The What, Why, and How of Probabilistic Verification
Part 1: Motivation and Models

Joost-Pieter Katoen

UNIVERSITY OF TWENTE.

CAV Invited Tutorial 2015, San Francisco
Roadmap of This Tutorial

Part 1. Motivation and Models
  ▶ More Than 5 Reasons for Probabilistic Analysis
  ▶ Elementary Models and Properties

Part 2. Algorithmic Foundations
  ▶ Reachability and Beyond in Discrete Markov Models
  ▶ Timed Reachability in Continuous Markov Models

Part 3. Treating Gigantic Markov Models
  ▶ Abstraction: Precise, Aggressive, and Compositional

Part 4. Recent Research Developments
  ▶ Parameter Synthesis and Model Repair
  ▶ Counterexample Generation
  ▶ Probabilistic Programming
Overview

The Relevance of Probabilities

Markov Models and Properties
The Relevance of Probabilities

Markov Models and Properties
The Relevance of Probabilities

More Than Five Reasons for Probabilities

1. Randomised Algorithms
2. Reducing Complexity
3. Probabilistic Programming
4. Reliability
5. Performance
6. Optimization
7. Systems Biology
Heads = “go left”; tails = “go right”.

Randomised Algorithms: Simulating a Die  
[Knuth & Yao, 1976]
Heads = “go left”; tails = “go right”. Does this model a six-sided die?
Distributed Computing

**FLP impossibility result**

[FLP impossibility result, Fischer et al., 1985]

In an asynchronous setting, where only one processor might crash, there is no distributed algorithm that solves the consensus problem—getting a distributed network of processors to agree on a common value.
Distributed Computing

FLP impossibility result

In an asynchronous setting, where only one processor might crash, there is no distributed algorithm that solves the consensus problem—getting a distributed network of processors to agree on a common value.

Ben-Or’s possibility result

If a process can make a decision based on its internal state, the message state, and some probabilistic state, consensus in an asynchronous setting is almost surely possible.
Reducing Complexity: Matrix Multiplication

Input: three $O(N^2)$ square matrices $A$, $B$, and $C$

Output: yes, if $A \times B = C$; no, otherwise
Reducing Complexity: Matrix Multiplication

[Freivald, 1977]

**Input:** three $O(N^2)$ square matrices $A$, $B$, and $C$

**Output:** yes, if $A \times B = C$; no, otherwise

**Deterministic:** compute $A \times B$ and compare with $C$

**Complexity:** in $O(N^3)$, best known complexity $O(N^{2.37})$
Reducing Complexity: Matrix Multiplication

Input: three $O(N^2)$ square matrices $A$, $B$, and $C$

Output: yes, if $A \times B = C$; no, otherwise

Deterministic: compute $A \times B$ and compare with $C$

Complexity: in $O(N^3)$, best known complexity $O(N^{2.37})$

Randomised: 1. take a random bit-vector $\vec{x}$ of size $N$
2. compute $A \times (B \vec{x}) - C \vec{x}$
3. output yes if this yields the null vector; no otherwise
4. repeat these steps $k$ times
Reducing Complexity: Matrix Multiplication [Freivald, 1977]

**Input:** three $O(N^2)$ square matrices $A$, $B$, and $C$

**Output:** yes, if $A \times B = C$; no, otherwise

**Deterministic:** compute $A \times B$ and compare with $C$

**Complexity:** in $O(N^3)$, best known complexity $O(N^{2.37})$

**Randomised:**
1. take a random bit-vector $\vec{x}$ of size $N$
2. compute $A \times (B \vec{x}) - C \vec{x}$
3. output yes if this yields the null vector; no otherwise
4. repeat these steps $k$ times

**Complexity:** in $O(k \cdot N^2)$, with false positive with probability $\leq 2^{-k}$
2013, DARPA launched a 48M (US dollar) program on
“Probabilistic Programming (PP) for Advanced Machine Learning (ML)”

“PP is a new programming paradigm for managing uncertain information. By incorporating it into ML, we seek to greatly increase the number of people who can successfully build ML applications, and make ML experts radically more effective”.
Probabilistic Programming: Once Upon a Time ....

C’ERA UNA VOLTA IL WEST

EURO INTERNATIONAL FILMS PRESENTA

Joost-Pieter Katoen
Duelling Cowboys

[McIver and Morgan, 2005]

```c
int cowboyDuel(float a, b) {
    int t := A [] t := B; // decide cowboy for first shooting
    bool c := true;
    while (c) {
        if (t = A) {
            (c := false [a] t := B); // A shoots B with prob. a
        } else {
            (c := false [b] t := A); // B shoots A with prob. b
        }
    }
    return t; // the survivor
}
```
The Relevance of Probabilities

Duelling Cowboys

[McIver and Morgan, 2005]

Claim: cowboy A wins the duel with probability at least

\[
\frac{(1-b) \cdot a}{a + b - a \cdot b}
\]
Claim: cowboy A wins the duel with probability at least \( \frac{(1-b) \cdot a}{a+b-a \cdot b} \)
Claim: cowboy A wins the duel with probability at least \( \frac{(1-b)\cdot a}{a+b-a\cdot b} \)

Usage: security, machine learning, approximate computing
The Relevance of Probabilities

Reliability Engineering
Reliability: (Dynamic) Fault Trees

(a) OR
(output)
(inputs)

(b) AND
(output)
(inputs)

(c) VOTING
(output)
(inputs)

(k/n)

(d) PAND
(output)
(inputs)

(e) SPARE
(output)
(inputs)

Primary
Spare

(f) FDEP
(dummy output)

(trigger)

Dependent events
A Fault Tree Example

Road trip fails

- Car fails
  - Phone
  - Tires fail
    - Engine
      - Tire
      - Tire
      - Tire
      - Tire
      - Spare
A Fault Tree Example

(D)FTs: one of—if not the—most prominent models for risk analysis
Aims: quantify system reliability and availability, MTTF, ...
The Relevance of Probabilities

Reliability: Architectural Languages

[Feiler et al., 2010]

```
[Device] Battery.imp: batt1
Nominal
charged
energy' = -0.02
energy' >= 20
voltage := f(energy)
empty => energy < 20
depleted
energy' = -0.03
voltage := f(energy)

Error
ok
empty
dead

Data
charged
energy' = -0.02
energy' >= 20
voltage := f(energy)
empty => energy < 20
depleted
energy' = -0.03
voltage := f(energy)
ok
dead

[Device] Battery.imp: batt2
Nominal
charged
energy' = -0.02
energy' >= 20
voltage := f(energy)
empty => energy < 20
depleted
energy' = -0.03
voltage := f(energy)
ok
dead

Error
ok
empty
dead

Data
charged
energy' = -0.02
energy' >= 20
voltage := f(energy)
empty => energy < 20
depleted
energy' = -0.03
voltage := f(energy)
energy init 100
```
Reliability: Architectural Languages

[Feiler et al., 2010]

```
error model BatteryFailure
features
  ok: initial state;
  dead: error state;
  batteryDied: out error propagation;
end BatteryFailure;

error model implementation BatteryFailure.Imp
events
  fault: error event occurrence poisson 0.01;
transitions
  ok -[fault]-> dead;
  dead -[batteryDied]-> dead;
end BatteryFailure.Imp;
```

Fault injection

In error state dead, voltage := 0
The early days:
Performance: GSPNs

[Jamone Marsan et al., 1984]

The early days:

More modern times: Petri nets with

- Timed transitions
- Immediate transitions
- Natural weights
Performance: GSPNs

The early days:

![Petri net diagram](image)

More modern times: Petri nets with

- Timed transitions
- Immediate transitions
- Natural weights

Aims: quantify arrivals, waiting times, QoS, soft deadlines, .......

GSPNs: very—if not the most—popular in performance modeling
Stochastic Scheduling
Stochastic Scheduling

JOSE NINO-MORA
Department of Statistics,
Universidad Carlos III de Madrid, Getafe, Spain

MSC2000: 90B36

Article Outline
Introduction
Models
  Scheduling a Batch of Stochastic Jobs
  Multi-Armed Bandits
  Scheduling Queueing Systems
References

Introduction

The field of stochastic scheduling is motivated by problems of priority assignment arising in a variety of systems where jobs with random features (e.g., arrival or optimal performance.

The theory of stochastic scheduling is a goal in the idealized models. Real-world random arrival or processing times among their probability distribution vary across several different scheduling policies considering interarrival and processing times. The arrangement of service objectives to be optimized are required to be nonanticipating and cannot make use of future information. Known total durations are yet finished.

Regarding solution methods, it seems fair to say that computational tools are yet available to design optimal policies across stochastic scheduling models. These problems can be cast in the following straightforward form:

...
Stochastic Scheduling

- Job processing times are subject to random variability
  - machine breakdowns and repairs, job parameters, ...
  - $N$ independent jobs with mean duration $\frac{1}{\mu_i}$
  - $M$ identical machines
  - job processing with (or without) pre-emption

- **Objective** = minimal expected makespan—finishing time of last job

- SEPT policy yields minimal expected makespan (Bruno et al., JACM 1981)
  “it is hard to calculate these expected values”

Which policy maximises the probability to finish all jobs on time?
Stochastic Scheduling \((N = 4; M = 2)\)

\[
\begin{align*}
\Pr\{\text{Job 4 finishes first}\} &= \frac{\mu_4}{\mu_3 + \mu_4} \\
\Pr\{\text{Job 3 finishes first}\} &= \frac{\mu_3}{\mu_3 + \mu_4}
\end{align*}
\]
Stochastic Scheduling \((N = 4; M = 2)\)

\[ \Pr\{\text{Job 1 finishes first}\} = \frac{\mu_1}{\mu_1 + \mu_2} \]

\[ \Pr\{\text{Job 2 finishes first}\} = \frac{\mu_2}{\mu_1 + \mu_2} \]
Stochastic Scheduling \((N = 4; M = 2)\)
Stochastic Model
Systems Biology
Systems Biology

Enzyme-catalysed substrate conversion

Enzyme kinetics is the investigation of how enzymes bind substrates and turn over data used in kinetic analyses are obtained from enzyme assays.

In 1902 Victor Henri proposed a quantitative theory of enzyme kinetics, but the equations were not useful because the significance of the hydrogen ion concentration was not understood. After Peter Lauritz Sørensen had defined the logarithmic pH-scale and introduced buffering in 1909, the German chemist Leonor Michaelis and his Canadian colleague Kenji Menten repeated Henri's experiments and confirmed his equation which is now known as the Henri-Michaelis-Menten kinetics (sometimes also Michaelis-Menten kinetics). The Michaelis-Menten equation was developed by G. E. Briggs and J. B. S. Haldane, who derived kinetic equations still used today.

The major contribution of Henri was to think of enzyme reactions in two stages. In the first, the substrate binds reversibly to the enzyme to form an enzyme-substrate complex. This is sometimes called the Michaelis complex. The enzyme then catalyzes the chemical step in the reaction to form products.

Enzymes can catalyze up to several million reactions per second. For example, the reaction catalyzed by the enzyme carbonic anhydrase is

\[
2 
\text{CO}_2 \rightarrow \text{H}_2 \text{CO}_3 
\]

which is a reversible reaction, but under certain conditions can be effectively irreversible.
Stochastic Chemical Kinetics

- Types of reaction described by **stochastic equations**:
  \[ E + S \overset{k_1}{\underset{k_2}{\rightleftharpoons}} C \overset{k_3}{\rightarrow} E + P \]

- \( N \) different types of molecules that **randomly collide**
  where state \( X(t) = (x_1, \ldots, x_N) \) with \( x_i = \# \) molecules of sort \( i \)
Stochastic Chemical Kinetics

- Types of reaction described by stochiometric equations:
  \[ E + S \overset{k_1}{\underset{k_2}{\rightleftharpoons}} C \overset{k_3}{\rightarrow} E + P \]

- \( N \) different types of molecules that randomly collide
  where state \( X(t) = (x_1, \ldots, x_N) \) with \( x_i = \# \) molecules of sort \( i \)

- Reaction probability within infinitesimal interval \( [t, t+\Delta) \):
  \[ \alpha_m(\vec{x}) \cdot \Delta = \Pr\{\text{reaction } m \text{ in } [t, t+\Delta) \mid X(t) = \vec{x}\} \]
  where \( \alpha_m(\vec{x}) = k_m \cdot \# \) possible combinations of reactant molecules in \( \vec{x} \)
Stochastic Chemical Kinetics

- Types of reaction described by stochiometric equations:
  \[ E + S \overset{k_1}{\underset{k_2}{\rightleftharpoons}} C \overset{k_3}{\rightarrow} E + P \]

- \( N \) different types of molecules that randomly collide
  where state \( X(t) = (x_1, \ldots, x_N) \) with \( x_i = \# \) molecules of sort \( i \)

- Reaction probability within infinitesimal interval \([t, t+\Delta)\):
  \[ \alpha_m(\vec{x}) \cdot \Delta = \Pr\{\text{reaction } m \text{ in } [t, t+\Delta) \mid X(t) = \vec{x}\} \]
  where \( \alpha_m(\vec{x}) = k_m \cdot \# \) possible combinations of reactant molecules in \( \vec{x} \)

- Process has the Markov property and is time-homogeneous
Substrate Conversion in the Small

States:
- init: enzymes 2, substrates 4, complex 0, products 0
- goal: enzymes 2, substrates 0, complex 0, products 4

Transitions: $E + S \xrightleftharpoons[1]{1} C^{0.001} \rightarrow E + P$

$e.g., (x_E, x_S, x_C, x_P) \xrightarrow{0.001 \cdot x_C} (x_E + 1, x_S, x_C - 1, x_P + 1)$ for $x_C > 0$
The Relevance of Probabilities

Markov Models and Properties
Common Feature

All these applications consider Markov models$^1$

$^1$Non-exponential distributions are approximated by phase-type distributions.
Discrete-Time Markov Models

A Markov chain for Knuth-Yao’s algorithm

A Markov decision process for the cowboy program
Continuous-Time Markov Models

A Markov chain for substrate conversion

A Markov decision process for the GSPN
Fault Trees are Continuous-Time MDPs

Markov models of a cold, warm and hot basic event
(dormancy factor $\mu = \alpha \cdot \lambda$)
Markov decision process for stochastic scheduling

Markov decision process\(^a\) for a SPARE gate

\(^a\)In fact, an interactive Markov chain.
# Markov Models

<table>
<thead>
<tr>
<th></th>
<th>Nondeterminism</th>
<th>Nondeterminism</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Discrete time</strong></td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>discrete-time Markov chain (DTMC)</td>
<td></td>
<td>Markov decision process (MDP)</td>
</tr>
<tr>
<td><strong>Continuous time</strong></td>
<td>CTMC</td>
<td>CTMDP</td>
</tr>
</tbody>
</table>

Other models: e.g., probabilistic variants of (priced) timed automata
### Properties

<table>
<thead>
<tr>
<th></th>
<th>Logic</th>
<th>Monitors</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Discrete time</strong></td>
<td>probabilistic CTL</td>
<td>deterministic automata</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(safety and LTL)</td>
</tr>
<tr>
<td><strong>Continuous time</strong></td>
<td>probabilistic timed CTL</td>
<td>deterministic timed automata</td>
</tr>
</tbody>
</table>
Markov Models and Properties

Properties

<table>
<thead>
<tr>
<th></th>
<th>Logic</th>
<th>Monitors</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Discrete time</strong></td>
<td>probabilistic CTL</td>
<td>deterministic automata (safety and LTL)</td>
</tr>
<tr>
<td><strong>Continuous time</strong></td>
<td>probabilistic timed CTL</td>
<td>deterministic timed automata</td>
</tr>
</tbody>
</table>

Core problem: computing (timed) reachability probabilities