

# Hybrid Statistical Model Checking Technique for Reliable Safety Critical Systems

Youngjoo Kim, Moonzoo Kim  
Computer Science Department  
Korea Advanced Institute of Science and Technology (KAIST)  
South Korea  
jerry88@cs.kaist.ac.kr, moonzoo@cs.kaist.ac.kr

## ABSTRACT

Reliability of safety critical systems such as nuclear power plants and automobiles becomes a significant issue to our society. As more computing systems are utilized in these safety critical systems, there are high demands for verification and validation (V&V) techniques to assure the reliability of such complex computing systems. However, as the complexity of computing systems increases, conventional V&V techniques such as testing and model checking have limitations to achieve the reliability of complex safety critical systems, since such systems often control highly complex continuous dynamics to interact with physical environments. To improve the reliability of such systems, *statistical model checking* (SMC) techniques have been proposed. SMC techniques can check if a target system satisfies given requirements through statistical methods. In this paper, we propose a new hybrid SMC technique that integrates sequential probability ratio test (SPRT) technique and Bayesian interval estimation testing (BIET) technique to achieve precise verification result fast. In addition, we demonstrate effectiveness and efficiency of this hybrid SMC technique by applying the hybrid SMC technique to three safety critical systems on automobile domain.

## 1. INTRODUCTION

As computing hardware advances rapidly, various areas of our life utilize computing systems such as smart phones, medical devices, and automobile controllers. Consequently, the reliability of computing systems becomes a significant issue to our society and various international standards are proposed and applied to assure reliability of such systems. For example, avionics domain has DO-178C [15] as a standard for reliable software, automobile domain has a functional safety standard ISO 26262 [7], and medical electrical equipment domain has IEC 60601 [6] as a technical standard for the safety and effectiveness.

However, as computing power increases, the complexity of computing systems also increases rapidly, which causes many challenges to assure reliability of computing systems. In particular, the size and complexity of software in a computing system has increased quickly. Although software reliability has been studied actively [14], conventional verification and validation (V&V) tech-

niques for software such as testing and model checking [2] have limitations to assure the reliability of complex safety critical computing systems. One reason for this difficulty is that such systems often control highly complex continuous dynamics to interact with physical environments. In addition, since safety critical systems consist of both hardware and software and interact with a physical environment that often behaves non-deterministically (e.g., condition of road surface for automobiles or wind speed for airplanes), we should analyze target hardware and software with its environment together as a stochastic process [16]. However, conventional V&V techniques for software are difficult to analyze a target system in such contexts.

As a remedy to improve the reliability of safety critical systems, *statistical model checking* (SMC) techniques [21, 19, 20, 5, 23, 3, 8] have been proposed. SMC techniques approximately compute probabilities for a target system to satisfy given requirements based on randomly sampled execution traces. Thus, SMC techniques can assure the reliability of a complex target system statistically without analyzing the internal logic of a target system.

In our previous work [9], we empirically evaluated the effectiveness (i.e., precision of verification) and efficiency (i.e., time cost of verification) of the four state-of-the-art SMC techniques including single sampling plan (SSP) [19], sequential probability ratio test (SPRT) [21], Bayesian hypothesis testing (BHT) [8], and Bayesian interval estimation testing (BIET) [23]. Through the empirical study, we observed that these SMC techniques have different strong points and weak points so that they can complement one another. From this observation, we develop a new hybrid SMC technique which combines SPRT which is the fastest SMC technique and BIET which is the most precise SMC technique among the four SMC techniques studied. This hybrid SMC technique achieves precise verification result fast. To demonstrate the effectiveness and efficiency of this hybrid SMC technique, we have applied this hybrid SMC technique to three safety critical systems on automobile domain - automatic transmission control system (ATCS), anti-lock braking system (ABS), and fault-tolerant fuel control system (FFCS). Through the experiments, we confirm that our hybrid SMC technique improves effectiveness and efficiency compared to a single SMC technique.

The organization of the paper is as follows. Section 2 overviews related SMC techniques (i.e., SPRT and BIET). Section 3 describes a new hybrid SMC algorithm. Section 4 explains the three target systems - ATCS, ABS, and FFCS. Section 5 describes the verification results by using the single SMC techniques and the hybrid technique on Simulink/Stateflow models of ATCS, ABS, and FFCS. Section 6 discusses issues from the empirical study. Section 7 concludes this paper with future work.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.  
Copyright 20XX ACM X-XXXXX-XX-X/XX/XX ...\$10.00.

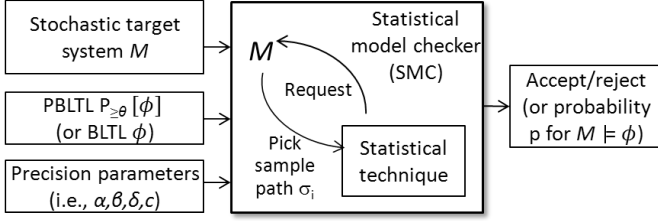


Figure 1: Framework of SMC techniques

## 2. OVERVIEW OF STATISTICAL MODEL CHECKING TECHNIQUES

### 2.1 SMC Framework

Figure 1 illustrates a framework of SMC techniques. There are two classes of statistical model checking techniques: *hypothesis testing* and *estimation testing*. A hypothesis testing technique receives a target system  $\mathcal{M}$ , a probabilistic bounded linear temporal logic (PBLTL) [23] formula  $P_{\geq\theta}[\phi]$  with probability threshold  $\theta$ , and precision parameters. A hypothesis testing technique produces an ‘accept’ answer if  $\mathcal{M} \models P_{\geq\theta}[\phi]$  which means that a probability for  $\mathcal{M}$  to satisfy  $\phi$  is greater than or equal to  $\theta$ ; a ‘reject’ answer, otherwise. An estimation testing technique receives a target system  $\mathcal{M}$  and a bounded linear temporal logic (BLTL) [22] formula  $\phi$  with precision parameters and produces an estimated probability  $p$  regarding  $\mathcal{M} \models \phi$ .

To produce an answer, both classes of SMC techniques pick a random sample path  $\sigma_i$  by executing  $\mathcal{M}$  and collect the result of checking  $\sigma_i \models \phi$ . SMC techniques request a sample path repeatedly until the information of sample paths are enough to determine if  $\mathcal{M} \models P_{\geq\theta}[\phi]$  or to calculate  $p$  for  $\mathcal{M} \models \phi$  with given precision parameters. Note that SMC techniques should determine a number of sample paths  $n$  to check if  $\mathcal{M} \models P_{\geq\theta}[\phi]$  or to calculate  $p$  for  $\mathcal{M} \models \phi$  using statistical techniques. Most SMC techniques calculate  $n$  dynamically through iterative sampling.

### 2.2 Probabilistic Bounded Linear Temporal Logic

We define a syntax and semantics of bounded linear temporal logic (BLTL) [22], and then extend BLTL to PBLTL [23]. For a target model  $\mathcal{M}$ ,  $SV$  is a finite set of real-valued state variables. A Boolean predicate over  $SV$  is a constraint of the form  $y \sim v$ , where  $y \in SV$ ,  $\sim \in \{\geq, \leq, =\}$ , and  $v \in \mathbb{R}$ . The syntax of the BLTL logic formula  $\phi$  is given by the following grammar:

$$\phi ::= y \sim v \mid (\phi_1 \vee \phi_2) \mid (\phi_1 \wedge \phi_2) \mid \neg\phi_1 \mid (\phi_1 \mathbf{U}^t \phi_2),$$

where  $y \in SV$ ,  $\sim \in \{\geq, \leq, =\}$ ,  $v \in \mathbb{R}$ , and  $t \in \mathbb{R}_{\geq 0}$ .

For other temporal operators, we can define  $\mathbf{F}^t \phi$  as  $\text{True } \mathbf{U}^t \phi$  and  $\mathbf{G}^t \phi$  as  $\neg \mathbf{F}^t \neg \phi$ . We denote a fact that an execution  $\sigma$  satisfies a property  $\phi$  as  $\sigma \models \phi$ . We use  $\sigma^k$  to denote a suffix trace of  $\sigma$  starting at step  $k$  ( $\sigma^0$  denotes the original execution  $\sigma$ ). We denote the value of a state variable  $y$  in  $\sigma$  at step  $k$  by  $V(\sigma, k, y)$ . We define  $t_k$  as a time at step  $k$  and  $t$  as a time bound. The semantics of BLTL on a trace  $\sigma^k$  is defined as follows:

- $\sigma^k \models y \sim v$  iff  $V(\sigma, k, y) \sim v$
- $\sigma^k \models \phi_1 \vee \phi_2$  iff  $\sigma^k \models \phi_1$  or  $\sigma^k \models \phi_2$
- $\sigma^k \models \phi_1 \wedge \phi_2$  iff  $\sigma^k \models \phi_1$  and  $\sigma^k \models \phi_2$
- $\sigma^k \models \neg\phi_1$  iff  $\sigma^k \not\models \phi_1$

- $\sigma^k \models \phi_1 \mathbf{U}^t \phi_2$  iff there exists  $i \in \mathbb{N}$  such that
  1.  $\sum_{0 \leq l < i} t_{k+l} \leq t$ ,
  2.  $\sigma^{k+i} \models \phi_2$ , and
  3. for each  $0 \leq j < i$ ,  $\sigma^{k+j} \models \phi_1$

A probabilistic bounded linear temporal logic (PBLTL) formula is a formula of the form  $P_{\geq\theta}[\phi]$ , where  $\phi$  is a BLTL formula and  $\theta \in (0, 1)$  is a probability threshold. We denote that a model  $\mathcal{M}$  satisfies PBLTL property  $P_{\geq\theta}[\phi]$  as  $\mathcal{M} \models P_{\geq\theta}[\phi]$ , which means that a probability for  $\mathcal{M}$  to satisfy  $\phi$  is greater than or equal to  $\theta$  (see [23] for detailed description).

### 2.3 Sequential Probability Ratio Test

Sequential probability ratio test (SPRT) which is one of the hypothesis testing techniques was introduced by Younes et al. [21]. SPRT [21, 19, 20, 17] determines a number of required sample paths dynamically at runtime. A main goal of SPRT is to decide if  $\mathcal{M} \models P_{\geq\theta}[\phi]$  with a small number of sample paths. If another sample path is needed, SPRT generates one more sample path by executing a target system. If the information from generated sample paths are enough, SPRT stops executing a target program and produces an answer regarding  $\mathcal{M} \models P_{\geq\theta}[\phi]$ . SPRT uses precision parameter inputs *error bounds*  $\alpha$  and  $\beta$ , and a half size of *indifference region*  $\delta$ . The detailed description of SPRT is as follows.

Before building a hypothesis for hypothesis testing of SPRT, we introduce the indifference region. Basically, we build a hypothesis  $H : p \geq \theta$  against an alternative hypothesis  $K : p < \theta$  where  $\theta$  is a threshold over  $(0, 1)$  and  $p$  is a *true probability* that  $\mathcal{M}$  satisfies  $\phi$ . Hypothesis testing checks if  $H$  is accepted or not based on the randomly sampled paths. For testing a hypothesis  $H$ , there are two types of errors such as false negative (also known as a type I error) which rejects a true hypothesis  $H$  and false positive (also known as a type II error) which accepts a false hypothesis  $H$ . We can bound an error probability of a false negative error within  $\alpha$ . Similarly, we can bound an error probability of a false positive error within  $\beta$ . We call  $\alpha$  and  $\beta$  as error bounds. The left side of Figure 2 presents the function of probability  $L_p$  of accepting the hypothesis  $H$  as a function of  $p$  with the probability of a type I error and type II error as exactly  $\alpha$  and  $\beta$ . However, we want to give similar probability  $L_p$  of  $p = \theta$  and  $p = \theta - \epsilon$  for arbitrarily small  $\epsilon > 0$  for reality. To solve this problem, we introduce indifference region  $(p_1, p_0)$  around  $\theta$  where  $p_0 = \theta + \delta$ ,  $p_1 = \theta - \delta$ , and  $\delta$  is a half size of indifference region (see right side function in Figure 2). Therefore, instead of testing  $H$  against  $K$ , we use the modified hypothesis

$$H_0 : p \geq p_0$$

against the alternative hypothesis

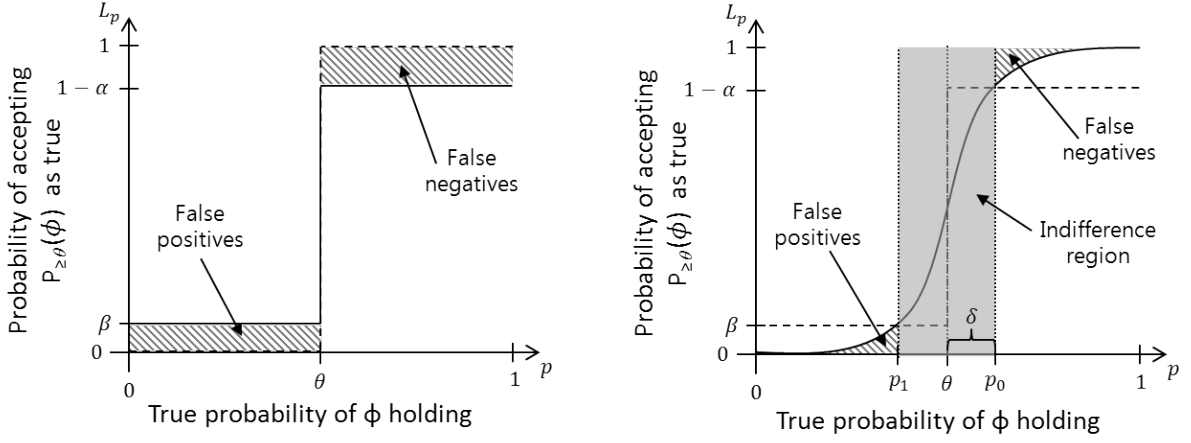
$$H_1 : p < p_1$$

If the probability  $p$  is in  $(p_1, p_0)$ , then  $p$  is sufficiently close to  $\theta$  so that we do not care which hypothesis is accepted.

Now, we describe the algorithm of SPRT. First, we obtain a sample path  $\sigma_i$  of a target system with simulating the target system and model-check if the sample path  $\sigma_i$  satisfies the given property  $\phi$  (see Section 2.1). After generating  $m$ th sample paths of the test, we calculate the quantity

$$\frac{p_{1m}}{p_{0m}} = \prod_{i=1}^m \frac{Pr[X_i = x_i | p = p_1]}{Pr[X_i = x_i | p = p_0]} = \frac{p_1^{d_m} (1 - p_1)^{m - d_m}}{p_0^{d_m} (1 - p_0)^{m - d_m}}$$

where  $d_m = \sum_{i=1}^m x_i$  and  $x_i$  is  $i$ th observation of  $\sigma_i \models \phi$ .  $p_{jm}$  is the probability of the sequence  $x_1, \dots, x_m$  with  $Pr[X_i = 1] = p_j$



**Figure 2: Function of probability  $L_p$  of accepting the hypothesis  $H : p \geq \theta$  (left side) and function of probability  $L_p$  of accepting the hypothesis  $H_0 : p \geq p_0$  with indifference region (right side).**

for  $j=0,1$ . Therefore, the above quantity makes the ratio of two probabilities, the *probability ratio*. The hypothesis  $H_0$  is accepted if

$$\frac{p_{1m}}{p_{0m}} \leq B,$$

and the hypothesis  $H_1$  is accepted if

$$\frac{p_{1m}}{p_{0m}} \geq A.$$

Otherwise, we should generate  $m + 1$ th sample path of the test.  $A$  and  $B$  are selected to bound error probability  $\alpha$  and  $\beta$ , with  $A > B$ . In practice, we choose  $A = \frac{1-\beta}{\alpha}$  and  $B = \frac{\beta}{1-\alpha}$  (detailed description is found in [17, 19]).

Note that SPRT can be imprecise with same indifference region value  $\delta$  when the threshold  $\theta$  is close to 1. The reason for the imprecise result of SPRT is due to the limited size of indifference region. For example, if the threshold  $\theta$  is 0.99 and  $\delta \geq 0.01$ , then  $p_0$  becomes 1, which causes the denominator of the probability ratio  $\frac{p_{1m}}{p_{0m}}$  to be 0 when one false sample path occurs, which can cause imprecise result. Therefore,  $\delta$  should be very small when  $\theta$  is close to 1, which requires large number of samples.

## 2.4 Bayesian Interval Estimation Testing

Bayesian interval estimation testing (BIET) is an estimation testing based SMC technique. Estimation testing can approximately compute  $p$ , the probability that the model  $\mathcal{M}$  satisfies the given property  $\phi$  expressed by bounded linear temporal logic (BLTL). With  $p$ , we can determine if the probabilistic bounded linear temporal logic (PBLTL) is satisfied. For that purpose, we use a following statistical estimation testing technique.

BIET [23] dynamically determines the number of sample paths for checking the satisfiability of the model  $\mathcal{M}$  with the property  $\phi$  during simulation as SPRT does. In Bayes' theorem, we get prior probability using current information first. After obtaining new information, we can obtain posterior probability refining prior probability. BIET uses the Bayes' theorem to determine the number of sample paths of the test.

BIET uses four precision parameter inputs such as a half-size  $\delta'$  of an estimation interval which will contain  $p$  with high probability, the coverage goal  $c$  of the estimation interval, and the parameters  $\alpha', \beta'$  of the Beta prior. In fact, BIET estimates interval around the probability  $p$  instead of estimating  $p$ , but we regard the mean of the

estimated interval as  $\hat{p}$ , the estimated value of *true probability*  $p$ , i.e., the estimated interval is  $(\hat{p} - \delta', \hat{p} + \delta')$ . We call the estimated interval as  $(t_0, t_1)$ . We have a *coverage goal* such that the probability that the probability satisfying  $\mathcal{M} \models \phi$  is in  $(t_0, t_1)$  should be over the coverage  $c \in (\frac{1}{2}, 1)$ . The exact description of the coverage goal is as follows:

$$\int_{t_0}^{t_1} f(u|x_1, \dots, x_n) du = c$$

where  $x_i$  is  $i$ th observation of  $\sigma_i \models \phi$  for  $i = 1, \dots, n$  and  $n$  is the number of sample paths. We call the coverage goal as a 100c percent *Bayesian interval estimate* of  $p$ . Since BIET uses the Bayes' theorem, we need prior information, i.e., prior density of  $p$  to obtain prior probability. For simplicity, we focus on the Beta prior with parameters  $\alpha', \beta'$  (See [23] for details).

At  $m$ th stage of the test, by Beta prior with  $\alpha', \beta'$ , we can calculate the quantity

$$\hat{p} = \frac{x + \alpha'}{m + \alpha' + \beta'}$$

where  $x = \sum_{i=1}^m x_i$  is the number of success sample paths during  $m$  number of sample paths. Next, using  $t_0 = \hat{p} - \delta', t_1 = \hat{p} + \delta'$ , we can calculate the quantity

$$\gamma = \int_{t_0}^{t_1} f(u|x_1, \dots, x_m) du$$

where  $\gamma$  is the coverage of  $m$  number of sample paths for checking  $\mathcal{M} \models \phi$ . If  $\gamma \geq c$ , then BIET stops the simulation and outputs  $t_0, t_1$ , and  $\hat{p}$ . Otherwise, BIET generates  $m + 1$ th sample path and repeats.

Note that BIET is fast when the estimated probability  $\hat{p}$  is close to 0 or 1 [23], whereas BIET is extremely slow (i.e., extremely larger number of samples is required) when  $\hat{p}$  is close to  $\frac{1}{2}$ . With this advantage of BIET, BIET can easily apply the problem for safety critical system since the probability standard of satisfiability for safety critical system should be usually close to 1 or 0.

## 3. HYBRID SMC ALGORITHM

We develop a hybrid SMC technique to improve both efficiency and effectiveness by combining SPRT whose verification speed is fast (i.e., small number of samples is required) and BIET whose

verification precision is high (i.e., the number of false positive and false negative results is small) [9]. Algorithm 1 describes how the hybrid SMC technique checks if a target system model  $\mathcal{M}$  satisfies a given property  $\phi$  in BLTL for a given probability threshold  $\theta$ <sup>1</sup> with precision parameters  $par_S$  for SPRT and  $par_B$  for BIET. The algorithm first applies SPRT multiple times with dynamically increasing probability threshold  $\theta_{SPRT}$  until a verification result is ‘reject’ (lines 15–18) or  $\theta_{SPRT}$  becomes larger than or equal to a user-given threshold  $th_{S2B}$  where  $0.5 < th_{S2B} \leq \theta$  (lines 5–20). If  $\theta_{SPRT}$  becomes larger than or equal to  $th_{S2B}$ , the algorithm applies BIET to obtain a precise verification result (lines 21–34).

The detail of the algorithm is as follows. First, the algorithm calls  $SPRT()$   $m_S$  times (lines 6–10), which applies SPRT to  $\mathcal{M}$  with regard to  $\phi$  and  $\theta_{SPRT}$  with  $par_S$  (line 8). A result of  $SPRT()$  is ‘accept’ (i.e., 1) or ‘reject’ (i.e., 0). After  $m_S$  trials of  $SPRT()$ , the algorithm calculates an average accept decision value  $accept_{avg}$  over the  $m_S$  trials (line 11). If  $accept_{avg}$  is less than a user-given accept decision threshold  $th_{acpt}$ , the algorithm decides that the verification result of  $\mathcal{M} \models P_{\geq \theta}(\phi)$  is ‘reject’ (line 16) and terminates (line 18). Otherwise (i.e.,  $accept_{avg} \geq th_{acpt}$ ), the algorithm increases  $\theta_{SPRT}$  from the initial value 0.5 (line 3) to 0.75, 0.875, 0.9375 and so on (line 14) until  $\theta_{SPRT}$  becomes larger than or equal to  $th_{S2B}$  through the while loop in lines 5–20.

If  $\theta_{SPRT}$  becomes larger than or equal to a user-given probability threshold  $th_{S2B}$  for applying BIET, the algorithm calls  $BIET()$  for  $m_B$  times (lines 23–27), which applies BIET to  $\mathcal{M}$  for  $\phi$  with precision parameters  $par_B$  (line 25). Based on the estimated probability  $p$  obtained from  $BIET()$ , the algorithm calculates an average estimated probability  $p_{avg}$  over the  $m_B$  trials (line 28). If  $p_{avg}$  is greater than or equal to  $\theta$ , then the algorithm decides that the verification result is ‘accept’ (lines 29–30); ‘reject’, otherwise (lines 31–32).

Note that the hybrid SMC algorithm can save a large amount of time cost compared to BIET, if a probability for  $\mathcal{M}$  to satisfy  $\phi$  is far from a given probability threshold  $\theta$ . For example, if the probability is less than 0.5, the algorithm terminates after executing  $SPRT()$  only once without executing  $BIET()$  whose time cost is very high (see Table 3). The algorithm executes  $BIET()$  if the probability is close to  $\theta$  (which is usually close to 1 for requirement properties of safety critical systems), which is necessary since SPRT becomes imprecise when  $\theta$  is close to 1 (Section 2.3).

#### 4. TARGET SAFETY CRITICAL SYSTEMS

This section presents an overview of the following three safety critical systems in automobile domain:

- Automatic transmission control system (ATCS) [13]
- Anti-lock braking system (ABS) [1]
- Fault-tolerant fuel control system (FFCS) [12]

We selected these systems as target systems to apply SPRT, BIET, and the hybrid statistical model checking (SMC) technique (Section 3) for the following reasons:

- These three automobile systems are safety critical systems whose reliability is very important. Many researchers are working to address the reliability issues on safety critical systems [12, 1, 13].

<sup>1</sup>We assume that  $\theta$  is close to 1, since we develop a hybrid SMC algorithm for safety critical systems whose reliability criteria are very high and, thus, requirement properties are given with high threshold values.

#### Input:

$\mathcal{M}$ : a model  
 $\phi$ : BLTL property  
 $\theta$ : probability threshold of  $\mathcal{M} \models \phi$   
 $par_S$ : precision parameters of SPRT  
 $par_B$ : precision parameters of BIET  
 $th_{acpt}$ : accept decision threshold over [0,1]  
 $th_{S2B}$ : probability threshold to change from SPRT to BIET  
 $m_S$ : a number of trials for SPRT  
 $m_B$ : a number of trials for BIET

#### Output:

$answer$ : result of  $\mathcal{M} \models P_{\geq \theta}(\phi)$   
 $p_{avg}$ : average estimated probability of  $\mathcal{M} \models \phi$  by BIET if BIET is applied; N/A otherwise

```

1  $SMC_{hyb}(\mathcal{M}, \phi, \theta, par_S, par_B, th_{acpt}, th_{S2B}, m_S, m_B)\{$ 
2  $accept_{sum} = 0;$  // sum of accept decisions by SPRT
3  $\theta_{SPRT} = 0.5;$  // initial probability threshold for SPRT
4 // SPRT for fast verification
5 while  $\theta_{SPRT} < th_{S2B}$  do
6   for  $i = 1 \rightarrow m_S$  do
7     // Checks  $\mathcal{M} \models P_{\geq \theta_{SPRT}}(\phi)$  using SPRT
8      $accept = SPRT(\mathcal{M}, \phi, \theta_{SPRT}, par_S);$ 
9     Add  $accept$  to  $accept_{sum};$ 
10  end
11  $accept_{avg} = accept_{sum}/m_S;$ 
12 if  $accept_{avg} \geq th_{acpt}$  then
13   // next probability threshold for SPRT
14    $\theta_{SPRT} = \theta_{SPRT} + (1 - \theta_{SPRT})/2;$ 
15 else
16    $answer = 'reject';$ 
17    $p_{avg} = N/A;$ 
18   return  $answer$  and  $p_{avg};$ 
19 end
20 end
21 // BIET for precise verification
22  $p_{sum} = 0;$  // sum of estimated probabilities by BIET
23 for  $i = 1 \rightarrow m_B$  do
24   // Checks  $\mathcal{M} \models \phi$  using BIET
25    $p = BIET(\mathcal{M}, \phi, par_B);$ 
26   Add  $p$  to  $p_{sum};$ 
27 end
28  $p_{avg} = p_{sum}/m_B;$ 
29 if  $p_{avg} \geq \theta$  then
30    $answer = 'accept';$ 
31 else
32    $answer = 'reject';$ 
33 end
34 return  $answer$  and  $p_{avg};$ 
35 }
```

Algorithm 1: Hybrid SMC algorithm

- The three automobile systems are complex real-world applications, not a toy example such as ones in probabilistic symbolic model checker (PRISM) [11] benchmarks.
- Simulink/stateflow models of the three automobile systems are publicly available in Matlab R2010a. Thus, it is convenient to build a prototype tool for the SMC techniques by using a Simulink/stateflow simulator.

#### 4.1 Automatic Transmission Control System

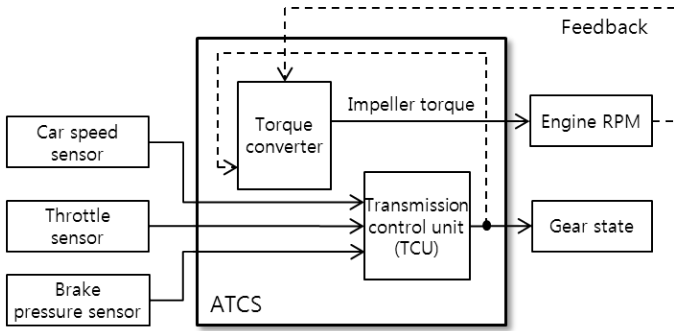


Figure 3: Block diagram of ATCS

An automatic transmission control system (ATCS) is a safety critical system which changes an engine gear automatically to drive smoothly. A main task of ATCS is to select a proper engine gear. As described in Figure 3, ATCS receives inputs regarding car speed, throttle, brake pressure (and engine RPM as a feedback) and calculates an engine RPM and a gear state. ATCS consists of a torque converter and a transmission control unit. The torque converter calculates an impeller torque value to deliver power to control the engine RPM based on the engine RPM and the gear state (i.e., if the impeller torque increases/decreases, the engine RPM increases/decreases). With the sensor inputs on car speed, throttle, and brake pressure, transmission control unit (TCU) selects a proper gear. Based on throttle and brake pressure values, TCU calculates a up-threshold and a down-threshold of a car speed. If a current car speed is greater than the up-threshold or less than the down-threshold, TCU changes the engine gear to keep the engine RPM in safe range (i.e., less than 6000).

The size and complexity of the Simulink/stateflow ATCS model in terms of the Halstead metrics [4] are described in Table 1. We counted each atomic block (i.e., a module of a mathematical function or control logic) as an operator and each input of an atomic block as an operand of the Simulink/stateflow ATCS model. The automatically generated C code from the model has 2353 LOC in 71 functions.

A requirement property for ATCS is that the probability that the engine RPM is less than 6000 for 30 seconds<sup>2</sup> should be greater than or equal to  $\theta$ . The property is important in real world, because if the engine RPM is constantly over 6000, the engine becomes overheated and can be damaged. The property can be expressed in PBLTL as follows:

$$P_{\geq\theta}[G^{30}(\text{engineRPM} < 6000)]$$

## 4.2 Anti-lock Braking System

An anti-lock braking system (ABS) is a safety system that repeatedly increases and decreases the brake pressure to allow the wheels of a car to interact with the road surface continuously as directed by a driver while braking. Thus, ABS can prevent the wheels from locking up and avoid skidding, which can enhance the safety of driving by improving vehicle control and decreasing stopping distances. As described in Figure 4, ABS has the following three sensors: a car speed sensor, a wheel speed sensor, and a brake pedal sensor. ABS receives data from these sensors and generates the brake pressure and slip as outputs, where slip indi-

<sup>2</sup>We set the time duration to monitor as 30 seconds, since a default simulation time of the Simulink model of ATCS included in Matlab R2010a is 30 seconds.

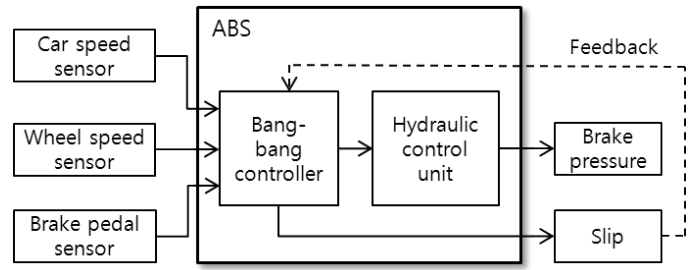


Figure 4: Block diagram of ABS

cates how properly a wheel of a car is controlled. ABS consists of a bang-bang controller and a hydraulic control unit. The bang-bang controller receives data from the three input sensors and commands the hydraulic control unit to increase/decrease the brake pressure. In addition, when the brake pedal is pressed, the bang-bang controller calculates slip as follows:

$$\text{slip} = 1 - \frac{\text{wheelspeed}}{\text{carspeed}}$$

When the wheel speed is equal to the car speed, slip becomes zero. When the wheel speed is zero (i.e., the wheel is locked), slip becomes one, which means that the driver loses his control of the car. There is an ideal slip value (which is 0.2) that maximizes the adhesion between the wheel and the road and minimizes the stopping distance with available friction. The bang-bang controller tries to adjust slip close to the ideal slip value by controlling the hydraulic control unit.

The size and complexity of the Simulink/stateflow ABS model in terms of the Halstead metrics are described in Table 1. The automatically generated C code from the model has 3443 LOC in 27 functions.

A requirement property for ABS is that the probability that, for 17 seconds<sup>3</sup>, when the brake pedal is pressed and the car speed is greater than 5 m/s, slip is less than or equal to 0.9, should be larger than or equal to  $\theta$ . The property is important in real world, because if slip becomes close to 1 when a car is driving, the wheel can be locked and a driver loses control of the car. The property can be expressed in PBLTL as follows:

$$P_{\geq\theta}[G^{17}((\text{brakepressed} \wedge \text{carspeed} > 5) \rightarrow \text{slip} \leq 0.9)]$$

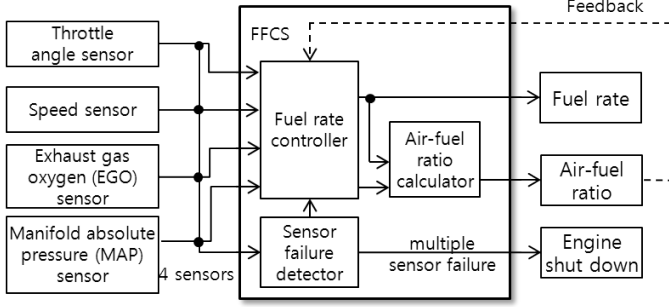
## 4.3 Fault-tolerant Fuel Control System

Figure 5 is an overall diagram of a fault-tolerant fuel control system (FFCS). FFCS [12] controls a fuel rate to inject fuel based on sensor data for best performance, detects a sensor fault, and shuts down an engine for safety in the presence of multiple sensor failures. FFCS has the following four sensors: throttle angle sensor, speed sensor, exhaust gas oxygen (EGO) sensor, and manifold absolute pressure (MAP) sensor. FFCS receives these four sensor inputs and generates a proper fuel rate and an air-fuel ratio. FFCS consists of the following three components: a fuel rate controller, an air-fuel ratio calculator, and a sensor failure detector. The fuel rate controller receives the four sensor data and calculates a proper fuel rate to make an air-fuel ratio optimal (i.e., 14.6). The air-fuel ratio calculator receives EGO sensor data and a fuel rate and calculates the air-fuel ratio. The sensor failure detector receives all four

<sup>3</sup>We set the time duration to monitor as 17 seconds, since a default simulation time of the Simulink model of ABS included in Matlab R2010a is 17 seconds.

**Table 1: Size and complexity of the Simulink models of ATCS, ABS, and FFCS in Halstead metrics**

Target system	$N_1$ : # of operators	$N_2$ :# of operands	$n_1$ :# of distinct operators	$n_2$ :# of distinct operands	$N$ :program length (= $N_1 + N_2$ )	$n$ : program vocabulary (= $n_1 + n_2$ )	$V$ : program volume ( $N \times \log n$ )	$D$ : program difficulty (= $n_1/2 \times N_2/n_2$ )	$E$ : program effort (= $D \times V$ )
ATCS	31	46	27	39	77	66	465.4	15.9	7410.9
ABS	27	36	19	36	63	55	364.2	9.5	3460.1
FFCS	65	111	35	94	176	129	1234.0	20.7	25500.0



**Figure 5: Block diagram of FFCS**

sensor data and controls the fuel rate controller to increase/decrease the fuel rate in the presence of a single sensor fault or shuts down the engine if multiple sensors fail, since the air-fuel ratio cannot be controlled with failures of multiple sensors.

The size and complexity of the Simulink/stateflow FFCS model in terms of the Halstead metrics are described in Table 1. The automatically generated C code from the model has 8266 LOC in 222 functions.

A requirement property for FFCS is that a probability that the fuel rate does not become zero for one second in 100 seconds should be greater than equal to threshold  $\theta$ . The property is crucial in a real world, because if the fuel rate is zero for one second, then the engine stops and can cause a serious accident. This property can be expressed by PBLTL as follows [23]:

$$P_{\geq \theta}[\neg(F^{100}G^1(\text{fuelrate} = 0))]$$

## 5. EXPERIMENTAL STUDY

We have applied SPRT, BIET, and the hybrid SMC technique to ATCS, ABS, and FFCS with precision parameters as independent variables to check if these target systems satisfy the given requirement properties in PBLTL. In addition, we have compared the results of the hybrid SMC technique with the results of SPRT and BIET. We used Simulink/stateflow models of the three systems included in the Matlab R2010a example directory.

### 5.1 Experiment Setup

#### 5.1.1 Environment Setup

We used the input value generation modules provided in the Simulink/stateflow models of FFCS, ATCS, and ABS without modification. In addition, we built the stochastic environments for the three automobile systems as follows:

- **ATCS**: we built a stochastic environment to ATCS by modeling a random delay to transfer the engine RPM value from

the engine to the torque converter.<sup>4</sup> This random delay is modeled by exponential distribution [10]. We selected a ‘passing maneuver’ scenario from the options of the ATCS model, which simulates a situation that a driver opens the throttle 100% after 15 seconds. We utilize the following four delay rates (i.e., mean delay times of transmission in seconds)  $\lambda \in \{0.01, 0.02, 0.03, 0.04\}$ .

- **ABS**: we built a stochastic environment of ABS that generates random delay to the command from the bang-bang controller to the hydraulic control unit.<sup>5</sup> The random delay of the command is modeled by exponential distribution [10]. We use a model of ABS representing a single wheel, which can be duplicated multiple times to create a model for a multi-wheel vehicle. We utilize the following four delay rates (in seconds)  $\lambda \in \{0.001, 0.003, 0.005, 0.007\}$ .
- **FFCS**: we built a stochastic environment model for FFCS that generates random faults at the EGO, MAP, and speed sensors as Zuliani et al. [23] did. The random faults are modeled by three independent Poisson processes with different arrival rates [18]. We assume one fault event remains for one second. When a fault event occurs in a sensor, FFCS remains in a failure mode in one second and returns to a normal mode. We utilize the following four inter-arrival fault rates (i.e., mean inter-arrival times of sensor fault) to the three sensors: (3,7,8), (10,8,9), (20,10,20) and (30,30,30).

#### 5.1.2 Precision Parameter Setup

We use the following precision parameters for SPRT and BIET:

- **SPRT**:
  - a half-size of indifference region  $\delta \in \{0.01, 0.03, 0.05\}$
  - error bounds  $\alpha, \beta \in \{0.1, 0.01, 0.001\}$
- **BIET**:
  - interval coverage  $c \in \{0.9, 0.99, 0.999\}$
  - a half-size of estimation interval  $\delta' \in \{0.01, 0.03, 0.05\}$
  - parameters of Beta prior  $\alpha' = \beta' = 1$  (since we assume the prior density to be a uniform density over (0, 1))

<sup>4</sup>This random delay is a real factor, not an artificial one. ATCS has an electronic circuit to deliver data from one sub-component to another and the data transfer can be delayed non-deterministically due to non-deterministic scheduling and bus contention among multiple sub-component.

<sup>5</sup>This random delay is a real factor for the similar reason of the one in ATCS.

**Table 2: Experiment result of SPRT for ATCS with  $\lambda=0.03$  and  $\delta = 0.03$  for the five trials**

$\alpha, \beta$	threshold $\theta$											
	0.5			0.7			0.9			0.99		
	$n$	$acpt$	$time$	$n$	$acpt$	$time$	$n$	$acpt$	$time$	$n$	$acpt$	$time$
0.1	110	1.0	69.9	215	1.0	143.9	343	0.0	221.8	18	1.0	12.6
0.01	270	1.0	171.0	375	1.0	301.1	410	0.0	347.1	41	1.0	27.1
0.001	395	1.0	249.0	563	1.0	361.1	985	0.0	636.7	45	1.0	30.2

**Table 3: Experiment result of BIET for ATCS with  $\lambda = 0.03$  for the five trials**

$\delta'$	interval coverage $c$								
	0.9			0.99			0.999		
	$n$	$\hat{p}$	$time$	$n$	$\hat{p}$	$time$	$n$	$\hat{p}$	$time$
0.05	630	0.8594	416.6	1550	0.8654	1011.9	2665	0.8636	1753.2
0.03	1845	0.8544	1208.6	3340	0.9000	2181.1	6475	0.8805	4356.5
0.01	14150	0.8810	9551.8	36540	0.8740	26281.2	58870	0.8762	42945.1

We performed each experiment five times to obtain average verification result over  $[0, 1]$  regarding if the hypothesis  $H_0$  is accepted where  $H_0$ : a probability for  $\mathcal{M}$  to satisfy  $\phi$  is greater than or equal to  $\theta + \delta$ . For the experiments, we used  $\theta \in \{0.5, 0.7, 0.9, 0.99\}$ . In addition, we measured the total verification time and total number of samples for each experiment.

For the hybrid SMC technique, we set  $\theta=0.99$ . This is because the hybrid SMC technique targets safety critical systems which require high reliability, which can be specified with PBLTL with high  $\theta$  values. We use the following precision parameters which are similar to those of the SPRT and BIET experiments:

- precision parameters for SPRT  $par_S$ :  $\delta \in \{0.01, 0.03, 0.05\}$ ,  $\alpha, \beta \in \{0.1, 0.01, 0.001\}$ .
- precision parameters for BIET  $par_B$ :  $c \in \{0.9, 0.99, 0.999\}$ ,  $\delta' \in \{0.01, 0.03, 0.05\}$ ,  $\alpha' = \beta' = 1$ .
- threshold for accept decision over  $[0, 1]$   $th_{acc}=0.5$
- the probability threshold to apply BIET instead of SPRT  $th_{S2B}=0.95$
- the number of trials for SPRT  $m_S = 5$
- the number of trials for BIET  $m_B = 5$

### 5.1.3 Experiment Platform

We built a statistical model checker as a Matlab module, which executes the Simulink/stateflow models for FFCS, ATCS, and ABS and monitors inputs and outputs of the models to check if  $\phi$  is satisfied on a current sample path. After each execution of the models, the SMC module calculates a required number of samples dynamically based on the precision parameters and the number of success/fail samples generated so far. If a number of the generated samples reaches the required number, the SMC module generates a verification result. The SMC module for SPRT is around 80 lines long. The SMC module for BIET is around 70 lines long. The

hybrid SMC module is around 200 lines long. We used Matlab R2010a for the experiments. All experiments were performed on 64 bit Windows 7 Professional equipped with a 3 GHz Intel processor and 16 gigabytes of memory.

## 5.2 Results of SPRT and BIET

Tables 2 and 3 describe the experiment results of applying SPRT with  $\delta = 0.03$  and BIET to ATCS respectively when the delay rate  $\lambda=0.03$ .<sup>6</sup> In Tables 2 and 3,  $n$  is a total number of required sample execution paths for the five trials and  $time$  is total verification time taken for the five trials in seconds.  $acpt$  in Table 2 is an average result over  $[0, 1]$  regarding the hypothesis  $H_0$  where 0 is ‘reject’ and 1 is ‘accept’.  $\hat{p}$  in Table 3 is an estimated probability for  $\mathcal{M} \models \phi$ .

Table 2 shows that the probability for ATCS with  $\lambda=0.03$  and  $\delta = 0.03$  to satisfy the requirement property  $\phi$  ( $=G^{30}(engineRPM < 6000)$ ) is between 0.7 and 0.9. This is because  $acpts$  are 1.0 when  $\theta \leq 0.7$  while  $acpts$  are 0.0 when  $\theta = 0.9$  in Table 2 (the verification result of SPRT with a high  $\theta$  value like 0.99 should not be trusted due to the characteristics of SPRT [21]).

In addition, we can conclude that the probability is close to 0.9, since  $n$  of SPRT increases as  $\theta$  increases from 0.5 to 0.9 and decreases sharply from 0.9 to 0.99. For example, Table 2 shows that  $n$  becomes 110, 215, 343, and 18 as  $\theta$  becomes 0.5, 0.7, 0.9, and 0.99 with  $\alpha=\beta=0.1$ . This tendency of  $n$  indicates that the true probability for ATCS with  $\lambda=0.03$  to satisfy  $\phi$  is close to 0.9, since SPRT requires a large number of sample paths to check a given hypothesis  $H_0$  if a true probability is close to  $\theta$  [21]. Furthermore, the verification result of BIET coincides with that of SPRT, since Table 3 shows that the estimated probability  $\hat{p}$  is between 0.8544 (with  $c = 0.9$  and  $\delta' = 0.03$ ) and 0.9000 (with  $c = 0.99$  and  $\delta' = 0.03$ ).

<sup>6</sup>Due to page limit, we cannot describe full experiment data in the paper. Full experiment data of applying SPRT and BIET to ATCS, ABS, and FFCS is available at <http://pswlab.kaist.ac.kr/data/issre2012-expr-results.zip>

For the verification speed, Tables 2 and 3 show that SPRT is much faster than BIET. For example, the maximum time spent by SPRT in Table 2 is 636.7 seconds with  $\theta = 0.9$  and  $\alpha=\beta=0.001$ , which is less than time costs of BIET in Table 3 except when BIET is applied with low precision parameters  $\delta' = 0.05$  and  $c = 0.9$  (416.6 seconds).

Thus, if a given PBLTL formula has a high  $\theta$  value like 0.99, it is a good idea to apply SPRT first with low  $\theta$  values (SPRT result with high  $\theta$  value should not be trusted) in hope to eliminate the necessity of applying BIET. For example, suppose that we should check  $P_{\geq\theta}[G^{30}(\text{engineRPM} < 6000)]$  for ATCS with  $\lambda=0.03$  and  $\theta = 0.99$ . With  $\alpha=\beta=0.1$ , SPRT takes 435.6 seconds in total ( $=69.9+143.9+221.8$ ) to conclude that ATCS does not satisfy the given PBLTL formula with  $\theta = 0.99$  by checking cases with  $\theta$  as 0.5, 0.7, and 0.9 in order (Table 2); the verification result with  $\theta = 0.9$  is ‘reject’, which makes the result with  $\theta = 0.99$  as ‘reject’ consequently. However, if we apply BIET, we will obtain the same verification result with higher time cost except a case with  $\delta' = 0.05$  and  $c = 0.9$  (416.6 seconds (Table 3)). The hybrid SMC technique (Algorithm 1) is developed to utilize this observation for precise and fast verification.

### 5.3 Results of the Hybrid SMC Technique

Tables 4-6 present the experiment results of the hybrid SMC technique on ATCS, ABS, and FFCS for  $\theta = 0.99$  with  $\delta = 0.03$ ,  $\delta' = 0.01$ , and  $c = 0.99$ , respectively.  $n$  is a total number of sample paths required by SPRT and BIET in the hybrid algorithm for each experiment.  $\hat{p}$  is an estimated probability obtained by BIET for each experiment. If BIET is not applied because SPRT rejects a hypothesis  $H_0$  before reaching  $th_{S2B}$ , then  $\hat{p}$  is N/A. *acpt* is a result over  $[0,1]$  regarding the hypothesis  $H_0$  where 0 is ‘reject’ and 1 is ‘accept’. *time* is total verification time taken for each experiment in seconds.

#### 5.3.1 Verification Results

For ATCS, Table 4 shows that the corresponding hypothesis  $H_0$  with  $\theta = 0.99$  is accepted for two delay rates  $\lambda \in \{0.01, 0.02\}$  (i.e.,  $\mathcal{M} \models P_{\geq\theta}[G^{30}(\text{engineRPM} < 6000)]$ ) and rejected for delay rates  $\lambda \in \{0.03, 0.04\}$ . For the experiments with  $\lambda \in \{0.03, 0.04\}$ , SPRT rejected  $H_0$  and BIET was not applied; thus, corresponding  $\hat{p}$ s are marked as ‘N/A’. This result coincides with the results of SPRT and BIET, since SPRT concludes that ATCS with  $\lambda=0.03$  does not satisfy the PBLTL formula with  $\theta = 0.9$  (i.e., *acpts* are all 0.0 in Table 2) and BIET concludes that the probability for ATCS with  $\lambda=0.03$  to satisfy  $G^{30}(\text{engineRPM} < 6000)$  is between 0.8544 and 0.9000 (Section 5.2).

An interpretation of this result is that ATCS may not operate correctly if an engine RPM value is transferred from the engine to the torque converter with long delay (i.e., delay rate  $\lambda$  in exponential distribution is larger than or equal to 0.03 seconds), since long delay of the data transfer can prevent ATCS from operating promptly. In addition, we can obtain a practical implication that, to achieve required high reliability specified by the PBLTL formula with  $\theta = 0.99$ , ATCS should use a data-transfer component that transfers data from the engine to the torque converter with delay rate  $\lambda \leq 0.02$  or revise the ATCS design to satisfy the PBLTL formula with  $\theta = 0.99$  even with long delay of the data transfer.

Similarly, for ABS, Table 5 shows that the corresponding hypothesis  $H_0$  with  $\theta = 0.99$  is accepted for delay rate  $\lambda=0.001$  (i.e.,  $\mathcal{M} \models P_{\geq\theta}[G^{17}((\text{brakepressed} \wedge \text{carspeed} > 5) \rightarrow \text{slip} \leq 0.9))$ ), and is rejected for larger delay rates. For FFCS, Table 6 shows that the corresponding hypothesis  $H_0$  with  $\theta = 0.99$  is accepted for fault ratios (20,10,20) (except  $\alpha=\beta=0.001$ ) and

(30,30,30) (i.e.,  $\mathcal{M} \models P_{\geq\theta}[\neg(F^{100}G^1(\text{fuelrate} = 0))]$ ), and is rejected for more frequent fault ratios (3,7,8) and (10,8,9).

#### 5.3.2 Verification Speeds

The hybrid SMC technique shows an order of magnitude faster verification speed compared to BIET for the experiments where the probability for  $\mathcal{M} \models \phi$  is less than  $th_{S2B}$ .<sup>7</sup> For example, for ATCS with  $\lambda=0.03$ , the hybrid technique spent 698.9 seconds (with  $\alpha=\beta=0.1$ ,  $\delta=0.03$ ,  $\delta' = 0.01$ , and  $c=0.99$ ) to 6020.4 seconds (with  $\alpha=\beta=0.001$ ,  $\delta = 0.03$ ,  $\delta' = 0.01$ , and  $c=0.99$ ) (Table 4), while BIET spent 26281.2 seconds for the same precision parameters (i.e.,  $\delta' = 0.01$ ,  $c = 0.99$ ) (Table 3). The hybrid technique is much faster than BIET for ATCS with  $\lambda=0.03$ , since SPRT of the hybrid technique concludes that ATCS with  $\lambda=0.03$  does not satisfy the PBLTL formula with  $\theta_{SPRT} = 0.9375$ . Since  $\theta_{SPRT} = 0.9375 < th_{S2B} = 0.95$ , the hybrid technique does not apply BIET and conclude that ATCS with  $\lambda = 0.03$  does not satisfy the given PBLTL formula with  $\theta = 0.99$ . As BIET takes an order of magnitude larger time cost than SPRT (Tables 2–3), the hybrid technique can reduce a large amount of time cost by removing the time cost of BIET.

However, for the experiments where the probability for  $\mathcal{M} \models \phi$  is larger than  $th_{S2B}$ , the hybrid technique shows slower verification speed compared to BIET. For example, for ATCS with  $\lambda=0.02$ , the hybrid technique spent 1173.7 seconds (with  $\alpha=\beta=0.1$ ,  $\delta=0.03$ ,  $\delta'=0.01$ , and  $c=0.99$ ) to 2102.7 seconds (with  $\alpha=\beta=0.001$ ,  $\delta=0.03$ ,  $\delta'=0.01$ , and  $c=0.99$ ) (Table 4), while BIET spent 820.1 seconds for the same precision parameters (i.e.,  $\delta'=0.01$  and  $c=0.99$ ) (see <http://pswlab.kaist.ac.kr/data/issre2012-expr-results.zip>). This larger time cost of the hybrid technique is due to the additional applications of SPRT for  $\theta_{SPRT} \in \{0.5, 0.75, 0.875, 0.9375\}$ .

For ABS and FFCS, we make similar observations to the experiments for ATCS. For the cases where the probability for  $\mathcal{M} \models \phi$  is less than  $th_{S2B}$ , the hybrid technique is much faster than BIET. For the other cases, the hybrid technique is slower than BIET.

## 6. DISCUSSION

### 6.1 Effective and Efficient Hybrid SMC Technique

Through the empirical evaluation of the hybrid statistical model checking technique on ATCS, ABS, and FFCS, we found that the hybrid technique is faster and more accurate than a single SMC technique (Section 5.3). This improvement is achieved by utilizing the different advantages of SPRT and BIET selectively, namely fast verification speed of SPRT and precise verification result of BIET (Section 5.2).

The hybrid SMC technique applies SPRT and BIET selectively, because significance of verification speed and that of verification precision do vary depending on a probability  $p$  for  $\mathcal{M}$  to satisfy a requirement property  $\phi$ . Suppose that if  $p$  is distant from  $\theta$  (e.g.,  $|\theta - p| \geq 0.1$ ), precision may not be very important, because small error (e.g. +0.01 or -0.01) in an estimated probability does not affect an accept/reject decision on  $H_0$ . In this case, the hybrid technique applies SPRT for fast verification without concern to precision much. If  $p$  is close to  $\theta$ , however, precision becomes important, because a small error (e.g. +0.01 or -0.01) may affect an accept/reject decision on  $H_0$  easily. In this case, the hybrid technique

<sup>7</sup>Comparison between the verification speed of the hybrid technique and that of SPRT is not meaningful, since SPRT result is imprecise for a large  $\theta$  value like 0.99.



**Table 4: Experiment result of the hybrid SMC for ATCS with  $\theta = 0.99$ ,  $\delta = 0.03$ ,  $\delta' = 0.01$ ,  $c = 0.99$** 

$\alpha, \beta$	delay rate $\lambda$ from engine to torque convertor															
	0.01				0.02				0.03				0.04			
	$n$	$\hat{p}$	$acpt$	$time$	$n$	$\hat{p}$	$acpt$	$time$	$n$	$\hat{p}$	$acpt$	$time$	$n$	$\hat{p}$	$acpt$	$time$
0.1	1710	0.9956	1	1256.1	1710	0.9956	1	1173.7	1066	N/A	0	698.9	1334	N/A	0	858.9
0.01	2315	0.9956	1	1740.8	2315	0.9956	1	1642.6	4795	N/A	0	3081.9	2946	N/A	0	1884.6
0.001	2905	0.9956	1	2320.2	2905	0.9956	1	2102.7	7804	N/A	0	6020.4	3833	N/A	0	2952.4

**Table 5: Experiment result of hybrid SMC for ABS with  $\theta = 0.99$ ,  $\delta = 0.03$ ,  $\delta' = 0.01$ ,  $c = 0.99$** 

$\alpha, \beta$	delay rate $\lambda$ from bang-bang controller to hydraulic control unit															
	0.001				0.003				0.005				0.07			
	$n$	$\hat{p}$	$acpt$	$time$	$n$	$\hat{p}$	$acpt$	$time$	$n$	$\hat{p}$	$acpt$	$time$	$n$	$\hat{p}$	$acpt$	$time$
0.1	1814	0.9953	1	986.5	6511	0.9826	0	2905.9	8247	0.9773	0	3854.4	952	N/A	0	382.4
0.01	2417	0.9953	1	1344.8	8006	0.9806	0	3619.4	9151	0.9770	0	4290.1	2238	N/A	0	890.2
0.001	3179	0.9950	1	1815.3	8541	0.9810	0	3906.1	9326	0.9791	0	4334.0	3684	N/A	0	1465.5

**Table 6: Experiment result of hybrid SMC for FFCS with  $\theta = 0.99$ ,  $\delta = 0.03$ ,  $\delta' = 0.01$ ,  $c = 0.99$** 

$\alpha, \beta$	sensor fault rates															
	(3, 7, 8)				(10, 8, 9)				(20, 10, 20)				(30, 30, 30)			
	$n$	$\hat{p}$	$acpt$	$time$	$n$	$\hat{p}$	$acpt$	$time$	$n$	$\hat{p}$	$acpt$	$time$	$n$	$\hat{p}$	$acpt$	$time$
0.1	1299	N/A	0	3359.6	14442	0.9575	0	36399.3	3180	0.9920	1	7990.0	2121	0.9944	1	5362.0
0.01	5369	N/A	0	13893.4	14130	0.9620	0	35894.1	4651	0.9906	1	11786.0	3747	0.9926	1	9556.4
0.001	7320	N/A	0	19059.9	16010	0.9592	0	41014.6	5809	0.9895	0	14792.1	3512	0.9939	1	9017.2

applies BIET for precise verification result.

Since we are targeting safety critical systems where PBLTL requirements often have  $\theta$  values close to 1 (e.g., 0.99 or 0.999) for high reliability, the hybrid SMC technique can apply SPRT for relatively low  $\theta_{SPRT}$  values first (e.g., 0.5, 0.75, etc.) in hope to conclude a ‘reject’ decision fast with little concern to precision (a case where  $p$  is distant from  $\theta$ ). If SPRT concludes ‘accept’ decisions for the relatively low  $\theta_{SPRTS}$  (i.e., a case where  $p$  is close to  $\theta$ ), the hybrid SMC algorithm applies BIET for precise verification result. Therefore, the hybrid SMC technique can produce a final verification result (i.e., accept/reject of  $H_0$ ) fast and precisely.

## 6.2 Independence between Complexity of Target System and SMC Cost

Through the empirical study on ATCS, ABS, and FFCS whose complexities are different, we found that the complexity of a target system does not affect the cost of the hybrid SMC technique. For example, although FFCS is more complex than the other systems (e.g., program effort  $E$  of FFCS is 25500.0, while those of ATCS and ABS are 7410.9 and 3460.1 respectively (Table 1)), for similar estimated probability  $\hat{p}$  with the same precision parameters, a number of sample execution paths  $n$  for FFCS is similar to those for ATCS and ABS.<sup>8</sup> For example, for the five experiments with  $\alpha=\beta=0.1$  in Tables 4-6 whose  $\hat{p} > 0.99$ , the numbers of execution paths  $n$ s for these experiments are similar.

- ATCS with  $\lambda=0.01$  or  $0.02$ :  $\hat{p} = 0.9956$  and  $n = 1710$

<sup>8</sup>For different target systems, we should use  $n$  as a measure of the SMC cost, not  $time$ , since  $time$  varies depending on the execution time of a target system.

- ABS with  $\lambda=0.001$ :  $\hat{p} = 0.9953$  and  $n = 1814$
- FFCS with the sensor fault rates (30,30,30):  $\hat{p} = 0.9944$  and  $n = 2121$
- FFCS with the sensor fault rates (20,10,20):  $\hat{p} = 0.9920$  and  $n = 3180$

As shown above, although the complexities of ATCS, ABS, and FFCS are different, the costs of the hybrid SMC technique for these target systems do not change much for similar  $\hat{p}$  (i.e., 0.9920–0.9956). A slightly increasing number of  $n$  from 1710 to 3180 for decreasing  $\hat{p}$  from 0.9956 to 0.9920 is due to the characteristics of BIET; BIET requires a more number of sample paths as  $\hat{p}$  decreases from 1 (Section 2.4). Therefore, we can expect that SMC techniques can be applied to large complex safety critical systems to assure their reliability.

## 6.3 SMC Techniques to Obtain a Certificate of Safety Standards

There are various international standards (e.g., DO-178C [15] for avionics domain, ISO-26262 [7] for automobile domain, IEC-60601 [6] for medical electrical equipment domain, etc.) to assure reliability of safety critical systems. Since products with a certificate can have a strong competitive power in market, manufacturers spend a large amount of man power and project time to acquire a high-level certificate for safety standards. For example, automobile manufacturers such as BMW and GM start to apply ISO 26262 standard for safety critical components. To obtain a high-level certificate, vendors should provide strong cases or ‘proof’ that their products achieve high reliability. For example, ISO-26262 requires

that a vendor of automobile components should apply formal verification techniques to the components to obtain a certificate of automotive software integrity level (ASIL) D. However, conventional formal verification techniques such as state model checking and theorem proving are difficult to apply for the purpose due to the state space explosion problem and lack of field engineers who are proficient in deductive proof.

From our experience of applying various SMC techniques for safety critical systems on automobile domain such as ATCS, ABS, and FFCS, we expect that the hybrid SMC technique can be applied successfully to obtain a high-level certificate of ISO 26262. A main reason is that the hybrid SMC technique is reasonably fast and precise (Section 6.1). For example, it takes less than 12 hours to verify FFCS with most precise parameters with regard to the PBLTL formula. Since most of the time cost is due to the simulation cost, SMC itself will take much less time to check other PBLTL formulas if any by utilizing saved sample traces. Second reason is that the cost of the SMC techniques is independent of the complexity of a target system (Section 6.2), since SMC techniques do not analyze the complex internal logic of a target system.

## 7. CONCLUSION AND FUTURE WORK

We have developed a new hybrid SMC technique which integrates SPRT and BIET which have different advantages and weaknesses depending on the range of a probability for a target system to satisfy a given requirement. By applying this new hybrid technique to three safety critical systems on automobile domain (i.e., ATCS, ABS, and FFCS), we have demonstrated that the hybrid SMC technique achieves precise verification results fast compared to a single SMC technique - SPRT or BIET.

As future work, we will collaborate with Hyundai motor company to apply the hybrid SMC technique to real control components of automobiles. We believe that the hybrid technique can provide more scientific assurance about the reliability of components than conventional testing techniques. In addition, we plan to use this hybrid technique in a process to obtain an ISO-26262 certificate.

## Acknowledgments

This work was supported by the ERC of Excellence Program of Korea Ministry of Education, Science and Technology(MEST) / National Research Foundation of Korea) (Grant 2011-0000978), and Basic Science Research Program through the NRF funded by the MEST (2010-0005498). We would like to thank Yunho Kim and Shin Hong for their constructive comments on the experiments.

## 8. REFERENCES

- [1] D. Antic, V. Nikolic, and D. Mitic. Sliding mode control of anti-lock braking system: An overview. In *Automatic Control and Robotics*, volume 9, pages 41–58, 2010.
- [2] E. Clarke, A. Biere, R. Raimi, and Y. Zhu. Bounded model checking using satisfiability solving. In *J. Formal Methods Syst. Design (FMSD)*, pages 7–34, 2001.
- [3] E.M. Clarke and P. Zuliani. Statistical model checking for cyber-physical systems. In *ATVA*, volume 6996, pages 1–12, 2011.
- [4] M.H. Halstead. Elements of software science. In *Amsterdam: Elsevier North-Holland, Inc.*, 1977.
- [5] T. Herault, R. Lassaigne, F. Magniette, and S. Peyronnet. Approximate probabilistic model checking. In *VMCAI*, volume 2937, pages 73–84, 2004.
- [6] International Electrotechnical Commission (IEC). IEC 60601: Medical electrical equipment - part 1: General requirements for basic safety and essential performance, 2005.
- [7] International Organization for Standardization (ISO). ISO 26262: Road vehicles – functional safety, 2011. [http://www.iso.org/iso/catalogue\\_detail?csnumber=43464](http://www.iso.org/iso/catalogue_detail?csnumber=43464).
- [8] S.K. Jha, E.M. Clarke, C.J. Langmead, A. Legay, A. Platzer, and P. Zuliani. A bayesian approach to model checking biological systems. In *CMSB*, volume 5688, pages 218–234, 2009.
- [9] Y. Kim, M. Kim, and T. Kim. Statistical model checking for safety critical systems: An empirical evaluation. In *Intl. Conf. on Formal Engineering Methods (ICFEM)*, 2012. under review.
- [10] C.W. Kirhwood. System dynamics methods: A quick introduction. In *Arizona State University*, 1998.
- [11] M. Kwiatkowska, G. Norman, and D. Parker. Prism 4.0: Verification of probabilistic real-time systems. In *Int Proc. 23rd International Conference on Computer Aided Verification*, volume 6806, pages 585–591, 2011.
- [12] J. Lauber, T.M. Guerra, and M. Dambrine. Air-fuel ratio control in a gasoline engine. In *International Journal of Systems Science*, volume 42, pages 277–286, Feb. 2011.
- [13] G. Li and J. Hu. Modeling and analysis of shift schedule for automatic transmission vehicle based on fuzzy neural network. In *World Congress on Intelligent Control and Automation*, pages 4839–4844, July 2010.
- [14] Michael R. Lyu. Software reliability engineering: A roadmap. In *2007 Future of Software Engineering, FOSE '07*, pages 153–170, 2007.
- [15] Radio Technical Commission for Aeronautics (RTCA). Do-178c: Software considerations in airborne systems and equipment certification, 2012.
- [16] X. Teng, H. Pham, and D. R. Jeske. Reliability modeling of hardware and software interactions, and its applications. 55, 2006.
- [17] A. Wald. Sequential tests of statistical hypotheses. In *Annals of Mathematical Statistics*, volume 16, pages 117–186, Jun. 1945.
- [18] S. Yi, J. Heo, Y. Cho, and J. Hong. Adaptive mobile checkpointing facility for wireless sensor networks. In *ICCSA*, pages 701–709, 2006.
- [19] H.L.S. Younes. Verification and planning for stochastic processes with asynchronous events. In *PhD thesis, Carnegie Mellon*, Jan. 2005.
- [20] H.L.S. Younes, M. Kwiatkowska, G. Norman, and D. Parker. Numerical vs. statistical probabilistic model checking. In *Int. Journal on Software Tools for Technology Transfer*, volume 8, pages 216–228, 2006.
- [21] H.L.S. Younes and D.J. Musliner. Probabilistic plan verification through acceptance sampling. In *AIPS Workshop on Planning via Model Checking*, pages 81–88, 2002.
- [22] H.L.S. Younes and R.G. Simmons. Statistical probabilistic model checking with a focus on time-bounded properties. In *Information and Computation*, volume 204, pages 1368–1409, 2006.
- [23] P. Zuliani, A. Platzer, and E.M. Clarke. Bayesian statistical model checking with application to stateflow/simulink verification. In *HSCC*, pages 243–252, 2010.