Statistical-based Response-Time Analysis of Systems with Execution Dependencies between Tasks

Yue Lu, Thomas Nolte, Johan Kraft and Christer Norström
Mälardalen Real-Time Research Centre
Mälardalen University, Västerås, Sweden
{yue.lu, thomas.nolte, johan.kraft, christer.norstrom}@mdh.se

Abstract

This paper presents a novel statistical-based approach to Worst-Case Response-Time (WCRT) analysis of complex system models. These system models have been tailored to capture intricate execution dependencies between tasks, inspired by real industrial control systems. The proposed WCRT estimation algorithm is based on Extreme Value Theory (EVT) and produces both WCRT estimates together with a predictable probability of being exceeded (i.e., 10^{-9}*). By using the tools developed, we validate the proposed method by evaluating a model taken from the real industrial control system, and we show the results in comparison with other four analysis methods.*

1 Introduction

To date, most existing embedded real-time software systems have been developed in a traditional code-oriented manner, i.e., making extensive use of legacy software. Many such systems are maintained over extended periods of time, sometimes spanning decades, during which the systems become larger and increasingly complex. The result is that these systems are difficult and expensive to maintain and verify. There are many embedded systems existing in industry which consist of millions of lines of C code, corresponding to 50, or 100 tasks or more, where many tasks have real-time constraints. The example of such systems is the robotic control systems developed by ABB [1]. Looking closer at these systems, contrary to the assumption in most real-time theory, tasks exhibit strong temporal dependencies, e.g., asynchronous message-passing, globally shared state variables and runtime changeability of periods and priorities of tasks, which vary the execution time of the tasks radically.

One desirable approach to avoid the timing-related errors in such complex systems is to use schedulability analysis methods, such as Response-Time Analysis (RTA) [2]. Nevertheless, RTA (and other schedulability analysis techniques), although providing the prediction about timing behavior of execution in worst-case scenarios, rely on the existence of a fixed Worst-Case Execution-Time (WCET) of the tasks. Correspondingly, the quality of the analysis is directly correlated to the quality of the WCET estimates. Unfortunately, in the above described systems, the WCET of tasks obtained by static WCET analysis techniques may not easily be bounded. Sometimes a pessimistic WCET bound can be calculated based on maximum queue lengths. While in other cases the WCET is completely unbounded until the behavior of dependent tasks is known. Consider the following example in Figure 1, taken from an industrial robotic control system, where a task reads all messages buffered in a message queue and processes them accordingly:

```
msg = recvMessage(MyMessageQueue);
  while (msg != NO_MESSAGE){
  process_msg(msg);
  msg = recvMessage(MyMessageQueue);
}
```

Figure 1. Iteration-loop wrt. message passing

By using static WCET analysis, the upper bound of number of messages actually consumed is equal to the maximum queue size. Furthermore, other tasks with a higher priority may preempt the execution of the loop and refill the queue at runtime. Looking further at the corresponding task periodicity dependencies, the analysis performed at RTA level also contributes to the pessimism as the number of loop iterations is not supposed to be bounded by the maximum queue size when preemption occurs.

The other approach, which avoids the state-space explosion issue raised by model checkers such as UPPAAL [3] and TIMES [4], for instance, is to use simulation-based methods that sample the state space. The first type of simulation technology to use is Monte Carlo simulation, which can be described as keeping the highest result from a set of randomized simulations. Several frameworks already exist in this realm, such as the commercial tool *VirtualTime* [5] and the academic tool *ARTISST* [6]. However, the main drawback of using Monte Carlo simulation is the low state-space test coverage, which subsequently decreases the confidence in the results of finding rare worst-case scenarios. The other category is to apply an optimization algorithm (e.g., (meta)heuristic search algorithm), on top of Monte Carlo simulation, as in [7] and [8], which yield substantially better results, i.e., tighter lower bounds of the WCRT estimation.

Another approach is to use stochastic analysis of hybrid task sets in priority-driven soft real-time systems, as in [9]. Nevertheless, this approach does not allow for dependencies between tasks in the analysis, and the priority of jobs (a task is comprised by a sequence of jobs) and task periods are fixed.

In this paper, we present a novel statistical-based approach to response time analysis of systems with intricate execution dependencies between tasks. The proposed method uses samples collected by running Monte Carlo simulation as the input, and produces WCRT estimates on tasks along with a predictable probability of being exceeded, i.e., 10^{-9}.

2 Modeling of Complex Real-Time Systems

The system model used in this work describes the detailed execution dependencies between tasks with respect to resource usage and interaction, e.g., Inter-Process Communication (IPC), CPU execution time and logical resource usage. Practically, the model is specified by the modeling language used in RTSSim [10], which can be considered as a domain-specific language describing both architecture and behavior of task-oriented systems developed in C, and running on a single processor. Its syntax and semantics are as expressive as the C programming language, and include the typical RTOS services to the task models, such as task scheduling (e.g. Fixed-Priority Preemptive Scheduling), IPC via message passing and synchronization (semaphore). RTSSim employs a hierarchical model to specify the system structure consisting of a number of *tasks*. Each task is characterized by a *period*, a *constant offset*, a *maximum jitter*, and a *priority*. Periods and priorities can be changed at any time by any task in the application. Finally, each task is composed of a number of *jobs* and invoked RTOS services. The interested reader can refer to [10] for a thorough description of RTSSim.

3 Extreme Value Theory

Extreme Value Theory (EVT) [11] is a separate branch of statistics for dealing with the tail behavior of a distribution. It is used to model the risk of the extreme, rare events, without the vast amount of sample data required by a brute-force approach. The example applications are hydrology, material sciences, telecommunications etc.

There are three models in EVT, i.e., the Gumbel (type I), Frechét (type II) and Weibull distributions (type III), which are intended to model random variables that are the maximum or minimum of a large number of other random variables. It is worth noting that the Frechét distribution is bounded on the lower side ($x > 0$) and has a heavy upper tail, while the Weibull model relates to minima (i.e., the smallest extreme value). Since the purpose of this work is to find the higher response time of the tasks in rare worst-case scenarios, we therefore use the maximum case in the Gumbel distribution, referred to as the Gumbel Max in the reminder of the paper.

4 WCRT Estimation Based on EVT

The proposed method, *WCRTEVT* is shown in Algorithm 1. It is a recursive procedure which takes as argument m data sets, of which each contains N samples of the response time of the task under analysis. The algorithm returns the WCRT estimation with a predictable probability of being exceeded (i.e., 10^{-9}). It consists of the following two steps: 1) construction of the referenced data sets, 2) WCRT estimation of the referenced data sets using EVT.

4.1 The Referenced Data Sets

In order to construct the input data sets to the *WCRTEVT*, there are m Monte Carlo simulations in RTSSim to run at first. Then the n best simulations with the highest maximum value of response times, are selected as the referenced data sets. For each referenced data set, there are N (i.e., N is no less than 9 000) samples of the response time taken from the task under analysis. This sufficiently ensures making a good estimate. The construction is showed in rows 1-3 in Algorithm 1, where x_i in line 3 is the highest response time of the task under analysis observed in simulation per each data set.

4.2 WCRT Estimation of the Referenced Data Sets

4.2.1 Blocking of N Samples

In order to avoid the risk of mistakenly fitting raw response time data that may not be from random variables, to the Gumbel distribution, we use the method of block maxima [11], as proposed in [12]. This is done by grouping N response time samples in each referenced data set into k blocks of size is b, and then choosing the maximum value from each block to construct a new set of sample "block maximum" values, i.e., $Y \leftarrow y_{i,1}, ..., y_{i,k}$, $y_{i,k} \leftarrow maxima(S) \leftarrow N_{(k-1)\times b+1}, ..., N_{kb}$ as shown in row 6, 9 and 10 in Algorithm 1. The samples at the end of the execution sequence in a simulation that do not completely fill a block are discarded. For instance, if there are 9 samples per data set, i.e., $\{1119, 1767, 2262, 2287, 1792, 2687, 1942, 1842, 1692\}$, and b (i.e., the size of the blocks) is 2, then the last sample (i.e., 1692) in the sequence is discarded since it can not be grouped in the 4 (i.e., $\left\lfloor \frac{9}{2} \right\rfloor$) blocks. Furthermore, the initial value of b is 100.

4.2.2 The Best-fit Gumbel Max Parameters Estimation

The estimation of the parameters of the Gumbel Max distribution is the core of *WCRTEVT*, which is also an iterative procedure as shown in rows 8-35 in Algorithm 1. The selection of b is a trade-off between the quality of fit to the Gumbel Max distribution, and the number of blocks (i.e., k) in each data set available used in the estimation of the Gumbel parameters. In this paper, we introduce two procedures using two different search algorithms, i.e., *lwbsearch* and *upbsearch* which could find the proper value of b producing the best-fit Gumbel Max parameters estimation. The algorithm *lwbsearch* is invoked at first as shown in rows 8-26 in Algorithm 1, which focuses on searching for the value of b to be as low as possible. In this way, there are more blocks, i.e., the bigger value of k, used as samples in the estimation. However, in some cases, *lwbsearch* may fail in finding such value of b in best-fit tests. If this is the case, then *upbsearch* will be adopted, which is showed in rows 27-35 in Algorithm 1. Moreover, the best-fit test is in terms of examining the estimated Gumbel parameters by using a goodness-of-fit (GOF) test, i.e., Chi-square test. Note that other more advanced (meta)heuristic search algorithms can be applied. While the empirical results including the one presented in Section 5 and the ones have not been included in this paper due to space limitations, show that the two proposed algorithms work well enough to reach the goal. There is one more interesting point to highlight, i.e., the generally accepted value of k is 30 as introduced in [12]. Therefore, in this work, the size of blocks b should be smaller than $\left\lfloor \frac{N}{30} \right\rfloor$. For the sake of space, we can not give the detailed explanation about each search algorithm, as well as their implementation.

4.2.3 The WCRT Estimations Formula

The two parameters of the Gumbel Max distribution: a location parameter μ and a scale parameter β, are used in the Gumbel percent-point function, which returns the WCRT estimation that the block maximum Y cannot exceed with a certain probability q, as shown in Equation 1.

$$est = \mu - \beta \times log(-log((1 - P_e)^b)) \tag{1}$$

Algorithm 1 $WCRTEVT(m)$

1: $RT \leftarrow rt_1, ..., rt_m \leftarrow MonteCarlo(m, rnd\_inst())$
2: $n \leftarrow \frac{m}{100}$
3: $X \leftarrow x_1, ..., x_i, ..., x_n \leftarrow selectHRT(n, RT)$
4: **for all** x_i such that $1 \leq i \leq n$ **do**
5: $b \leftarrow 100$
6: $k \leftarrow \left\lfloor \frac{N}{b} \right\rfloor$
7: $success \leftarrow false$
8: **while** $k \geq 30$ **and** $success = false$ **do**
9: $S \leftarrow s_{i,1}, ..., s_{i,k} \leftarrow segment(N, b)$
10: $Y \leftarrow y_{i,1}, ..., y_{i,k} \leftarrow maxima(S)$
11: **if** $passChiSquareTest(Y) > 0$ **then**
12: $lwb \leftarrow \frac{b}{2}$
13: $upb \leftarrow b$
14: $b \leftarrow \left\lfloor \frac{lwb + upb}{2} \right\rfloor$
15: **while** $success = false$ **do**
16: $success \leftarrow lwbsearch(b, Y)$
17: **if** $success = true$ **then**
18: $l, s \leftarrow ChiSquareTest(Y)$
19: $est_i \leftarrow wcrtevt(b, l, s)$
20: **end if**
21: **end while**
22: **else**
23: $b \leftarrow 2 \times b$
24: $k \leftarrow \left\lfloor \frac{N}{b} \right\rfloor$
25: **end if**
26: **end while**
27: $upb \leftarrow b$
28: $b \leftarrow \frac{b + \frac{b}{2}}{2}$
29: **while** $success = false$ **do**
30: $success \leftarrow upbsearch(b, Y)$
31: **if** $success = true$ **then**
32: $l, s \leftarrow ChiSquareTest(Y)$
33: $est_i \leftarrow wcrtevt(b, l, s)$
34: **end if**
35: **end while**
36: **end for**
37: $EST \leftarrow est_i, ..., est_n$
38: $rt_{est} \leftarrow min(EST)$
39: **return** rt_{est}

4.2.4 Selecting the Lowest WCRT Estimation

As the last step in *WCRTEVT*, the lowest WCRT estimate is selected as the WCRT estimate on all m data sets. This is also confirmed by the empirical results presented in Section 5.

5 Empirical Results

A validation model inspired by a real industrial control system is constructed with the purpose to investigate how close the response time given by *WCRTEVT* is to the exact WCRT achieved by the simulation optimization-based method, i.e., HCRR in [8]. Moreover, in order to make the model analyzable by using basic RTA, the adhering task execution dependencies are simplified in that the execution time of the tasks is only varied by asynchronous message-passing with the loop bounds manually added to the simulation model. The results of five different methods are showed in Table 1.

Table 1. The results comparison for the MV.

	MC	*MABERA*	*HCRR*	*Basic RTA*	*WCRTEVT*
MV	4332	4332	4332	5982	**4574.556**

Clearly, the WCRT estimation achieved by *WCRTEVT* is 5.6% (i.e., $(4\,574.556 - 4\,332)/4\,332 \times 100\%$) more pessimistic than the exact value derived by HCRR and MC (Monte Carlo simulation), but 23.5% (i.e., $(5\,982 - 4\,574.556)/5\,982 \times 100\%$) less pessimistic when compared to the value obtained by basic RTA. Hence, we believe that *WCRTEVT* has the potential to provide meaningful results, i.e., tighter upper bounds of the WCRT estimation in the analysis of the real-time systems with more complex execution dependencies between tasks.

6 Conclusions and Future Work

This paper has presented ongoing work towards performing response time analysis for system models with intricate execution dependencies between tasks, by using the proposed statistical-based method based on extreme value theory. Specially, we have presented and validated the method by using a model inspired by real industrial control systems, which shows the benefit over basic RTA, in terms of reduced pessimism. Contrary to existing stochastic real-time analysis, the proposed method is not restricted by the assumption that tasks are independent, that the job-level priority is fixed and that the worst-case scenario only happens in the case of the critical instance. As part of future work, the evaluation on models with more complex execution dependencies between tasks will be conducted.

Acknowledgment

This work was supported by the Swedish Foundation for Strategic Research via the strategic research centre PROGRESS.

References

[1] "Website of ABB Group," www.abb.com.

[2] N. Audsley, A. Burns, R. Davis, K. Tindell, and A. Wellings, "Fixed priority pre-emptive scheduling: an historical perspective," *Real-Time Systems*, vol. 8, no. 2/3, pp. 129–154, 1995.

[3] G. Behrmann, A. David, and K. G. Larsen, "A tutorial on UPPAAL," in *Formal Methods for the Design of Real-Time Systems: 4th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM-RT 2004*, ser. LNCS, M. Bernardo and F. Corradini, Eds., no. 3185. Springer–Verlag, September 2004, pp. 200–236.

[4] T. Amnell, E. Fersman, L. Mokrushin, P. Pettersson, and W. Yi, "Times - a tool for modelling and implementation of embedded systems," in *TACAS '02: Proceedings of the 8th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*. London, UK: Springer-Verlag, 2002, pp. 460–464.

[5] "Rapita systems, www.rapitasystems.com, 2008."

[6] D. Decotigny and I. Puaut, "ARTISST: an extensible and modular simulation tool for real-time systems," in *Proc. of the 5th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC '02)*, 2002, p. 0365.

[7] J. Kraft, Y. Lu, C. Norström, and A. Wall, "A metaheuristic approach for best effort timing analysis targeting complex legacy real-time systems," in *Proc. of the 14th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 08)*, April 2008, pp. 258–269.

[8] M. Bohlin, Y. Lu, J. Kraft, P. Kreuger, and T. Nolte, "Simulation-based timing analysis of complex real-time systems," in *The 15th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, RTCSA 09*, August 2009, pp. 321–328.

[9] G. A. Kaczynski, L. L. Bello, and T. Nolte, "Deriving exact stochastic response times of periodic tasks in hybrid priority-driven soft real-time systems," in *Proceedings of 12th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA'07)*. IEEE Industrial Electronics Society, September 2007, pp. 101–110.

[10] J. Kraft, "RTSSim - A Simulation Framework for Complex Embedded Systems," Mälardalen University, Technical Report, March 2009.

[11] J. S. J. Beirlant, Y. Goegebeur and J. Teugels, *Statistics of Extremes: Theory and Applications*. Wiley Press, 2004.

[12] S. H. J. Hansen and G. Moreno, "Statistical-based wcet estimation and validation," in *9th Int'l Workshop on Worst-Case Execution Time Analysis*, 2009, pp. 123–133.