Simulation + Hypothesis Testing for Solving the Probabilistic Model Checking Problem

Axel Legay

November 29, 2011
Our dream

Being able to verify that a system behaves correctly:

- Easy to formulate;
- Not easy to describe (properties, quantities, ...);
- Not easy to handle.

Several solutions: testing, formal verification, statistical model checking.
Testing

Approach:
- Define test cases that represent user’s behaviors;
- Check for the absence of errors;

Problems:
- Difficult to generate test cases;
- Not precise enough;

Solution: Formal Verification
Formal Verification

Approach:
- Build a mathematical model of the system;
- Explore the state-space of the model;
- Guarantee the absence of errors.

Problems:
- How to build the model?
- State-space explosion?
- How to handle undecidability?
- How to handle complex data (embedded systems, ...)
  This is not a Boolean world anymore!

Solution: a trade off between the two approaches (Statistical Model Checking).
1. Stochastic Systems
2. The Algebraic Approach
3. Statistical Model Checking
4. Experiments
5. Bayesian Model Checking
6. What’s next?
Stochastic Systems (1)

**Definition**

A stochastic system is a process that evolves over time, and whose evolution can be predicted in terms of probability (pure) and nondeterministic choices (nonpure).

**Where can they be found?**

- Embedded systems
- Heterogenous systems
- Economy
- Networking
- Systems biology

Axel Legay

Simulation + Hypothesis Testing for Solving the I
A stochastic system is a process that evolves over time, and whose evolution can be predicted in terms of probability (pure) and nondeterministic choices (nonpure).

- Embedded/heterogenous systems;
Definition

A stochastic system is a process that evolves over time, and whose evolution can be predicted in terms of probability (pure) and nondeterministic choices (nonpure).

Where can they be found?

- Embedded/heterogenous systems;
- Economy;
A stochastic system is a process that evolves over time, and whose evolution can be predicted in terms of probability (pure) and nondeterministic choices (nonpure).

Where can they be found?

- Embedded/heterogenous systems;
- Economy;
- Networking;
A stochastic system is a process that evolves over time, and whose evolution can be predicted in terms of probability (pure) and nondeterministic choices (nonpure).

Where can they be found?

- Embedded/heterogenous systems;
- Economy;
- Networking;
- Systems Biology;
A stochastic system is a process that evolves over time, and whose evolution can be predicted in terms of probability (pure) and nondeterministic choices (nonpure).

Where can they be found?

- Embedded/heterogenous systems;
- Economy;
- Networking;
- Systems Biology;
- ...
Our Objectives

We will have four objectives:

1. Getting more knowledge about how to verify stochastic systems;
We will have four objectives:

1. Getting more knowledge about how to verify stochastic systems;
2. Studying how statistical techniques can be applied in this area;
Our Objectives

We will have four objectives:

1. Getting more knowledge about how to verify stochastic systems;
2. Studying how statistical techniques can be applied in this area;
3. New applications (systems and properties);
We will have four objectives:

1. Getting more knowledge about how to verify stochastic systems;
2. Studying how statistical techniques can be applied in this area;
3. New applications (systems and properties);
4. Going further than algebraic approaches (PRISM, LIQOR, ...).
Those we will consider

- Any model of pure stochastic systems;
Stochastic Systems (2) : Models

Those we will consider

- Any model of pure stochastic systems;
- Example : Markov Chains.
Those we will consider

- Any model of pure stochastic systems;
- Example: Markov Chains.
### Those we will not consider

- Any model which mixes nondeterministic and probabilistic choices;
Those we will not consider

- Any model which mixes nondeterministic and probabilistic choices;
- Example: Markov Decision Processes (MDPs).
Those we will not consider

- Any model which mixes nondeterministic and probabilistic choices;
- Example: Markov Decision Processes (MDPs).
Verification Process

**Question**

Does $\mathcal{S} \models P_{\geq \theta}(\phi)$?

where:

$\mathcal{S}$ is a Stochastic system

$P_{\geq \theta}(\phi)$ means "the probability for $\phi$ to happen should be greater or equal to $\theta$"

$\theta$ is a probability threshold

$\mathcal{S}$ are executions of $\mathcal{S}$, which are sequences of states and random variables that can be reached by following the transitions from an initial state

$\phi$ is some execution-based property, specified in a specification language
Verification Process

Question

Does $S \models P_{\geq \theta}(\phi)$?

where:

- $S$ is a Stochastic system;
Verification Process

Question

\[ S \models P_{\geq \theta}(\phi) \]

where:

- \( S \) is a Stochastic system;
- \( P_{\geq \theta}(X) \) means: “the probability for \( X \) to happen should be greater or equal to \( \theta \);”
- \( \theta \) is a probability threshold.
Verification Process

Question

Does $S \models P_{\geq \theta}(\phi)$?

where:

- $S$ is a Stochastic system;
- $P_{\geq \theta}(X)$ means: “the probability for $X$ to happen should be greater or equal to $\theta$”;
- $\theta$ is a probability threshold.
- Executions of $S$ are sequences of states (random variables) that can be reached by following the transitions from an initial state;
Verification Process

**Question**

Does $S \models P_{\geq \theta}(\phi)$?

where:

- $S$ is a Stochastic system;
- $P_{\geq \theta}(X)$ means: “the probability for $X$ to happen should be greater or equal to $\theta$”;
- $\theta$ is a probability threshold.
- Executions of $S$ are sequences of states (random variables) that can be reached by following the transitions from an initial state;
- $\phi$ is some execution-based property (specification language);
Verification Process

Question

Does $S \models P_{\geq \theta}(\phi)$?

where:

- $S$ is a Stochastic system;
- $P_{\geq \theta}(X)$ means: “the probability for $X$ to happen should be greater or equal to $\theta$”;
- $\theta$ is a probability threshold.
- Executions of $S$ are sequences of states (random variables) that can be reached by following the transitions from an initial state;
- $\phi$ is some execution-based property (specification language);
Outline

1. Stochastic Systems
2. The Algebraic Approach
3. Statistical Model Checking
4. Experiments
5. Bayesian Model Checking
6. What’s next?
Main idea

Overview

- Assume the existence of a probability space;
Main idea

Overview

- Assume the existence of a probability space;
- Compute the probability $p$ for $S$ to satisfy $\phi$;
Main idea

Overview

- Assume the existence of a probability space;
- Compute the probability $p$ for $S$ to satisfy $\phi$;
- Compare $p$ with $\theta$. 
Main idea

Overview
- Assume the existence of a probability space;
- Compute the probability $p$ for $S$ to satisfy $\phi$;
- Compare $p$ with $\theta$.

Difficulty
- Algorithms to compute $p$. 
Computing $p$ (1)

- One (or several) matrix represents the transition relation.
One (or several) matrix represents the transition relation:

Example

\[
A = \begin{pmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 0.01 & 0.01 & 0.98 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 
\end{pmatrix}
\]
Computing $p$ (1)

- One (or several) matrix represents the transition relation:

Example

\[
A = \begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0.01 & 0.01 & 0.98 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\]

- $A[x, y]$ is the probability to go from state $x$ to state $y$. 

At least 3 modal operators to consider

- Next: $P_{\geq \theta}(\Box \phi_1)$
At least 3 modal operators to consider

- **Next**: $P_{\geq \theta}(\bigcirc \phi_1)$
- **Bounded Until**: $P_{\geq \theta}(\phi_1 U_{\geq k} \phi_2)$
At least 3 modal operators to consider

- Next: $P_{\geq \theta}(\Box \phi_1)$
- Bounded Until: $P_{\geq \theta}(\phi_1 U_{\geq k} \phi_2)$
- Until: $P_{\geq \theta}(\phi_1 U \phi_2)$
Computing $p$ (2)

At least 3 modal operators to consider

- Next: $P_{\geq \theta}(\bigcirc \phi_1)$
- Bounded Until: $P_{\geq \theta}(\phi_1 \mathcal{U}^{\geq k} \phi_2)$
- Until: $P_{\geq \theta}(\phi_1 \mathcal{U} \phi_2)$

Definition

$\text{Sat}(P_{\geq \theta}(\phi))$ is the set of states from which $P_{\geq \theta}(\phi)$ is satisfied
Computing $p$ for next

**Idea**

$$P(s, \bigcirc \phi_1) = \sum_{s' \in \text{Sat}(\phi_1)} P(s, s');$$
Computing $p$ for next

**Idea**

- $P(s, \Diamond \phi_1) = \sum_{s' \in \text{Sat}(\phi_1)} P(s, s')$;
- Compute a vector $\text{Prob}(\Diamond \phi_1)$ of probabilities for all states $s$;
Computing $p$ for next

**Idea**

- $P(s, \bigcirc \phi_1) = \sum_{s' \in \text{Sat}(\phi_1)} P(s, s')$;
- Compute a vector $\text{Prob}(\bigcirc \phi_1)$ of probabilities for all states $s$;
- $\text{Prob}(\bigcirc \phi_1) = A \cdot v_{\phi_1}$, where $v_{\phi_1}(s) = 1$ iff $s$ satisfies $\phi_1$.

**Question**

Model check $P \geq 0.9(\bigcirc(\neg \text{try} \lor \text{succ}))$
Computing $p$ for next

**Solution**

\[ \text{Sat}(\neg \text{try} \lor \text{succ}) = \{S0, S2, S3\} \]
The Algebraic Approach

Description of the approach

Computing $p$ for next

Solution

- $\text{Sat}(\neg \text{try} \lor \text{succ})) = \{S0, S2, S3\}$
- $\text{Prob}(\circ (\neg \text{try} \lor \text{succ})) =$

\[
\begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0.01 & 0.01 & 0.98 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
\end{pmatrix}
\begin{pmatrix}
1 \\
0 \\
1 \\
1 \\
\end{pmatrix}
= \begin{pmatrix}
0 \\
0.99 \\
1 \\
1 \\
\end{pmatrix}
\]
Computing $p$ for next

Solution

- $\text{Sat}(\neg \text{try} \lor \text{succ}) = \{S0, S2, S3\}$
- $\text{Prob}(\Box(\neg \text{try} \lor \text{succ})) =
\begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0.01 & 0.01 & 0.98 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix} \cdot \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0.99 \\ 1 \\ 1 \end{pmatrix}$

- Result: $\text{Sat}(P_{\geq 0.9}(\Box(\neg \text{try} \lor \text{succ}))) = \{S1, S2, S3\}$
Computing $p$ for (bounded) until

- The situation gets complicated as it involves to solve linear equations;
Computing $p$ for (bounded) until

- The situation gets complicated as it involves solving linear equations;
- The problem is that these equations depend on the number of states of the chain;
Computing $p$ for (bounded) until

- The situation gets complicated as it involves to solve linear equations;
- The problem is that these equations depend on the number of states of the chain;
- In real applications, we generally consider billions of states.
Advantages of the algebraic approach

- High accuracy in result (up to some digit precision);
Advantages of the algebraic approach

- High accuracy in result (up to some digit precision);
- Exists for nonpure models such as MDPs;
Advantages of the algebraic approach

- High accuracy in result (up to some digit precision);
- Exists for nonpure models such as MDPs;
- Well-established tools: PRISM, LIQUOR, PMAUD, ...
Advantages of the algebraic approach

- High accuracy in result (up to some digit precision);
- Exists for nonpure models such as MDPs;
- Well-established tools: PRISM, LIQUOR, PMAUD, ...
- ...
Disadvantages of the algebraic approach

There are at least 6 disadvantages
Disadvantages of the algebraic approach

There are at least 6 disadvantages

1. Memory intensive (consider the entire system);
Disadvantages of the algebraic approach

There are at least 6 disadvantages

1. Memory intensive (consider the entire system);
2. Limited to certain classes of systems and properties (finite-state, ..);
Disadvantages of the algebraic approach

There are at least 6 disadvantages

1. Memory intensive (consider the entire system);
2. Limited to certain classes of systems and properties (finite-state, ..);
3. No unique solution (one per model!);
Disadvantages of the algebraic approach

There are at least 6 disadvantages

1. Memory intensive (consider the entire system);
2. Limited to certain classes of systems and properties (finite-state, ..);
3. No unique solution (one per model!);
4. Complex algorithms:
   One Difficulty: How to find efficient data structures;
Disadvantages of the algebraic approach

There are at least 6 disadvantages

1. Memory intensive (consider the entire system);
2. Limited to certain classes of systems and properties (finite-state, ..);
3. No unique solution (one per model!);
4. Complex algorithms:
   - One Difficulty: How to find efficient data structures;
5. Difficult to parallelize.
Disadvantages of the algebraic approach

There are at least 6 disadvantages:

1. Memory intensive (consider the entire system);
2. Limited to certain classes of systems and properties (finite-state, ..);
3. No unique solution (one per model!);
4. Complex algorithms:
   - **One Difficulty**: How to find efficient data structures;
5. Difficult to parallelize.
6. What do we do if the problem is undecidable?
Disadvantages of the algebraic approach

There are at least 6 disadvantages

1. Memory intensive (consider the entire system);
2. Limited to certain classes of systems and properties (finite-state, ..);
3. No unique solution (one per model!);
4. Complex algorithms:
   - One Difficulty: How to find efficient data structures;
5. Difficult to parallelize.
6. What do we do if the problem is undecidable?
Outline

1. Stochastic Systems
2. The Algebraic Approach
3. Statistical Model Checking
4. Experiments
5. Bayesian Model Checking
6. What’s next?
Learning from a Simple Problem

From a simple problem

- Consider a machine that flips a (possibly biased) coin;
- Is the probability $p$ of having a head greater or equal to some $\theta$?
Learning from a Simple Problem

From a simple problem
- Consider a machine that flips a (possibly biased) coin;
- Is the probability $p$ of having a head greater or equal to some $\theta$?

A solution
- Do several flips and deduce the answer from them;
- If the number of flips is infinite, our answer will be correct up to some type error.
Learning from a Simple Problem

From a simple problem
- Consider a machine that flips a (possibly biased) coin;
- Is the probability $p$ of having a head greater or equal to some $\theta$?

A solution
- Do several flips and deduce the answer from them;
- If the number of flips is infinite, our answer will be correct up to some type error.

This is the statistical model checking approach!
Hypothesis Testing

Test $H_0 : P(\text{having a head}) \geq \theta$ against $H_1 : P(\text{having a head}) < \theta$

With (Type error):
Hypothesis Testing

Test $H_0 : P(\text{having a head}) \geq \theta$ against $H_1 : P(\text{having a head}) < \theta$

With (Type error):

1. $\alpha$ : the probability to accept $H_1$ while $H_0$ is true;
2. $\beta$ : the probability to accept $H_0$ while $H_1$ is true.
Performance of Test

Needs an infinite number of samples to get ideal performances!
If \( p \in [\theta - \delta, \theta + \delta] \), we say we are *indifferent* to know if \( p \geq \theta \).
We want to test:

\[ H_0 : p \geq p_0 \text{ against } H_1 : p \leq p_1, \text{ where} \]
\[ p_0 = \theta + \delta \text{ and } p_1 = \theta - \delta. \]

With:
We want to test:

\[ H_0 : p \geq p_0 \text{ against } H_1 : p \leq p_1, \text{ where} \]
\[ p_0 = \theta + \delta \text{ and } p_1 = \theta - \delta. \]

With:

- Type errors \( \alpha \) and \( \beta \), and
- Indifference region \( 2\delta \).
Bernouilli Variables for experiments

**Bernouilli variable $X_i$ of parameter $p$**

- Takes two values: $X_i = 0$ or $X_i = 1$;
- $P[X_i = 1] = p$ and $P[X_i = 0] = 1 - p$;
- Realization is denoted $x_i$. 
Bernouilli Variables for experiments

**Bernouilli variable** $X_i$ of parameter $p$
- Takes two values: $X_i = 0$ or $X_i = 1$;
- $P[X_i = 1] = p$ and $P[X_i = 0] = 1 - p$;
- Realization is denoted $x_i$.

**Experiments**
- We assume independent trials;
- We can generate as much trials as we want;
- $p$ is the probability to get a head;
- Associate a bernouilli variable $X_i$ to each trial;
- $X_i = 1$ iff the trial is a tail.
Two Algorithms

Algorithm 1: Single Sampling plan

- Pre-compute a number $n$ of experiments;
- $n$ depends on $\delta, \alpha$, and $\beta$. 
Two Algorithms

**Algorithm 1 : Single Sampling plan**
- Pre-compute a number $n$ of experiments;
- $n$ depends on $\delta, \alpha,$ and $\beta.$

**Algorithm 2**
Basically a on-the-fly version of the Single Sampling Plan
Choose $n$ and $c$ with $c \leq n$;
Single Sampling plan: Principles

- Choose \( n \) and \( c \) with \( c \leq n \);
- \( n \) observations \( x_1, \ldots, x_n \) for \( n \) samplings \( X_1, \ldots, X_n \);
Single Sampling plan: Principles

- Choose \( n \) and \( c \) with \( c \leq n \);
- \( n \) observations \( x_1, \ldots, x_n \) for \( n \) samplings \( X_1, \ldots, X_n \);
- \( Y = \sum_{i=1}^{n} x_i \);
Choose \( n \) and \( c \) with \( c \leq n \);

- \( n \) observations \( x_1, \ldots, x_n \) for \( n \) samplings \( X_1, \ldots, X_n \);
- \( Y = \sum_{i=1}^{n} x_i \);
- Accept \( H_0 \) if \( Y \geq c \) and \( H_1 \) otherwise;
Choose $n$ and $c$ with $c \leq n$;
$n$ observations $x_1, \ldots, x_n$ for $n$ samplings $X_1, \ldots, X_n$;
$Y = \sum_{i=1}^{n} x_i$;
Accept $H_0$ if $Y \geq c$ and $H_1$ otherwise;
Single Sampling plan: Principles

- Choose $n$ and $c$ with $c \leq n$;
- $n$ observations $x_1, \ldots, x_n$ for $n$ samplings $X_1, \ldots, X_n$;
- $Y = \sum_{i=1}^{n} x_i$;
- Accept $H_0$ if $Y \geq c$ and $H_1$ otherwise;

**Difficulty**: Find $n$ and $c$ such that $\alpha$ and $\beta$ are satisfied.
Single Sampling plan: \( \alpha \) and \( \beta \)

**Definition**

Binomial distribution

\[
P[Y \leq c] = F(c; n; p) = \sum_{i=0}^{c} C_i^n p^i (1 - p)^{n-i}.
\]
Single Sampling plan: $\alpha$ and $\beta$

**Definition**

Binomial distribution

$$P[Y \leq c] = F(c; n; p) = \sum_{i=0}^{i=c} C^n_i p^i (1 - p)^{n-i}.$$  

**Definition**

$$F(c; n; \theta) : \text{probability to accept } H_1.$$
Single Sampling plan: $\alpha$ and $\beta$

**Definition**

Binomial distribution

$$P[Y \leq c] = F(c; n; p) = \sum_{i=0}^{c} C^n_i p^i (1 - p)^{n-i}.$$  

**Definition**

$$F(c; n; \theta) : \text{probability to accept } H_1.$$  

**Definition**

- $F(c; n; \theta + \delta) \leq \alpha$;
- $1 - F(c, n; \theta - \delta) \leq \beta$.  

Axel Legay  
Simulation + Hypothesis Testing for Solving the I  
29/73
Single Sampling plan: Disadvantages

- Difficult to find $c$ and $n$: No unique solution;
Single Sampling plan: Disadvantages

- Difficult to find $c$ and $n$: No unique solution;
- Difficult to minimize $n$;

Approximation algorithms exist (Haakan Youness).

Better for black-box systems.
Optimality (Hasting)

<table>
<thead>
<tr>
<th>Thresholds</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\theta - \delta = 0$</td>
<td>$\theta + \delta = 1$</td>
</tr>
<tr>
<td>$\theta - \delta = 0$</td>
<td>$\theta + \delta &lt; 1$</td>
</tr>
<tr>
<td>$\theta - \delta &gt; 0$</td>
<td>$\theta + \delta = 1$</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sequential Hypothesis Testing

- Check hypothesis after each sample and stop as soon as possible.
- We can find an acceptance line and a rejection line given $\alpha$, $\beta$, $\theta$, $\delta$. 
Sequential Hypothesis Testing

- Check hypothesis after each sample and stop as soon as possible
- We can find an acceptance line and a rejection line given $\alpha, \beta, \theta, \delta$. 

```
Continue

positive

Generate
```
Compute

\[ W = \prod_{i=1}^{m} \frac{Pr(X_i = x_i \mid p = \theta - \delta)}{Pr(X_i = x_i \mid p = \theta + \delta)} = \frac{(\theta - \delta)^{d_m}}{(\theta + \delta)^{d_m}} \left(1 - \theta + \delta\right)^{m-d_m} \left(1 - \theta - \delta\right)^{m-d_m}, \]

where \(d_m = \sum_{i=1}^{m} x_i\).
Wald’s Testing

Compute

\[ W = \prod_{i=1}^{m} \frac{\Pr(X_i = x_i \mid p = \theta - \delta)}{\Pr(X_i = x_i \mid p = \theta + \delta)} = \frac{(\theta - \delta)^{d_m} (1 - \theta + \delta)^{m-d_m}}{(\theta + \delta)^{d_m} (1 - \theta - \delta)^{m-d_m}}, \]

where \( d_m = \sum_{i=1}^{m} x_i \).

Stop when:

- \( W \geq (1 - \beta)/\alpha \) : \( H_1 \) is accepted;
- \( W \leq \beta/(1 - \alpha) \) : \( H_0 \) is accepted.
More Mathematics

- **In theory**: the test does not guarantee $\alpha$ and $\beta$!
- New parameters $\alpha'$ and $\beta'$ such that

\[
\alpha' \leq \alpha_1 - \beta \quad \text{and} \quad \beta' \leq \beta_1 - \alpha
\]
More Mathematics

- **In theory**: the test does not guarantee $\alpha$ and $\beta$!
- New parameters $\alpha'$ and $\beta'$ such that
  - $\alpha' \leq \frac{\alpha}{1-\beta}$ and $\beta' \leq \frac{\beta}{1-\alpha}$
  - $\alpha' + \beta' \leq \alpha + \beta$;
- **In practice**: one observes that $\alpha$ and $\beta$ are almost often guarantee, and it may even be better!

**Example**

Let $p_0 = 0.5$, $p_1 = 0.3$, $\alpha = 0.2$, $\beta = 0.1$:

- **In theory**: $\alpha' \leq \frac{0.2}{0.9} = 0.222\ldots$ and $\beta' \leq \frac{0.1}{0.8} = 0.125$;
- **Computer simulation**: $\alpha' = 0.175$ and $\beta' = 0.082$. 
Performances (1)

- Single sampling plan can be better than SPRT!
- SPRT is, in practice, more efficient;
- Expected sample size $E_p$ (Wald’s formula):
Performances (1)

- Single sampling plan can be better than SPRT!
- SPRT is, in practice, more efficient;
- Expected sample size $E_p$ (Wald’s formula):
  - SPRT minimizes $E_p$ at $\theta + \delta$ and $\theta - \delta$;
  - $E_p$ increases from 0 to $\theta - \delta$;
  - $E_p$ decreases from $\theta + \delta$ to 1;
  - Between $\theta - \delta$ and $\theta + \delta$: increase and then decrease.
Performances (2) : SPRT

<table>
<thead>
<tr>
<th>Indifference region 2(\delta)</th>
<th>Number of executions</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>55</td>
</tr>
<tr>
<td>0.05</td>
<td>106</td>
</tr>
<tr>
<td>0.02</td>
<td>228</td>
</tr>
<tr>
<td>0.01</td>
<td>627</td>
</tr>
<tr>
<td>0.005</td>
<td>1056</td>
</tr>
</tbody>
</table>

Number of trajectories against \(2\delta\) \((\alpha = \beta = 0.02)\)

- \(m\) increases linearly if \(\delta\) decreases.
Performances (3) : SPRT

<table>
<thead>
<tr>
<th>Test strength $\alpha (= \beta)$</th>
<th>Number of executions</th>
</tr>
</thead>
<tbody>
<tr>
<td>$1 \times 10^{-2}$</td>
<td>335</td>
</tr>
<tr>
<td>$1 \times 10^{-4}$</td>
<td>502</td>
</tr>
<tr>
<td>$1 \times 10^{-6}$</td>
<td>857</td>
</tr>
<tr>
<td>$1 \times 10^{-8}$</td>
<td>1301</td>
</tr>
<tr>
<td>$1 \times 10^{-10}$</td>
<td>1467</td>
</tr>
</tbody>
</table>

Number of trajectories against $\alpha$ ($\beta = \alpha$ and $2\delta = 0.02$)

- $m$ increases logarithmically if $\alpha$ and/or $\beta$ decrease.
Fact

Flipping a coin is nothing more than testing whether a finite execution satisfies a property.
From Flipping a coin to Model Checking

Fact
Flipping a coin is nothing more than testing whether a finite execution satisfies a property.

Results
Consequence: Wald’s testing “directly” (to be discussed) applies to model check properties of white-box stochastic systems.
From Flipping a coin to Model Checking

Fact

Flipping a coin is nothing more than testing whether a finite execution satisfies a property.

Results

Consequence: Wald’s testing “directly” (to be discussed) applies to model check properties of white-box stochastic systems.

Properties

- **Natural**: those that can be checked on finite executions: next, bounded until;
- **Going further**: until.
What do we bring to the picture?

- Monte Carlo and simulations are used in industry since decades
- We give a formal semantic to these approaches: Now we have a proof that their results are correct!
- We have our mathematical models that give us insights about system’s behavior and permit to improve statistical techniques
- We provide our runtime procedures and our competences in terms of modelization and reasoning;
- Sometimes the link between formal methods and SMC is more complex!
Why are nonpure systems forbidden?

- We sample a unique distribution;
- Sampling several distributions would require to distinguish between them;
- This cannot be done on the sole basis of running the system;
- Solution: let's see what they have been doing in formal verification!
Advantages

- Easy to parallelize (independent sampling, unbiased distributed sampling);
Advantages

- Easy to parallelize (independent sampling, unbiased distributed sampling);
- Independent of system’s size;
Advantages

- Easy to parallelize (independent sampling, unbiased distributed sampling);
- Independent of system’s size;
- Independent of system’s probability distribution;
Advantages

- Easy to parallelize (independent sampling, unbiased distributed sampling);
- Independent of system’s size;
- Independent of system’s probability distribution;
- Easy to trade accuracy for speed;
Advantages

- Easy to parallelize (independent sampling, unbiased distributed sampling);
- Independent of system’s size;
- Independent of system’s probability distribution;
- Easy to trade accuracy for speed;
- Uniform approach;
Advantages

- Easy to parallelize (independent sampling, unbiased distributed sampling);
- Independent of system’s size;
- Independent of system’s probability distribution;
- Easy to trade accuracy for speed;
- Uniform approach;
- Easy to implement (well depends on the runtime procedure!)
Advantages

- Easy to parallelize (independent sampling, unbiased distributed sampling);
- Independent of system’s size;
- Independent of system’s probability distribution;
- Easy to trade accuracy for speed;
- Uniform approach;
- Easy to implement (well depends on the runtime procedure!)
- Can approach undecidability;
A Note on Parallelization

- Observations are generated by different machines;
- Observations must be independent;
A Note on Parallelization

- Observations are generated by different machines;
- Observations must be independent:
  - Using different seeds is not sufficient: it only determines initial numbers, not the way the sequence is generated;
- Solution:
A Note on Parallelization

- Observations are generated by different machines;
- Observations must be independent;
  - Using different seeds is not sufficient: it only determines initial numbers, not the way the sequence is generated;
- Solution:
  
  encode process ID directly is the generator.
A Note on Parallelization

- Observations are generated by different machines;
- Observations must be independent;
  - Using different seeds is not sufficient: it only determines initial numbers, not the way the sequence is generated;
  - Solution:
    
    encode process ID directly is the generator.

- Slave - master: experiments are collected in ring-order.
3 interesting experiments

- $\Delta - \Sigma$ Modulator (conversion: analogue to digital). Interesting as the property cannot be expressed in temporal logic (reasoning on an execution);
- Systems biology. Interesting as those systems are of huge size (CMACS project: Pancreatic Cancer!);
- Heterogenous systems. Airplan - concrete case study!
Model Checking mixed-signal circuits

- Mature for digital designs but still new for analog and mixed design
- Difficult due to continuous and hybrid state variables
Model Checking mixed-signal circuits

- Mature for digital designs but still new for analog and mixed design
- Difficult due to continuous and hybrid state variables

Probabilistic Model Checking

- Stochastic systems and/or stochastic uncertainties
- Exact solution is a difficult problem in general
Model Checking mixed-signal circuits

- Mature for digital designs but still new for analog and mixed design
- Difficult due to continuous and hybrid state variables

Probabilistic Model Checking

- Stochastic systems and/or stochastic uncertainties
- Exact solution is a difficult problem in general

Statistical Approach

- Use of numerical simulation
- Approximate solution with bounds on errors
Outline

1. Stochastic Systems
2. The Algebraic Approach
3. Statistical Model Checking
4. Experiments
5. Bayesian Model Checking
6. What’s next?
States
Stochastic Signal Discrete Time Event System (SSDES)

States

\[ s_0 \xrightarrow{p_0} s_1 \xrightarrow{p_1} s_2 \xrightarrow{p_2} \cdots \xrightarrow{p_N} s_N \]

execution
Stochastic Signal Discrete Time Event System (SSDES)

States

\[ s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \ldots \rightarrow s_N \]

analog

\[ \text{pia}_0 \rightarrow \text{pia}_1 \rightarrow \text{pia}_2 \rightarrow \ldots \rightarrow \text{pia}_N \]

time

execution
Stochastic Signal Discrete Time Event System (SSDES)

States

\[ s_0 \rightarrow s_1 \rightarrow s_2 \rightarrow \cdots \rightarrow s_N \]

execution

analog

\[ p_{ia0}, p_{ia1}, p_{ia2}, \ldots, p_{iaN} \]

digital

\[ p_{id0}, p_{id1}, p_{id2}, \ldots, p_{idN} \]
Logics: LTL formulas

Let $B$ be a set of predicates. The following defines an LTL formula:

$$\phi ::= T | F | b \in B | \neg \phi | \phi_1 \lor \phi_2 | \Box \phi | \phi_1 U \phi_2.$$
Let $B$ be a set of predicates. The following defines an LTL formula:

$$\phi ::= T | F | b \in B | \neg \phi | \phi_1 \lor \phi_2 | \bigcirc \phi | \phi_1 U \phi_\in.$$ 

Let $\omega = s_1s_2...s_k$, $|\omega| = k$, $\omega^i = s_is_{i+1}...s_k$, $\omega(i) = s_i$ and $L$ be a mapping from $S$ to $2^B$. We have:

- $\omega \models T$, $\omega \not\models F$ and $\omega \models \neg \phi$ iff $\omega \not\models \phi$
- $\omega \models b$ with $b \in B$ iff $b \in L(\omega(0))$
- $\omega \models \phi_1 \lor \phi_2$ iff $\omega \models \phi_1$ or $\omega \models \phi_2$
- $\omega \models \bigcirc \phi$ iff $|\omega| > 1$ and $\omega^1 \models \phi$
- $\omega \models \phi_1 U \phi_\in$ iff there exists $0 \leq i \leq |\omega| - 1$ such that $\omega^i \models \phi_2$, and for each $0 \leq j < i$, $\omega^j \models \phi_1$

Additionally, we use the eventually operator $\bigdiamond$ defined as

$$\bigdiamond \phi = F U \phi.$$ 

Note that we only consider finite executions.
Logics: LTL formulas

Let $\mathcal{B}$ be a set of predicates. The following defines an LTL formula:

$$\phi ::= T | F | b \in \mathcal{B} | \neg \phi | \phi_1 \lor \phi_2 | \diamond \phi | \phi_1 U \phi_2.$$ 

Let $\omega = s_1s_2...s_k$, $|\omega| = k$, $\omega^i = s_is_{i+1}...s_k$, $\omega(i) = s_i$ and $L$ be a mapping from $S$ to $2^\mathcal{B}$. We have:

- $\omega \models T$, $\omega \not\models F$ and $\omega \models \neg \phi$ iff $\omega \not\models \phi$
- $\omega \models b$ with $b \in \mathcal{B}$ iff $b \in L(\omega(0))$
- $\omega \models \phi_1 \lor \phi_2$ iff $\omega \models \phi_1$ or $\omega \models \phi_2$
- $\omega \models \diamond \phi$ iff $|\omega| > 1$ and $\omega^1 \models \phi$
- $\omega \models \phi_1 U \phi_2$ iff there exists $0 \leq i \leq |\omega| - 1$ such that $\omega^i \models \phi_2$, and for each $0 \leq j < i$, $\omega^j \models \phi_1$

Additionally, we use the eventually operator $\diamond$ defined as $\diamond \phi = F U \phi$. Note that we only consider finite executions.
Logics: Execution Predicates

**Definition (Execution Predicate)**

Let $\Sigma(S)$ be the set of all the executions of an SSDES $S$. An execution predicate $p$ for $S$ is a mapping $p : \sigma \in \Sigma(S) \mapsto p(\sigma) \in \{T, F\}$. 

**Example Execution Predicate**

A execution predicate $p$ that decides whether the mean value of the analog signal associated with $\sigma$ is $\geq 0.7$:

$$p(\sigma) = \text{true} \iff \frac{1}{N-1} \sum_{k=0}^{N-1} \pi_a(\sigma(k)) \geq 0.7$$

More complex functionals such as the Fourier transform can be used.
Logics: Execution Predicates

**Definition (Execution Predicate)**

Let $\Sigma(S)$ be the set of all the executions of an SSDES $S$. An *execution predicate* $p$ for $S$ is a mapping $p : \sigma \in \Sigma(S) \mapsto p(\sigma) \in \{T, F\}$.

**Example**

Execution predicate $p$ that decides whether the mean value of the analog signal associated with $\sigma$ is $\geq 0$:

$$p(\sigma) = T \quad \text{iff} \quad \frac{1}{N} \sum_{k=0}^{N-1} \pi_a(\sigma(k)) \geq 0.$$

More complex functionals such as the Fourier transform can be used.
Logics: Execution Predicates

Definition (Execution Predicate)
Let $\Sigma(S)$ be the set of all the executions of an SSDES $S$. An execution predicate $p$ for $S$ is a mapping $p: \sigma \in \Sigma(S) \mapsto p(\sigma) \in \{T, F\}$.

Example
Execution predicate $p$ that decides whether the mean value of the analog signal associated with $\sigma$ is $\geq 0$:

$$p(\sigma) = T \quad \text{iff} \quad \frac{1}{N} \sum_{k=0}^{N-1} \pi_a(\sigma(k)) \geq 0.$$ 

More complex functionals such as the Fourier transform can be used.

Claim
Let $S$ be an SSDES and $\phi$ be a Boolean combination of LTL formulas and execution predicates. One can always associate a probability with the set of executions of $S$ that satisfy $\phi$. 
Outline

1. Stochastic Systems
2. The Algebraic Approach
3. Statistical Model Checking
4. Experiments
5. Bayesian Model Checking
6. What’s next?
Δ – Σ Modulators for Dummies

**Analog to Digital converters (ADC)**

- Converts analog signal into digital signals
- Used in many electrical devices interfacing with a physical environment
Δ – Σ Modulators for Dummies

Analog to Digital converters (ADC)

- Converts analog signal into digital signals
- Used in many electrical devices interfacing with a physical environment

Δ – Σ modulators

- Widely used family of ADCs
- Efficient processing of the quantization error, i.e., the difference between the analog input and the digital output
A Simple Discrete-Time $\Delta - \Sigma$ Modulator

Principle Control of quantization error using a feedback loop
A Simple Discrete-Time $\Delta - \Sigma$ Modulator

Principle Control of quantization error using a feedback loop

\[ u(k) \xrightarrow{v(k)} e(k) \xrightarrow{\sum} x(k) \xrightarrow{\text{quantizer}} v(k) = v(k) \]

Input

Output

$u(k)$

$e(k)$

$x(k)$

$v(k)$
A Simple Discrete-Time $\Delta - \Sigma$ Modulator

Principle Control of quantization error using a feedback loop

- The quantization error is the difference between the input and the output
A Simple Discrete-Time $\Delta - \Sigma$ Modulator

**Principle Control of quantization error using a feedback loop**

- The **quantization error** is the difference between the input and the output.
- The **integrator** stores the summation of $\delta$s in a state variable $x$. 

$$u(k) \rightarrow v(k)$$ 
$$v(k) \rightarrow e(k) \rightarrow x(k) \rightarrow v(k)$$ 

**Diagram:**

- Input $u(k)$
- Output $v(k)$
- Error $e(k)$
- Integrator $x(k)$
- Quantizer $v(k)$
A Simple Discrete-Time $\Delta - \Sigma$ Modulator

Principle Control of quantization error using a feedback loop

- The quantization error is the difference between the input and the output
- The integrator stores the summation of $\delta$s in a state variable $x$
- The quantizer produces the output based on the sign of $x$
Higher Order $\Delta - \Sigma$ Modulators

- More complex designs use more than one integrator
- The *order* of a $\Delta - \Sigma$ modulator is the number of integrators used
- Beginning from order three, a *stability* issue appears
- i.e. the integrators states can reach a *saturation* threshold ($|x| > 1$) compromising the analog to digital conversion
Experiments

Experiments with a Third Order $\Delta – \Sigma$ Modulator

Outline

1. Stochastic Systems
2. The Algebraic Approach
3. Statistical Model Checking
4. Experiments
5. Bayesian Model Checking
6. What’s next?
Questions and Existing Results

First Question
When does saturation occur?

Second Question
Does saturation always imply a bad conversion?
Questions and Existing Results

First Question
When does saturation occur?

Second Question
Does saturation always imply a bad conversion?

Existing Results
- Hybrid system model;
- Some answer to the first question for a limited horizon;
- Nothing for the second question (Fourier transform!).
- We get a stochastic system by randomly choosing the inputs $u(k)$
- State $s_k$ is the tuple $(u(k), x_1(k), x_2(k), x_3(k), v(k))$
- The next state $s_{k+1}$ is determined by the random choice of $u(k+1)$ and computed by the Simulink engine
- For all $k$, $u(k)$ is chosen uniformly in $[-u_{\text{max}}, u_{\text{max}}]$
A third order $\Delta - \Sigma$ modulator, Simulink model

- We get a stochastic system by randomly choosing the inputs $u(k)$
- State $s_k$ is the tuple $(u(k), x_1(k), x_2(k), x_3(k), v(k))$
- The next state $s_{k+1}$ is determined by the random choice of $u(k+1)$ and computed by the Simulink engine
- For all $k$, $u(k)$ is chosen uniformly in $[-u_{\text{max}}, u_{\text{max}}]$
A third order $\Delta - \Sigma$ modulator, Simulink model

- We get a stochastic system by randomly choosing the inputs $u(k)$
- State $s_k$ is the tuple $(u(k), x_1(k), x_2(k), x_3(k), v(k))$
- The next state $s_{k+1}$ is determined by the random choice of $u(k + 1)$ and computed by the Simulink engine
- For all $k$, $u(k)$ is chosen uniformly in $[−u_{\text{max}}, u_{\text{max}}]$

$\Rightarrow$ Statistical analysis for all input signals of amplitude bounded by $u_{\text{max}}$
Saturation Analysis

Probability of saturation occurrence for different values of $u_{\text{max}}$?
Saturation Analysis

Probability of saturation occurrence for different values of $u_{\text{max}}$?

- Let $Satur$ be a boolean predicate
- For all state $s = (u, x_1, x_2, x_3, v)$, let $L(s) = \{Satur\}$ iff $|x_3| \geq 1$

We can then evaluate the formula $Pr_{\geq \theta}(\diamond Satur)$. 
Saturation Analysis

Probability of saturation occurrence for different values of $u_{\text{max}}$?

- Let $Satur$ be a boolean predicate
- For all state $s = (u, x_1, x_2, x_3, v)$, let $L(s) = \{Satur\}$ iff $|x_3| \geq 1$

We can then evaluate the formula $Pr_{\geq \theta}(\Diamond Satur)$.

A tool:
- A routine checking $\sigma \models \Diamond Satur$
- The sequential ratio testing algorithm which decides whether $S \models Pr_{\geq \theta}(\phi)$ given $\theta, \alpha, \beta$ and $\delta$
- A simple bisection procedure which tries to maximize the value of $\theta$ for which the answer is true
**Experimental Results**

<table>
<thead>
<tr>
<th>$u_{\text{max}}$</th>
<th>Hypothesis Accepted</th>
<th>Number of executions</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>$p \leq 0$</td>
<td>416</td>
</tr>
<tr>
<td>0.15</td>
<td>$p \geq 0.0938$</td>
<td>4967</td>
</tr>
<tr>
<td>0.2</td>
<td>$p \geq 0.640625$</td>
<td>17815</td>
</tr>
<tr>
<td>0.25</td>
<td>$p \geq 0.984375$</td>
<td>416</td>
</tr>
<tr>
<td>0.3</td>
<td>$p \geq 1$</td>
<td>688</td>
</tr>
</tbody>
</table>

Table of results for $p = Pr(\sigma \models \diamond Satur)$, with $\alpha = \beta = 1e^{-3}$ and $\delta = 1e^{-2}$
**Experimental Results**

<table>
<thead>
<tr>
<th>$u_{\text{max}}$</th>
<th>Hypothesis</th>
<th>Number of executions</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>$p \leq 0$</td>
<td>416</td>
</tr>
<tr>
<td>0.15</td>
<td>$p \geq 0.0938$</td>
<td>4967</td>
</tr>
<tr>
<td>0.2</td>
<td>$p \geq 0.640625$</td>
<td>17815</td>
</tr>
<tr>
<td>0.25</td>
<td>$p \geq 0.984375$</td>
<td>416</td>
</tr>
<tr>
<td>0.3</td>
<td>$p \geq 1$</td>
<td>688</td>
</tr>
</tbody>
</table>

Table of results for $p = P_r(\sigma \models \Diamond Satur)$, with $\alpha = \beta = 1e^{-3}$ and $\delta = 1e^{-2}$

- Consistent with results formally obtained in [Dang Donze Maler 04] but on a much larger horizon (24000 as compared to 31)
## Experimental Results

<table>
<thead>
<tr>
<th>( u_{\text{max}} )</th>
<th>Hypothesis</th>
<th>Number of executions</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>( p \leq 0 )</td>
<td>416</td>
</tr>
<tr>
<td>0.15</td>
<td>( p \geq 0.0938 )</td>
<td>4967</td>
</tr>
<tr>
<td>0.2</td>
<td>( p \geq 0.640625 )</td>
<td>17815</td>
</tr>
<tr>
<td>0.25</td>
<td>( p \geq 0.984375 )</td>
<td>416</td>
</tr>
<tr>
<td>0.3</td>
<td>( p \geq 1 )</td>
<td>688</td>
</tr>
</tbody>
</table>

Table of results for \( p = Pr(\sigma \models \Diamond Satur) \), with \( \alpha = \beta = 1e^{-3} \) and \( \delta = 1e^{-2} \)

- Consistent with results formally obtained in [Dang Donze Maler 04] but on a much larger horizon (24000 as compared to 31)
- The expected number of simulations grows logarithmically w.r.t. the inverse of \( \alpha \) and \( \beta \) and polynomially w.r.t. the inverse of \( \delta \)
Experiments with a Third Order $\Delta - \Sigma$ Modulator

Frequency Domain

F1

F2

quantization error

nu
• Quantization pushes error towards high frequencies;
• **Suggestion**: Check for quality under small frequencies.
Execution Predicate in the Frequency Domain

- Let $F_u(\sigma)$ and $F_v(\sigma)$ be the Fourier Transforms (FTs) of the input signal associated with $\sigma$.
- Let $d_f^{\nu_0}(\hat{\xi}_1, \hat{\xi}_2)$ be a measure of the distance between two FTs $\hat{\xi}_1$ and $\hat{\xi}_1$ for frequencies smaller than $\nu_0$.
- Then we can derive an execution predicate $p_f$ such that

\[ p_f(\sigma) = T \text{ iff } d_f^{\nu_0}(F_u(\sigma), F_v(\sigma)) \leq \epsilon, \]

For $\nu_0 = 100Hz$ and $\epsilon \leq .1$ the predicate discriminates between “correct” and “failed” conversions.
**Frequency Domain Predicate, Experimental Results**

<table>
<thead>
<tr>
<th>$u_{\text{max}}$</th>
<th>Hypothesis Accepted</th>
<th>Number of Executions</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.8</td>
<td>$p \geq 1$</td>
<td>688</td>
</tr>
<tr>
<td>0.9</td>
<td>$p \geq 0.984375$</td>
<td>612</td>
</tr>
<tr>
<td>1.0</td>
<td>$p \geq 0.984375$</td>
<td>1248</td>
</tr>
<tr>
<td>1.1</td>
<td>$p \geq 0.875$</td>
<td>6388</td>
</tr>
<tr>
<td>1.2</td>
<td>$p \geq 0.578125$</td>
<td>15507</td>
</tr>
</tbody>
</table>

Table of results for $p = Pr(p_f)$, with $\alpha = \beta = 1e^{-3}$ and $\delta = 1e^{-2}$
Experiments Interpretation

The previous results show that

- For $u_{\text{max}} \geq 0.3$ the system satisfies $\diamond Satur$ with probability 1
- For $u_{\text{max}} \leq 0.8$ the system satisfies $p_f$ with probability 1

Thus we statistically established that for $0.3 \leq u_{\text{max}} \leq 0.8$, the formula $\diamond Satur \land p_f$ is satisfied with probability 1, meaning that saturation can occur without a dramatic decrease in the conversion quality.

This extends the results in [Gupta Krogh Rutenbar 04] and [Dang Donze Maler 04] where it was conservatively assumed that the absence of saturation was necessary for a proper behavior.
Conclusion on $\Delta - \Sigma$ Modulator

Summary

- A framework for the statistical probabilistic Model Checking of mixed-signal circuits
- The simulation-based approach makes it easier to deal with functionals on executions such as the Fourier transform
- Application to a non-trivial case study for which we improved previous results
Conclusion on $\Delta - \Sigma$ Modulator

Summary

- A framework for the statistical probabilistic Model Checking of mixed-signal circuits
- The simulation-based approach makes it easier to deal with functionals on executions such as the Fourier transform
- Application to a non-trivial case study for which we improved previous results

Future work

- Extension to unbounded execution and dense time using appropriate monitoring techniques
- Logic mixing temporal properties and partial execution predicates
- More precise definitions and specifications for frequency domain properties based on the need of analog designers
In presence of a few species, reactions are defined in terms of stochastic processes;

In such context, one wants to exercise the master equation that governs system’s evolution.

**Definition**

The master equation: phenomenological set of first-order differential equations describing the time evolution of the probability of a system to occupy each one of a discrete set of states.
Situation

- We want to Solve stochastic equations, but
- Many stochastic equations are numerically intractable!
BionetGen and Gillespie

- **BionetGen Toolset:**
  Very simple language to model proteins and proteins-proteins interactions:
  Uses rewriting rules like the k-calculus

- Gillespie algorithm simulates rule applications (Continuous-timed Markov Chains);

- Systems can be big: more than 6 hours for a simulation!  
  $\Rightarrow$ distributed implementation.
The language allows to describe:

- Molecules and functional;
- States of functional;
- Binding between functional and molecules;
- Chemical reactions;
- ....

Available at:

http://bionetgen.org/index.php/Main_Page
Example

Molecule

R(l,d,Y~P)

Example

Chemical reaction

L(r) + R(l,d) \leftrightarrow L(r!1).R(l!1,d) \; kp1, \; km1
Biolab

- Combine BionetGen with SPRT;
- A logic for biologist;
- Formal validation of observations (T-Cell model, ...).
Architecture of BIOLAB

Experiments
System's Biology

Sequential Hypothesis Testing Algorithm

BioNetgen Model

BioNetgen

Simulation Trace

Temporal Logic Property

Trace Verifier

Property Verified / Failed by Trace

Verified / Failed Property by Trace

Simulation T Hypothesis Testing for Solving the Probabilistic Model Checking Problem
The T-Cell model

- Detect antigen and should react properly;
- Should not react to non-pathological proteins;
- Property: the system can alternate between reactive and nonreactive states.
Bayesian Testing

1. Prior probability (informative Vs. non informative) on $H_0$ and $H_1$;
2. Prior information is used to decrease the number of experiments;
3. Bayesian Testing is more driven towards compound hypothesis than statistical hypothesis testing!

**Future Work**: Bayesian risk and nested operators.
Non Informative Prior

<table>
<thead>
<tr>
<th>$\theta$</th>
<th>$Bayes' Factor Test$</th>
<th>$SPRT (\delta = 0.01)$</th>
<th>$SPRT (\delta = 0.001)$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$H_1$</td>
<td>$H_0$</td>
<td>$H_1$</td>
</tr>
<tr>
<td>0.95</td>
<td>2</td>
<td>275</td>
<td>349</td>
</tr>
<tr>
<td>0.9</td>
<td>8</td>
<td>610</td>
<td>608</td>
</tr>
<tr>
<td>0.8</td>
<td>35*</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.7</td>
<td>81*</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.6</td>
<td>591*</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.5</td>
<td>272</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.4</td>
<td>156</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.3</td>
<td>16</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0.2</td>
<td>5</td>
<td>909</td>
<td>929</td>
</tr>
<tr>
<td>0.1</td>
<td>9</td>
<td>446</td>
<td>468</td>
</tr>
<tr>
<td>0.05</td>
<td>2</td>
<td>201</td>
<td>189</td>
</tr>
</tbody>
</table>
What else could you learn?

- Learning state space of embedded systems with statistics;
- Timed Stochastic systems (uppaal approach)
- Model Checking PCTL* using hypothesis testing;
- Nested probability operators;
- Black-box systems.