Probabilistic Model Checking

An Introduction

Sebastian Junges
sjunges@berkeley.edu

These slides are partially based on slides by Joost-Pieter Katoen (with permission) from http://i-cav.org/2015/tutorials/
From Model Checking to Probabilistic Model Checking

General scheme from model checking

Given a model, exists a path to eventually reach a dangerous state?

Given a model, exists a path to eventually reach a dangerous state?

Given a model, what is the probability to eventually reach a dangerous state?

Probabilistic Model Checking

Model

Property "Reachability"

Answer

yes/no/probabilities

Given a model, is the probability to eventually reach a dangerous state above a threshold?

Given a model,

probabilities

eventually reach a dangerous state?
Knuth-Yao Die

In every state, flip a coin

Final states: die outcomes

1/2

1/2

1/2

1/2

1/2

1/2

1/2

1/2

1/2

1/2

1/2

1/2

1/2

1/2

1/2

1/6

1/6

1/6

1/6

1/6

1/6

1/6

1/6

1/6
Plan for today

3 Parts. Let’s see how far we get.

- Why Probabilistic Model Checking
- Basics of Probabilistic Model Checking
- Current Topics in Probabilistic Model Checking

Take home:

Probabilistic model checking on Markov models = graph-algorithms + equation system solving
Why probabilities?

Randomisation is everywhere

- Systems include randomization to solve more tasks,
- or tasks more efficiently.
- Stochastic processes can be an adequate abstraction of complex processes
- either technical, in nature, or both.

- Methods require to actually compute numbers, which is often hard, but
- humans are bad in reasoning under uncertainty, so automatic reasoning is helpful.
Distributed computing

Randomization is required to break symmetries

**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.

**Ben-Or’s possibility result**  
[Ben-Or, 1983]  
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.
Analysing Posteriors of (discrete) Probabilistic Programs

“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”.

```c
int cowboyDuel(float a, b) { // 0 < a < 1, 0 < b < 1
    int t := A [] t := B; // decide cowboy for first shooting
    turn
    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
}
```

cowboy A wins the duel with probability at least $\frac{(1-b)\cdot a}{a+b-a\cdot b}$
Fault Tree Analysis

The Prominent Reliability Engineering Model

Fault tree analysis

Given a system failure, what are its root causes in terms of component faults

Dynamic Fault Trees allow for typical but complex state-dependent failure propagation

Spare management, sequential failures

Quantitative Analysis: Given failure rates of the components, what is the mean time to failure, or the probability of mission success

NASA

US NRC
Stochastic Job Scheduling

Schedule $N$ independent jobs to $M$ servers, where the mean job duration is given by random variables

- $\text{job}_1 \lambda_1 = 4$
- $\text{job}_2 \lambda_2 = 1$
- $\text{job}_3 \lambda_3 = 10$
- $\text{job}_4 \lambda_4 = 6$
- $\text{job}_5 \lambda_5 = 8$

job scheduler

processor$_1$

processor$_2$

Inverse of expected execution times

• How do we optimize expected execution time (easy), probability of finishing $K$ jobs before a deadline (hard), or both (harder)?
Job Scheduling Example

State based model for 4 Jobs and 2 Servers
Markov Population Models

Prominent Model in Epidemiology, Social Networks, and Chemical Reactions

All these systems can be modelled with Markov models and we are always interested in reaching some configurations.
Why probabilities?

Randomisation is everywhere

- Systems include randomization to solve more tasks
- or tasks more efficiently
- Stochastic processes can be an adequate abstraction of complex processes
- Either technical, in nature, or both

- Methods require to actually compute numbers, which is often hard, but
- humans are bad in reasoning under uncertainty, so formal reasoning is helpful
Plan for today

3 Parts. Let’s see how far we get.

- Why Probabilistic Model Checking
- Basics of Probabilistic Model Checking
- Current Topics in Probabilistic Model Checking
# Markov Models

## Overview

<table>
<thead>
<tr>
<th>Nondeterminism</th>
<th>Discrete Time</th>
<th>Continuous Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>DTMC</td>
<td>CTMC</td>
</tr>
<tr>
<td>Nondeterminism</td>
<td>MDP</td>
<td>IMC/CTMDP</td>
</tr>
</tbody>
</table>
Discrete-time Markov Chains (DTMCs)

Formal Definition

- **States** \( S \)
- **Initial distribution** \( \tau \in \text{Distr}(S) \)
- **Transitions** \( P : S \rightarrow \text{Distr}(S) \)

Consider transitions as a transition matrix \( P \) with probabilities as entries \( P(s, s') \) or just an initial state.

- We may add atomic propositions and a (state)labelling to define sets of states.
Reachability in DTMCs

Sum over all paths

Problem statement
Consider a MC with finite state space $S$, $s \in S$ and $G \subseteq S$. Aim: determine $\Pr(s \models \diamond G) = \Pr_s\{ \pi \in \text{Paths}(s) \mid \pi \models \diamond G \}$

Paths

<table>
<thead>
<tr>
<th>States</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1$</td>
<td>$\frac{1}{8}$</td>
</tr>
<tr>
<td>$S_2$</td>
<td>$\frac{1}{128}$</td>
</tr>
<tr>
<td>$S_3$</td>
<td>$\frac{1}{8}$</td>
</tr>
<tr>
<td>$S_4$</td>
<td>$\frac{1}{4}$</td>
</tr>
<tr>
<td>$S_5$</td>
<td>$\frac{1}{8}$</td>
</tr>
<tr>
<td>$S_6$</td>
<td>$\frac{1}{8}$</td>
</tr>
<tr>
<td>$S_7$</td>
<td>$\frac{1}{8}$</td>
</tr>
</tbody>
</table>

$\frac{1}{8} \sum_{i=0}^{\infty} \left( \frac{1}{4} \right)^i$
Reachability in DTMCs

Characterization

Let $x(s)$ denote the probability to reach some target state from $s$. It holds that:

- If $s$ is a target state:
  $$x(s) = 1$$
- If there is no path from $s$ to some target state:
  $$x(s) = 0$$
- Otherwise:
  $$x(s) = \sum_{s' \in S} P(s, s') \cdot x(s')$$

Notice that these equations together have a unique solution.
Reachability in DTMCs

Example Equation System

\[
\begin{align*}
    x_1 &= 0 \\
    x_2 &= 1 \\
    x_3 &= \frac{1}{2} \cdot x_3 + \frac{1}{2} \cdot x_2 \\
    x_4 &= \frac{1}{2} \cdot x_5 + \frac{1}{2} \cdot x_1 \\
    x_5 &= 0
\end{align*}
\]

What is the probability to reach the red state?
Reachability in DTMCs

Characterization

Let \( x(s) \) denote the probability to reach some target state from \( s \). It holds that:

- If \( s \) is a target state:
  \[ x(s) = 1 \]

- If there is no path from \( s \) to some target state:
  \[ x(s) = 0 \]

- Otherwise:
  \[ x(s) = \sum_{s' \in S} P(s, s') \cdot x(s') \]

- Notice that these equations together have a unique solution
Long-run behavior and repeated reachability

Elementary property

Long-run Theorem

The set of all states in a terminal strongly connected component is reached with probability one.

In a terminal strongly connected component, each state is visited infinitely often with probability one.

For repeated reachability (globally eventually target set):

- Determine the terminal SCCs
- Consider those that contain at least one target state
- Determine the probability to reach these SCCs
Towards LTL Properties I

Automata Based Model Checking

Example: What is the probability for after throwing Heads initially, and throwing no more than two Tails total?

Label all states with init, heads, tails

All missing transitions go to a sink state!
Towards LTL Properties II

Product Construction

Intro to probabilistic model checking
LTL?

From finite automata to omega-regular finite automata

- LTL formula $\phi$ describes set of infinite words $[[\phi]]$,
- $[[\phi]]$ is omega-regular
- We aim for a product construction. Nondeterministic automata are tricky…
- There exists a **deterministic Rabin automaton** (DRA) that accepts $[[\phi]]$. 
Deterministic Rabin Automata

- A **deterministic Rabin automaton** (DRA) is a finite automaton with acceptance sets:
  \[ \mathcal{F} = \{(F_1, K_1), \ldots, (F_n, K_n)\} \]
- A run is accepting iff there exists an index \( i \) such that:
  States in \( F_i \) are visited only finitely often **and** some state in \( K_i \) is visited infinitely often.

This automaton accepts ‘eventually globally \( a \)’:

\[ \mathcal{F} = \{(F_0, K_0)\}, \quad F_0 = \{q_1\} \quad K_0 = \{q_2\} \]

There is no deterministic Büchi Automaton that accepts this language.
LTL model checking

Product automaton and repeated reachability

Model checking Omega-regular properties

For a finite DTMC $\mathcal{D}$ with state $s$ and a DRA $\mathcal{A}$:

$$\Pr_{\mathcal{D}}(s \models \mathcal{A}) = \Pr_{\mathcal{D} \otimes \mathcal{A}}(\langle s, q \rangle \models \Diamond U)$$

Where $U$ is the union of accepting terminal SCCs in $\mathcal{D} \otimes \mathcal{A}$

A terminal SCC is accepting iff

for some $i$ it contains no $L_i$ and some $K_i$ state

LTL Model checking: Build a DRA, take the product, find the accepting terminal SCCs by means of graph algorithm, solve reachability with a linear equation system
Beyond Reachability and LTL

Further properties

- Expected rewards:
  “What is the expected energy consumption?”

- Long-run average:
  “What are the expected costs of operation, in the long run”

- Cost-bounded reachability:
  “What is the probability that we arrive without an empty battery”

- Conditional reachability:
  “What is the probability that we reach the airport, when we also visit the train station”

- PCTL
## Markov Models

### Overview

<table>
<thead>
<tr>
<th>Nondeterminism</th>
<th>Discrete Time</th>
<th>Continuous Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Nondeterminism</td>
<td>DTMC</td>
<td>CTMC</td>
</tr>
<tr>
<td>Nondeterminism</td>
<td>MDP</td>
<td>IMC/CTMDP</td>
</tr>
</tbody>
</table>
Markov decision processes

Markov chains with nondeterminism

- **action choices, interleaving due to concurrency**

**MDP**

<table>
<thead>
<tr>
<th>States</th>
<th>$S$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial distribution</td>
<td>$\text{Distr}(S)$</td>
</tr>
<tr>
<td>Actions</td>
<td>Act</td>
</tr>
<tr>
<td>Transitions</td>
<td>$P: \ S \times \text{Act} \rightarrow \text{Distr}(S)$</td>
</tr>
</tbody>
</table>
Policies

Or Schedulers, Strategies, Adversaries

Resolve the nondeterminism:

Map histories on distributions over actions: $\text{States}^* \rightarrow \text{Distr(Actions)}$

- Deterministic $\text{States}^* \rightarrow \text{Actions}$
- Positional (or stationary or Markov or memoryless) $\text{States} \rightarrow \text{Distr(Actions)}$
Induced MC

Applying a policy to an MDP

Policy: alternate Act a and Act b

States are paths in the MDP: Generally countably infinite MC

- \( \Pr_\mathcal{M}^\sigma (s \models \Diamond G) = \Pr_{\mathcal{M}[\sigma]} (s \models \Diamond G) \)
Positional policies suffice for reachability

Essential simplification

Max Reachability in MDPs

For any finite MDP and with target set $G$:
There exists a positional policy $\sigma$ s.t. for any state $s$:

$$Pr^\sigma_M(s \models \diamond G) = \sup_{\sigma' \in \Sigma} Pr^{\sigma'}_M(s \models \diamond G)$$

• Thus, we can talk about the maximum reachability

Adaption for min reachability exists
Reachability in MDPs

Bellman Equations

Max Reachability in MDPs

Let $x(s)$ denote the maximal probability to reach some target state from $s$. It holds that:

- If $s$ is a target state: 
  $$x(s) = 1$$

- If there is no path from $s$ to some target state: 
  $$x(s) = 0$$

- Otherwise:
  $$x(s) = \max_{a \in \mathcal{A}} \sum_{s' \in S} P(s, a, s') \cdot x(s')$$

- Notice that these equations together have a unique solution

Adaption for min reachability exists
Bellman equations

Example

\[ x_v = 1 \]
\[ x_u = 0 \]
\[ x_s = \max \{ 0.5 \cdot x_s + 0.3 \cdot x_u + 0.2 \cdot x_t, 0.5 \cdot x_t + 0.5 \cdot x_u \} \]
\[ x_t = \max \{ 1 \cdot x_v, 0.5 \cdot x_v + 0.5 \cdot x_u \} \]
Reachability in MDPs

Three solution methods

- Linear Program (LP) (next slide)

- Value iteration (VI) (guess a solution to the Bellman equations, apply Bellman equations, repeat)

- Policy iteration (PI) (guess a positional policy, solve MC, change policy where improvements are possible)

- Linear Program is the only polynomial time method. VI and PI are fastest in practice.
Reachability in MDPs

Formulation as a linear program

\[
\begin{align*}
\min x_s \\
x_v &= 1 \\
x_u &= 0 \\
x_s &\geq 0.5 \cdot x_s + 0.3 \cdot x_u + 0.2 \cdot x_t \\
x_s &\geq 0.5 \cdot x_t + 0.5 \cdot x_u \\
x_t &\geq 1 \cdot x_v \\
x_t &= 0.5 \cdot x_v + 0.5 \cdot x_u
\end{align*}
\]
LTL

With MDPs

- As with Markov chains:
- Long run theorem (requires an adaption of SCCs)
- Construct automaton and cross product
- Optimal policy depends on state in the product
# Markov Models

## Overview

<table>
<thead>
<tr>
<th>Nondeterminism</th>
<th>Discrete Time</th>
<th>Continuous Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>No Nondeterminism</td>
<td>DTMC</td>
<td>CTMC</td>
</tr>
<tr>
<td>Nondeterminism</td>
<td>MDP</td>
<td>IMC/CTMDP</td>
</tr>
</tbody>
</table>
### Exponential distributions

Some facts

**Density of exponential distribution**

The density of an *exponentially distributed* r.v. $Y$ with *rate* $\lambda \in \mathbb{R}_{>0}$ is:

$$f_Y(x) = \lambda e^{-\lambda x} \quad \text{for } x > 0 \quad \text{and } f_Y(x) = 0 \text{ otherwise}$$

The cumulative distribution of r.v. $Y$ with rate $\lambda \in \mathbb{R}_{>0}$ is:

$$F_Y(d) = \int_0^d \lambda e^{-\lambda x} \, dx = [-e^{-\lambda x}]_0^d = 1 - e^{-\lambda d}.$$ 

The rate $\lambda \in \mathbb{R}_{>0}$ uniquely determines an exponential distribution.
Continuous-time Markov Chains

Two equivalent views

DTMC + exit-rate function $r(s)$

DTMC with transition rate matrix $R(s,s') = P(s,s')r(s)$ instead of transition probabilities $P(s,s')$
CTMC Semantics

Essential probabilities

- Probability to leave state $s$ within $t$ time:

\[\int_0^t r(s) \cdot e^{-r(s) \cdot x} dx = 1 - e^{-r(s) \cdot t}\]

- Probability to move from $s$ to $s'$ between now and time $t$:

\[
\frac{R(s,s')}{r(s)} \cdot \left(1 - e^{-r(s) \cdot t}\right)
\]
Timed reachability

What is the probability to reach a state within $T$ time:

$x_u(\tau) = 0 \quad x_v(\tau) = 1$

$x_s(\tau) = \int_0^{\tau} \frac{75}{4} \cdot e^{-25\cdot x} \cdot x_s(\tau - x)dx + \int_0^{\tau} \frac{25}{4} \cdot e^{-25\cdot x} \cdot x_v(\tau - x)dx$

$x_t(\tau) = \int_0^{\tau} \frac{4}{2} \cdot e^{-4\cdot x} \cdot x_u(\tau - x)dx + \int_0^{\tau} \frac{4}{2} \cdot e^{-4\cdot x} \cdot x_v(\tau - x)dx$
Reachability properties

Two types

- What is the probability of eventually reaching some set of states?
- What is the expected time to reach some set of states?
- What is the expected fraction of time in some set of states?

**Solution:** Calculate on the embedded DTMC

- What is the probability of eventually reaching some set of states within $T$ time.

**System of ODE equations — Solve via a technique called uniformization**

1) Baier et al., Model Checking Algorithms for continuous-Time Markov Chains, TSE 2003
# Markov Models

## Overview

<table>
<thead>
<tr>
<th>Nondeterminism</th>
<th>Discrete Time</th>
<th>Continuous Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>DTMC</td>
<td>CTMC</td>
</tr>
<tr>
<td>Nondeterminism</td>
<td>MDP</td>
<td>IMC/CTMDP</td>
</tr>
</tbody>
</table>
Plan for today

3 Parts. Let’s see how far we get.

Why Probabilistic Model Checking

Basics of Probabilistic Model Checking

Current Topics in Probabilistic Model Checking
Beyond Markov Chains

A Zoo of Models

Markov Automaton

CTMDP

IMC

LTS

CTMC

Kripke Structure

DTMC

Stochastic games

POPTA

POMDP

POSG

Hidden Markov Models

Timed automata (TA)
## Tools for probabilistic model checking

Various modern and mature (but academic) tools

- Various Domain Specific Languages as Input
- Common Language: JANI (easy for machines, hard for humans)

<table>
<thead>
<tr>
<th>Prism:</th>
<th>Storm:</th>
<th>Modest:</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ GUI</td>
<td>+ Performance</td>
<td>+ Extensive language</td>
</tr>
<tr>
<td>+ JAVA binary for major platforms</td>
<td>+ Docker container</td>
<td>+ Discrete event simulation</td>
</tr>
<tr>
<td>+ Extension to games</td>
<td>+ Python API</td>
<td>+ Combination of hybrid and stochastic</td>
</tr>
</tbody>
</table>

- QComp: Competition for most prominent model checking tasks

![Graph showing comparison of tools](image)
Probabilistic Model Checking vs Model Checking

(almost) all ideas from this lecture have been applied in the context of probabilistic model checking

- Bisimulations, Simulations, Partial Order Reduction, CEGAR, CEGIS, …

Counterexamples are more complex objects (sets of paths)

Some paths just do not matter that much…..

Counterexample to red state is reached with high probability contains all paths to red state

Not relevant to show that green states are reached with a probability less than 0.5
Multi-objective Model Checking

Pareto front

Recall:

- Optimal policies use memory and randomisation
- Performant implementations use reachability analyses in a loop.
Parameter Synthesis

‘Symbolic probabilities’

- Probabilities unknown, use some symbolic values instead
- For what values does the Markov chain satisfy some property?
Some more current research topics

A very long list….

- Variations to interval iteration: sound value iteration, optimistic value iteration, …
- Cost-bounded model checking, risk-bounded model checking
- Extensions to stochastic games, equilibria, …
- Extensions to partial observation models
- Connections to exact inference, Bayesian networks, …
- Connections to model counting
- Connections to reinforcement learning techniques
Want to know more?

Contact me at sjunges@berkeley.edu