Testing a probabilistic FSM using interval estimation

22 June 2010

Iksoon Hwang and Ana Cavalli
TELECOM SudParis
Contents

■ Problem statement & Preliminaries
■ Test generation
■ Testing of probability
■ Test sequence repetition numbers
■ Testing of a PFSM
■ Fault coverage evaluation
■ Application: A probabilistic non-repudiation protocol
■ Conclusions
Problem statement

- **Conformance testing**
  - We do not know the internal details of implementations.
  - An implementation is considered as a black-box.
  - Test sequences are derived from its specification so that they can check the conformance of an implementation with respect to the specification.

- **How to model?**
  - State machines, e.g. FSM (Finite-State Machine) and FA (Finite Automata) have been widely used to model system behaviors.
  - State machines can be either deterministic or nondeterministic.
  - A number of extension has been done by adding data (Extended FSM), time (timed automata, TEFSM), and probability (Probabilistic FSM, Probabilistic FA).
Problem statement

Testing problem

- A lot of publications for testing DFSMs and NFSMs.
- A number of work have been done for testing timed systems and probabilistic systems but they are still under study.
- Most of the work on probabilistic models focused on model checking.
- There are relatively few studies on testing whether the probabilities of an implementation are correctly implemented with respect to its specification.

In this work, the following questions are considered.

- How to generate test cases from PFSMs?
- How can we test the probabilities of an implementation that is modeled by a PFSM?
- What can be addressed after testing?
- How many times each test should be repeated when we test probabilities?
Preliminaries

 Definitions

• A PFSM $M$ is defined by 5-tuple $(S, Li, Lo, s_0, P_T)$ where
  - $S$ is a finite set of states;
  - $Li$ is the input alphabet;
  - $Lo$ is the output alphabet and may include an empty output $\lambda$;
  - $s_0$ is the initial state;
  - $P_T$ assigns transition probabilities $P_T : (S \times Li \times S \times Lo) \to [0,1]$ where a transition $t \in S \times Li \times S \times Lo$.

• For a given state and input pair $(s_i, a) \in S \times Li$, transition probabilities satisfy $\sum_{s_j \in S, b \in Lo} P_T(s_i, a, s_j, b) = 1$.

• The function $P_T$ can be extended to $P_T^*$ to be applied to an input sequence $u \in Li^*$ and an output sequence $y \in Lo^*$.

• A PFSM $M$ is deterministic if there is no more than one transition for every state and input pair.
Preliminaries

- A PFSM $M$ is *completely specified* if for each state it has at least one transition for each input symbol.
- A PFSM $M$ is said to be *initially connected* if every state is reachable from the initial state of $M$.
- Two states $s_i$ and $s_j$ in a PFSM are *trace equivalent* written $s_i =_{\text{trace}} s_j$ if the two states produce the same set of possible output sequences for every input sequence.
- Two states $s_i$ and $s_j$ in a PFSM are *probabilistically trace equivalent* written $s_i =_{\text{ptrace}} s_j$ if the two states produce the same set of possible output sequences for every input sequence and have the same probability for each output sequence.
- A PFSM $M$ is said to be *minimal* if none of its states is probabilistically trace equivalent.
- Two PFSMs $M$ and $M_I$ with initial states $s_0$ and $i_0$ are said to be *probabilistically trace equivalent* if $s_0 =_{\text{ptrace}} i_0$. 
Preliminaries

- An input sequence $ts$ of finite length is said to be a test sequence.
- A test sequence $ts_1$ is said to be a prefix of $ts_2$ if $ts_2 = ts_1.ts_3$ for some test sequence $ts_3$ where $ts_1.ts_3$ is the concatenation of $ts_1$ with $ts_3$.
- For a given input sequence set $V \subseteq L_i^*$, two states $s_i$ and $s_j$ in a PFSM are $V$-equivalent written $s_i =_v s_j$ if two states produce the same set of possible output sequences for every input sequence $v \in V$.
- For a given input sequence set $V$, two states $s_i$ and $s_j$ in a PFSM are probabilistically $V$-equivalent written $s_i =_{pV} s_j$ if two states produce the same set of possible output sequences for every input sequence $v \in V$ and have the same probability for each output sequence.
- Two PFSMs $M$ and $M_i$ with initial states $s_0$ and $i_0$ are said to be probabilistically $V$-equivalent if $s_0 =_{pV} i_0$.
- A PFSM $M$ is said to be observable if for every state the next state is uniquely determined by the output symbol when an input symbol is applied.
Preliminaries

Assumptions

- An implementation is modeled by an unknown PFSM $M_I$ where the specification PFSM $M$ and $M_I$ have the same input alphabet.
- Both specification $M$ and implementation $M_I$ are completely specified, initially connected, minimal, and observable PFSMs.
- There is an upper bound $m$ on the number of states in a PFSM $M_I$.
- The probability of each transition is static and dependent on the current state, i.e. independent of the history that leads to the current state.
- A reliable reset $r$ is assumed to be present such that upon receiving $r$ in any state the machine returns to its initial state.
- Spontaneous transitions are not considered.
Contents

- Problem statement & Preliminaries
- Test generation
- Testing of probability
- Test sequence repetition numbers
- Testing of a PFSM
- Fault coverage evaluation
- Application: A probabilistic non-repudiation protocol
- Conclusions
Test generation

Transformation of a PFSM into an observable PFSM

- An algorithm is presented that transforms some classes of non-observable PFSMs into probabilistically trace equivalent observable PFSMs.

PFSM $M_1$

Observable PFSM $M_2$

- $ptrace$
Test generation

- However, not all non-observable PFSMs can be transformed into a probabilistically trace equivalent observable PFSM.

- The number of states in $M'_3$ cannot be finite since $(p_1)^n \neq (p_1)^m$ if $m \neq n$. 

$\sum_{i=1}^{n} (p_1)^{-1} \neq \sum_{i=1}^{m} (p_1)^{-1}$
Test generation

Applying the generalized Wp-method by Luo et al.[1]

Definition 3.1 A characterization set $W \subseteq L_i^*$ of a PFSM $M = (S, L_i, L_o, s_0, P_f)$ is a set of input sequences where for any two states $s_i, s_j \in S$, $i \neq j$, there are some $w \in W$, $y \in L_o^*$, and $s_k, s_l \in S$ such that $P_f^*(s_i, w, s_k, y) \neq P_f^*(s_j, w, s_l, y)$.

Example) $W = \{a\}$ for $M_2$

<table>
<thead>
<tr>
<th>State</th>
<th>Output sequences with probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_0$</td>
<td>$e(1)$</td>
</tr>
<tr>
<td>$S_1$</td>
<td>$f(1)$</td>
</tr>
<tr>
<td>$S_2$</td>
<td>$e(0.8), f(0.2)$</td>
</tr>
<tr>
<td>$S_3$</td>
<td>$e(0.5), f(0.5)$</td>
</tr>
</tbody>
</table>
Test generation

- A test suite consists of state verification and transition testing.
  - State verification checks if each state in the implementation can be identified by the characterization set.
  - Transition testing checks whether each transition produces the expected output for a given input and moves to the expected next state.

- Some notations
  - $Q$-set: the set of input sequences that take the machine from the initial state $s_0$ to each state;
  - $P$-set: the set of input sequences that lead the machine to each outgoing transition of all states from the initial state;
  - $R$-set: $R = P \setminus Q$;
  - A state identification set $W_i \subseteq W$ is a set of input sequences which is used to identify the state $s_i$.
  - $\oplus$ operator: $V \oplus \{W_0, W_1, \ldots, W_{n-1}\} = \bigcup_{s_0=v \Rightarrow s_i} \{v\} . W_i$ where $V \subseteq L_i^*$. 
Test generation

- A test suite is given as $\Pi = \Pi_1 \cup \Pi_2$ where
  \[
  \Pi_1 = Q.(\{\varepsilon\} \cup L_i \cup L_i^2 \cup \cdots \cup L_i^{m-n}).W
  \]
  \[
  \Pi_2 = R.L_i^{m-n} \oplus \{W_0, W_1, \ldots, W_{n-1}\}
  \]

**Theorem 3.2** Let $M$ and $M_I$ be PFSMs where the number of states in $M$ is $n$ and the number of states in $M_I$ is bounded by $m$ ($n \leq m$). Let $\Pi$ be the test suite generated by the generalized Wp-method for $M$. We have $M = p_{\text{trace}} M_I$ iff $M = p_{\Pi} M_I$. 
Contents

- Problem statement & Preliminaries
- Test generation
- Testing of probability
- Test sequence repetition numbers
- Testing of a PFSM
- Fault coverage evaluation
- Application: A probabilistic non-repudiation protocol
- Conclusions
Testing of probability

- Testing is experiments.

- Selecting a transition between $t_1$ and $t_2$ for execution can be considered as an experiment.
- If $Y$ represents the number of times $t_1$ was selected after $n$ independent experiments, $Y$ is said to be a binomial random variable with parameters $(n, p)$.
- The observed data after $n$ independent experiments is called a sample where $n$ is the sample size.
- A number of methods have been proposed to estimate $p$ with a certain degree of confidence, which is called the confidence interval, with the sample after $n$ independent experiments.
Testing of probability

**Interval estimation**

- Wald interval with $1 - \alpha$ confidence

\[
\hat{p} - z_{\alpha/2} \sqrt{\frac{\hat{p}(1 - \hat{p})}{n}} < p < \hat{p} + z_{\alpha/2} \sqrt{\frac{\hat{p}(1 - \hat{p})}{n}}
\]

where $\hat{p} = Y/n$, $0 < \alpha < 1$ and $z_{\alpha/2}$ is such that $P\{|Z| < z_{\alpha/2}\} = 1 - \alpha$ where $Z$ is unit normal random variable (e.g. $z_{\alpha/2} = 1.96$ when $\alpha = 5\%$)

- Known problems of the Wald interval
  - Very poor for $p$ near 0 or 1
  - Gives erratic results for $p$ not near the boundaries when $n$ is not large enough.
  - Even though $p$ is not near the boundaries and $n$ is large enough, sometimes, it gives poor coverage probability*.

* The coverage probability of a confidence interval is the probability that the confidence interval contains its real parameter value,
Testing of probability

■ Interval estimation

• Agresti-Coull interval with $1 - \alpha$ confidence

$$\tilde{p} - \kappa \sqrt{p(1-p)} \leq \hat{p} \leq \tilde{p} + \kappa \sqrt{p(1-p)}$$

where $\tilde{Y} = Y + \kappa^2 / 2$, $\tilde{n} = n + \kappa^2$, $\tilde{p} = \tilde{Y} / \tilde{n}$, $\tilde{q} = 1 - \tilde{p}$, and $\kappa$ is such that $P\{|Z| \leq \kappa\} = 1 - \alpha$ where $Z$ is unit normal random variable. When $\alpha = 5\%$, the value 2 is used instead of 1.96 for $\kappa$ in the Agresti-Coull interval.

• The advantages of the Agresti-Coull interval [2]
  - If $n \geq 40$, the Agresti-Coull interval provides good coverage probability even for $p$ very close to 0 or 1.
  - For relatively small $n$, it gives a larger confidence interval length than other methods though it still gives good coverage.
  - One of the recommended intervals by Brown et al. [2].
Testing of probability

- Comparison of coverage probabilities of Wald interval and Agresti-Coull interval by simulation where $\alpha = 5\%$, $p=0.95$, and $n=40 \sim 18,000$
Testing of probability

The basic idea behind testing

- The meaning of the confidence interval for the probability $p'$ with 95% confidence is that
  
  “The technique used to obtain this interval is such that 95% of the time that it is employed it will result in an interval that contains $p'[3]$."

- Some notations
  - $p$ is the probability in the specification PFSM $M$.
  - $p'$ is the probability in an implementation PFSM $M_I$.
  - $d$ is the half of the obtained confidence interval length.
  - $P_{EP}$ is the test-pass probability of equivalent machines.
  - $P_{NEF}$ is the test-fail probability of faulty machines.
Testing of probability

• The criterion for testing
  - *Pass*, if $p$ is included in the obtained confidence interval for $p'$;
  - *Fail*, otherwise.

• Expected results for various $p'$
  Case I) $p' = p$
    95% of the time, the implementation will have a pass verdict ($P_{EP} = 95\%$).

  Case II & III) $p' > p + 2d$ and $p' < p - 2d$
    Approximately 97.5% of the time, the implementation will have a fail verdict ($P_{NEF} = 97.5\%$).

  Case IV) $0 < |p' - p| \leq 2d$
    The implementation will have a fail verdict from 5% to 97.5% of the time according to the difference of probabilities.
Testing of probability

- Therefore, we can assert that
  
  \[ P_{EP} \geq (1 - \alpha) \]
  
  \[ P_{NEF} \geq (1 - \alpha / 2) \]  
  where \( |p' - p| > 2d \)
  
  as far as we have good coverage probabilities.
Contents

- Problem statement & Preliminaries
- Test generation
- Testing of probability
- Test sequence repetition numbers
- Testing of a PFSM
- Fault coverage evaluation
- Application: A probabilistic non-repudiation protocol
- Conclusions
Test sequence repetition numbers

Without faults in probability

- A method has been proposed by Hwang et al.[4] to determine the test sequence repetition numbers from NFSMs which ensure that $P_{EP}$ and $P_{NEF}$ are not less than a given value.
- If there are $k_1$ test sequences that have more than one output sequence in a test suite and a test sequence $ts$ has $m$ observable output sequences. The test sequence repetition number $n$ which ensures that $P_{EP}$ is greater than a given value, $(1 - \alpha)$ is given as

$$\sum_{i_1} (1 - p_{i_1})^n - (-1)^2 \sum_{i_1} \sum_{i_2 > i_1} \{1 - (p_{i_1} + p_{i_2})\}^n - \cdots$$

$$- (-1)^{m-1} \sum_{i_1} \sum_{i_2 > i_1} \cdots \sum_{i_{m-1} > i_{m-2}} \{1 - (p_{i_1} + \cdots + p_{i_{m-1}})\}^n \leq 1 - \frac{k_1}{\sqrt{1 - \alpha}}$$

(1)

where $p_{i_j} (1 \leq i_j \leq m)$ is the probability of $i_j$-th output sequence.

- In order to ensure $P_{NEF}$, test sequence repetition numbers should be refined so that states are identified correctly against transfer faults.
Test sequence repetition numbers

\[ W = \{a\} \text{ for } M_2 \]

<table>
<thead>
<tr>
<th>State</th>
<th>Output sequences with probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S_0 )</td>
<td>( e (1) )</td>
</tr>
<tr>
<td>( S_1 )</td>
<td>( f (1) )</td>
</tr>
<tr>
<td>( S_2 )</td>
<td>( e (0.8), f (0.2) )</td>
</tr>
<tr>
<td>( S_3 )</td>
<td>( e (0.5), f (0.5) )</td>
</tr>
</tbody>
</table>

- When we test the transition \( b/f \) from state \( S_0 \), the next state can be identified by applying the input \( a \).
- In order to detect the transfer fault, the input sequence \( b.a \) should be repeated a number of times.
Test sequence repetition numbers

- **Two executable transitions**
  - We can always ensure $P_{EP}$ as far as the coverage probability is above the nominal confidence level $(1 - \alpha)$.
  - In order to have $P_{NEF} \geq 1 - \alpha / 2$, the test sequence repetition number $n$ should satisfy the following two conditions:
    \[
    n > \left( \frac{\kappa}{d} \right)^2 \hat{p}(1 - \hat{p}) - \kappa^2
    \]  
    \(2\)

    \[
    \hat{p} = \begin{cases} 
    p + d & \text{if } p \leq 0.5 \\
    p - d & \text{otherwise}
    \end{cases}
    \]  
    \(3\)

- **Evaluation of the coverage probabilities by simulation**
  - A simple machine with two transitions are considered.
  - $p$ was increased from 0.01 to 0.5 and $r_p$ from 5% to 50% where $r_p$ is the ratio of $d$ to $\min(p, 1-p)$. 
Test sequence repetition numbers

- For each $p$ and $r_p$ pair, $n$ was calculated with 95% confidence level and $P_{EP}$ and $P_{NEF}$ are evaluated.
- For faulty machines, the difference of probability is set to $2d$. 

![Graphs showing $P_{EP} < 0.95$ and $P_{NEF} < 0.975$](image)
Test sequence repetition numbers

- Comparison of $P_{EP}$ with Agresti-Coull interval and Wald interval

$$P_{EP}$$

Agresti-Coull interval

Wald interval
Test sequence repetition numbers

- In order to have better coverage probability, the following conditions are recommended:

\[
\begin{align*}
\text{For } P_{EP}, & \\
\begin{cases}
  r_p \leq 10\% & \text{if } 0.4 < \min(p, 1-p) \leq 0.5; \\
  r_p \leq 20\% & \text{if } 0.2 < \min(p, 1-p) \leq 0.4; \\
  r_p \leq 30\% & \text{if } 0.1 < \min(p, 1-p) \leq 0.2; \\
  r_p \leq 35\% & \text{if } 0 < \min(p, 1-p) \leq 0.1.
\end{cases}
\]

(4)

\[
\begin{align*}
\text{For } P_{NEF}, & \\
\begin{cases}
  r_p \leq 10\% & \text{if } 0.4 < \min(p, 1-p) \leq 0.5; \\
  r_p \leq 15\% & \text{if } 0.3 < \min(p, 1-p) \leq 0.4; \\
  r_p \leq 25\% & \text{if } 0.2 < \min(p, 1-p) \leq 0.3; \\
  r_p \leq 50\% & \text{if } 0 < \min(p, 1-p) \leq 0.2.
\end{cases}
\]

(5)
Example)
There are two transitions from a state where the probabilities are 0.8 and 0.2, respectively. What is the test sequence repetition number $n$ that ensures a 95% $P_{EP}$ and a 97.5% $P_{NEF}$ where faulty implementations are such that the difference of the probability is more than 10% of 0.2?

For $P_{NEF}$, since $r_p = 10\%$, it satisfies the conditions given in (5). By using (2) and (3), we can obtain $n \geq 1,713$. If we follow the conditions given in (4), we can have good $P_{EP}$ when $d \leq 0.3 \times 0.2 = 0.06$ as $\min(p, 1-p) = 0.2$. Therefore, we can obtain $n \geq 174$ by using (2) and we can finally have $n=1,713$. 
Test sequence repetition numbers

What happen if we have less $n$?

<table>
<thead>
<tr>
<th>$p$</th>
<th>$n$</th>
<th>$p-2d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>40</td>
<td>0.5137</td>
</tr>
<tr>
<td>0.8</td>
<td>60</td>
<td>0.5675</td>
</tr>
<tr>
<td>0.8</td>
<td>100</td>
<td>0.6222</td>
</tr>
<tr>
<td>0.8</td>
<td>200</td>
<td>0.6769</td>
</tr>
<tr>
<td>0.8</td>
<td>500</td>
<td>0.7241</td>
</tr>
<tr>
<td>0.8</td>
<td>1000</td>
<td>0.7472</td>
</tr>
<tr>
<td>0.8</td>
<td>1713</td>
<td>0.7600</td>
</tr>
</tbody>
</table>

If we have $n=40$, we can ensure a 97.5% $P_{NEF}$ for the machines such that $p' < 0.5137$. 
Test sequence repetition numbers

More than two executable transitions

- The condition for $\kappa$, i.e. $P\{|Z| \leq \kappa\} = 1 - \alpha$ is generalized to apply the Agresti–Coull interval to the case when there are $j$ executable transitions.

- Since all probabilities should be simultaneously contained in their own confidence intervals, the following condition should be satisfied:

$$P\{|Z| < \kappa\} = \begin{cases} 1 - \alpha & \text{if } j = 2 \\ (1 - \alpha)^{1/j} & \text{if } j > 2 \end{cases} \quad (6)$$
Test sequence repetition numbers

Transitions from a non-initial state

- In order to test the probabilities of $b/e$ and $b/f$ from $S_2$, the observation of the output sequences $f.e$ and $f.f$ should be considered.
- Let $2d’$ and $2d$ be the confidence interval lengths for $q$ and $r$ respectively. $n$ should satisfy the following two conditions where $\kappa’=2$:

  $n > \left( \frac{\kappa’}{d’} \right)^2 q(1-q) - \kappa’^2$ \hspace{1cm} (7)

  $n > \left( \frac{\kappa}{d} \right)^2 r(1-r) - \kappa^2 \right) \sqrt{(q-d')}$ \hspace{1cm} (8)
Test sequence repetition numbers

- Refinement of the condition for $\kappa$
  - If there are $j$ probabilities to test in a test sequence $ts$ and there are $k_2$ test sequences in a test suite which have at least two probabilities to test, $\kappa$ should satisfy the following condition:

$$P(|Z| < \kappa) = \begin{cases} 
(1 - \alpha)^{1/k_2} & \text{if } j = 2 \\
(1 - \alpha)^{1/(j+k_2-1)} & \text{if } j > 2 
\end{cases}$$

(9)
Test sequence repetition numbers

Transition without nondeterminism

- It is necessary to apply the test input more than once since the probability can be changed by extra transitions.
- Test sequence repetition number $n$ to ensure that $P_{NEF}$ is never less than a given value $(1 - \alpha / 2)$ should satisfy

$$\left(1 - 2d\right)^n \leq \alpha / 2 \quad (10)$$

- If the transition for testing is from a non-initial state $s_i$, $n$ should satisfy

$$\left(1 - 2pd\right)^n \leq \alpha / 2 \quad (11)$$

where $p$ is the probability of the output sequence that leads the machine from the initial state to $s_i$. 
Contents

- Problem statement & Preliminaries
- Test generation
- Testing of probability
- Test sequence repetition numbers
- Testing of a PFSM
- Fault coverage evaluation
- Application: A probabilistic non-repudiation protocol
- Conclusions
Testing of a PFSM

Testing hypotheses

\[ H_0: \text{the implementation is an equivalent machine} \]
\[ H_1: \text{the implementation which has at most } m \text{ states is not an equivalent machine such that:} \]
- it has at least one output fault or transfer fault; or
- among the given probabilities for testing, there is at least one probability \( p'_i \) in implementation that satisfies \( |p'_i - p_i| > 2d_i \) for given \( d_i \) and correct probability \( p_i \).

- The method ensures that the probability that the hypothesis \( H_0 \) is accepted when it is true (\( P_{EP} \)) is not less than a given value and that the hypothesis \( H_1 \) is accepted when it is true (\( P_{NEF} \)) is not less than a given value.
- \( (1-P_{EP}) \) and \( (1-P_{NEF}) \) correspond to type I and type II errors of hypothesis \( H_0 \) respectively.
Testing of a PFSM

Test sequence repetition numbers for state identification

- In order to distinguish $p_{in}$ and $p_{ln}$, there should be no intersection between the two confidence intervals. Therefore, it should satisfy

$$d_{i_n} \leq \left| p_{i_n} - p_{j_n} \right|/2$$  \hspace{1cm} (12)

- Test sequence repetition numbers $n_{in}$ can be calculated where $j=2$ can be used in (9). $n_i$ to identify the state $s_i$ can be obtained as $n_i = \min(n_{i_1}, n_{i_2}, \ldots, n_{i_m})$. 
Testing of a PFSM

Example)

Let us consider state identification of \( S_2 \) from PFSM \( M_2 \). Suppose that we apply a test sequence \( a.a \) to an implementation and then check if the implementation is in state \( S_2 \). What is the test sequence repetition number for this test sequence?

We can have the same output symbols with different probabilities from \( S_3 \). Let us denote by \( p_{i1} \) and \( p_{l1} \) the probabilities of output symbol \( e \) from \( S_2 \) and \( S_3 \), respectively. As \( p_{i1}=0.8 \) and \( p_{l1}=0.5 \), we can easily obtain \( d_{i1} \leq (0.8 - 0.5) / 2 = 0.15 \).

If we follow the conditions given in (5), \( d_{i1} \) should satisfy \( d_{i1} \leq 0.5 \times 0.2 = 0.1 \) for \( P_{NEF} \). Therefore, we can have \( n \geq 81 \) for a 97.5% \( P_{NEF} \) by using (2), (3) and (9).

If we follow the condition given in (4), we have \( d_{i1} \leq 0.3 \times 0.2 = 0.1 \) for \( P_{EP} \). Therefore, we can obtain the test sequence repetition number for a 95% \( P_{EP} \) as \( n \geq 174 \) by using (2) and (9). Finally, we have \( n=174 \).
Testing of a PFSM

Requirements for testing probabilities

- T-set \((T \subseteq S \times L_i^*)\) is assumed to be provided as a requirement for testing of probabilities. For each \((s, u) \in T\), we test the probabilities of output sequences from the state \(s\) when \(u\) is applied.

Two groups of test sequences

- \(\Pi_T\) is a set of \((ts, n)\) where \(ts\) is a test sequence and \(n\) is the test sequence repetition number. It is checked that if all possible output sequences are observed without any erroneous output.

- \(\Pi_p\) is a set of \((v, y, u, n)\) where \(v\) and \(y\) are input and output sequences that take the machine to the start state \(s\) for testing of probabilities, \(ts = v.u\) is the test sequence, and \(n\) is the test sequence repetition number. It is checked that if the probabilities of a given set of output sequences are correctly implemented.
Testing of a PFSM

Algorithm for test generation and determination of test sequence repetition number

Input: a PFSM $M=(S, Li, Lo, s_0, P_T)$, upper bound $m$ on the number of states in implementations, confidence level $(1-\alpha)$, a common ratio for confidence interval length $r_p$, $T \subseteq S \times Li^*$ for testing of probabilities

Output: $\Pi_T$ and $\Pi_P$

- Step 1: Set $\Pi'_p = \emptyset$ and $\Pi'_T = \emptyset$.
- Step 2: Construct characterization set $W$, $P$-set, $Q$-set, $R$-set, $\Pi1$, and $\Pi2$.
- Step 3: Set $k_2=1$. Construct $\Pi'_p$, $\Pi_T$, and $\Pi_P$. 
Testing of a PFSM

- Step 4: Eliminate redundant test sequences in $\Pi_p$, $\Pi'_p$, and $\Pi_T$ if they are prefixes of other test sequences. Test sequences eliminated from $\Pi_T$ are stored in $\Pi'_T$ and then test sequence repetition numbers are adjusted later (in Step 6).
- Step 5: Calculate the test sequence repetition numbers.
- Step 6: If $\Pi'_T \neq \emptyset$, adjust the test sequence repetition numbers in $\Pi_T$ and $\Pi_p$ if necessary. Otherwise, go to Step 7.
- Step 7: Eliminate redundant test sequences in $\Pi_p$ if they are prefixes of other test sequences and the probability for testing is 1.
Testing of a PFSM

Example) Let us consider test generation and determination of test sequence repetition numbers from PFSM $M_2$. Suppose that $m=4$ and we test the probability $p_i$ of each transition with a 95% confidence level and $r_p=10\%$. Then we have $T=\{(s_0, a), (s_0, b), (s_1, a), (s_1, b), (s_2, a), (s_2, b), (s_3, a), (s_3, b)\}$.

Step 1-2: $W=\{a\}$ and $W_0=W_1=W_2=W_3=\{a\}$. $Q$-set, $P$-set, $R$-set, $\Pi_1$, and $\Pi_2$ are $Q=\{\varepsilon, a, a.a, a.a.a\}$, $P=\{\varepsilon, a, b, a.a, a.b, a.a.a, a.a.b, a.a.a.b\}$, $R=\{b, a.b, a.a.b, a.a.a.a, a.a.a.b\}$, $\Pi_1=\{a, a.a, a.a.a, a.a.a.a\}$, and $\Pi_2=\{b.a, a.b.a, a.a.b.a, a.a.a.a.a, a.a.a.b.a\}$, respectively.

Step 3: Set $k_2=1$. Test sequences $a.a.a$ and $a.a.a.a$ from $\Pi_1$ and $a.a.a.a.a$ from $\Pi_2$ are included in $\Pi'_P$, as it is necessary to test the probabilities of output sequences when we identify state $S_2$ or $S_3$. For the test sequence $a.a.a.a$ from $\Pi_1$, two members $(a.a.a, e.f.e, a, 519)$ and $(a.a.a, e.f.f, a, 999)$ are created as it is necessary to identify two states $S_3$ and $S_2$ after application of input sequence $a.a.a$. Finally, $\Pi'_P=\{(a.a, e.f, a, 174), (a.a.a, e.f.e, a, 519), (a.a.a, e.f.f, a, 999), (a.a.a.a, e.f.e.e, a, 1071), (a.a.a.a, e.f.e.f, a, 490)\}$, $\Pi_T=\{(a, 0), (a.a, 0), (b.a, 0), (a.b.a, 0), (a.a.b.a, 0), (a.a.b.a, 0)\}$, and $\Pi_P=\{(\varepsilon, \lambda, a, 17), (\varepsilon, \lambda, b, 17), (a, e, a, 17), (a, e, b, 17), (a.a, e.f, a, 1713), (a.a, e.f, b, 1713), (a.a.a, e.f.e, a, 519), (a.a.a, e.f.e, b, 519)\}$.
Testing of a PFSM

Step 4: After elimination of redundant test sequences in $\Pi_p$, $\Pi'_p$ and $\Pi_T$, e.g. $(a.a, e.f, a, 174)$ and $(a.a.a, e.f.e, a, 519)$ from $\Pi'_p$, and $(a, 0)$ and $(a.a, 0)$ from $\Pi_T$, we can get $k'_1 = 2$ and $k'_2 = 7$ where $k'_1$ is the number of test sequences which have more than one output sequence in $\Pi_T$ and $k'_2$ is the number of test sequences which have at least two probabilities to test in $\Pi_p \cup \Pi'_p$. Therefore, we have $k_1 = k_2 = k'_1 + k'_2 = 9$.

Step 5-7: After re-calculation of test sequence repetition numbers, we can eliminate two test sequences $(a, e, a, 17)$ and $(a.a, e, a, 17)$ from $\Pi_p$ since the probability for testing has value 1, i.e. the transitions for testing do not have nondeterminism, and each test sequence is a prefix of another test sequence in $\Pi_T$.

Finally, we can obtain test sequences which ensure that $P_{EP}$ and $P_{NEF}$ are never less than 95% and 97.5% respectively with $r_p = 10\%$ as $\Pi_T = \{(b.a, 14), (a.b.a, 14), (a.a.b.a, 74), (a.a.a.b.a, 127)\}$ and $\Pi_p = \{(\varepsilon, \lambda, b, 17), (a, e, b, 17), (a.a, e.f, a, 3285), (a.a, e.f, b, 3285), (a.a.a, e.f.e, a, 982), (a.a.a, e.f.e, b, 982), (a.a.a, e.f.f, a, 1842), (a.a.a.a, e.f.e.e, a, 2010), (a.a.a.a, e.f.e.f, a, 909)\}$.
### Testing of a PFSM

- Test sequences in a tree form

#### $\Pi_T$

<table>
<thead>
<tr>
<th>Index</th>
<th>Test sequence</th>
<th>Index</th>
<th>Test sequence</th>
<th>Index</th>
<th>Test sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>b/f</td>
<td>4</td>
<td>a/e → a/f → a/e → b/e → a/e : 0.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>a/e → b/f → a/e</td>
<td>5</td>
<td>a/e → a/f → a/e → b/e → a/e : 0.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>a/e → a/f → b/e → a/e : 0.8</td>
<td>6</td>
<td>a/e → a/f → a/e → b/e → a/e : 0.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>b/f → a/e : 0.2</td>
<td></td>
<td>a/f → a/e → a/f → a/e : 0.04</td>
</tr>
</tbody>
</table>

#### $\Pi_P$

<table>
<thead>
<tr>
<th>Index</th>
<th>Test sequence</th>
<th>Index</th>
<th>Test sequence</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>b/f</td>
<td>8</td>
<td>a/e → a/e → a/e → a/e : 0.2</td>
</tr>
<tr>
<td>2</td>
<td>a/e → b/f → a/e</td>
<td>9</td>
<td>a/e → a/e → a/e : 0.2</td>
</tr>
<tr>
<td>3</td>
<td>a/e → a/f → a/e → a/f : 0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>a/e → a/f → b/e → a/e : 0.8</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>a/f → a/e : 0.032</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>a/f → a/e : 0.008</td>
</tr>
<tr>
<td>5</td>
<td>a/e → a/f → a/e → a/f → a/e : 0.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>a/e → a/f → a/e → b/e → a/e : 0.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>a/e → a/f → a/e → a/e → a/e : 0.4</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Testing of a PFSM

Scalability

- The most important parameter that affects the size of the repetition number is the confidence interval length.

Recommendations

- It is necessary to have proper confidence interval length; and
- For small probabilities, it is preferable to have larger confidence interval length.

The test sequence repetition numbers used to evaluate $P_{EP}$.
Testing of a PFSM

Comments

- Although the size of the repetition number is very large if the probability for testing is small, in the industry, they do it if it is really necessary.

- Performance testing of a UMTS Mobile Switching Center (MSC)
  - An MSC can be highly abstracted by a PFSM with a single state and two transitions: one with call_attempt/success, the other call_attempt/fail.
  - Performance requirement of inadequately handled call attempts of internal call is 99.9%, which is the requirement from an operator.
  - If we consider a 95% $P_{EP}$ and a 97.5% $P_{NEF}$ where $r_p = 5\%$, i.e. $P_{NEF}$ is ensured not less than 97.5% when $p' \leq 0.9988$, we can get the repetition number $n=1,758,060$.
  - In practice, more than 3 million call attempts are applied within 48 hours thanks to the automated tool support.
Contents

- Problem statement & Preliminaries
- Test generation
- Testing of probability
- Test sequence repetition numbers
- Testing of a PFSM
- Fault coverage evaluation
- Application: A probabilistic non-repudiation protocol
- Conclusions
Fault coverage evaluation

- **Fault types**
  - Type 1: change of the output of a transition
  - Type 2: change of the next state of a transition
  - Type 3: change of both the output and the next state of a transition
  - Type 4: change of the probabilities of a pair of transitions
  - Type 5: a single extra transition
  - Type 6: a single missing transition
### Fault classes (11 additional classes to the work of [5])

<table>
<thead>
<tr>
<th>Class</th>
<th>Description</th>
<th>Class</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>one type 1 fault</td>
<td>12</td>
<td>one type 4 fault</td>
</tr>
<tr>
<td>2</td>
<td>one type 2 fault</td>
<td>13</td>
<td>one type 5 fault</td>
</tr>
<tr>
<td>3</td>
<td>two type 1 faults</td>
<td>14</td>
<td>two type 4 faults</td>
</tr>
<tr>
<td>4</td>
<td>two type 2 faults</td>
<td>15</td>
<td>two type 5 faults</td>
</tr>
<tr>
<td>5</td>
<td>one type 1 fault, one type 2 fault, no type 3 fault</td>
<td>16</td>
<td>one type 4 fault, one type 5 fault</td>
</tr>
<tr>
<td>6</td>
<td>one type 3 fault</td>
<td>17</td>
<td>one type 6 fault</td>
</tr>
<tr>
<td>7</td>
<td>two type 3 faults</td>
<td>18</td>
<td>two type 6 faults</td>
</tr>
<tr>
<td>8</td>
<td>one type 2 fault, two type 1 faults, no type 3 fault</td>
<td>19</td>
<td>one type 4 fault, two type 5 faults</td>
</tr>
<tr>
<td>9</td>
<td>two type 2 faults, two type 1 faults, no type 3 fault</td>
<td>20</td>
<td>two type 4 faults, one type 5 fault</td>
</tr>
<tr>
<td>10</td>
<td>three type 2 faults, two type 1 faults, no type 3 fault</td>
<td>21</td>
<td>one type 2 fault, one type 4 fault</td>
</tr>
<tr>
<td>11</td>
<td>one type 2 fault, one type 3 fault</td>
<td>22</td>
<td>one type 2 fault, one type 5 fault</td>
</tr>
</tbody>
</table>
Fault coverage evaluation

Simulation environment

- For each fault class 1,000,000 faulty machines were generated where faulty transitions, outputs, next states, and newly added transitions were taken from independent pseudo-random sequence.
- A common $r_p = 10\%$ was used when we calculate test sequence repetition numbers.
- When it is necessary to change the probability of a transition $(s_i, a, s_j, b_i) \in S \times Li \times S \times Lo$, we find a $(s_k, b_j) \in S \times Lo$ such that $0 < P_T(s_i, a, s_k, b_j) \leq P_T(s_i, a, s_i, b_k)$ for any $(s_i, b_k) \in S \times Lo$ and the difference $(2d)$ was chosen as $2 \times r_p \times P_T(s_i, a, s_k, b_j)$.
Fault coverage evaluation

Simulation results

- Very high $P_{NEF}$ for all fault classes is due to large $\kappa$ to ensure that all probabilities fall into their own confidence intervals simultaneously.

<table>
<thead>
<tr>
<th>Fault Class</th>
<th>$P_{NEF}$</th>
<th>Fault Class</th>
<th>$P_{NEF}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100</td>
<td>12</td>
<td>99.897</td>
</tr>
<tr>
<td>2</td>
<td>99.850</td>
<td>13</td>
<td>98.914</td>
</tr>
<tr>
<td>3</td>
<td>98.578</td>
<td>14</td>
<td>100</td>
</tr>
<tr>
<td>4</td>
<td>100</td>
<td>15</td>
<td>99.993</td>
</tr>
<tr>
<td>5</td>
<td>100</td>
<td>16</td>
<td>99.998</td>
</tr>
<tr>
<td>6</td>
<td>100</td>
<td>17</td>
<td>100</td>
</tr>
<tr>
<td>7</td>
<td>99.835</td>
<td>18</td>
<td>100</td>
</tr>
<tr>
<td>8</td>
<td>99.996</td>
<td>19</td>
<td>100</td>
</tr>
<tr>
<td>9</td>
<td>100</td>
<td>20</td>
<td>100</td>
</tr>
<tr>
<td>10</td>
<td>100</td>
<td>21</td>
<td>100</td>
</tr>
<tr>
<td>11</td>
<td>100</td>
<td>22</td>
<td>100</td>
</tr>
</tbody>
</table>
Contents

- Problem statement & Preliminaries
- Test generation
- Testing of probability
- Test sequence repetition numbers
- Testing of a PFSM
- Fault coverage evaluation
- Application: A probabilistic non-repudiation protocol
- Conclusions
Application: A probabilistic non-repudiation protocol

■ Concepts

- Non-repudiation services are intended to prevent the originator or the recipient of a message from denying having sent or received the message.
- A non-repudiation protocol is said to be fair if either both parties receive their expected items or none of them has received any valuable information about their expected items at each step of the protocol run.
- How to implement?
  - Non-repudiation of originator can be achieved by digital signature.
  - Non-repudiation of recipient can be achieved by an acknowledge message that confirms the reception.
  - A trusted third party which obliges the recipients to send an acknowledgement for every received message.
- A probabilistic non-repudiation protocol [6] was proposed without the need of a trusted third party.
Application: A probabilistic non-repudiation protocol

- Procedures
  1. \( R \to O: \text{Sign}_R(\text{request}, R, O, t) \) (The originator checks \( t \), chooses a probability \( p \) and a ciphering key \( K \), and computes the \( \{m\}_K \).
  2. \( O \to R: \text{Sign}_O(\{m\}_K, O, R, t) (=m_1) \)
  3. \( R \to O: \text{Sign}_R(\text{ack}_1) \)
  4a. \((1-p) \ O \to R: \text{Sign}_O(m_r, R, O, t) (=m_i) \)
      \( R \to O: \text{Sign}_R(\text{ack}_i) \)
      goto step 4
  4b. \((p) \ O \to R: \text{Sign}_O(K, R, O, t) (=m_n) \)
  5. \( R \to O: \text{Sign}_R(\text{ack}_n) \)

- The protocol is not fair only when the recipient does not send the last \( \text{ack} \) after receiving the key \( K \) and this depends on the probability \( p \) chosen by the originator.
**Application:** A probabilistic non-repudiation protocol

- **PFSM** $M_5$ is the state transition diagram of the originator where $p=0.1$.
- The probability of each transition in PFSM $M_5$ is tested with 95% confidence level and $r_p=10\%$.
- We can obtain test sequences as follows: $\Pi_T=\{(b.b, 1), (c.b, 1), (a.a.b, 1), (a.c.b, 1), (a.b.a.b, 42), (a.b.b.b, 59), (a.b.c.b, 42)\}$ and $\Pi_P=\{(e, \lambda, b, 17), (e, \lambda, c, 17), (a, x, a, 17), (a, x, b, 6113), (a, x, c, 17), (a.b, x.z, a, 183), (a.b, x.z, b, 183), (a.b, x.z, c, 183)\}$.
Contents

- Problem statement & Preliminaries
- Test generation
- Testing of probability
- Test sequence repetition numbers
- Testing of a PFSM
- Fault coverage evaluation
- Application: A probabilistic non-repudiation protocol
- Conclusions
Conclusions

Answers to questions

- How to generate test cases from PFSMs?
  It was shown that the generalized Wp-method for NFSMs can be applied to PFSMs with a little extension;
- How can we test the probabilities of an implementation that is modeled by a PFSM?
  A method was proposed to test probabilities by using interval estimation.
- What can be addressed after testing?
  Test-pass probability of correct implementations and test-fail probability of restricted class of faulty implementations are ensured.
- How many times each test should be repeated when we test probabilities?
  As an extension of the work of Hwang et al.[4], a method to calculate the test sequence repetition numbers was presented.
Conclusions

Some comments

• Very large total test size of the test suite generated by the proposed method is due to the fact that testing probabilities is inherently costly.

• If we do not have such a total test size, the test-fail probability of faulty implementations can be guaranteed for a smaller class of incorrect implementations, i.e., for implementations where the difference of probability is large.

• In order to have an appropriate cost for testing, therefore, it is recommended that probabilities which require high accuracy or small probabilities be avoided when we design specifications.
Conclusions

Future works

- Optimization of the total test size.
- Extension to non-observable PFSMs
- To use adaptive strategies when generate test cases
- To consider testing of time.
### References


