Evaluations of Push Forward: Global Fixed-Priority Scheduling of Arbitrary-Deadline Sporadic Task Systems (Artifact)

Jian-Jia Chen
Department of Informatics, TU Dortmund University, Germany
jian-jia.chen@tu-dortmund.de
https://orcid.org/0000-0001-8114-9760

Georg von der Brüggen
Department of Informatics, TU Dortmund University, Germany
georg.von-der-brueggen@tu-dortmund.de
https://orcid.org/0000-0002-8137-3612

Niklas Ueter
Department of Informatics, TU Dortmund University, Germany
niklas.ueter@tu-dortmund.de
https://orcid.org/0000-0002-6722-4805

Abstract

This artifact provides the experimental details and implementations of all the facilitated schedulability tests used in the reported acceptance ratio based evaluations as documented in the related paper *Push Forward: Global Fixed-Priority Scheduling of Arbitrary-Deadline Sporadic Task Systems*.

2012 ACM Subject Classification Computer systems organization → Real-time systems

Keywords and phrases global fixed-priority scheduling, schedulability analyses, speedup bounds

Digital Object Identifier 10.4230/DARTS.4.2.6

Related Article Jian-Jia Chen, Georg von der Brüggen, and Niklas Ueter, "Push Forward: Global Fixed-Priority Scheduling of Arbitrary-Deadline Sporadic Task Systems", in Proceedings of the 30th Euromicro Conference on Real-Time Systems (ECRTS 2018), LIPIcs, Vol. 106, pp. 8:1–2:24, 2018.
`http://dx.doi.org/10.4230/LIPIcs.ECRTS.2018.8`

Related Conference 30th Euromicro Conference on Real-Time Systems (ECRTS 2018), July 3–6, 2018, Barcelona, Spain

1 Scope

This provided artifact relates to the implementations and evaluations of the schedulability tests proposed in the paper *Push Forward: Global Fixed-Priority Scheduling of Arbitrary-Deadline Sporadic Task Systems* [3]. That is, we provide all implementations of the schedulability tests and experimental setups that were used to retrieve the reported acceptance ratio results in the paper.

The intent is to prove the validity and reproducibility of the presented and claimed results by providing all relevant information about the facilitated experimental setups as well as the implementations of the compared to algorithms. Moreover, we want to demonstrate the existence of an efficient implementation of the schedulability test shown in Theorem 4.4 of the related paper [3] as was claimed by the authors by providing the source code for a reference implementation as well as a detailed description of the algorithm in the appendix.

2 Content

The content of this artifact is structured into three folders, namely *algorithms*, *utility* and *experiments*.

Algorithms: The algorithms folder contains all implementations of the evaluated schedulability tests, namely:

- *OUR-4.4*: Theorem 4.4 in the related paper.
- *OUR-4.6*: Theorem 4.6 in the related paper.
- *OUR-4.7*: Theorem 4.7 in the related paper.
- *HC*: The sufficient schedulability test presented in Corollary 2 by Huang and Chen in *W.-H. Huang and J.-J. Chen. Response time bounds for sporadic arbitrary-deadline tasks under global fixed-priority scheduling on multiprocessors. In RTNS, 2015.* [5].
- *LOAD*: The load-based analysis for deadline-monotonic scheduling by Baruah and Fisher presented in *S. K. Baruah and N. Fisher. Global fixed-priority scheduling of arbitrary-deadline sporadic task systems. In Distributed Computing and Networking, 9th International Conference, ICDCN, pages 215–226, 2008.* [2].
- *BAK*: Theorem 11 in *T. P. Baker. An analysis of fixed-priority schedulability on a multiprocessor. Real-Time Systems, 32(1-2):49–71, 2006.* [1].

Utility: The utility folder contains scripts to generate sporadic arbitrary-deadline task sets for multiprocessor systems. More precisely, the task generator draws periods uniformly from a $(min, max, False)$ range (to be specified in the argument of the respective function), where the boolean flag specifies whether the period should be truncated to integers or not. Each deadline is constructed as $D_i = \alpha_i T_i$, where α_i is drawn uniformly from the range (min, max) as specified in the arguments. For the generation of uniformly distributed utilizations, the randfixedsum algorithm [4] is used. Further, this folder contains helpful scripts for plotting.

Experiments: Lastly, the experiments folder contains the scripts to generate schedulability test evalutions in terms of acceptance ratios. Each experiment is placed in it's unique folder, where the experimental setup is to be specified in the *experiments.py* file. All associated results are stored in the sub directory *results*.

3 Getting the artifact

The artifact endorsed by the Artifact Evaluation Committee is available free of charge on the Dagstuhl Research Online Publication Server (DROPS).

4 Tested platforms

The provided artifact was tested on a desktop computer using 64-bit Linux Ubuntu 16.04 LTS, Intel Core i5 (4 Cores) and 8GB of RAM. In general, the artifact should be feasible on any system that can provide Python ($\geq$ *Python* 2.7.12) with the following packages:

- Matplotlib 1.5.1
- Tkinter (Python-tk 2.7.12)
- Numpy 1.11.0

5 License

The artifact is available under the MIT License.

6 MD5 sum of the artifact

6ca9de53b717d130c40c289a82ab61d5

7 Size of the artifact

712 KiB

A Polynomial Time implementation

▶ **Theorem 1** (Theorem 4.4 [3]). *Task τ_k is schedulable by the given global fixed-priority scheduling if*

$$\forall \ell \in \mathbb{N}, \exists 1 \geq \rho \geq \ell C_k/((\ell-1)T_k + D_k)$$
$$\frac{\ell C_k}{D'_k} + \sum_{\tau_i \in \mathbf{T}^*} \frac{\gamma_i U_i D_i}{D'_k} + \sum_{i=1}^{k-1} \left(\frac{C_i - C_i U_i}{D'_k} + U_i \right) \leq \mu_k, \tag{1}$$

where $\mu_k = M - (M-1)\rho$ with $1 \geq \rho \geq \ell C_k/((\ell-1)T_k + D_k)$, D'_k is $(\ell-1)T_k + D_k$,

$$\gamma_i = \begin{cases} 1 & \text{if } U_i > \rho \\ 0 & \text{if } U_i \leq \rho \end{cases} \tag{2}$$

and $\mathbf{T}^*$ is the set of the $\lceil \mu_k \rceil - 1$ tasks among the $k-1$ higher-priority tasks with the largest values of $\gamma_i U_i D_i$. Note that $|\mathbf{T}^*|$ can be smaller than $\lceil \mu_k \rceil - 1$ if the number of tasks with $U_i > \rho$ is less than $\lceil \mu_k \rceil - 1$. If $D_k \leq T_k$, we only need to consider $\ell = 1$.

In Theorem 1 we need to find a ρ with $1 \geq \rho \geq \ell C_k/((\ell-1)T_k + D_k)$ for each $\ell \in \mathbb{N}$ such that

$$\frac{\ell C_k}{D'_k} + \sum_{\tau_i \in \mathbf{T}^*} \frac{\gamma_i U_i D_i}{D'_k} + \sum_{i=1}^{k-1} \left(\frac{C_i - C_i U_i}{D'_k} + U_i \right) \leq \mu_k \tag{1}$$

holds, where $\mu_k = M - (M-1)\rho$ and $D'_k = (\ell-1)T_k + D_k$. Furthermore $\gamma_i = 1$ if $U_i > \rho$ and $\gamma_i = 0$ if $U_i \leq \rho$. As C_i, D_i, T_i, and U_i are given for all tasks, whether Eq. (1) holds or not only depends on the values of ρ (and hence μ_k), ℓ, and $\mathbf{T}^*$. Let $\sum_{\tau_i \in \mathbf{T}^*} \frac{\gamma_i U_i D_i}{D'_k}$ be denoted as $G(\mu)$. Note that $\mathbf{T}^*$ depends on ρ and hence μ_k. If we assume μ_k and hence $G(\mu)$ to be a constant, the left hand side of Eq. (1), denoted as $F(\ell, \mu)$, is either an increasing or a non-increasing function with respect to ℓ, i.e., $\ell \to \infty$.

We will use ∞ as a value here for notational brevity, where $F(\infty, \mu)$ is the limit of the function $F(\ell, \mu)$ when $\ell \to \infty$. Knowing this, for a given μ_k we have the following cases:

- $F(\ell,\mu)$ is increasing with respect to ℓ
  - $F(1,\mu) > \mu_k \Rightarrow$ Eq. (1) never holds.
  - $F(\infty,\mu) \leq \mu_k \Rightarrow$ Eq. (1) always holds.
  - $F(1,\mu) \leq \mu_k$ and $F(\infty,\mu) > \mu_k$. This means, we can calculate a value $l_{\mu_k} \in \mathbb{R}$ with $F(l_{\mu_k},\mu) = \mu_k \Rightarrow$ Eq. (1) holds for $1 \leq \ell \leq l_{\mu_k}$ but not for $\ell > l_{\mu_k}$.
- $F(\ell,\mu)$ is not increasing with respect to ℓ
  - $F(1,\mu) \leq \mu_k \Rightarrow$ Eq. (1) always holds.
  - $F(\infty,\mu) > \mu_k \Rightarrow$ Eq. (1) never holds.
  - $F(1,\mu) > \mu_k$ and $F(\infty,\mu) \leq \mu_k$. This means, we can calculate a value $l_{\mu_k} \in \mathbb{R}$ with $F(l_{\mu_k},\mu) = \mu_k \Rightarrow$ Eq. (1) holds for $l_{\mu_k} \leq \ell$ but not for $\ell < l_{\mu_k}$.

As a result, for a given ρ we can calculate the interval where Eq. (1) holds by calculating the values for $\ell = 1$, $\ell = \infty$, and $l_{\mu_k} \in \mathbb{R}$ with $F(l_{\mu_k},\mu) = \mu_k$. Note that this interval must be further reduced due to the condition $1 \geq \rho \geq \ell C_k/((\ell-1)T_k + D_k)$, i.e., some (or all) values of ℓ are not allowed for a given μ_k. However, when each $\ell \in \mathbb{N}$ is covered by at least one interval, the task is schedulable according to Theorem 1. By testing only a finite number of μ_k values, we can implement the schedulability condition in Theorem 1 efficiently.

When we only look at the right hand side of Eq. (1), we would want to reduce ρ as much as possible to get the largest possible value for μ_k, thus making the condition easier. However, increasing μ_k will lead to a larger value of $G(\mu)$, i.e., a bigger left hand side. This happens either due to an additional summand in the summation or due to new tasks available to be summed up, i.e, the reduction of ρ leads to $U_i > \rho$. For the number of summands, due to the ceiling function, we only have to test integer values of μ_k, as they maximize the right hand side of Eq. (1) for a given number of summands. As $\mu_k = M - (M-1)\rho$, the number of integer values for μ_k is bounded by the number of processors, i.e., we only have to test a finite number of ρ values to cover the situation where the number of summands in $G(\mu)$ increases. In addition, only tasks τ_i with $U_i > \rho$ are allowed in $G(\mu)$. As ρ gets smaller, the number of tasks with $U_i > \rho$ increases and vice versa. However, if we test all values with $U_i = \rho$, where τ_i that has higher priority than τ_k, in an increasing order, we only have to test a finite number of additional ρ values, depending on the number of tasks.

Therefore, we only have to test those $O(M+k)$ possible ρ values. As discussed above, each of them forms an interval I_ρ of the integer values of ℓ that can be covered by the specified ρ value. For each interval $I_\rho = [left_\rho, right_\rho)$, Eq. (1) holds when $\ell = left_\rho, left_\rho + 1, \ldots, right_\rho - 1$, where $left_\rho$ is an positive integer and $right_\rho$ is either positive integer or ∞. Note that such an interval I_ρ does not exist if Eq. (1) never holds, and such ρ values are discarded from further considerations. Deriving all these *valid* intervals needs $O(M+k)$ time in the amortized manner, provided that the higher-priority tasks are sorted by their utilization in $O(k \log k)$ and stored in a list in advance. We need to pay some attention if an interval I_ρ does not have a limited upper bound, called an *unbounded interval* here, i.e., Eq. (1) holds for any $\ell \geq left_\rho$. Note that we do need the existence of at least such an unbounded interval to cover sufficiently large ℓ. Among the unbounded intervals, we take the minimum left endpoint, called ℓ_{max}. This step takes $O(M+k)$. The remaining intervals I_ρ that are not unbounded are called *bounded intervals*. Verifying whether $\ell = 1, 2, \ldots, \ell_{max} - 1$ are covered can be done by checking whether the union of these bounded intervals provides the coverage, which is achievable in $O((M+k)\log(M+k))$ with Klee's algorithm [6].

Due to the above discussions, we can efficiently implement the schedulability test in Theorem 1 with a time complexity of $O((M+k)\log(M+k))$.

References

1 Theodore P Baker. An analysis of fixed-priority schedulability on a multiprocessor. *Real-Time Systems*, 32(1-2):49–71, 2006.

2 Sanjoy K. Baruah and Nathan Fisher. Global fixed-priority scheduling of arbitrary-deadline sporadic task systems. In *Distributed Computing and Networking, 9th International Conference, ICDCN*, pages 215–226, 2008.

3 Jian-Jia Chen, Georg von der Brüggen, and Niklas Ueter. Push forward: Global fixed-priority scheduling of arbitrary-deadline sporadic task systems. In *Euromicro Conference on Real-Time Systems, ECRTS*, pages 8:1–8:24, 2018.

4 Paul Emberson, Roger Stafford, and Robert I Davis. Techniques for the synthesis of multiprocessor tasksets. In *International Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (WATERS 2010)*, pages 6–11, 2010.

5 Wen-Hung Huang and Jian-Jia Chen. Response time bounds for sporadic arbitrary-deadline tasks under global fixed-priority scheduling on multiprocessors. In *International Conference on Real Time Networks and Systems, RTNS*, pages 215–224, 2015.

6 Victor Klee. Can the measure of $\cup_{i=1}^{n}[a_i, b_i]$ be computed in less than $O(nlogn)$ steps? *The American Mathematical Monthly*, 84(4):284–285, 1977.