Revisiting Slot-Shifting's Offline Acceptance Test for Sporadic Tasks: A Technical Note

Mohammad Ibrahim Alkoudsi
RPTU Kaiserslautern-Landau, Kaiserslautern, Germany

Damir Isovic
Mälardalen University, Västerås, Sweden

Gerhard Fohler
RPTU Kaiserslautern-Landau, Kaiserslautern, Germany

Abstract

The *Slot-Shifting* algorithm presents a solution to combine the benefits of offline and online scheduling in time-triggered systems. It dynamically adjusts the allocation of time slots to tasks in the scheduling tables to accommodate aperiodic tasks at runtime. In this note, we revisit an extension to Slot-Shifting that enables it to handle sporadic task sets. In particular, we clarify the assumptions required for the correct application of its offline acceptance test, identify sources of pessimism within it, and address its schedulability analysis interval.

2012 ACM Subject Classification Computer systems organization → Real-time systems; Software and its engineering → Real-time schedulability

Keywords and phrases real-time systems scheduling, time-triggered systems, offline acceptance test of sporadic tasks

Digital Object Identifier 10.4230/LITES.10.1.4

Funding The work presented in this paper was supported in part by the Fonds National de la Recherche Luxembourg (FNR) and the Deutsche Forschungs Gemeinschaft (DFG) through the Core-Inter project ReSAC.

Acknowledgements The authors gratefully acknowledge Anaïs Finzi, Silviu S. Craciunas, and Marc Boyer, whose evaluation presented in [3] inspired our review of the original work, and thank them for comments and valuable discussions. We also would like to thank the reviewers for their constructive comments that contributed to the quality of the text.

Received 2025-05-16 **Accepted** 2025-10-27 **Published** 2025-12-09

1 Introduction

The *time-triggered (TT)* paradigm of activation has gained acceptance in many safety critical systems by ensuring reliability and predictability. It is particularly well-suited for applications with complex timing constraints, e.g., precedence, mutual exclusion, and distributed end-to-end deadlines. By triggering activities at predetermined points in time and letting them execute exclusively in dedicated time intervals (*slots*), extra-functional properties w.r.t. timeliness and dependability, e.g., determinism, reliability, and temporal partitioning, can be efficiently guaranteed.

The strict adherence of native TT implementations to offline defined scheduling tables prevents runtime flexibility. Slot-Shifting [4] is a method to handle independent *event-triggered (ET)* tasks, not already included in the scheduling tables at runtime, while maintaining the feasibility of applications and TT's extra functional properties. It does so by dynamically adjusting the allocation of time slots to tasks in the scheduling tables.

In [5], Isovic and Fohler extended Slot-Shifting from guaranteeing individual aperiodic tasks to sporadic task sets, i.e., tasks with unknown arrival times, but known minimum separation between consecutive instances. The extension includes an offline acceptance test for sporadic tasks, along with modified runtime scheduling rules and an updated online acceptance test for aperiodic tasks to account for the presence of sporadic tasks. In this note, we revisit the offline acceptance test of the extension, clarify the assumptions required for its correct application, identify sources of pessimism within it, and address its schedulability analysis interval.

2 Guaranteeing ET Tasks with Slot-Shifting

The TT scheduling approach can be split into two phases:

(i) offline: before runtime, a scheduling algorithm takes the complex timing constraints of tasks and their messages in the distributed system as input and generates scheduling tables for every node and communication controller in the system. Each scheduling table contains a sequence of time slots, each allocated exclusively to a given task or message on the respective resource. Notably, time slots align with task arrivals and deadlines, as all task parameters are configured to be integer multiples of the slot size.

(ii) online: the operating system at every resource (node or communication controller) invokes the local dispatcher at the beginning of every slot to execute the task or transmit the message allocated to it in the resource scheduling table.

2.1 Individual Aperiodic Tasks

The offline phase of Slot-Shifting extends that of the native TT approach by supplanting the fixed slot allocation with less rigid target execution windows, defined for each TT task on each node. It divides each scheduling table into disjoint *capacity intervals* created from the deadlines of the target execution windows of TT tasks. Each of these intervals is annotated with the as soon as possible (ASAP) available spare processing capacity, i.e. the number of free slots based on as late as possible (ALAP) execution of offline tasks. The spare capacity is derived from the length of the interval and the demand required by TT tasks within the interval for their safe execution. Since the offline scheduling algorithm successfully generated a schedule for the entire hyperperiod of TT tasks, the calculated ALAP spare capacity at the beginning of every hyperperiod is the same.

During runtime, the local dispatcher on each node uses and maintains these spare capacities for the safe handling and scheduling of hard and soft aperiodic tasks arriving into the system, in a slot by slot basis and based on the earliest deadline first (EDF) policy. An online acceptance test analyzes if aperiodic tasks can be safely included in the system while maintaining TT's extra functional properties. It verifies if the demand of an aperiodic task can be retrofitted in the free slots of capacity intervals that are located within its execution window.

2.2 Sporadic Tasks

In [5], Isovic and Fohler extended Slot-Shifting from guaranteeing individual aperiodic tasks to sporadic task sets. The extension includes an offline acceptance test for analyzing the feasibility of extending the TT task set with sporadic tasks, along with the necessary adaptations for runtime scheduling. This involves extending Slot-Shifting's runtime rules for the dynamic allocation of slots to tasks and modifying the admission test for aperiodic tasks to account for the presence of both TT and offline-guaranteed sporadic tasks.

The offline acceptance test considers the worst-case arrival pattern of sporadic tasks for schedulability analysis. In this scenario, all sporadic tasks arrive synchronously at the same slot, with subsequent instances released at their maximum frequency. Consequently, the sporadic task set can be modeled as a synchronous periodic task set during the test, where all tasks share a global offset and each task's period is equal to its minimum inter-arrival time, as proposed in [2].

The offline test analyzes whether the worst-case demand of sporadic tasks can be accommodated within the available capacity intervals, under the assumption that sporadic tasks may arrive synchronously at every slot in the schedule. In other words, each slot is treated as a potential global offset value for sporadic tasks. Isovic and Fohler proposed an optimization to reduce the runtime of the offline acceptance test by limiting the number of global offset candidates for sporadic tasks. They prove that selecting a single representative slot per capacity interval (termed the *critical slot* in [5]) is sufficient to analyze schedulability.

3 Revisiting Slot-Shifting's Offline Acceptance Test for Sporadic Tasks

3.1 Clarification of Assumptions

The offline acceptance test assumes that TT tasks experience maximum delay and are forced to complete their execution ALAP. Consequently, the identification of critical slots within each capacity interval for the simultaneous arrival of sporadic tasks, and in turn the correctness of the acceptance test, is only established under this assumption[1]. Every slot in the schedule represents a global offset candidate for sporadic tasks. The critical slots defined in [5] serve to reduce the number of candidate slots to a single slot per capacity interval. However, in [5], there is no proof of the applicability of critical slots to ASAP scheduling. Identifying critical slots for the ASAP scheduling case is beyond the scope of this note.

3.2 Identification of the Pessimism

We now turn to identifying the pessimism in the offline acceptance test, which analyzes the feasibility of adding the worst-case sporadic demand alongside the ALAP-serviced TT demand based on the offline computed capacity intervals, and assuming that sporadic tasks arrive synchronously at the critical slot of each capacity interval in the TT schedule.

The runtime mechanisms defined in the Slot-Shifting extension prioritize ready ET tasks whenever spare capacity is available in the current interval. Consequently, ET tasks can impose forced delays on the execution of TT tasks, even when they do not have the earliest deadlines. In the worst-case scenario, TT tasks execute ALAP, while still meeting their deadlines. However, when no ET tasks are ready, TT tasks execute ASAP on a slot-by-slot basis and according to EDF policy.

Since the considered acceptance test is performed offline and aperiodic tasks are only admitted online, analyzing the schedulability of the combined TT and sporadic task set offline does not need to take aperiodic tasks into account. This implies that, online, TT tasks execute ASAP prior to the synchronous arrival of sporadic tasks at a critical slot. Hence, the actual spare capacity values at a given slot deviate from those originally computed under the assumption of ALAP execution. Since we can precisely determine which TT tasks execute prior to the critical slot and for how many slots (based on their release times, worst-case execution times (WCET), and EDF-based priorities), we can precisely calculate the change in capacity values. Each slot used by a TT task reduces its remaining demand by one. This does not consume a free slot but may shift it to a later point in time, meaning that there can be more available spare capacity online compared to the offline calculated values based on ALAP execution in the test. Consequently, assuming less available spare capacity than actually present based on ALAP execution of TT

[1] The reasoning in the proof of Theorem 1 in [5] is grounded in ALAP reservation of slots for TT tasks. This is evidenced by the absence of any change in spare capacity within the considered arrival interval, expressed as $\alpha = 0$, when the arrival time of sporadic tasks is shifted forward.

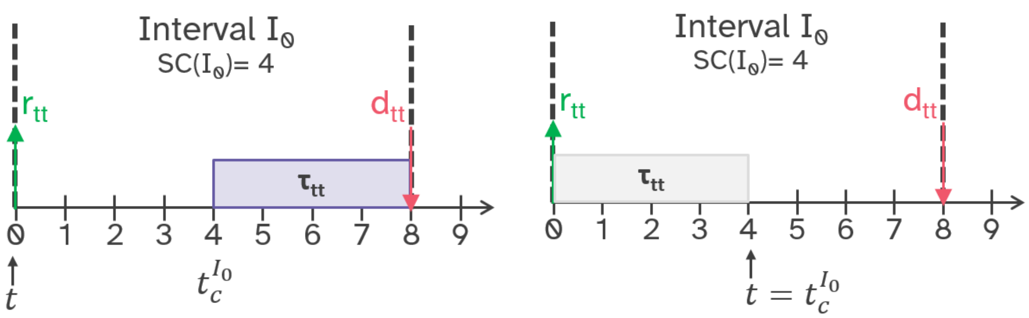
tasks introduces pessimism into the test. In particular, the test can produce false negatives by incorrectly classifying some schedulable task sets as unschedulable. However, this assumption does not lead to false positives, i.e., any task set deemed schedulable by the test is indeed schedulable. In Table 2, we list simple modifications to make the test exact.

Detailed Description and Example. In [5], Isovic and Fohler define, for each capacity interval I_i, a critical time slot $t_c^{I_i}$. They prove that if the sporadic task set is feasible when its tasks arrive synchronously at $t_c^{I_i}$, then it is also feasible when its tasks arrive synchronously at every other time slot t_k located within the same interval I_i. Isovic and Fohler denote the start of capacity interval I_i by $start(I_i)$ and the spare capacity within it by $sc(I_i)$. They calculate the critical slot of interval I_i as follows: $t_c^{I_i} = start(I_i) + sc(I_i)$. In other words, since spare capacities in Slot-Shifting are calculated based on an ALAP execution of TT tasks in order to service aperiodic tasks ASAP, a critical slot of a capacity interval is the first time slot, where the spare capacity of the interval is completely used or wasted.

Let us consider the simple example task set system in Table 1, showing the parameters of a single TT task τ_{tt} after performing Slot-Shifting's offline phase (which resolves its complex constraints and transforms it into an independent task) and a single sporadic task τ_{sp} with which we seek to extend the task set. Figure 1a shows the schedule for τ_{tt} with ALAP execution, annotated with its single spare capacity interval I_0 and its spare capacity $sc(I_0) = 4$. The critical slot for this interval is $t_c^{I_0} = 4$.

Table 1 Example task set, showing the parameters of a single TT task τ_{tt} after performing Slot-Shifting's offline phase and a single sporadic task τ_{sp}.

Task	Type	Offset ϕ	WCET $\mathcal{C}$	Deadline D	Period/MIT T
τ_{tt}	Periodic	0	4	8	8
τ_{sp}	Sporadic	-	1	4	4



(a) ALAP Schedule following execution until $t = 0$. **(b)** ALAP Schedule following execution until $t = 4$.

Figure 1 Schedule for task τ_{tt} from Table 1, illustrating ALAP execution relative to time slot t, under the assumption that tasks are executed ASAP prior to t. For both cases, the capacity interval I_0 is indicated and annotated with its spare capacity $sc(I_0)$.

Each sporadic task instance can be guaranteed depending on the availability of spare capacities within its execution window. If we let the sporadic task τ_{sp} arrive at the first critical slot, i.e., at $t = 4$, and attempt to guarantee it within the ALAP schedule in Figure 1a, we find that the first instance of τ_{sp} has no available free slots in its execution window interval $[4, 8]$. Therefore, it cannot be guaranteed according to the test in [5].

However, since Slot-Shifting's runtime scheduler executes tasks ASAP at each time slot, the corresponding ALAP schedule at $t_c^{I_0} = 4$ is shown in Figure 1b. By then, the task τ_{tt} has already finished executing, and all free slots have been relocated to the interval $[4, 8]$. Hence, the first instance of τ_{sp} can be guaranteed in the interval $[4, 8]$; there are 4 free-slots, and it needs only one. In fact, the task set in Table 1 follows the implicit deadline model, where the necessary and sufficient condition for EDF-schedulability is that the total utilization of the task set must be less than or equal to one [2]. Since this condition is satisfied, the task set is schedulable.

Therefore, the test in [5] is pessimistic. The ASAP execution of TT tasks, shifts and does not consume free time slots within capacity intervals. Removing this pessimism requires recalculating the spare capacity values of all intervals based on the ASAP execution of TT tasks prior to the candidate slot for the global arrival of sporadic tasks. A possible approach is by simulating Slot-Shifting's online scheduling behavior on TT tasks up to the critical slot, thereby ensuring that spare capacities are updated accordingly across the entire schedule. In Table 2, we list simple modifications to make the test exact.

3.3 Addressing the Analysis Interval

In [5], Isovic and Fohler perform the test relative to each critical slot for the duration of the hyperperiod of sporadic tasks $\mathcal{H}_S$, i.e., the least common multiple (LCM) of their periods. However, this is the proven analysis interval without the co-presence of TT periodic tasks [2]. Isovic and Fohler do not provide a formal proof for their proposed analysis interval and do not specify the deadline model assumed for sporadic tasks. Here, we propose adopting formally proven bounds within scheduling theory and have been shown to hold under ASAP scheduling: TT tasks are inherently periodic and may have offsets and complex timing constraints, such as end-to-end deadlines and mutual exclusion. As explained in Section 2, during the offline phase of Slot-Shifting, an offline scheduler resolves these constraints, effectively converting TT tasks into independent ones (see [4] for details). This transformation allows reusing established and safe schedulability bounds for periodic tasks with offsets [1, 2, 6] to modify the analysis interval of the offline acceptance test.

Let ϕ_{max} denote the largest offset among all periodic tasks, $\mathcal{H}_{TT}$ be their hyperperiod, Γ be the combined set of time-triggered and sporadic tasks, and $\mathcal{C}_i$, T_i, and D_i be the worst-case execution time, period, and relative deadline of task τ_i, respectively. It was shown in [1] that it is sufficient to analyze schedulability over all possible release times t_1 and deadlines t_2, subject to the inequalities:

$$\phi_{max} \le t_1 < \phi_{max} + \mathcal{H}_{TT} \tag{1}$$

$$t_1 < t_2 < t_1 + B_{TTS} \tag{2}$$

$$B_{TTS} = \frac{\sum_{\tau_i \in \Gamma} (T_i - D_i) \frac{\mathcal{C}_i}{T_i}}{1 - \sum_{\tau_i \in \Gamma} \frac{\mathcal{C}_i}{T_i}} \tag{3}$$

Since the bounds on t_1 and t_2 are established under the assumption of constrained deadlines, we adopt the same deadline model and adapt the terms to align with those used in Slot-Shifting's offline test as follows:

1. Set t_1 to the critical slot t_c, marking the beginning of an analysis iteration.
2. Omit t_2 as an explicit variable and have the test evaluate all capacity intervals starting from t_c up to $t_1 + B_{TTS}$

With this modification, the acceptance test is expected to run longer, which is acceptable for offline use.

3.4 Properties Summery

We summarize the properties of the offline acceptance test, including its modified analysis interval, in Table 2. In addition, we outline minor adjustments that align the test with the exact formulation presented in [1], specifically by incorporating the ASAP scheduling of TT tasks prior to the synchronous release of sporadic tasks at the candidate global offset slot. Experimental results in [1] show that the exact test achieves comparable runtime performance to the original sufficient test, while offering substantial improvements in schedulability.

Table 2 Properties of the offline acceptance test and simple modifications to the test to account for the ASAP scheduling of TT tasks.

	Acceptance test in [5]	**Modifications**
Candidate slots as global offset of sporadic tasks	Critical slot at every interval I, calculated as $start(I) + sc(I)$	Every slot that coincides with the release of a TT task in the schedule
During the test, available spare capacity is calculated	Once, based on ALAP execution of TT tasks	Once for every candidate slot, based on ASAP execution of TT tasks prior to the candidate slot
Analysis interval	Subject to the inequalities 1, 2 and 3	
Test type	Sufficient	Exact
Runtime performance	Comparable (see [1])	

4 Conclusion

In this technical note, we identified pessimism in the offline test of Slot-Shifting for extending a feasible time-triggered task sets with a set of sporadic tasks, clarified assumptions for its use, and addressed the interval considered for schedulability analysis.

References

1 Mohammad Ibrahim Alkoudsi, Sanjoy Baruah, Pontus Ekberg, Gerhard Fohler, Joël Goossens, and Maryam Moslehi. Extending periodic task sets with sporadic tasks: Computational complexity and exact feasibility tests. In *Proceedings of the 33rd International Conference on Real-Time Networks and Systems (RTNS '25)*, Pisa, Italy, 2025.

2 Sanjoy K. Baruah, Louis E. Rosier, and Rodney R. Howell. Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor. *Real-Time Systems*, 2, 1990. doi:10.1007/BF01995675.

3 Anaïs Finzi, Silviu S. Craciunas, and Marc Boyer. Integrating sporadic events in time-triggered systems via affine envelope approximations. In *2024 IEEE 30th Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 15–28, 2024. doi:10.1109/RTAS61025.2024.00010.

4 G. Fohler. Joint scheduling of distributed complex periodic and hard aperiodic tasks in statically scheduled systems. In *Proceedings 16th IEEE Real-Time Systems Symposium*, pages 152–161, 1995. doi:10.1109/REAL.1995.495205.

5 Damir Isovic and Gerhard Fohler. Handling mixed sets of tasks in combined offline and online scheduled real-time systems. *Real-Time Syst.*, 43(3):296–325, November 2009. doi:10.1007/s11241-009-9088-3.

6 Joseph Y.-T. Leung and M.L. Merrill. A note on preemptive scheduling of periodic, real-time tasks. *Information Processing Letters*, 11(3):115–118, 1980. doi:10.1016/0020-0190(80)90123-4.