Trace-Based Statistical Response-Time Analysis of Complex Real-Time Embedded Systems

Yue Lu[1], Thomas Nolte[1], Iain Bate[2] and Liliana Cucu-Grosjean[3]
[1]Mälardalen Real-Time Research Centre (MRTC), Västerås, Sweden
[2]Department of Computer Science, University of York, York, United Kingdom
[3]INRIA Nancy-Grand Est, Nancy, France
yue.lu@mdh.se

Abstract

Real-time embedded systems are becoming ever more complex, and we are reaching the stage where even if static Response-Time Analysis (RTA) was feasible from a cost and technical perspective, the results are overly pessimistic making them less useful to the practitioner. When combined with most timing analysis tends to be statistical in nature, this suggests there should be a move toward statistical RTA. However, to make such analysis useful, it is imperative that we have evidence that the statistical RTA and the information analyzed is sufficiently accurate. In this paper we present and validate a technique for statistical RTA that can cope with systems that are complex from both a size and tasks' dependencies perspective. This claim is backed up by our evaluation using information from real industrial control systems.

1 Introduction

Many industrial embedded systems are very complex, large, flexible, and highly configurable software systems. Such systems often consist of millions of lines of code, and contain hundreds of tasks, many are with real-time constraints and being triggered by other tasks in a complex and nested pattern. More importantly, in such systems, tasks may have intricate dependencies in their temporal behavior, such as 1) asynchronous message-passing and globally shared state variables, which may decide important control-flow conditions with major impact on task execution time as well as task response time, 2) task offsets, and 3) run-time changeability of priorities and periods of tasks. Consequently, the systems have a very complicated runtime behavior. We refer to systems with such characteristics as *Complex Real-Time Embedded Systems* (CRTES).

The severity of the failure about missing tasks' deadlines is grave for the CRTES in the safety-critical domain, where functional and temporal correctnesses are equally important. An example is that a deadline miss in an industrial robotic control system may lead to a failure, which would further lead to a significant economical loss, in terms of halting an entire production line in a factory for hours and reducing production rate. In order to determine that all such timing constraints are met in all circumstances, Response-Time Analysis (RTA) is often used in the context. Traditional RTA methods [1] are under the assumption that tasks are independent with each other, and use the Worst-Case Execution Time (WCET) of tasks in the analysis, which increases the degree of pessimism in the results and hence make these results cannot be practically applied. Probabilistic approaches [3, 4, 5] can reduce the pessimism, in which tasks' execution times are modeled by *independent and identically distributed (i.i.d)* discrete random variables, which, however, cannot be applied to model the execution time of tasks in the CRTES, due to the existence of adhering intricate task execution dependencies that we mentioned previously. Other methods which adopt resource reservations algorithms such as the CBS [2] usually requires the exact knowledge of the entire distributions of the computation times and the inter-arrival times of the tasks, which are in generally impossible to obtain. Real-Time Queueing Theory (RTQT) [9] also provides a way to compute tasks' response time distributions using various real-time scheduling algorithms, under the *heavy traffic assumption* which significantly restricts its application in practice. Moreover, preemption between customers is not permitted in RTQT.

Our previous work has used statistical analysis based on simulation [8] and model checking-based approaches [7]. However, each of these is limited by the validity of the models [6] that they heavily depend on, and the statistical techniques that make certain assumptions about the nature of the information being processed that may not hold in practice. Therefore in this paper, it is our intention to timing traces taken from real systems and then process them with more powerful statistical techniques by extending our prior work in [8], in terms of not only giving accurate worst-case behaviors, but also allowing the validity of the results to be considered.

2 Problem Overview

Throughout this work, we particularly propose a method to collect qualified analysis samples in timing traces taken from real systems, and we study the trade-offs between the improvement in the statistical inference in RapidRT [8], i.e., in terms of reducing the number of analysis samples and easing statistical constraints, and results accuracy. Such analyses are necessary and critical for many applications when RapidRT is used in contexts. Before focusing on the contributions in this work, we introduce RapidRT, its parameters and assumptions used through the rest of the paper.

RapidRT is based on Extreme Value Theory (EVT), which is used to model the risk of the extreme, rare events. Further, RapidRT is a recursive procedure:

1. As the first two arguments, it takes n reference data sets each of which contains m analysis samples containing tasks' response times, resulting in $n \times m$ analysis samples in total. In addition, the individuals in such analysis samples are assumed to be i.i.d. for the purpose of statistical inference.
2. For each reference data set, the algorithm returns the WCRT estimate of the task under analysis with a probability of being exceeded, the third algorithm argument P_e (e.g., 10^{-9} which is for instance adopted by Airbus at the highest development assurance level in the safety-critical system domain).
3. Next, RapidRT will verify if the sampling distribution consisting of n WCRT estimates for all n reference data sets (i.e., the EVT distribution hereafter) conforms to a normal distribution or not, according to the result given by the non-parametric Kolmogorov-Smirnov test (the KS test hereafter). If it is, then RapidRT will calculate the Confidence Interval (i.e., CI hereafter) of the EVT distribution, at the certain confidence level cl (e.g., 99.7%), and choose the upper bound on the CI as the final WCRT estimate. This invents a new hard statistical constraint, i.e., from the statistical perspective, given the modeled system, the possibility of the existence of a higher WCRT estimate (i.e., the actual WCRT of the task on focus) than the WCRT estimate given by RapidRT is no more than $P_e \times cl$. Otherwise, the *resampling* statistic *bootstrap* will be adopted to obtain the upper bound on the CI of the EVT distribution.

It is interesting to stress that the input to RapidRT consisting of a number of analysis samples in timing traces, can be taken from either the simulation model which models the target system [8] or the real system. For latter, we propose a sampling mechanism which is to be introduced as follows.

3 Contributions

3.1 The Sampling Mechanism for Collecting Timing Traces Taken from Real Systems

First, in order to eliminate bias on the sampling, which is a key issue of selecting samples from the population of all individuals concerning the desired information, the technique of Simple Random Samples (SRS) is adopted. The SRS gives every possible sample of a given size the same chance to be chosen. Practically, when such samples are taken from real systems, the SRS can be done by randomizing system inputs by using the *uniform distribution*. Secondly, we propose a sampling mechanism which first executes the real system for N times (i.e., N *sub-timing traces*) based on the SRS technique, and each of sub-timing traces contains m raw RT data for every adhering task. Next, per sub-timing trace, the highest value of raw RT data for each task recorded, will be chosen as the sample to construct the new sampling distributions of the RT data of the same task to be analyzed by the statistical inference in RapidRT (to be introduced in Section 3.2). Furthermore, since there are no dependencies between any maximum of the RT data of tasks from two independent sub-timing traces, as a result, all the individuals in the new reconstructed sampling distributions are mutually independent. Hence, the underline assumption i.i.d. is realistic and satisfied when such new reconstructed sampling distributions are used in the statistical inference.

3.2 Improvement in the Statistical Inference in RapidRT

In our prior work, the values of parameters in RapidRT are obtained according to empirical evidence, i.e., the value of four parameters in RapidRT (as introduced in Section 2) is statistically sufficient enough to produce an upper bound of tasks' WCRT estimates which, however, is way too pessimistic, i.e., 17.25% more pessimistic than the exact value of the task's WCRT in the validation model in [8]. In addition, there was very less confidence in drawing any conclusions about using RapidRT to perform RTA of CRTES by only evaluating two simulation models. Hence, it is necessary to improve the statistical inference procedure in RapidRT by reducing the number of samples and easing the statistical constraint and confidence level, as well as evaluate the improved procedure by using more case studies. In this work, such improvement can be done from the following perspectives centering around RapidRT parameters, i.e., the number of reference data sets n, the number of samples per each reference data set m, the statistical constraint P_e, and the confidence level cl. To be specific, we propose a number of algorithms, of which implementation is described by Algorithms 1, 2, 3, 4, 5, and 6. Moreover, since we are more interested in reducing the number of analysis samples i.e., $n \times m$, which is critical when such timing information is collected by executing the real system, the corresponding improvement is subsequently prioritized to other tasks about finding a lower value of P_e and cl, as shown in Rows 5 – 8 in Algorithm 1, where the new statistical inference procedure starts with some initial values as introduced in [8]. For space sake, some sufficiently enough details about such improvement procedure will be given in our full paper submission.

Algorithm 1 $RapidRT^{*}_{Val}(n, m, P_e, cl)$

1: $n \leftarrow 50$
2: $m \leftarrow 20000$
3: $P_e \leftarrow 10^{-9}$
4: $cl \leftarrow 0.997$
5: $m_{IMP} \leftarrow IMP_M()$
6: $n_{IMP} \leftarrow IMP_N()$
7: $P_{eIMP} \leftarrow IMP_{Pe}()$
8: $cl_{IMP} \leftarrow IMP_{CL}()$
9: $rt_{est} \leftarrow rapidRT(n_{IMP}, m_{IMP}, P_{eIMP}, cl_{IMP})$

Algorithm 2 $IMP_M(n, m, P_e, cl)$

$m \leftarrow 20000$
$rt_{est} \leftarrow rapidRT(n, m, P_e, cl)$
$b \leftarrow \frac{m}{2}$
$rt'_{est} \leftarrow rapidRT(n, b, P_e, cl)$
$upb \leftarrow m$
$success \leftarrow false$
while $rt'_{est} \geq rt_{exact}$ **and** $success = false$ **do**
  if $rt'_{est} < rt_{est}$ **then**
    $rt_{est} \leftarrow rt'_{est}$
    $upb \leftarrow b$
    $b \leftarrow \frac{b}{2}$
    $rt'_{est} \leftarrow rapidRT(n, b, P_e, cl)$
  else
    $lwb \leftarrow b$
    $b \leftarrow \frac{upb + lwb}{2}$
    $success \leftarrow true$
  end if
end while
$m_{IMP} \leftarrow upperbinarysearch(b, lwb, upb)$

4 Evaluation

4.1 Four Evaluation Models

We examine the idea by using four simulation models describing a fictive, representative industrial robotic control system developed by one of our industrial partners. Those models are designed to include some behavioral mechanisms from the industrial robotic control system: 1) Tasks with intricate dependencies in temporal behavior due to Inter-Process Communication (IPC); 2) The use of buffered message queues for IPC, which vary the execution time and response time of tasks dramatically; 3) Although fixed-priority preemptive scheduling is used as base, some task, may change its priority during runtime, in response to system events. More importantly, the exact value of WCRT of the task under analysis is known, as shown by Row *Exact WCRT* in Table 1. The detailed description of each model is introduced as follows:

1. Validation Model 1 (MV1): In MV1, the adhering tasks communicate with each other via bounded number of messages, at each task's invocation.
2. Validation Model 1 with 0.5 times CPU speed (MV1-0.5): When compared to MV1, the difference in MV1-

Algorithm 3 $upperbinarysearch(b, lwb, upb)$

$success \leftarrow false$
while $success = false$ **do**
  $rt'_{est} \leftarrow rapidRT(n, b, P_e, cl)$
  if $b - 1 = lwb$ **then**
    $success = true$
  else
    if $rt'_{est} < rt_{est}$ **then**
      $lwb \leftarrow b$
      $b \leftarrow \frac{upb + lwb}{2}$
    else
      $upb \leftarrow b$
      $b \leftarrow \frac{upb + lwb}{2}$
    end if
    $rt_{est} \leftarrow rt'_{est}$
  end if
end while
return b

Algorithm 4 $IMP_N(n, m_{IMP}, P_e, cl)$

$i \leftarrow 5$
$n \leftarrow 0$
$rt_{est} \leftarrow 0$
$tmp \leftarrow rapidRT(i, m_{IMP}, P_e, cl)$
for all i such that $5 \leq i \leq 50$ **do**
  $rt_{est} \leftarrow rapidRT(i, m_{IMP}, P_e, cl)$
  if $rt_{est} \geq tmp$ **then**
    $i \leftarrow i + 1$
  else
    $tmp \leftarrow rt_{est}$
    $n \leftarrow i$
    $i \leftarrow i + 1$
  end if
end for
return n

Algorithm 5 $IMP_{P_e}(n_{IMP}, m_{IMP}, P_e, cl)$

$i \leftarrow 9$
$rt_{est} \leftarrow 0$
$P_e \leftarrow P_{e1}, ..., P_{e9} \leftarrow 10^{-1}, 10^{-2}, ..., 10^{-9}$
$success \leftarrow false$
while $success = false$ **do**
  $rt_{est} \leftarrow rapidRT(n_{IMP}, m_{IMP}, P_{ei}, cl)$
  if $rt_{est} < rt_{exact}$ **then**
    $i \leftarrow i + 1$
    $success \leftarrow true$
  else
    $i \leftarrow i - 1$
  end if
end while
return P_{ei}

Algorithm 6 $IMP_{cl}(n_{IMP}, m_{IMP}, P_{eIMP}, cl)$

$rt_{est} \leftarrow 0$
$rt'_{est} \leftarrow 0$
$CL \leftarrow cl_1, cl_2 \leftarrow 0.95, 0.997$
$rt_{est} \leftarrow rapidRT(n_{IMP}, m_{IMP}, P_{eIMP}, cl_1)$
$rt'_{est} \leftarrow rapidRT(n_{IMP}, m_{IMP}, P_{eIMP}, cl_2)$
if $rt_{est} \leq rt'_{est}$ **then**
  $cl \leftarrow cl_1$
else
  $cl \leftarrow cl_2$
end if
return cl

0.5 is that the execution time of jobs in tasks[1] is doubled. The intention is to check if the result given by RapidRT can still bound the known WCRT while keeping the accuracy of analysis results, when the execution time of jobs in tasks is increased.

3. Validation Model 2 (MV2): When compared to MV1, the behavior about modeling the change priority and period of the most complicated task at runtime is included in MV2.
4. Validation Model 2 with 0.5 times CPU speed (MV2-0.5): When compared to MV2, the only difference in MV2-0.5 is that the execution time of jobs in tasks is doubled. The intention is the same as the one introduced in MV1-0.5.

4.2 Evaluation Results

As shown in Rows *RapidRT* and $RapidRT^{*}_{VAL}$ (by using the improved statistical inference procedure), clearly, the results given by latter is more accurate than the analysis results given by former. Specifically, for the model MV1 and MV2, the most pessimism is 0.25%, while when the execution time of jobs in tasks is doubled, the most pessimism is increased to be 3.43%, which is still quite reasonable. Furthermore, as shown in Table 1, $P_{e,cl}$ is not necessarily to be at the highest development assurance level in the safety-critical system domain, i.e., 10^{-9} as presented in our prior work [8], and hence can be eased in order to produce more accurate analysis results.

[1] A task consists of a number of jobs.

Table 1. The results obtained by the new version of RapidRT can more accurately bound the actual WCRT of tasks in all four evaluation models, than its previous version.

Models	MV1	MV1-0.5	MV2	MV2-0.5
Exact WCRT	4 432	11 382	4 432	11 382
RapidRT	5 196.68	17 328.55	5 921.678	12 254.946
$RapidRT^{*}_{VAL}$	**4 438.606**	**11 623.203**	**4 443.300**	**11 772,390**
$P_{e,cl}$	9.5×10^{-7}	9.5×10^{-5}	9.5×10^{-4}	9.5×10^{-5}
Pessimism	0.15%	2.12%	0.25%	3.43%
Improvement	17.1%	50.13%	33.36%	4.14%

5 Summary

This paper has presented ongoing work toward using our prior statistical response-time analysis method *RapidRT* with timing traces for complex real-time embedded systems. In addition, our evaluation results showed that by using an improved statistical inference procedure, RapidRT can find much less pessimistic WCRT estimates of tasks when compared to the results given by our prior work, i.e., at most 50.13% less pessimistic. The main part of our future work will focus on extending such validation by using independent real-time system model with various task's execution time.

References

[1] *Handbook of Real-Time and Embedded Systems*. Chapman and Hall/CRC (July 23, 2007), 2007.
[2] L. Abeni et al. Integrating Multimedia Applications in Hard Real-Time Systems. In RTSS' 98, pages 4–13, 1998.
[3] A. Burns et al. A Probabilistic Framework for Schedulability Analysis. In EMSOFT' 03, pages 1–15, 2003.
[4] L. Cucu-Grosjean. Probabilistic real-time schedulability analysis: from uniprocessor to multiprocessor when the execution times are uncertain. Technical report, RR-INRIA, May 2009.
[5] J. M. López et al. Stochastic analysis of real-time systems under preemptive priority-driven scheduling. *Real-Time Syst.*, pages 180–207, Nov. 2008.
[6] Y. Lu et al. A statistical approach to simulation model validation in response-time analysis of complex real-time embedded systems. In SAC' 11. ACM, Mar. 2011.
[7] Y. Lu et al. Timing Analyzing for Systems with Task Execution Dependencies. In COMPSAC' 10. IEEE, July 2010.
[8] Y. Lu et al. A Statistical Approach to Response-Time Analysis of Complex Embedded Real-Time Systems. In RTCSA' 10, Aug. 2010.
[9] J. E. K et al. Real-time queueing theory: A tutorial presentation with an admission control application. *Queueing Systems*, 35(1-4):1–21, 2000.