On Validation of Simulation Models in Timing Analysis of Complex Real-Time Embedded Systems

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

***Abstract*—In this paper, we present work toward validating simulation models extracted from complex real-time embedded systems, from the perspective of response time and execution time of adhering tasks, by using the non-parametric two-sample Kolmogorov-Smirnov test. Moreover, we introduce a method of reducing the number of samples used in the analysis, while keeping the accuracy of results. The evaluation using a fictive but representative system model inspired by a real robotic control system with a set of change scenarios, shows a promising result: the proposed algorithm has the potential of assessing whether the extracted simulation model is a sufficiently accurate approximation of the target system.**

I. Introduction

To date, many industrial embedded systems are very large, flexible, highly configurable software systems, containing millions of lines of code and consisting of hundreds of tasks, many with real-time constraints and being triggered in complex, nested patterns. Examples of such systems include the robotic control systems developed by ABB, as well as several telecom systems. Further, the temporal dependencies between tasks in such systems vary the execution time and response time of tasks radically. We refer to such systems as *Complex Real-Time Embedded Systems* (CRTES).

Simulation-based analysis of CRTES has the potential of not only allowing for response-time analysis of such systems [1], [2], but also facilitating migration toward a component based real-time system by e.g. analyzing the timing properties of the existing code and wrapping it into components. Moreover, simulation-based methods can also be used in timing impact analysis [3], i.e. to analyze the impact of changes on a system's temporal behavior, before introducing changes to the system. A major issue when using simulation-based timing analysis is how to obtain the necessary analysis model, which describes the original software program focusing on behavior of significance for task scheduling, communication and allocation of logical resources. For many systems, manual modeling would be far too time-consuming and error-prone. Two methods for automated model extraction are proposed in [4]. A tool for automated model extraction is in development, named MXTC - Model eXtraction Tool for C. The MXTC tool targets large implementations in C, consisting of millions of lines of code, and is based on program slicing. It is worth noting that the detailed procedure of model extraction is not in discussion in this paper. Further, the output of MXTC is simulation models for the RTSSim simulation framework [5].

However, there is one important issue to be raised, i.e. *model validity*, which is defined as *the process of determining whether a simulation model is an accurate representation of the system, for the particular objectives of the study* [6]. As a model is an abstraction of the system, some system details may be omitted in the model, for instance when using probabilistic execution time modeling. Thus, the results from a simulation of such models may not be identical to the recordings of the system, e.g. with regard to the exact task response time. In order to convince system experts to use simulation-based methods, the models should reflect the system with a satisfactory level of significance, i.e. as a sufficiently accurate approximation of the actual system. Moreover, other threats to model validity are the configuration of the model extraction tool and bugs in the model extraction and analysis tools. Therefore, an appropriate validation process has to be performed before using the models.

There are various methods in the field of model validation; these methods are either objective or subjective. Subjective methods are often used for validation of simulation models; examples of subjective methods are Face Validation, Graphical Comparisons and Sensitive Analysis [7], which are highly dependent on domain expertise and hence error-prone. In this paper, we present an objective method using a statistical approach for validation of temporal simulation models extracted from such CRTES, by considering this particular problem as a statistical problem, which can be solved by using existing, mature methods from the field of statistics. Furthermore, we examine the idea by using simulation models inspired by a real industrial robotic control system with three change scenarios as introduced in Section IV.

II. Model Validation

A. RTSSim Simulation Models

RTSSim simulation framework [5] is used to model and analyze our target CRTES in this work. It is quite similar to *ARTISST* [8] and *VirtualTime* [9]. 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 of RTSSim is Fixed-Priority Preemptive Scheduling (FPPS) and each task has scheduling attributes such as priority,

978-1-4244-6850-8/10/$26.00 ©2010 IEEE

period, offset and jitter. RTSSim allows for three types of selections which are directly controlled by simulator input data: Selection of execution times in `execute` statements; Selection of task jitter; Selection of task behaviors, depending on the system environment, e.g. random number of external events generated by sensors. In RTSSim, Monte Carlo simulation is realized by providing randomly generated (conforming to the uniform distribution) input data. A more thorough description of RTSSim can be found in [5].

B. Problem Formulation

We are given a simulation model $S^{'}$ which is extracted from a real system (or modeled system) S containing a task set Γ including n tasks, where $n \in \mathbb{N}$. Let $RT_{samples}(S^{'}, \tau_i)$, $RT_{samples}(S, \tau_i)$, $ET_{samples}(S^{'}, \tau_i)$ and $ET_{samples}(S, \tau_i)$ denote the sampling distributions of the response time and execution time measured for a task τ_i in $S^{'}$ and S respectively. The goal of the problem is then to find: whether there are statistically significant differences between the system and model distributions with respect to response times and execution times of the adhering tasks, or can they be considered statistically equal (i.e. from the same population).

C. Descriptive Statistics of Raw RT and ET Data

Table I shows the numerical summary of the center and the spread (or variability) of sampling distributions of the response time (RT) data of tasks in Model 1 (M1) containing intricate execution dependencies, used for the evaluation in Section IV. In Table I, *Std. Dev*, *Q1* and *Q3* represents *standard deviation*, *first quartile* and *third quartile* of the sampling distribution respectively. Further, the outliers existing in raw RT data as well as ET data of all tasks cannot be removed since they are not generated due to system errors or hardware failures. Therefore, we have the reasoning to add the *five-number summary* introduced in [10] consisting of *Min*, *Q1*, *Median*, *Q3* and *Max* to Table I. Due to limited space, we only show the histogram of the sampling distribution of raw RT data of one task i.e. the CTRL task (with the most complicated temporal behavior) when the number of samples is large enough i.e. 199 990 in one simulation run (refer to row *Samples* for the CTRL task in Table I), as an example shown in Figure 1. Further, note that the outliers in the picture might not be clear enough to see, though in fact, they approximately exist in the range of $[4\,600, 6\,954]$ along with the horizontal axis.

D. Dependencies between Raw RT and ET Data of Tasks

In our case, due to intricate task execution dependencies in the system, an upcoming RT data may not be independent with the RT data previously recorded at each simulation run (we refer to such RT and ET data as *raw RT and ET data*). The same problem applies for raw ET data. Secondly, in the conventional statistical procedure (*parametric test*), e.g. t-test, analysis of variance (ANOVA), one important assumption is that the underline population is assumed to follow a normal distribution. However, such assumption cannot be made since the sampling distribution of either raw RT data or raw ET data of all tasks is often conforming to a multimodal distribution having several peaks (consider Figure 1 as an example). Specifically, because of such distinctive feature of our target industrial control system, it is difficult to bring conventional statistical methods into the context. A new way of constructing the sampling distributions of tasks' RT and ET data has to be introduced, in order to fulfill the basic requirement given by *probability distribution*, i.e. the variable described by a probability distribution is a `random variable`, of which value is a function of the outcome of a statistical experiment that has outcomes of equal probability. We will present the proposed mechanism in the following Section III-A.

TABLE I
DESCRIPTIVE STATISTICS OF SAMPLING DISTRIBUTIONS OF RAW RT DATA OF TASKS IN THE SYSTEM MODEL M1 USED IN THE EVALUATION.

	DRIVE	*IO*	*CTRL*	*PLAN*
Samples	199994	400000	199990	199988
Mean	222.08	125.0	1967.3	2002.9
Std. Dev	14.291	45.576	390.09	412.46
Min	220	0	1074	332
Q1	220	100	1594	1631
Median	220	125	1919	1931
Q3	220	150	2339	2376
Max	420	250	6954	45957

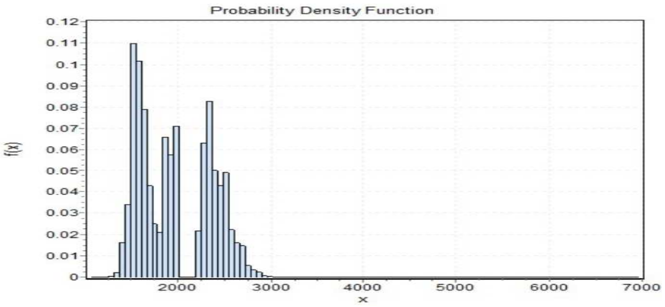

Fig. 1. The sampling distribution of raw RT data of the CTRL task in the evaluation model M1.

III. ALGORITHM

A. Reconstruction of New RT and ET Sampling Distribution

Firstly, 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) [10] is adopted. SRS gives every possible sample of a given size the same chance to be chosen. For instance, Monte Carlo simulation is used as a way of implementing SRS to collect sampling distributions of RT and ET data of tasks in the extracted RTSSim model. This is done by an embedded random number generator `rnd_inst()` in the RTSSim simulator (in lines 3, 5, 7 and 9 in Algorithm 1) which is an improved version of the Pseudo-random number generator used in C `rand()`. Moreover, empirical results showed that the distribution of random numbers given by `rnd_inst()` is conforming to the uniform distribution, which assures that for each selection in RTSSim input data, all possible values in any range are equally

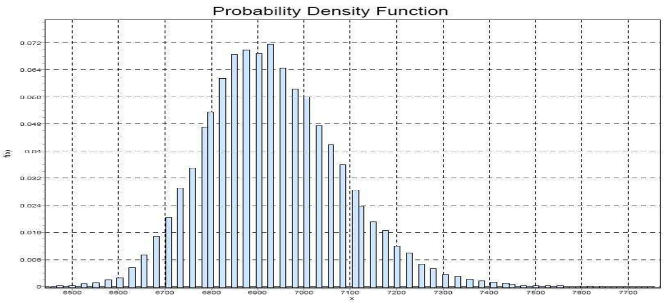

Fig. 2. A new reconstructed sampling distribution of RT data of the CTRL task in the evaluation model M1, by using maximum of each simulation run.

likely to be chosen. Analogously, the sampling distributions of RT and ET data of tasks in the real system can be collected based on measurements given a randomized system input. Some of the outliers (extreme values) which are caused, e.g. hardware failure or system errors, have to be removed from the sampling distributions.

Secondly, we propose a method by firstly running N Monte Carlo simulations conforming to SRS as introduced previously, with the purpose of eliminating the dependencies between raw RT and ET data of tasks due to intricate task temporal dependencies. Specifically, for each task in the task set Γ, the maxima of m samples RT data and m samples ET data recorded by each simulation, will be chosen to construct new sampling distributions of RT data and ET data. By doing this, the new reconstructed sampling distributions of RT and ET data of tasks can be considered from a random variable, since there are no dependencies between any maximum of RT and ET data of tasks between two independent simulations. Refer to Figure 2 as an example.

Nonetheless, as shown in Figure 2 clearly, the new reconstructed sampling distribution is *positive skewed* (i.e. the right tail is longer), in which the assumption required by the conventional statistical analyses i.e. normality of distributions cannot be satisfied. Hence, a parametric test cannot be reasonably applied in this work, we thereby consider using the two sample Kolmogorov-Smirnov (hereafter KS test) which is a non-parametric statistics making no assumptions on the underline population of a sampling distribution.

B. StatiVal

The proposed method, *StatiVal*, is shown in Algorithm 1. The algorithm returns the result concerning if there exists a statistically significant difference between the modeled system S and the simulation model $S^{'}$, in the view of system timing properties such as tasks' response time and execution time. Further, in this work, since we cannot perform the validation between the real modeled system and the extracted model, we will instead compare a system model S inspired by a real industrial robotic control system (considered as the modeled system) with a set of variations $S^{'}$ where a specific change scenario (as shown in Table IV) is applied to S. Both of S and $S^{'}$, are in this case simulation models, analyzed using Monte Carlo simulation which in Algorithm 1 is modeled as a function, *MTC*, with four parameters: m - the number of samples drawn from each simulation trace, τ_k - the task on focus in KS test, *Property* - either RT or ET of the task τ_k and *rnd_inst*() - a random number generator in RTSSim simulator. The outline of StatiVal is as follows:

1) Construct the sampling distribution of N RT and ET data of all the tasks in both the system S and the model $S^{'}$ by Monte Carlo simulation `MTC()` respectively (refer to lines 1 to 16 in Algorithm 1).
2) Use KS test to compare if sampling distributions of RT and ET data of each task τ_k in the task set Γ in both S and $S^{'}$ are statistically significant iteratively. If any of such results is H_0, then Algorithm 1 draws the conclusion H_0, i.e., *the model $S^{'}$ is not a sufficiently accurate approximation of the system S due to an improper model extraction process*, and finally, stops the validation process; Otherwise, the entire validation process will terminate after all the tasks are evaluated by KS test (refer to lines 18 to 33 in Algorithm 1). Note that the hypotheses used in this work are opposite to the ones in the traditional hypothesis test, the reasoning can be found in [11]. In practice, KS test is conducted by using a commercial software *XLSTAT*, which is a plug-in to EXCEL.

Algorithm 1 $StatiVal(\Gamma)$

```
 1: for all $\tau_k$ such that $1 \le k \le n$ in $\Gamma$ in both $S$ and $S^{'}$ do
 2:     for all $i$ such that $1 \le i \le N$ do
 3:         $X_i \leftarrow x_{i,1}, ..., x_{i,j}, ..., x_{i,m} \leftarrow MTC(m, \tau_k, RT, rnd\_inst())$
 4:         $X_{\tau_k,i} \leftarrow Max(X_i)$
 5:         $Y_i \leftarrow y_{i,1}, ..., y_{i,j}, ..., y_{i,m} \leftarrow MTC(m, \tau_k, ET, rnd\_inst())$
 6:         $Y_{\tau_k,i} \leftarrow Max(Y_i)$
 7:         $X^{'}_i \leftarrow x^{'}_{i,1}, ..., x^{'}_{i,j}, ..., x^{'}_{i,m} \leftarrow MTC(m, \tau_k, RT, rnd\_inst())$
 8:         $X^{'}_{\tau_k,i} \leftarrow Max(X^{'}_i)$
 9:         $Y^{'}_i \leftarrow y^{'}_{i,1}, ..., y^{'}_{i,j}, ..., y^{'}_{i,m} \leftarrow MTC(m, \tau_k, ET, rnd\_inst())$
10:         $Y^{'}_{\tau_k,i} \leftarrow Max(Y^{'}_i)$
11:     end for
12:     $X_{\tau_k} \leftarrow X_{\tau_k,1}, ..., X_{\tau_k,i}, ..., X_{\tau_k,N}$
13:     $Y_{\tau_k} \leftarrow Y_{\tau_k,1}, ..., Y_{\tau_k,i}, ..., Y_{\tau_k,N}$
14:     $X^{'}_{\tau_k} \leftarrow X^{'}_{\tau_k,1}, ..., X^{'}_{\tau_k,i}, ..., X^{'}_{\tau_k,N}$
15:     $Y^{'}_{\tau_k} \leftarrow Y^{'}_{\tau_k,1}, ..., Y^{'}_{\tau_k,i}, ..., Y^{'}_{\tau_k,N}$
16: end for
17: $ret \leftarrow 0$
18: for all $\tau_k$ such that $1 \le k \le n$ in $\Gamma$ in both $S$ and $S^{'}$ do
19:     $ret \leftarrow kstest(X_{\tau_k}, X_{\tau^{'}_k}, \alpha)$
20:     if $ret = H_a$ then
21:         $ret \leftarrow H_a$
22:     else
23:         $ret \leftarrow H_0$
24:         return $ret$
25:     end if
26:     $ret \leftarrow kstest(Y_{\tau_k}, Y_{\tau^{'}_k}, \alpha)$
27:     if $ret = H_a$ then
28:         $ret \leftarrow H_a$
29:     else
30:         $ret \leftarrow H_0$
31:         return $ret$
32:     end if
33: end for
34: return $ret$
```

C. A Method of Reducing Sample Size N

In [12], the rational way to choose a sample size is introduced by weighing the *benefits* in information against the *cost* of increasing the sample size. In our context, the benefit is to obtain the correct validation results given by the proposed method, and the cost is the number of simulations N in the analysis. We will illustrate the idea by referring to a concrete example shown in Table II using Case 3 in Table IV. According to our reasoning, when N is equal to 20 000, StatiVal can give the correct result H_0 as shown at Step 1 in Table II. In Column *Accuracy* in Table II, √ represents *the result given by StatiVal is correct when N is equal to the certain value*, while × denotes the opposite situation. Further, we decrease the number of N by four times, according to the important rule of thumb: To cut the error in half, the sample size must be *quadruple*. Therefore, at Step 2, N is set to 5 000 (i.e. $\lceil \frac{20\,000}{4} \rceil$). The results given by StatiVal are not wrong until when N is equal to 79 at Step 5. Consequently, the value of N can be safely reduced to 313 at Step 4, meanwhile keeping the accuracy. It is worth noting that the value of N could be further optimized by using for instance a lower-part binary search algorithm.

TABLE II
Illustration of reducing the number of samples required by the proposed algorithm.

Step	*N*	*Accuracy*	*Step*	*N*	*Accuracy*
1	*20000*	√	*4*	***313***	√
2	*5000*	√	*5*	*79*	×
3	*1250*	√	*6*	*20*	×

IV. Evaluation

In this work, we examine the idea by using a simulation model Model 1 (M1) describing a fictive, representative industrial robotic control system developed by ABB. It is designed to include some behavioral mechanisms from the ABB system: 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, which vary the execution time of tasks dramatically; 3) Although FPPS is used as base, one task, i.e. the CTRL task, changes its priority during runtime, in response to system events. Further, the task model is presented in Table III, where time unit is one *simulation time unit* (tu). The details of the model are described in [5].

TABLE III
Tasks and task parameters for M1. The lower numbered priority is more significant, i.e. 0 stands for the highest priority.

Task	Period (tu)	Offset (tu)	Priority
DRIVE	2000	12000	2
IO	5000	500	5
CTRL	10000 or 20000	0	6 or 4
PLAN	40000	0	8

The RT and ET data of tasks produced by the original simulation model M1 is used as reference, for comparing the impact of a set of *change scenarios* outlined in Column *Changes Description* in Table IV. Moreover, for Case 3, there is a DUMMY task added to the model S' with the certain priority, execution time and period (denoted as C and T in Table IV). Finally, we compare the outputs against the original model to investigate the performance of the method. Further, the number of samples used in StatiVal is obtained after optimization, i.e. 313 samples for both RT and ET sampling distributions. The results given by StatiVal and the *expected results* (ER) are shown in Table IV. More importantly, our evaluation shows a promising result, i.e. the proposed algorithm can correctly identify temporal differences between different evaluation models, hence it has the potential of being used a model validation technique by showing the evidence whether the extracted simulation model is a sufficiently accurate approximation of the target system.

TABLE IV
Results obtained by using StatiVal concerning different models according to change scenarios.

Change Scenarios	Changes Description	RT	ET	StatiVal	ER
Case 1	IO: C 23 → 46	H_0	H_0	H_0	H_0
Case 2	PLAN: Prio 8 → 9	H_a	H_a	H_a	H_a
Case 3	DUMMY: Prio = 7, T = 5 000, C = 25	H_0	H_a	H_a	H_0

V. Future Work

This paper has presented our ongoing work on validation of temporal simulation models extracted from complex real-time embedded systems. In particular, our evaluation showed that the proposed method has the potential to identify temporal differences between the modeled system and the extracted simulation models. As part of future work, an effort will be spent on evaluating more scenario changes on the evaluation model. Moreover, we will evaluate the method on real systems.

References

[1] 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*, April 2008, pp. 258–269.

[2] M. Bohlin, Y. Lu, J. Kraft, P. Kreuger, and T. Nolte, "Simulation-based timing analysis of complex real-time systems," in *RTCSA 09*, August 2009, pp. 321–328.

[3] J. Andersson, J. Huselius, C. Norström, and A. Wall, "Extracting simulation models from complex embedded real-time systems," in *Procs. of the Int. Conf. ICSEA'06*. IEEE, 2006.

[4] J. Kraft, J. Huselius, A. Wall, and C. Norström, "Extracting simulation models from complex embedded real-time systems," in *Real-Time in Sweden 2007*, August 2007.

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

[6] A. M. Law, "How to build valid and credible simulation models," in *WSC '08*. Winter Simulation Conference, 2008, pp. 39–47.

[7] O. Balci, "How to assess the acceptability and credibility of simulation results," in *WSC '89*. New York, NY, USA: ACM, 1989, pp. 62–71.

[8] 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, pp. 365–372.

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

[10] D. S. Moore, G. P. Mccabe, and B. A. Craig, *Introduction to the practice of statistics*, 6th ed. New York, NY 10010: W. H. Freeman and Company, 2009.

[11] A. Robinson and R. Froese, "Model validation using equivalence tests," *Elsevier, ScienceDirect 2004*, vol. 176, pp. 349–358, 2004.

[12] R. Schlaifer, *Applied statistical decision theory*. Wiley-Interscience, 1961.