57(5):561–569, 2011.
22. R. I. Davis and A. Burns. A survey of hard real-time scheduling for multiprocessor systems. *ACM Comput. Surv.*, 43(4).
23. R. I. Davis and A. Burns. Improved priority assignment for global fixed priority pre-emptive scheduling in multiprocessor real-time systems. *Real-Time Systems*, 47(1):1–40, 2011.
24. R. I. Davis, A. Burns, J. Marinho, V. Nelis, S. M. Petters, and M. Bertogna. Global and partitioned multiprocessor fixed priority scheduling with deferred preemption. *ACM Transactions on Embedded Computing Systems (TECS)*, 14(3):47, 2015.
25. P. Emberson, R. Stafford, and R. I. Davis. Techniques for the synthesis of multiprocessor tasksets. In *Proceedings of the 1st International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS), July 6, 2010, Brussels, Belgium*, pages 6–11.
26. N. Fisher, S. K. Baruah, and T. P. Baker. The partitioned scheduling of sporadic tasks according to static-priorities. In *Proceedings of the 18th Euromicro Conference on Real-Time Systems (ECRTS), July 5-7, 2006, Dresden, Germany*, pages 118–127.
27. G. Geeraerts, J. Goossens, and M. Lindström. Multiprocessor schedulability of arbitrary-deadline sporadic tasks: complexity and antichain algorithm. *Real-time systems*, 49(2):171–218, 2013.
28. N. Guan, Z. Gu, Q. Deng, S. Gao, and G. Yu. Exact schedulability analysis for static-priority global multiprocessor scheduling using model-checking. *Software Technologies for Embedded and Ubiquitous Systems*, pages 263–272, 2007.
29. N. Guan, M. Stigge, W. Yi, and G. Yu. New response time bounds for fixed priority multiprocessor scheduling. In *Proceedings of the 30th IEEE Real-Time Systems Symposium (RTSS), December 1-4, 2009, Washington, DC, USA*, pages 387–397.
30. N. Guan and W. Yi. General and efficient response time analysis for EDF scheduling. In *Design, Automation and Test in Europe Conference (DATE), March 24-28, 2014, Dresden, Germany*, pages 1–6.
31. A. Gujarati, F. Cerqueira, and B. B. Brandenburg. Schedulability analysis of the linux push and pull scheduler with arbitrary processor affinities. In *Proceedings of the 25th Euromicro Conference on Real-Time Systems (ECRTS), July 9-12, 2013, Paris, France*, pages 69–79.
32. W.-H. Huang and J.-J. Chen. Response time bounds for sporadic arbitrary-deadline tasks under global fixed-priority scheduling on multiprocessors. In *Proceedings of the 23rd International Conference on Real Time Networks and Systems (RTNS), November 4-6, 2015, Lille, France*, pages 215–224.
33. J. Lelli, D. Faggioli, T. Cucinotta, and G. Lipari. An experimental comparison of different real-time schedulers on multicore systems. *Journal of Systems and Software*, 85(10):2405 – 2416, 2012.
34. C. Liu and J. H. Anderson. Suspension-aware analysis for hard real-time multiprocessor scheduling. In *Proceedings of the 25th Euromicro Conference on Real-Time Systems (ECRTS), July 9-12, 2013, Paris, France*, pages 271–281.
35. C. Liu and J. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the Association for Computing Machinery*, 20(1):46–61, January 1973.
36. J. M. López, J. L. Díaz, and D. F. García. Utilization bounds for edf scheduling on real-time multiprocessor systems. *Real-Time Systems*, 28(1):39–68, 2004.
37. J. M. López, M. García, J. L. Díaz, and D. F. García. Worst-case utilization bound for EDF scheduling on real-time multiprocessor systems. In *Proceedings of the 12th Euromicro Conference on Real-Time Systems (ECRTS), June 19-21, 2000, Stockholm, Sweden, Proceedings*, pages 25–33.
38. C. E. Nemitz, K. Yang, M. Yang, P. Ekberg, and J. H. Anderson. Multiprocessor real-time locking protocols for replicated resources. In *Proceedings of the 28th Euromicro Conference on Real-Time Systems (ECRTS), July 5-8, 2016, Toulouse, France*, pages 50–60.

39. R. M. Pathan. Schedulability analysis of mixed-criticality systems on multiprocessors. In *Proceedings of the 24th Euromicro Conference on Real-Time Systems (ECRTS), July 11-13, 2012, Pisa, Italy*, pages 309–320.
40. M. Spuri. Analysis of deadline scheduled real-time systems. Research Report RR-2772, INRIA, 1996. Projet REFLECS.
41. Y. Sun. *Real-time schedulability analysis with formal techniques.* PhD thesis, Scuola Superiore S. Anna, 2015. `http://retis.sssup.it/?q=content/phd-theses`.
42. Y. Sun and M. Di Natale. Assessing the pessimism of current multicore global fixed-priority schedulability analysis. In *Proceedings of the 33rd Annual ACM Symposium on Applied Computing (SAC), April 9-13, 2018, Pau, France.*
43. Y. Sun and G. Lipari. Response time analysis with limited carry-in for global earliest deadline first scheduling. In *Proceedings of the 36th IEEE Real-Time Systems Symposium (RTSS), December 1-4, 2015, San Antonio, Texas, USA*, pages 130–140.
44. Y. Sun and G. Lipari. A weak simulation relation for real-time schedulability analysis of global fixed priority scheduling using linear hybrid automata. In *Proceedings of the 22nd International Conference on Real-Time Networks and Systems (RTNS), October 8-10, 2014, Versaille, France*, pages 35–44.
45. Y. Sun and G. Lipari. A pre-order relation for exact schedulability test of sporadic tasks on multiprocessor global fixed-priority scheduling. *Real-Time Systems*, 52(3):323–355, 2016.
46. Y. Sun, G. Lipari, N. Guan, and W. Yi. Improving the response time analysis of global fixed-priority multiprocessor scheduling. In *Proceedings of the 20th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), August 20-22, 2014, Chongqing, China*, pages 1–9.
47. M. Yang, A. Wieder, and B. B. Brandenburg. Global real-time semaphore protocols: A survey, unified analysis, and comparison. In *Proceedings of the 36th IEEE Real-Time Systems Symposium (RTSS), December 1-4, 2015, San Antonio, Texas, USA*, pages 1–12.
48. H. Zhu, S. Goddard, and M. B. Dwyer. Response time analysis of hierarchical scheduling: The synchronized deferrable servers approach. In *Proceedings of the 32nd IEEE Real-Time Systems Symposium (RTSS), November 29 - December 2, 2011, Vienna, Austria*, pages 239–248.