

A 10-Year Taxonomy on Prior Work on Sampling Period Selection for Resource-Constrained Real-Time Control Systems

Camilo Lozoya, Manel Velasco, Pau Martí
Automatic Control Department
Technical University of Catalonia
{camilo.lozoya,manel.velasco,pau.marti}@upc.edu

Abstract

In this paper we present a non-complete taxonomy on prior work on sampling period selection for resource-constrained real-time control systems. Selection of sampling periods for real-time control tasks determines resource utilization (or alternatively, task set schedulability) as well as overall control performance. At the end, it determines the sequence of control tasks instances' executions over time, that is, the schedule. Different schedules are obtained depending on which criterion is used to select sampling periods, what real-time paradigm is demanded in the underlying executing platform, who should decide which task to execute, when the decision is taken, and how the decision is enforced. To analyze all these aspects, ten papers from the last decade, one per year from 1998 to 2007, have been selected and categorized. The taxonomy, although being incomplete, reveals key tendencies and rises important research challenges.

1. Introduction

Traditionally, computer controlled systems are implemented using real-time periodic tasks. Each periodic task is statically assigned a period obtained following well-established procedures that mandate to sample and control periodically. However, the embedded systems market requires systems with more and better functionalities at lower prices. A consequence is that control applications must be implemented in platforms where resources are scarce and/or where increasing performance is a must. And the traditional static periodic approach to control systems implementation fails at minimizing resource utilization and maximizing control performance.

To provide solutions fulfilling the tight demands posed by modern embedded systems, the control and real-time communities have shown in recent years a renewed inter-

est on deriving novel sampling period selection methods for efficient implementation of real-time control systems. In this paper we present a non-complete taxonomy of some of these methods. Although most of them have focused on CPU, many can also be adapted to networks or battery limited platforms.

Many of the novel methods for sampling period selection go beyond than just finding the *best* values for control task periods. They provide complete real-time frameworks tailored to effective concurrent execution of control tasks. They are characterized by *which* criterion is used to select sampling periods, thus establishing *what* real-time paradigm is demanded in the underlying executing platform, *who* should decide which task to execute, *when* the decision is taken, and *how* the decision is enforced.

To analyze these frameworks and their key properties in terms of *Which*, *What*, *Who*, *When* and *How*, ten papers [1]-[10] from the last decade (1998-2007) have been arbitrarily selected and categorized. Papers are listed in chronological order in the references section to provide an initial view of the contributions over time. The selection of one paper per year, although being incomplete in terms of time coverage and number of papers, collects a wide variety of approaches while revealing the key existing tendencies on control task scheduling, and rising important research challenges.

2. Taxonomy

A key aspect of these new methods is the theoretical criterion used to obtain the set of sampling periods. From the criterion, key aspects are derived and analyzed in order to construct the taxonomy summarized in table 1.

2.1. Criterion

Two main criterion can be identified: optimization approach or bounding the intersampling dynamics.

Table 1. Taxonomy of sampling period selection approaches.

	Which	What	Who	When	How		
	Criterion	Triggering Paradigm	Triggering Entity	Solving the problem	Solution	Timing Constraints	Sched.
[1] Set98	Optimizat.	TT	Coord.	Off-line	Periods	Static periodic	EDF/FP
[2] Arz99	Bound dyn.	ET	Task	On-line	Periods	Aperiodic ET	Missing
[3] Reh00	Optimizat.	TT	Coord.	Off-line	Sequences	Static pseudo periodic	Cyc. Ex.
[4] Hri01	Bound dyn.	TT	Coord.	Off-line	Sequences	Static pseudo periodic	Cyc. Ex.
[5] Pal02	Optimizat.	TT	Coord.	Off-line	Periods	Static periodic	EDF
[6] Cha03	Optimizat.	TT	Coord.	Off-line	Periods	Static periodic	EDF/FP
[7] Mar04	Optimizat.	TT	Coord.	On-line	Periods	Varying periodic	EDF
[8] Hen05	Optimizat.	TT	Coord.	On-line	Periods	Varying periodic	EDF
[9] Tab06	Bound dyn.	ET	Task	On-line	Periods	Aperiodic TT	Missing
[10] Lem07	Bound dyn.	ET	Task	On-line	Periods	Aperiodic TT	Elastic sch.

In the optimization approaches ([1], [3], [5], [6], [7], [8]) sampling periods are selected to solve an optimization problem. They assume that there is a cost function parameterized in terms of control performance and sampling periods that has to be minimized or maximized depending on whether it denotes penalty or benefit. The optimization problem domain is restricted by closed loop stability and task set schedulability constraints.

In the approaches based on bounding the intersampling dynamics ([2], [4], [9], [10]), sampling periods are selected to keep each closed loop dynamics within predefined thresholds. Thresholds, which are derived from pure control theoretical approaches, are used to bound changes in the dynamics or to ensure closed loop stability.

It is important to identify whether the theoretical criteria capture the dual problem posed by modern embedded systems: minimizing resource utilization and maximizing control performance. In all the optimization approaches the duality is captured by the cost function and optimization constraints. However, the bounding approaches usually only capture control performance issues. Therefore, utilization, and at the end schedulability, is not addressed.

2.2. Triggering paradigm and entity

These two criteria influence whether the period selection solution requires a real-time architecture following a time-triggered (TT) or event-triggered (ET) paradigm. All the solutions to the optimization approaches require a time-triggered architecture while almost all the solutions based on bounding closed-loop dynamics require an event-triggered architecture, except for [4].

The classification considers who is in charge of selecting sampling periods (triggering entity). All solutions requiring a TT architecture are based on a global coordinator that decides the *best* periods for the set of control tasks. On the

contrary, in the solutions requiring an ET architecture control tasks are in charge of deciding their periods.

2.3. When to solve the problem

The previous classification (TT vs. ET) relates to whether the period selection is performed off-line or on-line. In all ET approaches ([2], [9], [10]) periods are derived on-line. However, in the TT approaches, some solutions have to be computed off-line ([1], [3], [4], [5], [6]) while the rest are on-line ([7], [8]).

It is important to identify when the sampling periods are selected for two main reasons. First, computational overhead must be considered, which may be a disadvantage for on-line approaches. Second, ability to adapt to workload changes, this means that varying available resources or varying demands from the control applications has to be also accounted for, which may be an advantage for on-line approaches.

2.4. Solution and its enforcement using real-time technology

Once periods are selected, they must be enforced by the underlying real-time architecture. Therefore, it is important to examine how the solutions are enforced. This can be analyzed in a three step procedure: first, looking at the solution in more detail; second, looking at the type of demanded timing constraints; and third, looking at scheduling policies capable of enforcing such timing constraints.

Solution. Although the taxonomy reviews methods for sampling period selection, two of the methods ([3], [4]) do not establish sampling periods. Rather they provide periodic sequences of ordered control task instances. All the others provide periods, time intervals or timing bounds

that establishes when tasks have to be executed, i.e., they provide diverse control tasks timing constraints.

Timing constraints. All solutions following a TT paradigm impose periodic timing constraints for control tasks. In particular, [1], [5], and [6] specify static periodic timing constraints for control tasks. That is, the outcome of solving off-line their particular methods is a set of periods for control tasks, which will not change at run-time, named *static periodic*. Similarly, [3] and [4] specify static periodic sequences of ordered tasks instances that will not change at run-time. Looking at a single task, this would be a *static pseudo-periodic* execution. Finally, the on-line solution to the frameworks presented by [7] and [8] provide periods for control tasks that will change at run-time, named *varying periodic*.

The timeliness of the solutions demanding an event-triggered architecture is in the general case aperiodic. However, different type of aperiodicities can be distinguished. In [2], the execution is purely *aperiodic* and it is triggered whenever an external event detected, using specific hardware (event detector), is identified. For control safety reasons, an upper bound is imposed in order to force an execution if no events are detected. The triggering condition can not be predicted. In [9], although tasks execute aperiodically, a lower bound on the inter-arrival of job executions is predicted at each job execution, thus indicating an sporadic task behavior. Finally, in [10], the execution is again aperiodic, but with the advantage that at each job execution the next job deadline is predicted. Both approaches [9] and [10] can be considered as *aperiodic time-triggered*.

Scheduling. All solutions demanding a time-triggered architecture can enforce the derived timing constraints for control tasks using well known scheduling strategies. In particular, [3] and [4] can exploit cyclic executives, while the rest can exploit scheduling policies for periodic tasks, such as earliest deadline first (EDF) and fixed priority (FP). Table 1 specifies for each of these solutions what scheduling policy can be applied. For the on-line approach [7] a specific resource allocator that computes on-line the sampling periods is required before the EDF dispatching. In the other on-line approach [8], EDF can be applied directly because the computation of the periods is performed periodically for a particular task named feedback scheduler task.

For the solutions demanding an event-based architecture, the scheduling policy, that can enforce the presented solution, is lacking in the general case. Only the result provided in [10] integrates the presented even-triggered control with existing scheduling theory. At each job execution the deadline for the following job is predicted and the elastic scheduling is invoked to accommodate the new timing

demands, considering the whole task set. However, if the elastic scheduling can not met them, problems may occur.

2.5. Discussion

The presented taxonomy mainly considers key real-time aspects of the reviewed methods. However, some other aspects have been omitted: Which task model is used in terms of avoiding/reducing sampling and latency jitters? In the optimization problem, are the solutions general or depend on each controlled system? Are they exact (closed forms) or approximates? Solving them also means to obtain the appropriated controller gains?. Looking more at control aspects, questions not analyzed are: Which type of controllers support the presented solutions? Are observers considered? Is noise also considered?

Overall, many questions have not been analyzed. However, the taxonomy is not closed, and all the previous questions can be incorporated. In addition, many existing papers on sampling period selection (and related issues) for real-time control systems have not been cited nor analyzed. They could be also included in the taxonomy. However, the main tendencies that the taxonomy reveals and that we analyze next would remain the same (or very similar).

3. Tendencies

To identify tendencies, we focus the attention to columns *When* and *What* of Table 1, reading them chronologically, from top to bottom.

The *When* column, that refers to whether the sampling period selection is performed off-line or on-line clearly shows a tendency to on-line approaches. This tendency reflects and aims at meeting the demands of modern embedded systems that are required to work in dynamic environments, being adaptive to the available resources that can change abruptly, or to the resource demands of control applications that can be considered as varying depending on the state of the controlled plants.

The *What* column refers to whether the presented solution requires a time or event triggered real-time architecture. The first conclusion that can be extracted is that TT solutions are more common than ET solutions. But more revealing, it shows a tendency toward event-triggered approaches. A reason for such trend could be to force up to the limit the logics of the periodic on-line approaches ([7] or [8]). In on-line approaches, sampling periods are hold until the sampling period selection procedure takes place, and new periods are derived. This logic has been shown to be effective at minimizing resource utilization and/or at maximizing control performance. Forcing this logic to the limit means executing the sampling period selection procedure each time a control task instance executes, which may

provide even better resource utilization and better control performance.

4. Research challenges

Key tendencies indicate a current interest on on-line event-triggered approaches. However, there is a recognized lack of scheduling support for event-based control theoretical results [2], problem that can be also identified by looking at the last column of table 1: scheduling solutions for [2] and [9] are missing, and the scheduling solution adopted by [10] may not be able to fulfill the demanded timing.

A key property of solutions demanding a TT architecture is that they include resource constraints in the theoretical criterion. Therefore, task set schedulability is already considered for example in terms of utilization tests, as mentioned in Section 2.1. However, the criteria used to derive sampling periods for the ET approaches do not consider resource constraints in the formulation. Therefore, the existing solutions do not implicitly solve the scheduling.

To overcome this limitation, several solutions can be envisioned. In the last two reviewed papers, [9] and [10], tasks decide their rate of progress in an event-triggered fashion. Specifically, at each control task instance execution, the same information used to compute the next value of the control signal is also used to decide when the control task has to be applied again. The latest decision could also consider resource utilization. As mentioned earlier, resource utilization is usually expressed in terms of the utilization factor. However, this measure is atemporal and pessimistic. A more appealing metric for considering when the next instance has to be executed may be the synthetic utilization factor, which depends on time and was developed for aperiodic scheduling. Future work will explore this approach.

A more radical solution envisions the dispatching of aperiodic jobs without characterizing them with timing constraints (periods and deadlines). The solutions presented in [9] and [10] translate at each job execution, control demands into next job timing constraints. However, an alternative vision could be to avoid translating control demands into timing constraints. The approach would require to have all control tasks always ready to execute, and to have a global coordinator that picks each time the most appropriated task for execution. In this case, concepts like task set schedulability will not hold, and equivalent concepts should be derived. For example, schedulability could mean stability in the sense of determining how many control loops can be kept stable. Future work will also explore this approach.

5. Conclusions

A 10-Year taxonomy on prior work on sampling period selection for resource-constrained real-time control systems

has been presented. The taxonomy shows that current trends point to on-line event-triggered approaches, that is, to select periods at run-time generating aperiodic task instances executions. In addition, research challenges for effective building these type of solutions have been presented and discussed.

References

- [1] D. Seto, J.P. Lehoczky, L. Sha, "Task Period Selection and Schedulability in Real-Time Systems", IEEE Real-Time Systems Symposium, 1998
- [2] K.-E. Årzén, "A Simple Event-Based PID Controller", 14th World Congress of IFAC, January, 1999.
- [3] H. Rehbinder, and M. Sanfridson, "Integration of off-line scheduling and optimal control", 12th Euromicro Conference on Real-Time Systems, 2000.
- [4] D. Hristu-Varsakelis, "Feedback control systems as users of a shared network: communication sequences that guarantee stability", 40th IEEE Conference on Decision and Control, 2001
- [5] L. Palopoli, C. Pinello, A. L. Sangiovanni-Vincentelli, L. Elghaoui and A. Bicchi, "Synthesis of Robust Control Systems under Resource Constraints", Hybrid Systems: Computation and Control, 2002.
- [6] R. Chandra, X. Liu, and L. Sha, "On the Scheduling of Flexible and Reliable Real-Time Control Systems", Real-Time Systems 24(2), March 2003.
- [7] P. Martí, C. Lin, S. Brandt, M. Velasco and J.M. Fustes, "Optimal State Feedback Based Resource Allocation for Resource-Constrained Control Tasks", 25th IEEE Real-Time Systems Symposium, Lisbon, Portugal, December 2004.
- [8] D. Henriksson, and A. Cervin, "Optimal On-line Sampling Period Assignment for Real-Time Control Tasks Based on Plant State Information", 44th IEEE Conference on Decision and Control and European Control Conference ECC 2005, December 2005.
- [9] P. Tabuada and X. Wang, "Preliminary results on state-triggered scheduling of stabilizing control tasks", 45th IEEE Conference on Decision and Control, December 2006.
- [10] M. Lemmon, T. Chantem, X. Hu, and M. Zyskowski, "On Self-Triggered Full Information H-infinity Controllers", Hybrid Systems: Computation and Control, April 2007