The Sixteenth IEEE International Conference on Embedded and Real-Time Computing Systems and Applications

A Statistical Approach to Response-Time Analysis of Complex Embedded Real-Time Systems

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 RapidRT, a novel statistical approach to Worst-Case Response-Time (WCRT) analysis targeting complex embedded real-time systems. The proposed algorithm combines Extreme Value Theory (EVT) and other statistical methods in order to produce a probabilistic WCRT estimate. This estimate is calculated using response time data from either Monte Carlo simulations of a detailed model of the system, or from response-time measurements of the real system. The method could be considered as a pragmatic approach intended for complex industrial systems with real-time requirements. The target systems contain tasks with many intricate dependencies in their temporal behavior, which violates the assumptions of traditional analytical methods for response time analysis and thereby makes them overly pessimistic. An evaluation is presented using two simulation models, inspired by an industrial robotic control system, and five other methods as reference.

1 Introduction

Many industrial embedded systems are very large, highly configurable software systems, containing many event-triggered tasks, triggered by other tasks in complex, nested patterns. Consequently, they have a very complex runtime behavior. Such systems may consist of millions of lines of code, and contain hundreds of tasks, many with real-time constraints. Examples of such systems include the robotic control system IRC 5, developed by ABB [1], as well as several telecom systems. In such systems, many tasks have intricate dependencies in their temporal behavior, which violates the assumptions made in most real-time theory, i.e. the tasks are independent in the analysis model. Such dependencies include asynchronous message-passing and globally shared state variables which may decide important control-flow conditions with major impact on task execution time. Other violations of these assumptions are runtime changeability of priorities and periods of tasks. We refer to systems with such characteristics as Complex Embedded Real-Time Systems (CERTS). For such systems, timing analysis methods, such as Response-Time Analysis (RTA) [3], are often not applicable, as their assumptions do not hold. They thereby become overly pessimistic; often too pessimistic to be useful. Moreover, methods like RTA relies on the existence of a Worst-Case Execution-Time (WCET) for each task. Correspondingly, the quality of the analysis is directly correlated to the quality of the WCET estimates. In order to perform a safe analysis covering system worst-case scenarios, static WCET analysis has to be adopted in the context, but today's WCET tools cannot analyze the complex high-performance CPUs used by many industrial systems.

An alternative approach is to use simulation-based methods, where the simulation model contains execution time data from measurements. The first type of simulation technique to use is Monte Carlo simulation, which can be described as keeping the highest result from a set of randomized simulations. Examples of tools implementing Monte Carlo simulation include the commercial tool VirtualTime [22] and the academic tool ARTISST [7]. However, the main drawback of Monte Carlo simulation is its 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. a (meta)heuristic search algorithm) on top of Monte Carlo simulation, as in [15] and [4], which yields substantially better results, i.e. tighter lower bounds, but not upper bounds of the WCRT estimation.

Another interesting approach features the use of stochastic task execution times in RTA of priority-driven soft real-time systems [13] and schedulability analysis [20]. Nonetheless, this approach currently does not allow for execution dependencies between tasks in the analysis. [5] presents another probabilistic framework extending RTA

1533-2306/10 $26.00 © 2010 IEEE
DOI 10.1109/RTCSA.2010.13

to incorporate a probabilistic characterization of task arrivals and execution times. However, task execution dependencies such as runtime changeability of task priorities and periods, and message-passing, are not taken into consideration. Other related work includes [11] presenting how likely a WCET estimate generated by Extreme Value Theory (EVT) [10] will be exceeded in the future. Here, the search algorithm concerning the best-fit Gumbel distribution parameters is done in a simple way, by only doubling the block size. A tool for statistical analysis of hard real-time scheduling algorithms is introduced in [9], where the data used for the statistical analysis can be collected from simulations. However, the simulation models described do not include task behavior and thereby do not capture execution dependencies between tasks.

In earlier work [19] we presented a first approach for using statistical response time analysis of complex embedded systems containing task execution dependencies, based on EVT. In this paper, we extend our work by bringing in a more systematic way of using a combination of EVT and other statistical methods with the purpose of producing a WCRT estimate of tasks on focus in system models under a hard statistic constraint, i.e. a certain probability of being exceeded. More importantly, the statistical conviction on using the new improved method as a tighter upper bound of WCRT estimate is carried out and confirmed by our evaluation on two models depicting a fictive but representative real industrial robotic control system.

Contributions: The contributions of this paper are four:

1. We introduce a new method of constructing a sampling distribution for best-fit Gumbel Max parameters estimation, by eliminating the dependencies between each response time data caused by tasks execution dependencies, which is not considered in our previous work.
2. We propose a new search procedure focusing on using more samples in the best-fit Gumbel Max parameters estimation, which could produce more precise estimated parameters.
3. We bring in a new systematic way of combining EVT with other statistics in order to assure that the result given by our proposed method can statistically be considered as a tight upper bound of the WCRT estimate of tasks under analysis in the system model. This also highlights how statistical analysis methods can be used to assess relevant issues in the real-time realm.
4. We evaluate the proposed method *RapidRT*, and show that it can find an accurate WCRT estimation in the cases where the true WCRT is known, and the highest WCRT estimate when compared to the ones obtained using other simulation-based methods when the true WCRT is unknown (i.e. when applying RTA is infeasible).

Organization: The remaining part of the paper is organized as follows: Section 2, at first, briefly presents a new type of system model used in our analysis, where the WCET of each data-driven task is represented as an expression containing parameters. Then we show how the modeling language is used in practice, and give the problem definition. Section 3 presents the proposed method, i.e. RapidRT, and Section 4 describes the implementation of our developed testbed and a tool chain. The evaluation by using two case-study models with five methods as reference is presented in Section 5, before conclusions are drawn in Section 6.

2 Modeling of CERTS

The novelty of the system model used in this paper is to represent the WCET of tasks as a symbolic formula centering around i) the number of messages in the buffers consumed by, or sent by, a task and ii) the value of the Globally Shared State Variables (GSSVs) used in selecting control branches. A WCET dependent on external context is often referred to as a parametric WCET [6]. Moreover, the system model uses job-level[1] WCET estimates, which makes static WCET analysis feasible on job-level as, even though there may be dependencies among tasks, there will be no execution dependencies inside jobs.

Hence, the system S contains a set of non-blocking tasks, each of which consists of n *jobs*, where $n \in \mathbb{N}$. Each deadline-constrained task τ_i is a tuple $\tau_i(T_i, C_i^p, D_i, O_i, J_i, P_i)$, where T_i is the task period with maximum jitter J_i, constant offset O_i and a priority P_i, C_i^p is the WCET expression as a function of b buffers (i.e. $U_{i,1}, ..., U_{i,b}$) and g GSSVs (i.e. $V_{i,1}, ..., V_{i,g}$) associated with task τ_i and execution time on jobs, D_i is the relative deadline ($max(C_i^p) \leq D_i \leq T_i$). For sake of space (interested readers can refer to [18]), we only give the WCET expression of task τ_i as shown in Equation 1:

$$C_i^p = \sum_{j=1}^{b} C_i^p(U_{i,j}) + \sum_{j=1}^{g} C_i^p(V_{i,j}) + \sum_{j=1}^{c} C_{i,nvj} \qquad (1)$$

where c is the number of non-volatile (NV) sections which do not contain any buffers U and GSSVs V in task τ_i.

In practice, such system models are described by the modeling language used by RTSSim, which describes both architecture and behavior of task-oriented systems developed in C. An RTSSim simulation model consists of a set of tasks, sharing a single processor. Each task in RTSSim is a C program, which executes in a "sandbox" environment with similar services and runtime mechanisms as a normal real-time operating system, e.g. task scheduling, inter-process communication (message queues) and synchronization (semaphores). The default scheduling policy

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

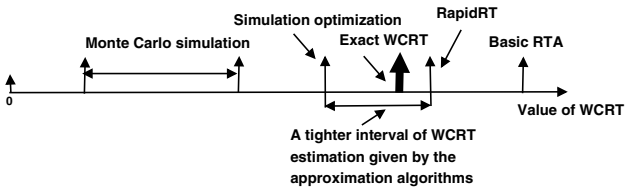

Figure 1. Illustration of applying different WCRT analysis methods in the system model presented in Section 2.

of RTSSim is Fixed-Priority Preemptive Scheduling (FPPS) and each task has scheduling attributes such as priority, period, offset and jitter. A more thorough description of RTSSim can be found in [14].

In this paper, we present a new approximation method, *RapidRT*, based on Extreme Value Theory (EVT) [10], with the purpose of determining a tight and meaningful upper bound of the WCRT of tasks in complex real-time systems, where basic RTA [12] cannot be applied in practice. Consequently, the problem can be defined as follows. We are given a model, which can be simulated as RTSSim simulation instance s. Let $R(s)$ denote the highest response time measured for the task under analysis in the simulation instance s. Given m simulation instances $s_1, ..., s_i, ..., s_m$ as the samples in space S, i.e. $S \leftarrow s_1, ..., s_j, ..., s_n$, where $n \in \mathbb{N}$, the goal of the problem is then to find an estimation that is bigger than any $R(s_j)$ in space S. Moreover, the relationship between the results obtained by different analysis methods and the exact value of the WCRT of the task on focus in the system model is illustrated in Figure 1.

3 RapidRT

Extreme Value Theory (EVT) was first codified in 1958 and 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. Example applications of EVT include risk management, insurance, hydrology, material sciences, telecommunications etc.

There are three models in EVT, i.e. 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 tasks concerning rare worst-case scenarios, we use the maximum case in the Gumbel distribution, referred to as Gumbel Max in the reminder of the paper. Further, the curve in Figure 2 shows the shape of Gumbel Max.

The proposed method, *RapidRT*, is shown in Algorithm 1. It is a recursive procedure which, as the first two arguments, takes n reference data sets each of which contains m samples of the response time of the task under analysis. For each reference data set, the algorithm returns the WCRT estimation with a probability of being exceeded, i.e. 10^{-9}, which is the third algorithm argument. For instance, Airbus [2] uses such a value 10^{-9} in the safety-critical system domain. Next, RapidRT will verify if the sampling distribution consisting of n WCRT estimates given by EVT for all n reference data sets (we refer to such a sampling distribution as *EVT distribution* hereafter) conforms to a normal distribution or not, according to the result given by the non-parametric Kolmogorov-Smirnov test [16] (KS test hereafter, and the reason for why we are using KS test in this work is given in Section 3.2). If it is, then RapidRT will calculate the confidence interval (i.e. CI hereafter) of the EVT distribution, at the given confidence level 99.7%, and choose the upper bound of CI as the final WCRT estimate. This invents a new hard statistic constraint, i.e. in the statistical perspective, given the modeled system, the possibility of the existence of a higher WCRT estimate than the WCRT estimate given by RapidRT is no more than 1.5×10^{-12} (i.e. $(100\% - 99.7\%)/2 \times 10^{-9}$). Otherwise, if the EVT distribution cannot be fitted to a normal distribution, a *resampling* statistic *bootstrap* will be adopted to obtain the upper bound of CI of the EVT distribution. Further, in our evaluation, the EVT distributions for both two evaluation models conform to a normal distribution, therefore the bootstrap test will not be introduced in this paper. However, interested readers can find the details in Chapter 16 in [21].

3.1 Algorithm Outlined

The outline of the RapidRT algorithm is as follows, which is discussed in greater detail in the following sections.

1. Construct n reference data sets for the WCRT estimates by running m Monte Carlo simulations for each reference data at first, and then choosing the highest maximum value of response time of the task under analysis in each simulation. Consequently, the sampling distribution of response-time (RT) data per reference data set consists of the m highest maximum RT data of m simulations.
2. Perform the WCRT estimates on the task under analysis per each reference data set.
   (a) Set the initial block size b to 1, for each reference data set.
   (b) If the number of blocks $k = \left\lfloor \frac{m}{b} \right\rfloor$ is less than 30, the algorithm stops as there are not enough samples to generate an estimate.

(c) Segment m response times into blocks of size b, and for each of the $\left\lfloor \frac{m}{b} \right\rfloor$ blocks find the maximum values.

(d) Estimate the best-fit Gumbel parameters μ and β to the block maximum values by using a proposed search algorithm introduced in Section 3.3.3.

(e) Calculate a WCRT estimate based on the best-fit Gumbel Max parameters estimated through Step *d*), i.e. μ, β, and a target acceptance probability P_e, i.e. 10^{-9}.

3. After verifying if the EVT distribution (i.e. $EST \leftarrow est_1, ..., est_i, ..., est_n$) can successfully be fitted to a normal distribution by using KS test, RapidRT will return a result, i.e. $\overline{EST} + 3\sigma_{EST}$ (the sum of *mean* value and 3 *standard deviation* of EST at the confidence level 99.7%).

3.2 Construction of the Reference Data Sets

Due to the execution dependencies between tasks in the system model introduced in Section 2, the sampling distribution for each reference data set cannot be constructed by just collecting RT data given by running a single shot of Monte Carlo simulation. The reason for this is inherent in that such RT data are not from a random variable due to task execution dependencies such as globally shared state variables. In order to assess this issue, in this paper, we propose to run m Monte Carlo simulations in RTSSim at first, then select the highest value of response time per each simulation to construct a new sampling distribution per each reference data set. The construction is showed in row 2 in Algorithm 1, where $rt_{i,1}$ in line 2 is the highest response time of the task under analysis observed in the first simulation instance out of m simulation runs for the reference data set i. Moreover, because the search algorithm (to be introduced in Section 3.3.3) is not fully implemented in our tool chain (presented in Section 4), we choose 50 as the value of n, which is required by the non-parametric KS test in the verification process of fitting the EVT distribution to a normal distribution.

3.3 WCRT Estimation of the Reference Data Sets

3.3.1 Blocking of m Samples per Reference Data Set

In order to avoid the risk of mistakenly fitting raw response time data for each reference data set, that may not be from random variables, to Gumbel Max, we use the method of block maxima [10] as proposed in [11]. This is done by grouping m response time samples in each reference data set into k blocks of size b, and then choosing the maximum value from each block to construct a new set of sample "block maximum" values, i.e. $Y \leftarrow y_{i,1}, ..., y_{i,k}$, $y_{i,k} \leftarrow maxima(S) \leftarrow m_{(k-1)\times b+1}, ..., m_{kb}$ as shown in line 7 and 8 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.

3.3.2 Search Space of Block Size b

The search space of block size b is the range of $[1, \left\lfloor \frac{m}{30} \right\rfloor]$. Since in order to use the Chi-squared test in finding the best-fit Gumbel Max parameters, the number of blocks should not be less than 30; Otherwise, there are not enough samples in block maxima Y used in the estimation (to be introduced in Section 3.3.3). It is also interesting to notice that the value of m also impacts the success ratio of using our search algorithm introduced in the following section. For instance, for the evaluation model M1, the value of m is 20 000, which is sufficient enough to give a good coverage of the underline population. However, if the same value of m is used in another model MV, then our proposed search algorithm is not efficient in finding the best-fit Gumbel Max parameters, in terms of having a few failures. Therefore, we propose a solution with the intention of diminishing the number of samples in a sampling distribution fitting to Gumbel Max, by decreasing the number of simulations by half, i.e. 10 000 (i.e. 20 000 ÷ 2) which ensures that the proposed search algorithm can find the best-fit parameters for each reference data set. Further, the cost of collecting 10 000 and 20 000 samples for each reference data set in the different evaluation models is quite reasonable, i.e. 27.126 seconds and 96.118 seconds respectively. Note that we use mean value here due to low variance of computation time cost by simulation.

3.3.3 Best-fit Gumbel Max Parameters

The estimation of the parameters of the Gumbel Max distribution is the core of RapidRT, which is also an iterative procedure as shown in rows 6-36 in Algorithm 1. The intention of such a search procedure is to focus on searching for the value of block size b to be as low as possible. In this way, there are more blocks, i.e. the bigger value of blocks k, used as samples in the best-fit Gumbel Max parameters estimation. To achieve this, in this paper, we propose a search algorithm including a simple procedure of doubling block size b and a lower-part binary search algorithm [23], which are invoked as shown in rows 10-35 in Algorithm 1. For a better understanding and sake of space, we will illustrate the

entire search procedure by using a concrete example based on the data shown in Table 1. Moreover, the best-fit test is, in terms of examining the estimated Gumbel parameters, a goodness-of-fit (GOF) test, i.e. Chi-squared test at α-value of 0.05. Chi-squared test is used to determine if a sample comes from a population with a specific distribution [8], i.e. the Gumbel Max distribution in this work. Further, the null and alternative hypotheses are:

- H_0: the data, i.e. the maxima of the blocks follow the Gumbel Max distribution;
- H_a: the data, i.e. the maxima of the blocks do not follow the Gumbel Max distribution.

Table 1. Illustration of using a proposed search algorithm to find the best-fit Gumbel Max parameters.

step	*b*	*ALG*	χ^2	*step*	*b*	*ALG*	χ^2
1	1	db	×	8	128	db	√
2	2	db	×	9	96	lwb	×
3	4	db	×	10	112	lwb	×
4	8	db	×	11	120	lwb	×
5	16	db	×	12	124	lwb	√
6	32	db	×	13	122	lwb	√
7	64	db	×	14	**121**	lwb	√

The columns *b*, *ALG* and χ^2 in Table 1 represents *block size*, *search algorithm* and *result of Chi-squared test at α-value of* 0.05 respectively (Chi-squared test at α-value of 0.05 is referred to as χ^2 test without indicating α-value of 0.05 in the following context). Moreover, *lwb* stands for the algorithm *lwbsearch*, *db* means the algorithm which doubles the block size *b*, √ is *not reject* χ^2 test and × is *reject* χ^2 test. At the beginning, Algorithm 1 will try to find a valid upper bound of the search space used later in the lower-part binary search, by doubling block size *b* as shown through Steps 1 to 8 in Table 1. At step 8, such bound is found which is equal to 128. Then *lwbsearch* will start searching for the lowest value of b^* in the range of [1, 128], under the condition that the corresponding χ^2 test is not rejected. b^* is used to estimate the parameters for the Gumbel Max distribution which are considered as the best-fit parameters. For example, at Step 8, b_{lwb} (i.e. the lower bound of *b*) is set to be 1 and b_{upb} (i.e. the upper bound of *b*) is 128. The new value of *b* to be verified by using χ^2 test at Step 9 is 64, i.e. $\left\lfloor \frac{1+128}{2} \right\rfloor$. However, since 64 is already checked at Step 7, therefore b_{lwb} is updated to 64. Correspondingly, the new *b* value to be verified at Step 9 is 96, i.e. $\left\lfloor \frac{64+128}{2} \right\rfloor$. *lwbsearch* will not stop searching for b^* until at Step 14, when the value of *b* is 121 which succeeds in χ^2 test. While at Step 11, its preceding ordered number 120 fails in χ^2 test. Hence b^* is ensured to be 121 (in **bold** in Table 1). The corresponding block maxima is shown in Figure 2.

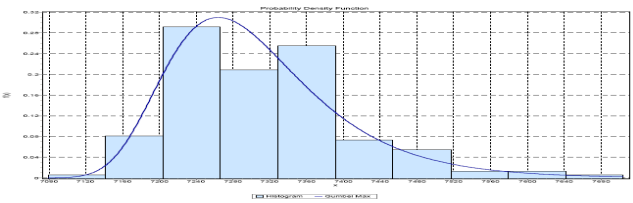

Figure 2. The block maxima obtained by our proposed search algorithm conforms to Gumbel Max distribution.

3.3.4 The WCRT Estimation 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 as shown in Equation 2, which returns the WCRT estimate that the block maximum *Y* cannot exceed with a certain probability P_e. Its implementation is the function *wcrtevt* with the arguments *b* (block size), *l* (location parameter), *s* (scale parameter) and P_e (acceptance probability) (refer to lines 13 and 18 in Algorithm 1). The process of determining the best-fit Gumbel Max parameters practically by using the tool chain we developed in this work, are explained in the following Section 4.

$$est = \mu - \beta \times log(-log((1 - P_e)^b)) \quad (2)$$

3.3.5 WCRT Estimation Given by RapidRT

Due to the different sampling distribution collected by running Monte Carlo simulation that are used in the best-fit Gumbel Max parameters estimation, the WCRT estimate given by EVT for each reference data set is therefore different with another. Consequently, it is worth applying other statistics on top of EVT, which ensures that the results given by EVT will not exceed a specific value at the certain confidence level that is considered as the final result given by RapidRT. In this work, we try to fit the EVT distribution to a normal distribution at first (one important underline assumption of the conventional statistical procedures is that the sampling distribution conforms to a normal distribution, which further enables application of parametric tests, such as analysis of variance (ANOVA) and t-test, to infer the parameters of the population), then use an upper bound of the CI of EVT distribution at the confidence level 99.7% as the final result given by RapidRT. Moreover, if the EVT distribution cannot be fitted to a normal distribution, which is not the case in the evaluation work later in this paper, we use a bootstrap test to obtain the upper bound of the CI of the EVT distribution.

4 Implementation

In this section, our testbed and the tool chain including the implemented tools are introduced in details. Our

testbed is running Microsoft Windows XP Professional, version 2002 with Service Pack 3. The computer is equipped with the Intel Core Duo CPU E6550 processor, 2GB RAM and a 4MB L2 Cache. The processor has 2 cores and 1 frequency level: 2.33 GHz.

RTSSim is a Monte Carlo simulation framework used to construct the sampling distribution for the WCRT estimate using EVT for each reference data set. It generates one text file *out.txt* which contains m lines of simulation results representing the highest value of response time for a specific task observed during each simulation in m simulation runs for each reference data set.

The core part of the tool chain, *ThinkStati*, is a prototype of RapidRT as an executable program with a simple user interface developed using Microsoft C# programming language and .NET framework 2.0. The software 1) reads one output of the RTSSim simulator, i.e. the reference data set file containing m (i.e. either 20 000 or 10 000 for the different evaluation models as explained in Section 3.2) samples of response times of the task on focus, at first, then 2) generates a text file *yblock.txt* for each reference data set after segmenting the samples as introduced in Section 3.3.1, then 3) produces the WCRT estimation on tasks under analysis according to the best-fit Gumbel Max parameters (verified and returned by *EasyFit* introduced in the following context) and the acceptance probability, i.e. 10^{-9} in this work. Due to limited time, the investigation on how to use the interfaces in EasyFit, concerning the verification result of the estimated Gumbel Max parameters, has not been done yet. Correspondingly, one proposed search algorithm has not been implemented in ThinkStati. The output of ThinkStati is a text file containing the EVT distribution, which is used by EXCEL 2007 to construct the confidence interval at the confidence level 99.7%. Last but not least, we choose the upper bound of such an interval as the final result given by RapidRT.

Concerning the Chi-squared test required by ThinkStati, it is done by using a commercial software *EasyFit* [8]. Specifically, given the text file *yblock.txt* which contains a certain number of samples generated by ThinkStati, as the input, the Chi-squared test engine embedded in EasyFit will return the results in terms of the success or failure of the hypothesis test concerning the acceptance of H_0 or *null hypothesis*.

5 Empirical Results

In this section, we firstly introduce two models used for method evaluation, including one validation model, and then we compare our solution against five other methods as reference: Monte Carlo simulation, MABERA, HCRR, basic RTA, and our previously proposed method WCRTEVT. Worth noting is that all models are inspired by real industrial control systems.

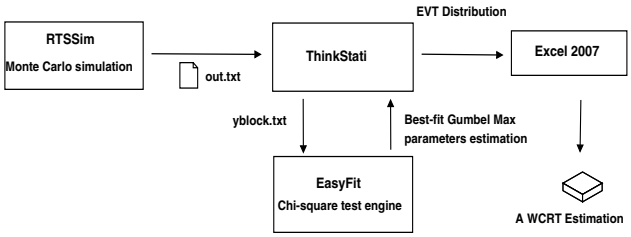

Figure 3. The toolchain in this work.

5.1 Evaluation Models

The two models, i.e. Model 1 (M1) and Model for Validation (MV), have similar architecture and analysis problems as one industrial real-time application in use at ABB [1]. M1 is representing a control system for industrial robots developed by ABB Robotics, which is not possible to analyze using methods such as RTA [3, 17]. We also use a simplified version of Model 1, making RTA is applicable, for validation (MV). The sole purpose of this model is to investigate how close the response time estimation given by RapidRT is to the true known WCRT derived by RTA. The scheduling policy is FPPS for all models, apart from M1 (where FPPS is used as base but one task changes its priority during runtime); MV uses fixed priorities. Furthermore, both M1 and MV can be described (modeled) by the system model proposed in Section 2.

This model represents a control system for industrial robotics, developed by ABB. M1 is designed to include some behavioral mechanisms from the ABB system which RTA can not take into account: 1) tasks with intricate dependencies in temporal behavior due to Inter-Process Communication (IPC) and globally shared state variables, 2) the use of buffered message queues for IPC, where triggering messages may be delayed, and 3) tasks that change scheduling priority or periods dynamically, in response to system events.

The modeled system controls a set of electric motors based on periodic sensor readings and aperiodic events. The calculations necessary for a real control system are, however, not included in the model; the model only describes behavior with a significant impact on the temporal behavior of the system, such as resource usage (e.g., CPU time), task interactions and important state changes. The details of the model are described in [14].

MV is constructed based on M1, but the adhering task execution dependencies are simplified in that 1) globally shared state variables have been removed, 2) priority and period are strictly static, 3) explicit loop bounds have been added manually, and 4) the constant offset of tasks is re-

moved. As a consequence, MV has considerably lower complexity, which makes both using the RTCF in basic RTA to calculate the WCRT of tasks under analysis, and achieving the exact WCRT by using simulation-based methods, e.g., Monte Carlo simulation and HCRR [4], feasible.

5.2 Results Comparison

For sake of space in Table 2 and Table 3, *MC* and *MAB* represent Monte Carlo simulation and MABERA respectively, and *RTA* stands for Basic RTA (without blocking). As shown in Table 2, when different simulation budgets (refer to Table 3) are given to different methods in order to obtain the highest WCRT estimate of the task under analysis, for MV, the result achieved by RapidRT is 19.96% (i.e. $(5\,196.68 - 4\,332)/4\,332 \times 100\%$) more pessimistic than the known WCRT estimate 4 332, but 13.13% (i.e. $(5\,982 - 5\,196.68)/5\,982 \times 100\%$) less pessimistic when compared to the value obtained by basic RTA. Further, when compared to the results given by the method proposed in our previous work WCRTEVT, RapidRT is 13.60% (i.e. $(5\,196.68 - 4574.56)/4574.56 \times 100\%$) more pessimistic than the result given by WCRTEVT. This is because in WCRTEVT, we only chose the lowest RT estimate among all reference data sets which could intentionally produce a lower WCRT estimate that is closer to the known WCRT, without any statistical confidence, e.g., CI. While in RapidRT, a more systematic way is adopted in terms of constructing a CI at the confidence level 99.7% of the EVT distribution for all reference data sets at first, then choosing the upper bound of such CI. This also brings in a harder statistic constraint, i.e. 1.5×10^{-12}, than the one in WCRTEVT, i.e. 10^{-9}. More importantly, for M1 where the true WCRT estimate of the task on focus is unknown, the result given by RapidRT does not only cover all the best results given by MC, MABERA and HCRR, but also is higher than the one given by WCRTEVT in terms of 1% more pessimistic (i.e. $(8698.289 - 8610.766)/8610.766 \times 100\%$). Further, not only because of the complexity of M1 (to which basic RTA cannot be applied) is much higher than MV, but also due to that RapidRT is designed in a more systematic way and is under a harder statistic constraint, we therefore believe that RapidRT can return a better (safer) WCRT estimate compared to the one achieved by WCRTEVT, specially when basic RTA cannot be applied.

Regarding the computation time consumed by each method used in the evaluation, Table 3 shows that the computation time cost by RapidRT is either 8.35% (i.e. $(1\,716.73 - 1\,573.36)/1\,716.73 \times 100\%$) (for MV) less than, or 180% (i.e. $(4\,805.90 - 1\,716.73)/1\,716.73 \times 100\%$) (for M1) more than, the computation time consumed by WCRTEVT. This is because the number of samples in each reference data set in MV and M1 required by RapidRT is different, as introduced in Section 3.3.2. However, in order to have a tighter WCRT estimate when compared to WCRTEVT, such extra computation time on the construction of sampling distribution is worthwhile and the cost is reasonably acceptable, i.e. 1.33 hours at most. Further, to optimize such number of samples used in total by RapidRT is part of our future work.

Table 2. Results comparison for two evaluation models, when the different simulation budgets, i.e. the number of simulations to run, are given to the methods.

	MC	MAB	HCRR	RTA	WCRTEVT	RapidRT
MV	4332	4332	4332	5982	4574.56	**5196.68**
M1	7682	8065	8474	NA	8610.766	**8698.289**

Table 3. The computation time corresponding to the number of simulations required to execute by each method.

	MC & MAB	HCRR	WCRTEVT	RapidRT
MV	36133.46 s	44.39 s	1716.73 s	**1573.36 s**
M1	36133.46 s	44.39 s	1716.73 s	**4805.90 s**

6 Conclusions and Future Work

In this paper, we have proposed a method for Worst-Case Response Time (WCRT) analysis for system models with intricate task execution dependencies using a statistical approach, which has been developed inspired by the complexity of real systems. For this purpose, we meticulously described an algorithm that combines Extreme Value Theory with other statistics to produce a tight upper bound of WCRT estimates, considering a probability of being exceeded commonly used in the industrial safety-critical system domain. We have evaluated the method on two models depicting a fictive but representative industrial control system. The focus of our future work includes addressing the optimization of the number of samples required by each reference data set in RapidRT, as well as tool chain automation. Moreover, we will investigate the possibility of evaluating RapidRT on real systems via non-trivial industrial case-studies, and applying more industrial standards on the tool chain.

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] Airbus, www.airbus.com/en/, 2009.

[3] N. Audsley, A. Burns, R. Davis, K. Tindell, and A. Wellings. Fixed priority pre-emptive scheduling: an historical perspective. *Real-Time Systems*, 8(2/3):129–154, 1995.

[4] M. Bohlin, Y. Lu, J. Kraft, P. Kreuger, and T. Nolte. Simulation-based timing analysis of complex real-time systems. In *RTCSA'09*, pages 321–328, August 2009.

[5] A. Burns, G. Bernat, and I. Broster. A probabilistic framework for schedulability analysis. In *EMSOFT'03*, pages 1–15, October, 2003.

[6] S. Bygde, A. Ermedahl, and B. Lisper. An efficient algorithm for parametric WCET calculation. In *RTCSA'09*, pages 13–21, August 2009.

[7] D. Decotigny and I. Puaut. ARTISST: an extensible and modular simulation tool for real-time systems. In *ISORC'02*, pages 365–372, April, 2002.

[8] Easyfit, www.mathwave.com/products/easyfit.html, 2010.

[9] J. Goossens and C. Hernalsteen. A tool for statistical analysis of hard real-time scheduling algorithms. In *SS'98*, page 58, 1998.

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

[11] J. Hansen and G. Moreno. Statistical-based WCET estimation and validation. In *WCET'09*, pages 123–133, June, 2009.

[12] M. Joseph and P. Pandya. Finding response times in a real-time system. *The Computer Journal (British Computer Society)*, 29(5):390–395, October 1986.

[13] G. A. Kaczynski, L. Lo. Bello, and T. Nolte. Deriving exact stochastic response times of periodic tasks in hybrid priority-driven soft real-time systems. In *ETFA'07*, pages 101–110, September 2007.

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

[15] J. Kraft, Y. Lu, C. Norström, and A. Wall. A metaheuristic approach for best effort timing analysis targeting complex legacy real-time systems. In *RTAS'08*, pages 258–269, April 2008.

[16] A. M. Law and D. M. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill Higher Education, 1999.

[17] C. Liu and J. Layland. Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment. *Journal of the ACM*, 20(1):46–61, 1973.

[18] Y. Lu, T. Nolte, I. Bate, and C. Norström. Timing analyzing for systems with task execution dependencies. In *COMPSAC'10*, July 2010.

[19] Y. Lu, T. Nolte, J. Kraft, and C. Norström. Statistical-based response-time analysis of systems with execution dependencies between tasks. In *ICECCS'10*, pages 169–179, March 2010.

[20] S. Manolache, P. Eles, and Z. Peng. Schedulability analysis of applications with stochastic task execution times. *ACM Trans. Embed. Comput. Syst.*, 3(4):706–735, 2004.

[21] D. S. Moore, G. P. Mccabe, and B. A. Craig. *Introduction to the practice of statistics*. W. H. Freeman and Company, New York, NY 10010, sixth edition, 2009.

[22] Rapita systems, www.rapitasystems.com, 2008.

[23] T. Cormen, C. Leiserson and C. Stein. *Introduction to Algorithms*. The MIT Press, 2nd revised edition edition, September 2001.

Algorithm 1 $ThinkStati(n, m, P_e)$

for all est_i such that $1 \leq i \leq n$ **do**
  $X_i \leftarrow rt_{i,1}, ..., rt_{i,m} \leftarrow MonteCarlo(m, rnd\_inst())$
  $b \leftarrow 1$
  $k \leftarrow \left\lfloor \frac{m}{b} \right\rfloor$
  $success \leftarrow false, bdouble \leftarrow 0, lwb \leftarrow 0, upb \leftarrow 0$
  while $k \geq 30$ **and** $success = false$ **do**
    $S_i \leftarrow s_{i,1}, ..., s_{i,k} \leftarrow segment(m, b)$
    $Y_i \leftarrow y_{i,1}, ..., y_{i,k} \leftarrow maxima(S_i)$
    if $passChiSquaredTest(Y_i, GumbelMax) > 0$ **then**
      if $b = 1$ **or** $b = 2$ **then**
        $success \leftarrow true$
        $l, s \leftarrow ChiSquaredTest(Y_i)$
        $est_i \leftarrow wcrtevt(b, l, s, P_e)$
      else
        if $b - 1 = lwb$ **then**
          $success \leftarrow true$
          $l, s \leftarrow ChiSquaredTest(Y_i)$
          $est_i \leftarrow wcrtevt(b, l, s, P_e)$
        else
          if $bdouble = 0$ **then**
            $upb \leftarrow b$
            $lwb \leftarrow 1$
            $b \leftarrow \frac{upb + lwb}{2}$
            $bdouble \leftarrow 1$
          else
            $upb \leftarrow b$
            $lwb \leftarrow \frac{b}{2}$
            $b \leftarrow \frac{upb + lwb}{2}$
          end if
        end if
      end if
    else
      $b \leftarrow 2b$
      $k \leftarrow \left\lfloor \frac{m}{b} \right\rfloor$
    end if
  end while
end for
$EST \leftarrow est_1, ..., est_i, ..., est_n$
if $passKS(EST, Normal)$ **then**
  $\overline{EST} \leftarrow \frac{1}{n} \times \sum_{i=1}^{n} est_i$
  $\sigma_{EST} \leftarrow \sqrt{\frac{1}{n} \sum_{i=1}^{n} (est_i - \overline{EST})^2}$
  $rt_{est} \leftarrow \overline{EST} + 3\sigma_{EST}$
else
  $rt_{est} \leftarrow bootstraptest(EST)$
end if
return rt_{est}