Assessment of Trace-Differences in Timing Analysis for Complex Real-Time Embedded Systems

Yue Lu[1], Thomas Nolte[1], Iain Bate[2], Johan Kraft[1] and Christer Norström[3]
[1]Mälardalen Real-Time Research Centre (MRTC), Västerås, Sweden
[2]Department of Computer Science, University of York, York, United Kingdom
[3]Swedish Institute of Computer Science (SICS), Kista, Sweden
yue.lu@mdh.se

Abstract

In this paper, we look at identifying temporal differences between different versions of Complex Real-Time Embedded Systems (CRTES) by using timing traces representing response times and execution times of tasks. In particular, we are interested in being able to reason about whether a particular change to CRTES will impact on their temporal performance, which is difficult to answer due to the complicated timing behavior such CRTES have. To be specific, we first propose a sampling mechanism to eliminate dependencies existing in tasks' response time and execution time data in the traces taken from CRTES, which makes any statistical inference in probability theory and statistics realistic. Next, we use a mature statistical method, i.e., the non-parametric two-sample Kolmogorov-Smirnov test, to assess the possible temporal differences between different versions of CRTES by using timing traces. Moreover, we introduce a method of reducing the number of samples used in the analysis, while keeping the accuracy of analysis results. This is not trivial, as collecting a large amount of samples in terms of executing real systems is often costly. Our evaluation using simulation models describing an industrial robotic control system with complicated tasks' timing behavior, indicates that the proposed method can successfully identify temporal differences between different versions of CRTES, if there is any. Furthermore, our proposed method outperforms the other statistical methods, e.g., bootstrap and permutation tests, that are often widely used in contexts, in terms of bearing on the accuracy of results when other methods have failed.

1 Introduction

Many industrial embedded systems are becoming ever larger, more flexible and highly configurable software systems, which often contain millions of lines of code and hundreds of tasks. Furthermore, many tasks in such systems have real-time constraints, and they are being triggered in complex, nested patterns. In addition, the tasks' intricate temporal dependencies vary their execution time and response time radically, and hence making systems' timing behavior quite complicated. Examples of such systems include the robotic control systems developed by ABB [1], as well as several telecom systems. We refer to such systems as *Complex Real-Time Embedded Systems* (CRTES).

To maintain, analyze and reuse CRTES is very important, difficult and expensive, which, nonetheless, offers high business value responding to great concern in industry. More importantly, due to the fact that many changes are made over years, as a consequence, the perspicuity of CRTES decreases, i.e., it becomes increasingly harder to predict whether a change, e.g., a new feature, will impact not only functional behavior, but also temporal correctness of the system. For instance, if the Worst-Case Execution Times (WCETs) or priorities of some tasks in the system have to be increased according to a certain requirement, then the overall system timing behavior (including adhering tasks' execution time and response time) will be significantly different so that a more thorough re-verification or regression testing of the system has to be done. Worse yet, since the size of the source code of CRTES is usually quite large, sometimes millions of lines of code, the complexity of using static analysis techniques based on source code or disassembly is far too great. To some degree, along with the lines of protecting intellectual property pertaining to some important parts of such CRTES, the companies owning such code may not be willing to make the source code available and public for researches conducted outside the company; it may even not be allowed to disassemble the corresponding object code. Enabling assessment of temporal differences in CRTES is thereof a problem of high industrial relevance.

In this paper, rather than relying on the source code or

the disassembly, we aim at tackling with the above issue by using timing traces taken from real systems, which can been generated for instance by an instrumented program during the test phase. For each such timing trace, the execution time and response time of adhering tasks in the system during runtime, are measured and recorded by using our proposed sampling mechanism based on the Simple Random Samples (SRS) technique [17]. It is interesting to notice that there are different types of tracing mechanisms that potentially can be used to generate and store a timing trace. However, these techniques are not in the scope of this paper, as in this work we assume that such timing traces exist, and they are the valid[1] inputs to our proposed method.

There has been a large body of work in the domain of simulation model validation, e.g., [13, 15], which is quite close to the research presented in this paper; these methods are either objective or subjective. Objective methods, such as statistical hypothesis testing, offers a framework which compares a sample of observations taken from the original system and the simulation model. In this paper, we consider our particular research problem as a statistical problem, which can be solved by using existing mature methods in the field of statistical hypothesis testing. The benefit is that such hypothesis testing is less dependent on domain expertise and hence less error-prone, when compared to subjective methods, such as *Face Validation* and *Graphical Comparisons* [4].

Contributions: The main contribution of this paper is how to perform an accurate analysis focusing on assessment of temporal differences, with regard to both tasks' execution time and response time, between different versions of CRTES, without their disassembly or source code. This is done in terms of analyzing sampling distributions of tasks' Response Time (RT) and Execution Time (ET) data in timing traces taken from CRTES using our proposed algorithm based on the non-parametric *two-sample Kolmogorov-Smirnov test* [12] (the two-sample KS test hereafter). In addition, we also present the process of reducing the number of samples used in the analysis, while keeping the accuracy of analysis results. Such optimization is necessary and meaningful, since collecting analysis samples in terms of executing industrial-size CRTES is usually costly. Moreover, our proposed method is evaluated and compared with other non-parametric statistical hypothesis tests including the ones that have been widely used in contexts (e.g., *resampling methods* [21]), by using an evaluation framework consisting of a set of simulation models inspired by an industrial robotic control system. The results indicate that our proposed method can successfully identify the temporal differences existing in different versions of CRTES, especially when other statistical methods either failed, or cannot be properly applied.

Organization: Section 2 introduces a simulation framework for modeling and timing analysis of CRTES, and gives descriptive statistics of the RT and ET data of tasks in the traces taken from an example of CRTES, as well as the problem formulation being defined in the same section. Next, Section 3 first presents a sampling mechanism to eliminate dependencies existing in timing data in traces, and then introduces the feasibility of using different statistical methods, including parametric and non-parametric ones, in the context of analyzing the sampled traces, and finally outlines our proposed algorithm *TTVal* which is based on the two-sample KS test. The evaluation framework and the corresponding results, and the related work appear in Section 4 and Section 5 respectively, before conclusions are drawn in Section 6.

2 Modeling and Timing Analysis of CRTES

This section is split into three parts: Section 2.1 presents the *RTSSim* simulation framework for modeling and timing analysis of CRTES, which is mainly for the evaluation purpose in this work, Section 2.2 introduces the descriptive statistics of timing data in the traces taken from an example of CRTES, and finally, Section 2.3 gives the problem definition together with non-traditional hypotheses used in our work.

2.1 Simulation of CRTES

The target CRTES are described by the modeling language in RTSSim simulation framework [11], which 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, 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 during simulation are called by the RTSSim framework. 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 and execution time of tasks are measured whenever the scheduler is invoked, which happens for example at IPC, task switches, operations on semaphores, task activations and when tasks end, etc. This, together with the simulation clock behavior, guarantees that the measured response time and execution time are exact.

[1] Such timing traces used in our analysis should NOT contain some invalid data caused by hardware or software errors in the system.

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 during simulation, and offset and jitter can be smaller, equal or larger than the period. Tasks with equal priorities are served on a first come first served basis.

2.2 Descriptive Statistics of Timing Data in Traces Taken From An Example of CRTES

In this work, for the evaluation purpose, we will use the model Model 1 in Section 4.1, which is manually designed to contain similar modeling and analysis challenges as an example of CRTES, i.e., a real industrial robotic control system developed by ABB, which contains intricate temporal dependencies between tasks. Further, due to the existence of intricate task execution dependencies in Model 1, an upcoming RT data may not be independent with the RT data previously recorded at each simulation run. The same problem happens for ET data. We refer to such RT and ET data as *raw RT and ET data of tasks* hereafter. More importantly, *outliers* existing in raw RT and ET 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 or ET datum beyond an *outer fence* [18] is considered as *(extreme) outlier*. For the sake of space, Table 1 only shows the numerical summary of the center and the spread (or variability) of sampling distributions of the raw RT and ET data of the CTRL task (which has the most complicated temporal behavior among all tasks) in Model 1, where *X*, *Y*, *Std. Dev*, *Q1* and *Q3* represents *response time*, *execution time*, *standard deviation*, *first quartile* and *third quartile* of the sampling distribution respectively. The presence of outliers leads us to consider using the *five-number summary* introduced in [17] consisting of *Min*, *Q1*, *Median*, *Q3* and *Max* to Table 1. Concerning outliers, by using the definitions given in [18], *inner fences* and *outer fences* for raw RT and ET data of the CTRL task are also shown in Table 1, where *LIF*, *UIF*, *LOF* and *UOF* represent *lower inner fence* (i.e., $Q1-1.5\times IQ$), *upper inner fence* (i.e., $Q3+1.5\times IQ$), *lower outer fence* (i.e., $Q1-3\times IQ$) and *upper outer fence* (i.e., $Q3+3\times IQ$) respectively. Note that $IQ = Q3 - Q1$.

Furthermore, the *Probability Density Function* (PDF) histograms of the sampling distributions of raw RT and ET data of the CTRL task, given a large[2] number at sample data, are shown in Figure 1 respectively. Note that the outliers in the picture are not clearly visible, though in fact, they exist in the range of $[4\,574, 6\,829]$ (for the RT sampling distribution) and $[3\,074, 5\,324]$ (for the ET sampling distribution).

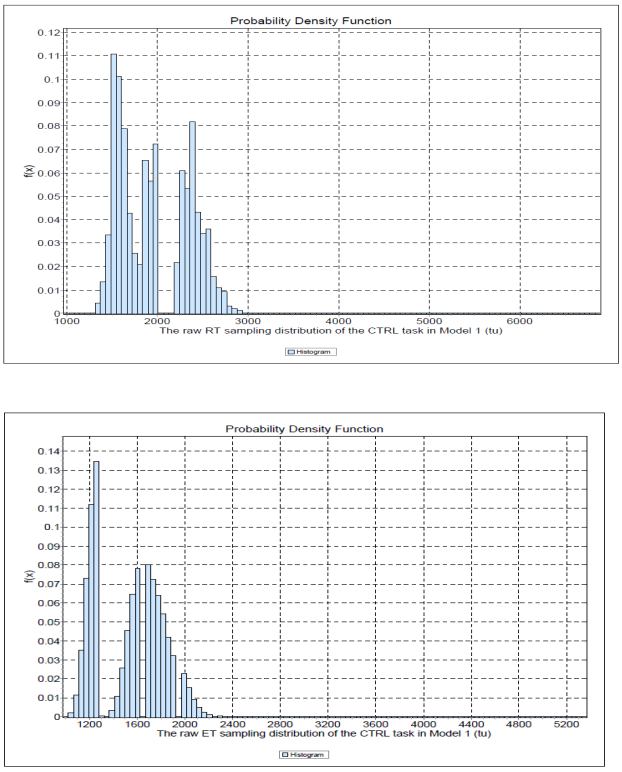


Figure 1. The Probability Density Function (PDF) histograms of raw RT data (on the upper part) and raw ET data (on the lower part) of sampling distributions of the CTRL task in Model 1.

In the conventional statistical procedure (*parametric test*), e.g., t-test, z-test and analysis of variance (ANOVA) [23], one important assumption is that the underlying population is assumed to follow a normal distribution. However, such an assumption cannot be made in our case, since the sampling distribution of either raw RT data or raw ET data of all tasks is often clearly conforming to a multi-modal distribution having several peaks (consider Figure 1 as examples). Specifically, because of such distinctive feature of our target CRTES, it is difficult to bring conventional statistical methods into the context. More importantly, due to the dependencies existing in both tasks' RT and ET data, a new way of constructing the sampling distributions of such tasks' timing data is necessary, and hence has to be introduced, in order to fulfill the basic assumption *Independent and Identically Distributed* (i.i.d.) in any statistical inference in probability theory and statistics. We will introduce the i.i.d. assumption together with our proposed sampling mechanism in Section 3.1.

[2]By running one simulation for the time up to the upper bound of the RTSSim simulation time, i.e., $2^{31}-1$, we have collected 199 990 samples of raw RT and ET data of the CTRL task.

Table 1. Descriptive statistics of sampling distributions of raw RT and ET data of the CTRL task in the evaluation of Model 1 with the same simulation trace.

	Samples	Mean	Std. Dev	Min	Q1	Median	Q3	Max	LIF	UIF	LOF	UOF
X_{CTRL}	199 990	1 967.3	389.98	1 024	1 594	1 919	2 339	6 829	476.5	3 456.5	-641	**4 574**
Y_{CTRL}	199 990	1 534.4	276.86	1 024	1 274	1 574	1 724	5 324	599	2 399	-76	**3 074**

2.3 Problem Formulation

We are given two versions of CRTES, S and $S^{'}$ which represent the original system and the changed system respectively. Further, each system contains a task set Γ which has the same number of tasks, i.e., n, where $n \in \mathbb{N}$. Let X_{τ_k}, $X^{'}_{\tau_k}$, and Y_{τ_k}, $Y^{'}_{\tau_k}$, denote sampling distributions drawn from underlying populations of RT and ET 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 $\langle X_{\tau_k}, X^{'}_{\tau_k} \rangle$, and $\langle Y_{\tau_k}, Y^{'}_{\tau_k} \rangle$ are significantly different, or if they can be considered statistically equal (i.e., from the same population), at the certain significance level [17]. 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 [22]. In addition, as introduced in [16] and [14], 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: *There are significant differences between different versions of CRTES, at the significance level* $\alpha = 0.05$*, from the perspective of response time and execution time distributions of the adhering tasks.*
- H_a: *There are no significant differences between different versions of CRTES, at the significance level* $\alpha = 0.05$*, from the perspective of response time and execution time distributions of the adhering tasks.*

3 Assessment of Temporal Differences in CRTES by Using Timing Traces

This section presents our proposed method of assessing temporal differences between different versions of CRTES by using the two-sample KS test. Section 3.1 first shows our proposed sampling mechanism used to eliminate dependencies existing in timing traces taken from CRTES, which is necessary to be done before any statistical methods are used. Section 3.2 introduces the feasibility of using different parametric and non-parametric statistical hypothesis tests in our research context. Finally, Section 3.3 and Section 3.4 introduce our proposed algorithm and the method of reducing the number of samples in our analysis while keeping the accuracy of analysis results, respectively.

3.1 The Sampling Mechanism

First, in order to eliminate bias on the sampling, which is a key issue of selecting samples from the population of all individuals concerning the desired information, the technique of Simple Random Samples (SRS) [17] is adopted. The SRS gives every possible sample of a given size the same chance to be chosen. Practically, when such samples are taken from real systems, the SRS can be done by randomizing system inputs by using the *uniform distribution*. Secondly, we propose a sampling mechanism which first executes the real system for N times (i.e., N *sub-timing traces*) based on the SRS technique, and each of sub-timing traces contains m raw RT and m raw ET data for every adhering task. Next, per sub-timing trace, the highest value of raw RT data and raw ET data for each task recorded, will be chosen as the sample to construct the new sampling distributions of the RT and ET data of the same task to be analyzed by our proposed algorithm (to be introduce in Section 3.3). Furthermore, since there are no dependencies between any maximum of the RT and ET data of tasks from two independent sub-timing traces, as a result, each individual in the new reconstructed sampling distributions consisting of such sub-timing traces can be considered to have the same probability distribution as the other individuals, and all the individuals are mutually independent. Hence, the underlying assumption i.i.d. is realistic and satisfied when the new reconstructed sampling distributions are used in the statistical inference. Refer to Figure 2 as examples about the PDF histograms of the RT and ET data of new sampling distributions of the CTRL task in Model 1 respectively. It is worth noting that Figure 1 shows the PDF histogram of the original (raw) RT and ET data of the CTRL task based on 199 990 samples obtained by executing a single run of the Model 1, while Figure 2 displays the PDF histogram of the new reconstructed RT and ET data of the CTRL task consisting of a maximum of 20 000 sub-timing traces.

Nonetheless, as shown in Figure 2 clearly, the new reconstructed sampling distributions of the RT and ET data of

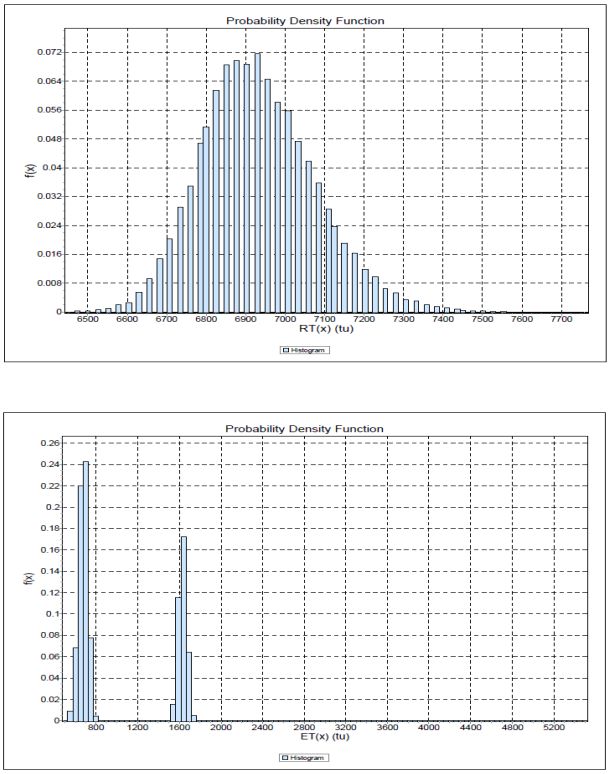


Figure 2. The PDF histograms of the RT (on the upper part) and ET (on the lower part) data of new sampling distributions of the CTRL task in Model 1.

the CTRL task is either *positive skewed* (i.e., the right tail is longer), or a multimodal distribution with outliers, which makes the assumption required by the conventional statistical methods, i.e., normality of distributions, not be satisfied. Hence, a parametric test cannot be reasonably applied in this work, and we therefore consider using non-parametric statistical methods, which are making no assumptions on the underlying population from which a sampling distribution is drawn. Looking at potential non-parametric statistical hypothesis tests, we have several candidates, including resampling methods (i.e., bootstrap and permutation tests), the χ^2 test, the Wilcoxon-Mann-Whitney test, and the two-sample KS test. In Section 3.2 we will discuss the feasibility of using these methods in our context.

3.2 The Feasibility of Using Statistical Methods in Timing Analysis of CRTES

3.2.1 Parametric Statistics

In order to determine if the conventional statistical procedure (*parametric test*), e.g., t-test, z-test and ANOVA, can be used to infer valid parameters of tasks' RT and ET data sampling distributions, it first needs to be checked whether the values conform to a normal distribution. In this work, we have performed this investigation by using a commercial statistic analysis software *EasyFit* [7], according to the results given by a Goodness of Fit (GOF) test, i.e., Chi-squared test [5] at α-value of 0.05. The results show that, for example, the new sampling distributions of RT and ET data of the CTRL task, which is the task with the most complicated timing behavior (i.e., the most interesting task) in the Model 1, do not conform to any of the 65 known distributions in [7], e.g., Normal, Uniform, Student's t, Lognormal etc. Hence, we conclude that parametric tests cannot be reasonably applied in this work, and we thereby consider to use non-parametric hypothesis tests which make no assumptions on the underlying population of a sampling distribution.

3.2.2 Resampling Methods

Without relying on the assumption on the normality of both underlying populations, the methods *bootstrap* and *permutation* tests are widely used to compare if two sampling distributions are from the same populations or not. The basic idea of resampling methods is to find out if its sampling distribution, at least approximately, from the original samples represent the population from which they were drawn [17]. Thus, resamples with or without *replacement*[3] from the original samples represent the underlying population if many samples are taken from the population. Consequently, the resampling distribution of a statistic (such as mean, median) represents the sampling distribution of the statistic, can thus be used to compare the statistics of the underlying populations. In this work, we compare both the mean and the median of resampling distributions of RT and ET data of all tasks in both different versions of CRTES, i.e., S and S', at the significance level $\alpha = 0.05$. The outline of the algorithm is shown as follows.

- **Resampling:** For each task τ_k in the task set Γ, two RT sampling distributions (i.e., X_{τ_k} and X'_{τ_k}) and two ET sampling distributions (i.e., Y_{τ_k} and Y'_{τ_k}) of task τ_k in both of the original system S (to which there are no changes applied) and the changed system S' are constructed in the way introduced in Section 3.1. Moreover, it is necessary to calculate the differences between either the *mean* or the *median* of RT sampling distributions X_{τ_k} and X'_{τ_k} (i.e., $X^{diff}_{\tau_k}$) and ET sampling distributions Y_{τ_k} and Y'_{τ_k} (i.e., $Y^{diff}_{\tau_k}$).

[3] *Sampling with replacement* means that after an observation from the original samples is randomly drawn, it will be placed back before drawing the next observation. While *Sampling without replacement* means that next time drawing will be done without putting the previous observation back.

- **Bootstrap or permutation distributions:** $X_{\tau_k}^{diff'}$, $Y_{\tau_k}^{diff'}$

  i) Draw two RT resamples $X_{\tau_k,l}^{bootstrap}$ (for the system S) and $X_{\tau_k,l}^{bootstrap'}$ (for the changed system S') with replacement (for bootstrap) or without replacement (for permutation test), from the RT sampling distributions X_{τ_k}, X'_{τ_k} being constructed at the previous step. The same actions are taken for ET resampling data of task τ_k.

  ii) Compute the statistic, i.e., the difference between either the mean or the median of $X_{\tau_k,l}^{bootstrap}$ and $X_{\tau_k,l}^{bootstrap'}$, as well as $Y_{\tau_k,l}^{bootstrap}$ and $Y_{\tau_k,l}^{bootstrap'}$.

  iii) Repeat this sub-process for 10 000 iterations, giving more accurate results than the number of iterations which is usually adopted, i.e., 1 000.

- **Hypothesis testing conclusion:**

  i) For each task τ_k in the task set Γ, the corresponding P values of bootstrap or permutation test distributions $X_{\tau_k}^{diff'}$, $Y_{\tau_k}^{diff'}$ (i.e., P_x and P_y respectively), are calculated to decide if the observed difference between the sample means or medians is large enough to reject the null hypothesis H_0, i.e., *probability value* of the number of samples in bootstrap or permutation test distributions $X_{\tau_k}^{diff'}$, $Y_{\tau_k}^{diff'}$ that are larger than the observed mean or median difference, i.e., $X_{\tau_k}^{diff}$, $Y_{\tau_k}^{diff}$.

  ii) Draw the conclusion that H_0[4] cannot be rejected if P_x or P_y is larger than 0.05 (i.e., the significance level $\alpha = 0.05$); otherwise H_a will be drawn.

However, both bootstrap and permutation tests failed in, for instance, identifying temporal differences existing in both the system S and the changed system S' concerning Case 4-1 (of which the brief scenario description is shown in Table 4) in our evaluation framework.

3.2.3 The χ^2 Test, WMW Test and Two-sample KS Test

In the domain of non-parametric statistical hypothesis tests, except for the resampling methods (introduced in the previous section), there are a few other methods such as *Chi-squared test*, *Wilcoxon-Mann-Whitney test* [25], *Kolmogorov-Smirnov test* (the χ^2 test, the WMW test and the two-sample 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 underlying population are compared to the observed frequencies of samples in the sampling distribution. Such expected frequencies are made subjectively, in terms of being either hypothesized as all equal or on the basis to some priori knowledge or experience. However, in the case of timing analysis for CRTES, it is either too subjective to support such a hypothesis, or not accurate enough due to limited a priori knowledge of the complicated tasks' runtime behavior caused by intricate task execution dependencies. The χ^2 test is not feasible to be applied in this work thereof.

Concerning the WMW test, it cannot compare 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 comparing two sampling distributions taken from two multi-modal 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 two-sample KS test uses the maximum vertical deviation between the two *Cumulative Fraction Plot* (CFP) curves[5] as the statistic D, to which the corresponding P value will suggest that if there is a significant difference or not. Specifically, the two-sample KS test does not limit itself to the location, and it can take into account more information, such as the variability. Hence, the two-sample 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 two-sample 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 [22]) in this work, and the corresponding hypotheses are as follows:

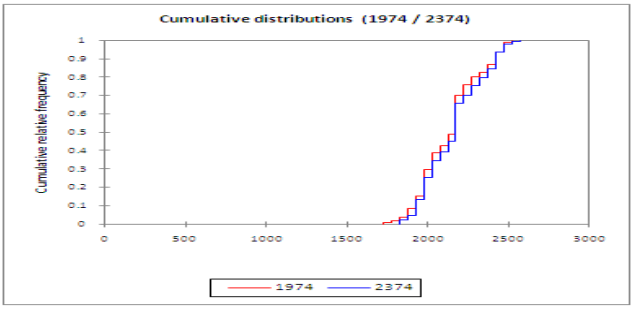
- $H_{0,ks}$: *There is no significant difference between two different versions of the target system in the view of response time and execution time distributions of the task on focus.*
- $H_{a,ks}$: *There is a significant difference between two different versions of the target system in the view of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

3.3 TTVal

The proposed method *TTVal* is shown in Algorithm 1. The algorithm returns the result given by the two-sample KS test, i.e., if there exists a statistically significant difference between the original system S and the changed system

[4] H_0: There is no difference between the system S and the changed system S' from the perspective of the adhering tasks' RT and ET. H_a is opposite to H_0.

[5] CFP curves are a graphical display of how the data in each sampling distribution is distributed. Figure 3 shows two examples that use the two-sample KS test to identify differences between two sampling distributions separately.



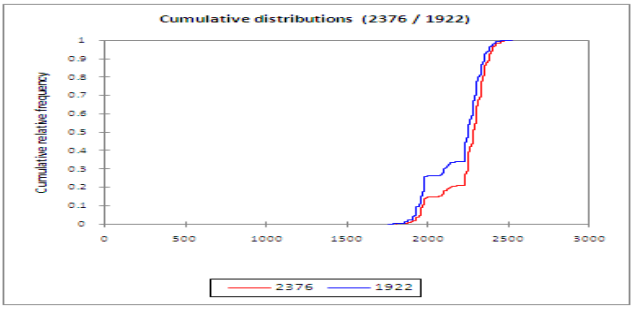


Figure 3. The upper sub-figure demonstrates that the two-sample KS test finds no significant difference between two sampling distributions. The lower sub-figure shows that the two-sample KS test does find a significant difference between the two sampling distributions. Moreover, all hypothesis tests are supported with the significance level of $\alpha = 0.05$.

S', in the view of system timing properties such as the adhering tasks' response time and execution time. For both of the original system S and the changed system S', the tasks' RT and ET data sampling distributions used in the analysis are collected by using our proposed sampling mechanism (as introduced in Section 3.1) based on the SRS, of which implementation in Algorithm 1 is corresponding to lines 1 to 16. In addition, the SRS is implemented as a function *SRS*, with three parameters: m - the number of samples in each task's timing trace, τ_k - a specific task in CRTES, and *Property* - either RT or ET of the task τ_k. The outline of the algorithm is as follows:

- Construct sampling distributions of N RT and ET data of all tasks in both S and S' by using our proposed sampling mechanism, i.e., executing the target CRTES for N times by using SRS, and then choosing the maximum of the RT and ET values per each run to construct the new RT and ET sampling distributions for analysis respectively (refer to lines 1 to 16 in Algorithm 1). It is interesting to stress that to our experience, when N is equal to 20 000, any two of the timing traces collected by using our proposed sampling mechanism, for the same system, are not significantly different with each other based around the two-sample KS test.
- Use the two-sample KS test to compare if sampling distributions of RT and ET data of each task τ_k in both S and S' are statistically significant iteratively. If for any task τ_k, the result given by the two-sample KS test draws the conclusion H_0 (as introduced in Section 2.3), then the algorithm will consider that there are temporal differences between S and S' with regard to tasks' RT and ET data, and the entire process will stop; otherwise, the entire process will not terminate until all tasks are evaluated by the two-sample KS test (refer to lines 18 to 33 in Algorithm 1). In practice, the two-sample KS test is conducted by using a commercial software *XLSTAT* [26], which is a plug-in to EXCEL.

Algorithm 1 $TTVal(\Gamma)$

1: **for all** τ_k such that $1 \le k \le n$ in Γ in both S and S' **do**
2: **for all** i such that $1 \le i \le N$ **do**
3: $X_i \leftarrow x_{i,1}, ..., x_{i,j}, ..., x_{i,m} \leftarrow SRS(m, \tau_k, RT)$
4: $X_{\tau_k,i} \leftarrow MAX(X_i)$
5: $Y_i \leftarrow y_{i,1}, ..., y_{i,j}, ..., y_{i,m} \leftarrow SRS(m, \tau_k, ET)$
6: $Y_{\tau_k,i} \leftarrow MAX(Y_i)$
7: $X'_i \leftarrow x'_{i,1}, ..., x'_{i,j}, ..., x'_{i,m} \leftarrow SRS(m, \tau_k, RT)$
8: $X'_{\tau_k,i} \leftarrow MAX(X'_i)$
9: $Y'_i \leftarrow y'_{i,1}, ..., y'_{i,j}, ..., y'_{i,m} \leftarrow SRS(m, \tau_k, ET)$
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** τ_k such that $1 \le k \le n$ in Γ 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

3.4 A Method of Reducing Sample Size N

In [20], 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 timing

traces N in the analysis. We will illustrate the idea by referring to a concrete example shown in Table 2 using Case 3 in Table 4. According to our reasoning, when N is equal to 20 000, TTVal can give the correct result H_0 as shown at Step 1 in Table 2. In Column *Accuracy* in Table 2, √ represents *the result given by TTVal 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., $\left\lceil \frac{20\,000}{4} \right\rceil$). The results given by TTVal 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 of analysis results. It is worth noting that the value of N could be further optimized by using for instance a lower-part binary search algorithm [6].

Table 2. Illustration of reducing the number of samples (per timing trace) required by the proposed algorithm TTVal.

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

4 Evaluation

In this section, we first introduce the evaluation models inspired by an example of CRTES, i.e., a real industrial control system. Then we give the description of *change scenarios* and the expected results, and finally, we show that our proposed algorithm TTVal can obtain accurate analysis results at the confidence level 95%, which is equal to the significance level 5%, representing the probability of mistakenly rejecting the null hypothesis in the two-sample KS test is no more than 5%, when it is true.

4.1 The Evaluation Models

We examine the idea by using a simulation model Model 1 describing a fictive, representative industrial robotic control system developed by one of our industrial partners. It is designed to include some behavioral mechanisms from the industrial robotic control system: 1) Tasks with intricate dependencies in temporal behavior due to Inter-Process Communication (IPC) 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. The details of Model 1 are described in [11]. Further, the tasks and task parameters in both Model 1 and a set of variations of Model 1[6] are presented in Table 3, where *DUMMY_H*, *PLAN_H*, *CTRL_H*, *CTRL_L*, *DUMMY_L*, *PLAN_O* and *PLAN_L* represent the DUMMY task with the priority 1, 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 3 is a *simulation time unit* (tu). Moreover, Row *Case* in the table denotes in which case or cases a specific task exists. For example, the DUMMY task only exists in Case 5 and Case 6. "–" represents the task appears in all cases.

4.2 Change Scenarios and Evaluation Results

Model 1 is considered as the original system S, and a set of variations S' (as shown in Table 4) are considered as changed systems. The description of the change scenarios and the corresponding Expected Results (ERs) are introduced as follows. In addition, the ERs are expressed in terms of using the hypotheses in our hypothesis test as introduced in Section 2.3, which are opposite to the ones in the traditional hypothesis test.

- Case 1: The execution time of the IO task is doubled, i.e., from 23 to 46 *tu*. Furthermore, due to that the IO task has a higher priority than the CTRL and PLAN task, therefore the ET and RT of not only the CTRL, PLAN task, but also the IO task itself will be changed. Thus H_0 is the expected result, i.e., there are significant differences between the S and S' at the significance level $\alpha = 0.05$, from the perspective of response time and execution time distributions of adhering tasks.
- Case 2: When the priority of the PLAN task is lowered in Case 2-1, the ET and 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 that the priority of the PLAN task is prompted to be higher than any other tasks except for the DRIVE task, then the ET and RT of other tasks will be changed. Furthermore, since the DRIVE task is released with an offset (i.e., 12 000 *tu*) which is longer than its period (i.e., 2 000 *tu*), its ET and RT might also be impacted. Thereby, the corresponding expected result is H_0.
- Case 3: When the period of the PLAN task which is the lowest priority task in the system is increased, the ET

[6] Such variations of the Model 1 are derived after the certain type of changes [24] is applied to the Model 1.

Table 3. Tasks and task parameters in evaluation models. 0 stands for the highest priority.

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

and RT of all the tasks are not supposed to be affected at all, therefore the expected result is H_a.

- Case 4: When there is a DUMMY task (refer to as DUMMY_L in Table 3) 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 the ET or RT of the PLAN task might be affected after the change. Therefore H_0 is expected to be drawn.

- Case 5: When increasing the priority of the DUMMY task to be higher than all other tasks in the system model, for instance the ET and RT of the CTRL and PLAN task are supposed to be changed. Correspondingly, H_0 is supposed to be drawn. Further, the ET of the DUMMY task is too short to give impact on the ET and RT of the DRIVE and IO task.

- Case 6: When the priority of the DUMMY task is lowered to be the one which is higher than any other tasks except for the PLAN task, then only the ET or RT of the PLAN task is supposed to be affected, therefore the expected result is H_0.

In Table 4, ✓ means that the analysis result given by TTVal at the confidence level 95% conforms to the corresponding known expected result; × will be given otherwise. Further, as shown in Columns *TTVal*, *ER*, *Accuracy* and *Confidence Level* in Table 4[7], clearly, the results given by our proposed method TTVal at the confidence level 95% are in line with the expected results. This indicates that our proposed method can be used to successfully identify the temporal differences between different versions of CRTES based on using timing traces. This is helpful and meaningful for engineers to predict if the temporal changes made during the years of maintenance for CRTES will impact on the system's overall performance from the perspective of the adhering tasks' response time and execution time. Traditionally, such prediction is difficult to be made, due to the complicated system's timing behavior which usually not only makes many of the existing statistical methods fail, but also drives the subjective methods to be infeasible in such context since the number of scenarios is too large to be practically considered. It is interesting to note that the reasons for why other parametric and non-parametric statistics cannot be applied to the context of our research have been outlined in details, in Section 3.2.

[7] In addition, the listing order of such columns is the same as the listing order in Table 4.

5 Related work

This section reviews the work that are not mentioned in previous sections, but related. In [2] 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 [10] 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-modal distributions, due to the adhering intricate task temporal dependencies. The key innovation that equivalence testing [19] 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. Huselius [9] presents the work about the automated validation of models extracted from real-time systems by checking if the model can generate the same event sequences as the recorded event sequences from the system using a model checker.

6 Conclusions and Future Work

To predict if changes applied to *Complex Real-Time Embedded Systems* (CRTES) would impact on the overall systems' temporal behavior is an area of interest, both in academia and in industry. This paper has presented our work on tackling the above issue by using existing mature statistical methods together with timing traces taken from real systems, which is a good approach not limited by system source code being withheld or restrictions in that the

Table 4. Results obtained by using TTVal concerning different models according to change scenarios, and the Expected Results (ER).

Cases S'	Changes Description	RT_{DR}	ET_{DR}	RT_{IO}	ET_{IO}	RT_{CT}	ET_{CT}	RT_{PL}	ET_{PL}	TTVal	Confidence Level	ER	Accuracy
Case 1	IO: C 23 → 46	H_a	H_a	H_0	H_0	H_0	H_0	H_0	H_0	H_0	95%	H_0	✓
Case 2-1	PLAN: Prio 8 → 9	H_a	H_a	H_a	H_a	H_a	H_a	H_a	H_a	H_a	95%	H_a	✓
Case 2-2	PLAN: Prio 8 → 3	H_0	H_0	H_0	H_0	H_0	H_0	H_0	H_0	H_0	95%	H_0	✓
Case 3	PLAN: T 40 000 → 80 000	H_a	H_a	H_a	H_a	H_a	H_a	H_a	H_a	H_a	95%	H_a	✓
Case 4-1	DUMMY: Prio = 7, T = 5 000, C = 25	H_a	H_a	H_a	H_a	H_a	H_a	H_0	H_a	H_0	95%	H_0	✓
Case 4-2	DUMMY: Prio = 7, T = 5 000, C = 50	H_a	H_a	H_a	H_a	H_a	H_a	H_0	H_a	H_0	95%	H_0	✓
Case 5	DUMMY: Prio = 1, T = 5 000, C = 25	H_a	H_a	H_a	H_a	H_0	H_0	H_0	H_0	H_0	95%	H_0	✓
Case 6	DUMMY: Prio = 1, T = 10 000, C = 50	H_a	H_a	H_a	H_a	H_0	H_0	H_0	H_0	H_0	95%	H_0	✓

relocatable object code cannot be disassembled in order to protect intellectual property. Our evaluation using a number of simulation models inspired by an example of CRTES, i.e., an industrial robotic control system, shows that the results given by our proposed method are in line with the expected results. This also indicates that the proposed method can successfully identify temporal differences between different versions of CRTES with regard to Response Time (RT) and Execution Time (ET) of the adhering tasks. In addition, we introduce a method of reducing the sample size while keeping the accuracy of analysis results, which is not trivial since collecting a large number of tasks' RT and ET data samples from real systems is usually quite costly.

Further work will investigate ways in which to evaluate our proposed method by using e.g., multi-media applications running on the *Andriod* 2.3 platform [3]. Moreover, we will consider using some optimization algorithm, e.g., Genetic Algorithm [8], to generate system inputs, which could contribute to collect the sampling distributions with a certain bias, i.e., higher RT and ET of adhering tasks, then to check if our proposed method can still identify the temporal differences. Finally, the work on addressing the issue if the temporal changes made to CRTES would be acceptable given certain previously defined timing constraints, will be conducted.

References

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

[2] J. Andersson, A. Wall, and C. Norström. Validating temporal behavior models of complex real-time systems. In *SERPS' 04*, September 2004.

[3] Website of Andriod. http://www.android.com/.

[4] O. Balci. How to assess the acceptability and credibility of simulation results. In *WSC' 89*, pages 62–71, New York, NY, USA, 1989. ACM.

[5] Chi-squared test, www.enviroliteracy.org/pdf/materials/1210.pdf, 2011.

[6] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to algorithms*. MIT Press, Cambridge, MA, USA, second edition, 2001.

[7] EasyFit, www.mathwave.com/products/easyfit.html, 2011.

[8] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley Professional, January 1989.

[9] J. Huselius, J. Andersson, H. Hansson, and S. Punnekkat. Automatic generation and validation of models of legacy software. In *RTCSA' 06*, pages 342–349, 2006. IEEE Computer Society.

[10] J. Kleijnen, R. Cheng, and B. Bettonvil. Validation of trace-driven simulation models: More on bootstrap tests, 2000.

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

[12] KS-test, www.physics.csbsju.edu/stats/ks-test.html, 2010.

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

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

[15] Y. Lu, J. Kraft, T. Nolte, and I. Bate. A statistical approach to simulation model validation in response-time analysis of complex real-time embedded systems. In *SAC' 11*. ACM, March 2011.

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

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

[18] What are outliers in the data? http://www.itl.nist.gov/div898/handbook/prc/section1/prc16.htm.

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

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

[21] J. L. Simon. *Resampling: The New Statistics*. ACM Press, second edition, 1997.

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

[23] t-test, z-test and ANOVA, http://mathworld.wolfram.com, 2011.

[24] A. WAll. *Architectural Modeling and Analysis of Complex Real-Time Systems*. PhD thesis, Mälardalen University, September 2003.

[25] Wilcoxon-Mann-Whitney test, http://www.slideshare.net/mhsgeography/mann-whitney-u-test-2880296, 2011.

[26] XLSTAT, www.xlstat.com, 2011.