A Statistical Approach to Simulation Model Validation in Response-Time Analysis of Complex Real-Time Embedded Systems

Yue Lu[1], Johan Kraft[1], Thomas Nolte[1], and Iain Bate[2]
[1]Mälardalen Real-Time Research Centre (MRTC), Västerås, Sweden
[2]Department of Computer Science, University of York, York, YO10 5DD
{yue.lu, johan.kraft, thomas.nolte}@mdh.se, iain.bate@cs.york.ac.uk

ABSTRACT

As simulation-based analysis methods make few restrictions on the system design and scale to very large and complex systems, they are widely used in, e.g., timing analysis of complex real-time embedded systems (CRTES) in industrial circles. However, before such methods are used, the analysis simulation models have to be validated in order to assess if they represent the actual system or not, which also matters to the confidence in the simulation results. This paper presents a statistical approach to validation of temporal simulation models extracted from CRTES, by introducing existing mature statistical hypothesis tests to the context. Moreover, our evaluation using simulation models depicting a fictive but representative industrial robotic control system indicates that the proposed method can successfully identify temporal differences between different simulation models, hence it has the potential to be considered as an effective simulation model validation technique.

Categories and Subject Descriptors

I.6.4 [**Simulation and modeling**]: Model validation and analysis; D.2.4 [**Software engineering**]: Software/program verification: Statistical methods; C.3 [**Special-purpose and application-based system**]: Real-time and embedded systems

Keywords

simulation model validation, response time, complex real-time embedded systems, non-parametric statistical hypothesis testing, two-sample Kolmogorov-Smirnov test

1. INTRODUCTION

Many of today's industrial embedded systems are large, flexible and highly configurable software systems which contain many event-triggered tasks being triggered by other tasks in complex, nested patterns, resulting in very complicated timing behavior. Furthermore, such systems often consist of millions of lines of code, and contain hundreds of tasks, many with real-time constraints. Examples of such systems are the robotic control system *IRC* 5 developed by ABB [1], as well as several telecom systems. More importantly, many tasks in such systems have intricate temporal dependencies, which may decide important control flow conditions with major impact on task execution time as well as task response time. We refer to the systems with such characteristics as *Complex Real-Time Embedded Systems* (CRTES).

Simulation-based timing analysis methods have expanded both in terms of Response-Time Analysis (RTA) for more complex systems [13, 5] and how the results are subsequently used, e.g., by analyzing the timing properties of the existing code and wrapping it into components, which facilitate migration towards a component-based real-time system. Simulation-based methods provide a powerful augmentation to RTA as they allow the user to analyze the impact of changes on a system's temporal behavior, before introducing changes to the system, which is referred to as *timing impact analysis* [2].

A major issue when using simulation-based timing analysis methods is *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* [11]. 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. Furthermore, 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 should be performed before using the models.

There is a large body of work having been done in the realm of simulation model validation; these methods are either objective or subjective. Examples of subjective methods are *Face Validation* and *Graphical Comparisons* [4], which are highly dependent on domain expertise and hence error-prone. The key contributions of the paper are to pro-

SAC'11 March 21-25, 2011, TaiChung, Taiwan.

vide a means of evaluating the validity in the context of RTA of CRTES, by considering this particular problem as a statistical problem, which could be solved by using existing, mature methods from the field of statistics and evaluate these using a fictive but representative industrial robotic control system.

Organization: Section 2 first introduces a simulation framework for modeling and timing analysis of CRTES, and then gives descriptive statistics of original RT data of tasks and problem formulation. Next, Section 3 first presents a mechanism to eliminate dependencies existing in original RT data, and then introduces the non-traditional hypothesis test and a non-parametric hypothesis test used in this paper, and finally, positions our proposed method *StatiVal*. The method evaluation and the related work appear in Section 4 and 5 respectively, before conclusions are drawn in Section 6.

2. SIMULATION OF CRTES

This section is split into three parts: Section 2.1 presents a simulation framework for modeling and timing analysis of CRTES, Section 2.2 introduces the descriptive statistics of tasks' original response time data in the evaluation models, and finally, Section 2.3 gives the problem definition.

2.1 The Simulation Framework RTSSim

The target CRTES are described by the modeling language in RTSSim simulation framework [10], which is quite similar to *ARTISST* [7] and *VirtualTime* [17], and allows for simulating system models containing detailed intricate execution dependencies between tasks, such as asynchronous message-passing, globally shared state variables, and runtime changeability of priority and period of tasks. In RTSSim, the system consists of a set of tasks, sharing a single processor. RTSSim provides typical RTOS services to tasks simulation model, such as Fixed Priority Preemptive Scheduling (FPPS), Inter-Process Communication (IPC) via message queues and synchronization (semaphores). The tasks in a model are described using C functions, which are called by the RTSSim framework. The framework provides an isolated "sandbox", where time is represented in a discrete manner using an integer simulation clock, which is only advanced explicitly by the tasks in the simulation model, using a special routine, `EXECUTE`. Calls to this routine models the tasks' consumption of CPU time.

All time-related operations in RTSSim, such as timeouts and activation of time-triggered tasks, are driven by the simulation clock, which makes the simulation result independent of process scheduling and performance of the analysis PC. The response time of tasks are measured whenever the scheduler is invoked, which happens e.g., at IPC, task switches, `EXECUTE` statements, operations on semaphores, task activations and when tasks end. This, together with the simulation clock behavior, guarantees that the measured response time is exact.

In RTSSim, a task may not be released for execution until a certain non-negative time (the offset) has elapsed after the arrival of the activating event. Each task also has a period, a maximum arrival jitter, and a priority. Periods and priorities can be changed at any time by any task in the application, and offset and jitter can both be larger than the period. Tasks with equal priorities are served on a first come first served basis. The framework allows for three types of selections which are directly controlled by simulator input data: 1) selection of execution times (for `EXECUTE`), 2) selection of task-arrival jitter, and 3) selection of task control flow, directly or indirectly based on environmental input stimulus. Monte Carlo simulation can be realized by providing randomly generated (conforming to the uniform distribution) simulator input data, and gives output in terms of a set of measured RT data of each task invocation during one simulation run. Furthermore, the evaluation model Model 1 in Section 4.2 is manually designed to contain similar modeling and analysis challenges as a real industrial robotic control system developed by ABB, i.e., intricate temporal dependencies between tasks. More importantly, sampling distributions of original response time of adhering tasks exhibit a distinctive characteristic (to be introduced in Section 2.2), which makes conventional statistical analyses difficult to analyze.

2.2 Descriptive Statistics of Original RT Data

Due to the existence of intricate task execution dependencies in the evaluation model *Model 1*, an upcoming RT datum may not be independent with the RT datum previously recorded at each simulation run. We refer to such RT data as *original RT data of tasks* hereafter. More importantly, *outliers* existing in original RT data of all tasks cannot be removed since they are not caused by system errors or hardware failures. The definition of such *outliers* referred in this work is introduced as follows: A RT datum beyond an *outer fence* [16] is considered as an *(extreme) outlier*. For the sake of space, Table 1 only shows the numerical summary of the center and the spread (or variability) of the original RT data sampling distribution of the CTRL task (with most complicated temporal behavior) in Model 1, where *X*, *Q1* and *Q3* represent *response time*, *first quartile* and *third quartile* of the sampling distribution respectively. In addition, by using the definitions for determining outliers given in [16], i.e., $Q3 + 3 \times IQ$ where $IQ = Q3 - Q1$, the corresponding outer fence for the CTRL task is found to be $4\,574$ (i.e., $2\,339 + 3 \times (2\,339 - 1\,594)$) *simulation time units (tu)*. Due to the presence of outliers, we consider using the *five-number summary* introduced in [15] consisting of *Min*, *Q1*, *Median*, *Q3* and *Max* in Table 1, in order to give the overall statistic descriptive of the sampling distribution of the original RT data of the CTRL task in Model 1. Furthermore, Figure 1 shows the *probability density function* (PDF) histogram of the original RT data sampling distribution of the CTRL task when the number of samples is large enough, i.e., 199 990 which is corresponding to execute one simulation run for the time up to the upper bound of the RTSSim simulation time, i.e., $2^{31} - 1$. Note that the outliers in Figure 1 might not be clear enough to see, though in fact, they approximately exist in the range of $[4\,574, 6\,829]$ along with the horizontal axis.

Table 1: Descriptive statistics of the sampling distribution of the original RT data of the CTRL task in Model 1 for one simulation run. The time unit is one tu.

	Std. Dev	Min	Q1	Median	Q3	Max	UOF
X_{CTRL}	389.98	1 024	1 594	1 919	2 339	6 829	**4 574**

In the conventional statistical procedure (i.e., *parametric test*), e.g., t-test and analysis of variance (ANOVA) [20], one important assumption is that the underline population

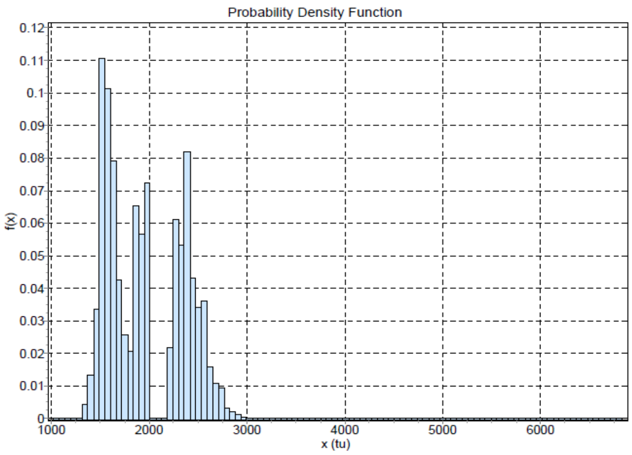

Figure 1: The probability density function (PDF) histogram of the original RT data sampling distribution of the CTRL task in Model 1.

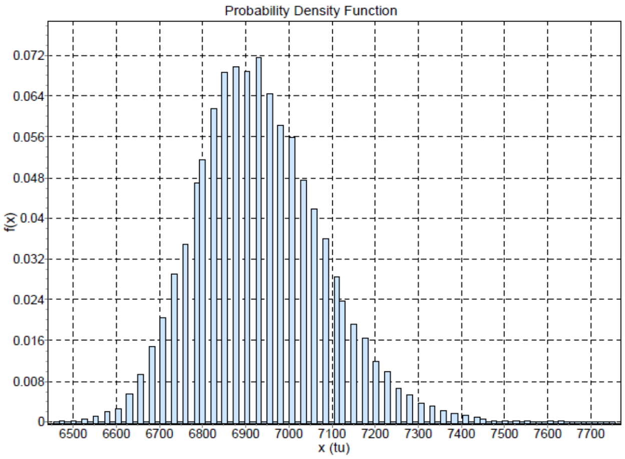

Figure 2: A new reconstructed sampling distribution of RT data of the CTRL task in Model 1.

is assumed to follow a normal distribution. However, such an assumption cannot be made in our case, since the sampling distribution of the original RT data of all tasks is often clearly conforming to a multimodal distribution having several peaks (refer to Figure 1 as an example). Such conventional statistical methods cannot be brought into the context thereof. In addition, the original RT data of tasks noted as above in Section 2.2 do not fulfill the basic requirement given by any statistics using *probability distribution*: 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. Consequently, a new way of constructing the qualified sampling distributions of tasks' RT data has to be invented.

2.3 Problem Formulation

We are given two RTSSim simulation models S and $S^{'}$ which represent the target system and the extracted simulation model respectively. Further, each model contains a task set Γ which has the same number of tasks, i.e., n, where $n \in \mathbb{N}$. Let X_{τ_k} and $X^{'}_{\tau_k}$ denote the sampling distributions drawn from underline populations of RT data of the same task τ_k in both S and $S^{'}$ separately, where $1 \leq k \leq n$. The goal of the problem is then to find: whether each paired X_{τ_k} and $X^{'}_{\tau_k}$ is significantly different, or can they be considered statistically equal (i.e., from the same population), at the certain significance level [15]. More typically, in this work, the significance level of a test is such that the probability of mistakenly rejecting the null hypothesis is no more than the stated probability, i.e., $\alpha = 0.05$ which is a typical value and based on preliminary assessments provides appropriate results [19].

3. THE ALGORITHM

This section first shows how to build up new sampling distributions of RT data of tasks in CRTES, by using our proposed method. Next, Section 3.2, 3.3 and 3.4 introduce problems with using conventional parametric statistics in analyzing the new RT data sampling distributions, the feasibility of applying different non-parametric statistical hypothesis tests to our case, and the non-traditional hypotheses used in this work, respectively. Finally, Section 3.5 describes our proposed algorithm.

3.1 New RT Sampling Distributions

A key issue of selecting samples from the population of all individuals concerning the desired information, is to eliminate bias on the sampling. We therefore propose to use the technique of simple random samples (SRS) [15], which gives every possible sample of a given size the same chance to be chosen. In this work, Monte Carlo simulation is used as a way of implementing SRS to collect sampling distributions of RT data of tasks in the extracted RTSSim model. This is done by an embedded random number generator `rnd_inst()` in the RTSSim simulator, which is an improved version of the Pseudo-random number generator used in C. 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 likely to be chosen.

In order to eliminate dependencies between original RT data of tasks caused by intricate task temporal dependencies, we propose a method by first running N Monte Carlo simulations conforming to SRS noted as above. For each task in the task set Γ, the highest value of m samples RT data recorded by each simulation, will be chosen to construct new sampling distributions of RT data. By doing this, the new constructed sampling distributions of RT data of tasks can be considered from a random variable, since there are no dependencies between any maximum value of RT data of tasks between two independent simulations. Refer to Figure 2 as an example.

Analogously, the sampling distributions of RT data of tasks in the real system can be collected based on measurements by randomizing inputs to the system at first, and then removing outliers from the sampling data that are caused by hardware failure or system errors during each system runtime observation, and finally, choosing the highest value of RT data of tasks in the system. Nevertheless, because such activity is also application-specific, we therefore will not discuss it in details in this work, due to limited space.

3.2 Problems with Using Parametric Statistics

In order to determine if the conventional statistical procedure (*parametric test*), e.g., t-test and ANOVA, can be used to infer valid parameters of tasks' RT data sampling distributions, it first needs to be checked whether the values conform to a normal distribution. In this work, it is done by using a commercial statistic analysis software *EasyFit* [8], according to the results given by a Goodness of Fit (GOF) test, i.e., Chi-squared test [6] at α-value of 0.05. The results show that the new sampling distributions do not conform to any of the 65 known distributions in [8], e.g., Normal, Uniform, Student's t, Lognormal etc. Since parametric tests cannot be reasonably applied in this work, we thereby consider to use non-parametric hypothesis tests which make no assumptions on the underline population of a sampling distribution.

3.3 The Two-sample KS Test

In the domain of non-parametric statistical hypothesis tests, there are a few methods such as *Chi-squared test*, *Wilcoxon-Mann-Whithney test* [21], *Kolmogorov-Smirnov test* (χ^2 test, WMW test and KS test respectively, hereafter), which are often used in identifying differences between two sampling distributions. The χ^2 test, in which the expected frequencies of samples in the underline population are compared to the observed frequencies of samples in the sampling distribution. Such expected frequencies are either hypothesized as all equal or on the basis to some priori knowledge or experience. However, in the case of RTA for CRTES, it is either too subjective to support such a hypothesis, or not accurate enough due to limited a priori knowledge of the tasks. The χ^2 test is not feasible to be applied in this work thereof. Concerning the WMW test, it disallows to compare for instance the variability of two sampling distributions, as it in fact compares the ranks of two samples by computing the ranks of two samples in the grouped sampling distribution. Therefore, the WMW test may fail in the comparing two sampling distributions taken from two multimodal distributions which are with identical mean but different variances: the WMW test will draw a conclusion that they are the same, but actually they are not. So the WMW test cannot be applied to our context neither. The KS test uses the maximum vertical deviation between the two *cumulative fraction plot* (CFP) curves[1] as the statistic D, to which the corresponding P value will suggest that if there is a significant difference or not. Hence, the KS test does not have the above issues raised by the χ^2 test and the WMW test, and it is also widely used in both academia research and industrial application. We therefore adopt the KS test at the confidence level 95% which is corresponding to the significance level $\alpha = 0.05$ (i.e., a typical value and based on preliminary assessments provides appropriate results [19]) in this work, and the corresponding hypotheses are as follows:

- $H_{0,ks}$: *There is no significant difference between the target system and the extracted model in the view of response time distributions of the task on focus.*

- $H_{a,ks}$: *There is a significant difference between the target system and the extracted model in the view of response time distributions of the task on focus.*

3.4 The Non-traditional Hypothesis Test

In [14] and [12], it is recognized that the application of traditional hypothesis tests is not appropriate for model validation. The reason is that traditional *null* hypothesis, i.e., *there is no difference between the means, for instance, of the populations in perspective of the same interesting property*, is tantamount to saying that the model meets the accuracy standard. Correspondingly, the burden of proof rests on the alternative hypothesis, i.e., the model is not acceptable. Moreover, this test strategy is also unsatisfactory because failure to reject the null hypothesis could be due to the model being acceptable, but it can also be interpreted as the user merely having chosen a test with lower power [18]. Therefore, we will use different hypotheses against the ones in the traditional hypothesis tests, i.e., the traditional *null* hypothesis should be reversed. The hypotheses used in this work can be formally introduced as follows:

- H_0: *The simulation model is not a sufficiently accurate approximation of the target system at the significance level* $\alpha = 0.05$, *from the perspective of response time distributions of adhering tasks.*

- H_a: *The simulation model is a sufficiently accurate approximation of the target system at the significance level* $\alpha = 0.05$, *from the perspective of response time distributions of adhering tasks.*

3.5 StatiVal

Our proposed method *StatiVal* is shown in Algorithm 1, which returns the simulation model validation results in terms of the hypotheses introduced in Section 3.4. Furthermore, in this work, since we choose not to perform the validation between the real system and the extracted model, we will instead compare an original system model S inspired by a real industrial robotic control system (considered as the modeled system) with a set of models $S^{'}$ where a specific change scenario (as shown in Section 4.2) is applied. Hence, in this case, both S and $S^{'}$ are simulation models being analyzed by 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 run, τ_k - the task on focus in the KS test, *Property* - task response time and *rnd_inst*() - a random number generator in RTSSim simulator. When the reference for comparison is a real system, the RT data sampling distribution is collected in the way as introduced in Section 3.1, which in Algorithm 1 is modeled by the function *measurement*, with four parameters: m - the number of samples drawn from each execution of the target system, τ_k - the task on focus in the analysis, *Property* - task response time and *rnd_testvector*() - a random test vector generator. The outline of StatiVal is as follows:

- Construct a sampling distribution of N RT data of all the tasks in both the actual system S and the model $S^{'}$ by using *measurement()* and Monte Carlo simulation *MTC()* respectively (refer to lines 1 to 10 in Algorithm 1).

- Use the KS test to compare if sampling distributions of RT data of each task τ_k in the task set Γ in both S and $S^{'}$ are statistically significantly different iteratively. If the result given by the KS test is $H_{a,ks}$, then

[1] CFP curves are graphical display of how the data in each sampling distribution is distributed.

Algorithm 1 draws the conclusion, i.e., *the model $S^{'}$ is not a sufficiently accurate approximation of the system S due to an improper model extraction process* (in other words, we should not reject the null hypothesis H_0 as introduced in Section 3.4), and finally, stops the validation process. Otherwise, the entire validation process will terminate after all the tasks are evaluated by the KS test (refer to lines 12 to 21 in Algorithm 1). In practice, the KS test is conducted by using a commercial software *XLSTAT* [22], which is a plug-in to EXCEL.

Algorithm 1 $StatiVal(\Gamma)$

```
 1: for all $\tau_k$ such that $1 \leq k \leq n$ in $\Gamma$ in both $S$ and $S^{'}$ do
 2:     for all $i$ such that $1 \leq i \leq N$ 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:         $X_i^{'} \leftarrow x_{i,1}^{'}, ..., x_{i,j}^{'}, ..., x_{i,m}^{'} \leftarrow$
            $Measurement(m, \tau_k, RT, rnd\_testvector())$
 6:         $X_{\tau_k,i}^{'} \leftarrow Max(X_i^{'})$
 7:     end for
 8:     $X_{\tau_k} \leftarrow X_{\tau_k,1}, ..., X_{\tau_k,i}, ..., X_{\tau_k,N}$
 9:     $X_{\tau_k}^{'} \leftarrow X_{\tau_k,1}^{'}, ..., X_{\tau_k,i}^{'}, ..., X_{\tau_k,N}^{'}$
10: end for
11: $ret \leftarrow 0$
12: for all $\tau_k$ such that $1 \leq k \leq n$ in $\Gamma$ in both $S$ and $S^{'}$ do
13:     $ret \leftarrow kstest(X_{\tau_k}, X_{\tau_k^{'}}, \alpha)$
14:     if $ret = H_{a,ks}$ then
15:         $ret \leftarrow H_0$
16:     else
17:         $ret \leftarrow H_a$
18:         return $ret$
19:     end if
20: end for
21: return $ret$
```

4. EVALUATION

In this section, we first introduce the evaluation models inspired by a real industrial control system, and our testbed. Then we give the description of *change scenarios* and the expected model validation results, and finally, we show that our proposed algorithm can obtain the accurate analysis results.

4.1 The Evaluation Models and Testbed

We examine the idea by using a set of simulation models which are 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 a basis, one task, i.e., the CTRL task, changes its priority during runtime, in response to system events. The details of Model 1 are described in [10]. Furthermore, the tasks and task parameters in both Model 1 and a set of variations of Model 1 are presented in Table 2, where *PLAN_H*, *CTRL_H*, *CTRL_L*, *DUMMY*, *PLAN_O* and *PLAN_L* represent the PLAN task with the priority 3, the CTRL task with the priority 4, the CTRL task with the priority 6, the DUMMY task with the priority 7, and the PLAN task with the priority 9 respectively. Note that the lower numbered priority is more significant, i.e., 0 stands for the highest priority. The time unit in Table 2 is a *simulation time unit* (tu) and Column *Case* denotes that in which case or cases a specific task exists. For example, the DUMMY task only exists in Cases 4-1 and 4-2. "−" represents the task appears in all cases. Moreover, 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.

Table 2: Tasks and task parameters in evaluation models. The lowest number stands for the highest priority.

	Priority	Period (tu)	Offset (tu)	Case
DRIVE	2	2 000	12 000	-
PLAN_H	3	40 000	0	2-2
CTRL_H	4	20 000	0	-
IO	5	5 000	500	-
CTRL_L	6	10 000	0	-
DUMMY	7	5 000	0	4-1,4-2
PLAN_O	8	40 000	0	-
PLAN_L	9	40 000	0	2-1

4.2 Change Scenarios and Evaluation Results

Model 1 is an original simulation model and is considered as the target system S, while a set of variations $S^{'}$ (in which either tasks' execution time, priority and period are changed, or some extra tasks are added) are considered as the extracted models from the system S. In addition, the description of change scenarios and the corresponding expected model validation results which are expressed by using the hypotheses introduced in Section 3.4, are presented as follows:

- Case 1: The execution time of the IO task is doubled from 23 to 46 *tu*. Moreover, because the IO task has a higher priority than the CTRL and PLAN task, therefore the RT of the two tasks as well as the IO task itself will be changed. H_0 is the expected result thereof.

- Case 2: In Case 2-1 where the priority of the PLAN task is lowered, the RT of other tasks will not be impacted because the priority of the PLAN task is the lowest among all tasks in the model. Consequently, the expected result is H_a. However, concerning Case 2-2 where the priority of the PLAN task is prompted to be higher than any other tasks except for the DRIVE task, the RT of such tasks will be changed. The corresponding expected result is H_0 thereof.

- Case 3: When the period of the PLAN task which is the lowest priority task in the system is increased, the RT of all the tasks is not supposed to be affected at all, therefore the expected result is H_a.

- Case 4: When there is a DUMMY task added to Model 1, with different execution times and priorities that are lower than any other tasks, except for the PLAN task in the system. Consequently, only RT of the PLAN task might be affected after the change. Therefore H_0 is expected to be drawn.

As shown in Rows *StatiVal*, *ER*, *Accuracy*[2] and *Confidence level* in Table 3, clearly, the results given by our proposed method StatiVal at the confidence level 95% are in line with the expected model validation results. This indicates that our proposed method has the potential to be considered as an effective simulation model validation method. It is interesting to note that the reasons for why other parametric and non-parametric statistics cannot be applied to the context of model validation for CRTES have been outlined in Section 3.2 and Section 3.3 respectively.

Table 3: The results obtained by using StatiVal concerning different models according to change scenarios, and the expected model validation results.

Cases	1	2-1	2-2	3	4-1	4-2
RT_{DRIVE}	$H_{0,ks}$	$H_{0,ks}$	$H_{a,ks}$	$H_{0,ks}$	$H_{0,ks}$	$H_{0,ks}$
RT_{IO}	$H_{a,ks}$	$H_{0,ks}$	$H_{a,ks}$	$H_{0,ks}$	$H_{0,ks}$	$H_{0,ks}$
RT_{CTRL}	$H_{a,ks}$	$H_{0,ks}$	$H_{a,ks}$	$H_{0,ks}$	$H_{0,ks}$	$H_{0,ks}$
RT_{PLAN}	$H_{a,ks}$	$H_{0,ks}$	$H_{a,ks}$	$H_{0,ks}$	$H_{a,ks}$	$H_{a,ks}$
StatiVal	H_0	H_a	H_0	H_a	H_0	H_0
ER	H_0	H_a	H_0	H_a	H_0	H_0
Accuracy	✓	✓	✓	✓	✓	✓
Confidence level	95%	95%	95%	95%	95%	95%

5. RELATED WORK

This section reviews the work that are not mentioned in previous sections, but related. In [3] Andersson presents the notion of model equivalence based on observable property equivalence which is used to compare results of a model and an actual system. Kleijnen [9] presents work about validation on trace-driven simulation models using bootstrap test on simulation sub-runs. However, it seems that there are no outliers existing in the sampling distributions used in their analysis. Moreover, the system and simulation models in their work are much less complex than the system models we are using, of which samples distributions used in the analysis are multi-model distributions, due to the adhering intricate task temporal dependencies. The key innovation that equivalence testing [18] relies upon is the subjective choice of a region within which differences between test and reference data are considered negligible. For example, a region of indifference might be nominated to be ±20% of the standard deviation, which introduces a measure of subjectivity and hence cannot be reasonably applied in our case where there are many outliers existing in the samples.

6. CONCLUSIONS AND FUTURE WORK

Simulation model validation is a vital issue when simulation-based methods are used in the timing analysis of Complex Real-Time Embedded Systems (CRTES). This paper has presented our work on validation of temporal simulation models from the perspective of tasks' response time. In particular, the evaluation using a number of simulation models describing a fictive but representative industrial robotic control system, shows that our proposed method can obtain the correct analysis results. This also indicates that our proposed method has the potential to be applied as an effective simulation model validation method. Further work will investigate ways in which to extend our method with the ability to evaluate tasks' execution time, as well as to evaluate the method by using more scenario changes and real systems.

7. REFERENCES

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

[2] J. Andersson, J. Huselius, C. Norström, and A. Wall. Extracting Simulation Models from Complex Embedded Real-Time Systems. In *Proc. of the ICSEA' 06*, pages 7–16, French Polynesia, Oct 2006.

[3] J. Andersson, A. Wall, and C. Norström. Validating Temporal Behavior Models of Complex Real-Time Systems. In *Proc. of the SERPS' 04*, Sep 2004.

[4] O. Balci. How to assess the acceptability and credibility of simulation results. pages 62–71, New York, NY, USA, 1989. ACM.

[5] M. Bohlin, Y. Lu, J. Kraft, P. Kreuger, and T. Nolte. Simulation-Based Timing Analysis of Complex Real-Time Systems. In *Proc. of the RTCSA' 09*, pages 321–328, Aug 2009.

[6] Chi-squared test, www.enviroliteracy.org/pdf/materials/1210.pdf, 2010.

[7] D. Decotigny and I. Puaut. ARTISST: An Extensible and Modular Simulation Tool for Real-Time Systems. In *Proc. of the ISORC' 02*, pages 365–372, 2002.

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

[9] J. Kleijnen, R. Cheng, and B. Bettonvil. Validation of trace-driven simulation models: More on bootstrap tests. In *Proc. of the WSC' 00*, pages 882–892, 2000.

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

[11] A. M. Law. How to build valid and credible simulation models. In *Proc. of the WSC' 08*, pages 39–47, 2008.

[12] C. Loehle. A hypothesis testing framework for evaluating ecosystem model performance. *Ecol. Modeling 97*, pages 153–165, 1997.

[13] Y. Lu, T. Nolte, J. Kraft, and C. Norström. A Statistical Approach to Response-Time Analysis of Complex Embedded Real-Time Systems. In *Proc. of the RTCSA' 10*, pages 153–160, Aug 2010.

[14] D. G. Mayer and D. G. Butler. Statistical validation. *Ecol. Modeling 68*, pages 21–32, 1993.

[15] 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.

[16] What are outliers in the data? http://www.itl.nist.gov/div898/handbook/prc/section1.

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

[18] A. Robinson and R. Froese. Model validation using equivalence tests. *Elsevier, ScienceDirect 2004*, 176:349–358, 2004.

[19] S. Stigler. Fisher and the 5% Level. *Journal of CHANCE*, 21(4):12, 2008.

[20] t-test and ANOVA, http://mathworld.wolfram.com, 2010.

[21] Wilcoxon-Mann-Whitney test. http://www.slideshare.net/mhsgeography/mann-whitney-u-test-2880296, 2010.

[22] XLSTAT, www.xlstat.com, 2010.

[2] ✓ means that the result given by StatiVal at the confidence level 95% conforms to the corresponding known model validation result; × will be given otherwise.