Performance and Reliability Analysis by Model Checking

— From Verification to Synthesis —

Joost-Pieter Katoen

RWTH AACHEN UNIVERSITY

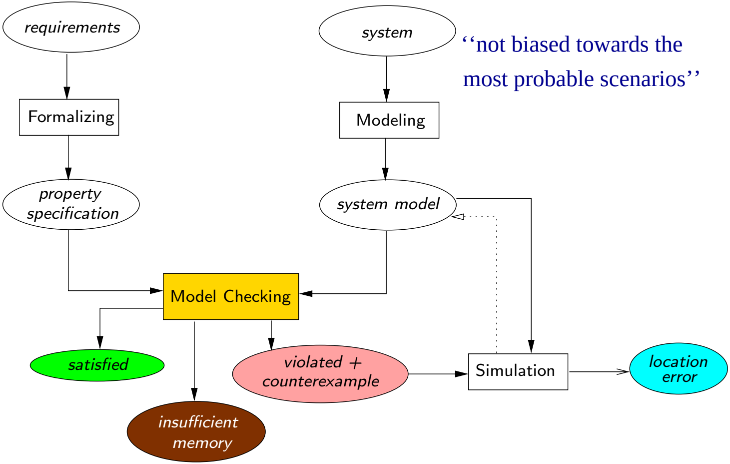
UNIVERSITY OF TWENTE.

7th oCPS Ph.D. School on Cyber-Physical Systems

Overview

Overview

Model Checking in a Nutshell



''not biased towards the most probable scenarios''

Gödel Prize 2000

Moshe Vardi

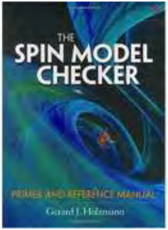
Pierre Wolper

“For work on model checking with finite automata.“

Some other winners: Shor, Sénizergues, Agrawal et al., Spielman and Teng, . . .

ACM System Software Award 2001

Gerard J. Holzmann



SPIN book

“SPIN is a popular open-source software tool, used by thousands of people worldwide, that can be used for the formal verification of distributed software systems.”

Some other winners: TeX, Postscript, UNIX, TCP/IP, Java, Smalltalk, . . .

ACM Turing Award 2007

Edmund Clarke

E. Allen Emerson

Joseph Sifakis

"For their role in developing model checking into a highly effective verification technology, widely adopted in the hardware and software industries."

Some other winners: Wirth, Dijkstra, Cook, Hoare, Rabin and Scott, . . .

Striking Examples

- Needham-Schroeder protocol
- IEEE cache coherence protocol
- Hardware property languages like PSL
- `C`, `.NET` code verification
- NASA space mission software
- Storm surge barrier Maeslantkering

- ....

Storm Surge Barrier Maeslantkering

Storm Surge Barrier Maeslantkering

Storm Surge Barrier Maeslantkering

What is This Lecture About?

"Probabilistic model checking is one of the main challenges for the future."

Edmund J. Clarke

The Birth of Model Checking, 2008

What is This Lecture About?

"A promising new direction in formal methods research these days is the development of probabilistic models, with associated tools for quantitative evaluation of system performance along with correctness."

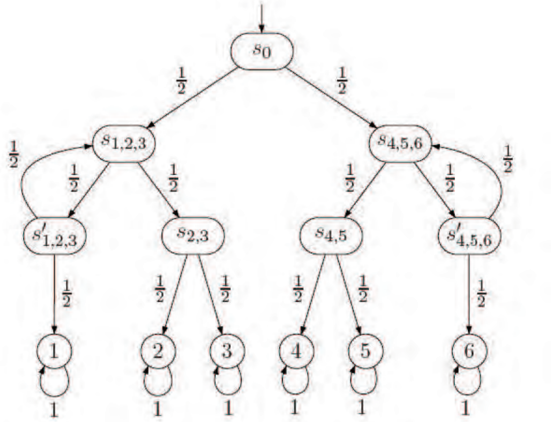
ACM SIGLOG News 2015

Overview

Almost Ten Reasons for Probabilities

1. Randomised Algorithms
2. Reducing Complexity
3. Avoiding the Impossible
4. Probabilistic Programs
5. Reliability
6. Performance
7. Robotics
8. Optimisation
9. Systems Biology

Randomised Algorithms: Simulating a Die [Knuth & Yao, 1976]



Heads = "go left"; tails = "go right". Does this model a six-sided die?

Avoiding the Impossible

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.

Avoiding the Impossible

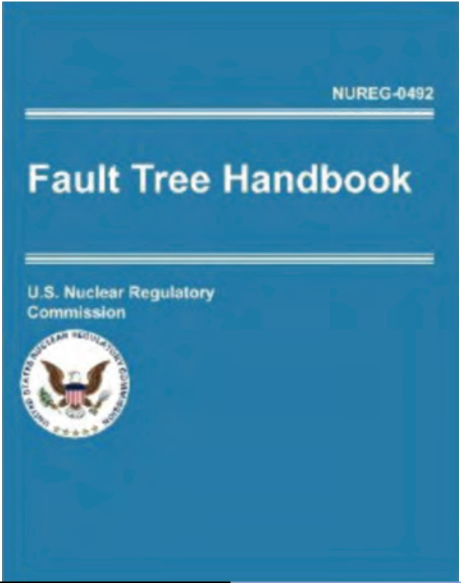
FLP impossibility result [Fischer *et al.*, 1985]

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.

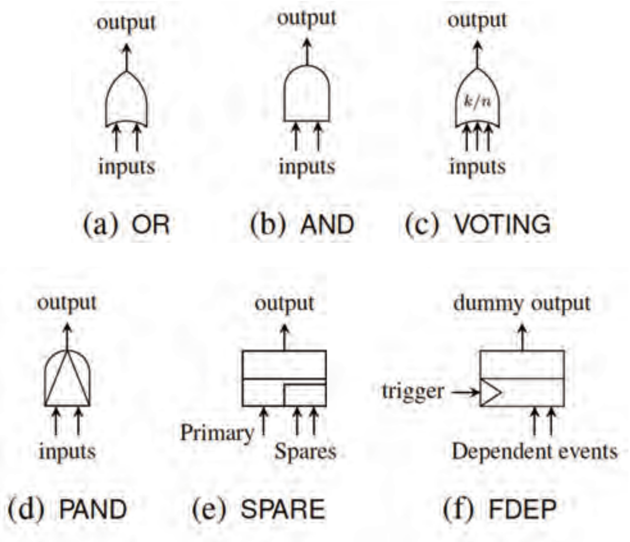
Ben-Or's possibility result [Ben-Or, 1983]

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.

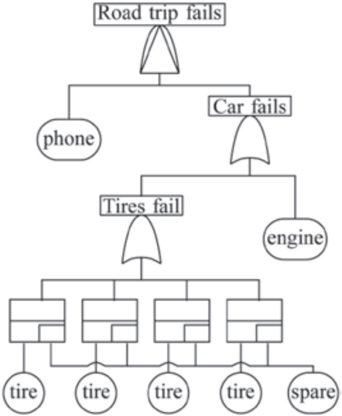
Reliability Engineering



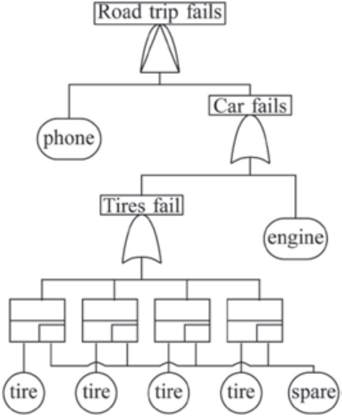
Reliability: Dynamic Fault Trees [Dugan *et al.*, 1990]



A Fault Tree Example



A Fault Tree Example

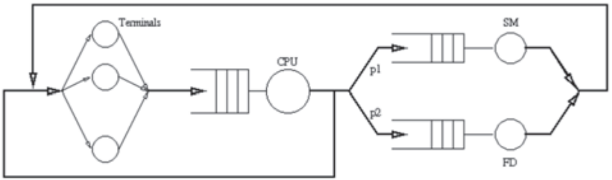


(D)FTs: one of —if not the— most prominent models for risk analysis

Aims: quantify system reliability and availability, MTTF, ...

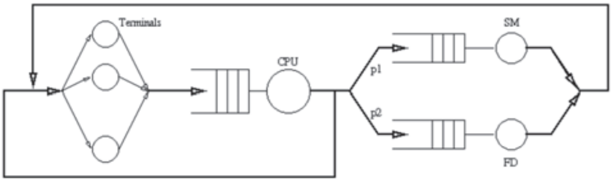
Performance: GSPNs [Ajmone Marsan et al, 1984]

The early days:



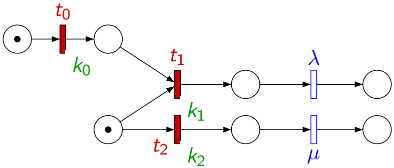
Performance: GSPNs [Ajmone Marsan et al, 1984]

The early days:



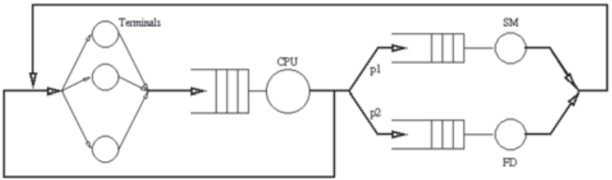
More modern times: Petri nets with

- Timed transitions
- Immediate transitions
- Natural weights



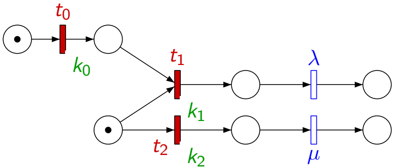
Performance: GSPNs [Ajmone Marsan et al, 1984]

The early days:



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

Encyclopedia of Optimisation 2008

Stochastic Scheduling

José Niño-Mora
Department of Statistics,
Universidad Carlos III de Madrid, Getafe, Spain

MSC2000: 90B36

Article Outline

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 stoc
a goal in the idealized
els. Real-world rand
rival or processing ti
ing their probability
vary across several d
scheduling policies cc
terarrival and proces
arrangement of servic
jective to be optimiz
are required to be nc
cannot make use of fu
known total duration
yet finished.
Regarding solutic
seems fair to say that
is yet available to des
optimal policies acro
tic scheduling model
can be cast in the fr
ming straightforward

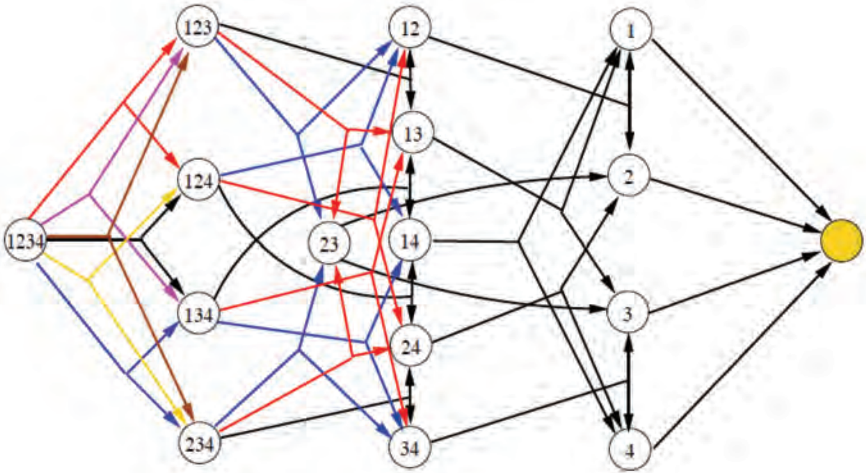
Stochastic Scheduling [Bruno et al., 1981]

- 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, i.e., finishing time of last job
- SEPT policy yields minimal expected makespan

  "it is hard to calculate these expected values"

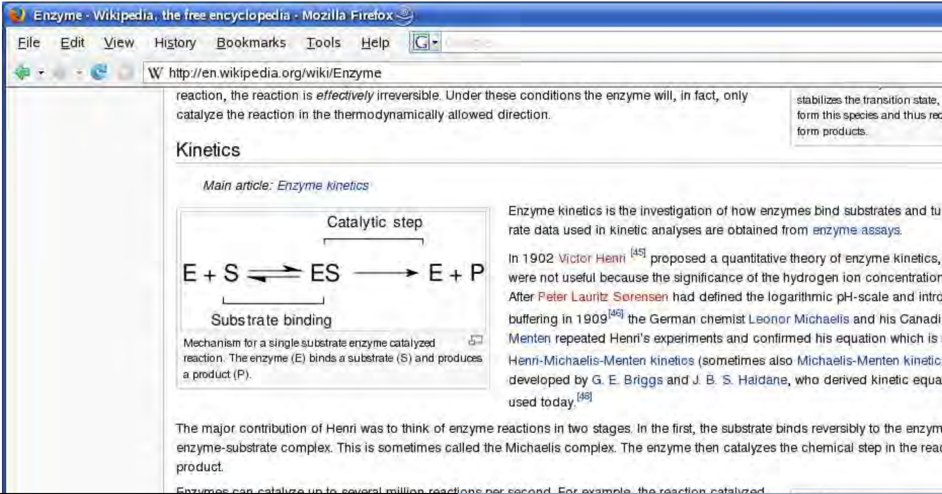
Which policy maximises the probability to finish all jobs on time?

Stochastic Model



Nature Behaviour: Systems Biology

Enzyme-catalysed substrate conversion



Enzyme - Wikipedia, the free encyclopedia - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://en.wikipedia.org/wiki/Enzyme

reaction, the reaction is *effectively* irreversible. Under these conditions the enzyme will, in fact, only catalyze the reaction in the thermodynamically allowed direction.

stabilizes the transition state, form this species and thus red form products.

Kinetics

Main article: Enzyme kinetics

Mechanism for a single substrate enzyme catalyzed reaction. The enzyme (E) binds a substrate (S) and produces a product (P).

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

In 1902 Victor Henri [45] proposed a quantitative theory of enzyme kinetics, were not useful because the significance of the hydrogen ion concentratior After Peter Lauritz Sørensen had defined the logarithmic pH-scale and intro buffering in 1909[46] the German chemist Leonor Michaelis and his Canadi Menten repeated Henri's experiments and confirmed his equation which is Henri-Michaelis-Menten kinetics (sometimes also Michaelis-Menten kinetic developed by G. E. Briggs and J. B. S. Haldane, who derived kinetic equa used today.[48]

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

Stochastic Chemical Kinetics

- Types of reaction described by stochiometric equations:

  $$E + S \underset{k_2}{\overset{k_1}{\rightleftharpoons}} C \overset{k_3}{\rightarrow} E + P$$

- 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:

$$E + S \underset{k_2}{\overset{k_1}{\rightleftharpoons}} C \overset{k_3}{\rightarrow} E + P$$

- 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:

$$E + S \underset{k_2}{\overset{k_1}{\rightleftharpoons}} C \overset{k_3}{\rightarrow} E + P$$

- N different types of molecules that randomly collide

  where state $X(t) = (x_1, \ldots, x_N)$ with $x_i = \#$ molecules of sort i

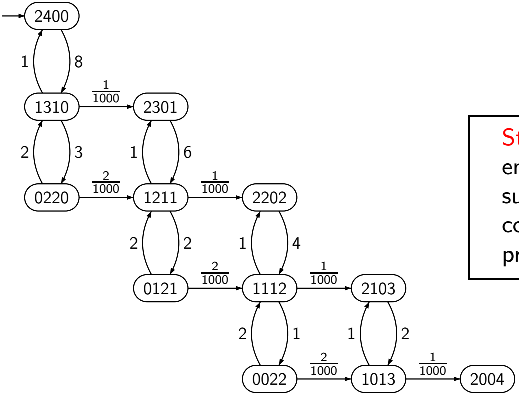
- Reaction probability within infinitesimal interval $[t, t+\Delta)$:

$$\alpha_m(\vec{x}) \cdot \Delta \;=\; \Pr\{\text{reaction } m \text{ in } [t, t+\Delta) \mid X(t) = \vec{x}\}$$

  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*	*goal*
enzymes	2	2
substrates	4	0
complex	0	0
products	0	4

Transitions: $E + S \underset{1}{\overset{1}{\rightleftharpoons}} C \overset{0.001}{\rightarrow} E + P$

e.g., $(x_E, x_S, x_C, x_P) \overset{0.001 \cdot x_C}{\rightarrow} (x_E + 1, x_S, x_C - 1, x_P + 1)$ for $x_C > 0$

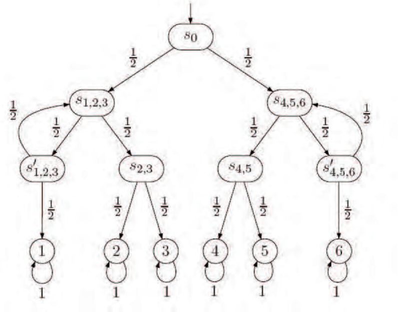
Overview

Common Feature

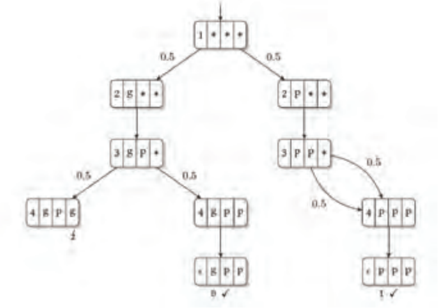
All these applications consider Markov models[1]

[1] Non-exponential distributions are approximated by phase-type distributions.

Discrete-Time Markov Models

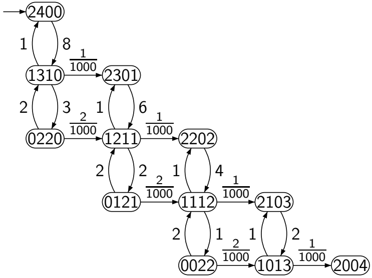


A Markov chain
for Knuth-Yao's algorithm

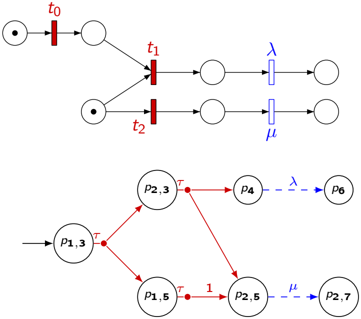


A Markov decision process
for a probabilistic program

Continuous-Time Markov Models

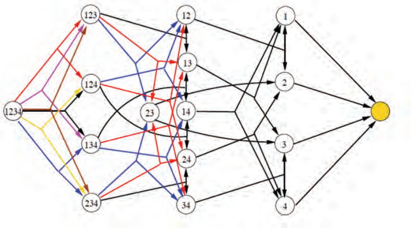


A Markov chain
for substrate conversion

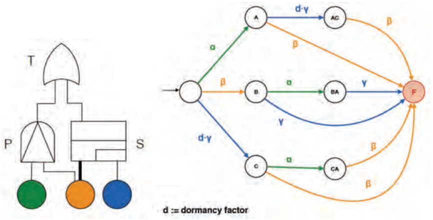


A Markov decision process
for the GSPN

Continuous-Time Markov Models



Markov decision process
for stochastic scheduling



Markov decision process for a DFT

Markov Models

	Discrete	Continuous
Deterministic	discrete-time Markov chain (DTMC)	continuous-time MC
Nondeterministic	Markov decision process (MDP)	CTMDP
Compositional	Segala's probabilistic automata (PA)	Markov automata (MA)

Other models: e.g., probabilistic timed automata, pVASS, pPDA, SGs, etc.

Overview

Properties

	Discrete	Continuous
Logic	probabilistic CTL	probabilistic timed CTL
Monitors	deterministic automata (safety and LTL)	deterministic timed automata (MITL fragments)

Others: e.g., conditional probs, multi-objective, rewards, quantiles, etc.

Core problem: computing (timed) reachability probabilities

Reachability Probabilities

Problem

Consider a finite MC with $s \in S$ and $G \subseteq S$.

Aim: determine $\Pr(s \vDash \Diamond G) = \Pr_s\{ \pi \in Paths(s) \mid \pi \vDash \Diamond G \}$

Reachability Probabilities

Problem

Consider a finite MC with $s \in S$ and $G \subseteq S$.

Aim: determine $\Pr(s \models \Diamond G) = \Pr_s\{ \pi \in Paths(s) \mid \pi \models \Diamond G \}$

Characterisation of reachability probabilities

- Let variable $x_s = \Pr(s \models \Diamond G)$ for any state s

Reachability Probabilities

Problem

Consider a finite MC with $s \in S$ and $G \subseteq S$.

Aim: determine $\Pr(s \models \Diamond G) = \Pr_s\{ \pi \in Paths(s) \mid \pi \models \Diamond G \}$

Characterisation of reachability probabilities

- Let variable $x_s = \Pr(s \models \Diamond G)$ for any state s
  - if G is not reachable from s, then $x_s = 0$

Reachability Probabilities

Problem

Consider a finite MC with $s \in S$ and $G \subseteq S$.

Aim: determine $\Pr(s \models \Diamond G) = \Pr_s\{\, \pi \in Paths(s) \mid \pi \models \Diamond G \,\}$

Characterisation of reachability probabilities

- Let variable $x_s = \Pr(s \models \Diamond G)$ for any state s
  - if G is not reachable from s, then $x_s = 0$
  - if $s \in G$ then $x_s = 1$

Reachability Probabilities

Problem

Consider a finite MC with $s \in S$ and $G \subseteq S$.

Aim: determine $\Pr(s \models \Diamond G) = \Pr_s\{ \pi \in \mathit{Paths}(s) \mid \pi \models \Diamond G \}$

Characterisation of reachability probabilities

- Let variable $x_s = \Pr(s \models \Diamond G)$ for any state s
  - if G is not reachable from s, then $x_s = 0$
  - if $s \in G$ then $x_s = 1$
- For any state $s \in \mathit{Pre}^*(G) \setminus G$:

Reachability Probabilities

Problem

Consider a finite MC with $s \in S$ and $G \subseteq S$.

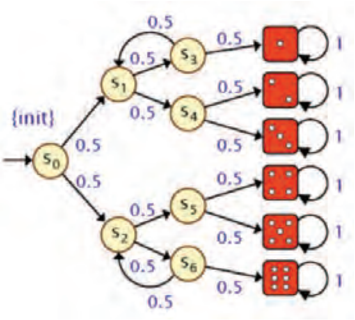
Aim: determine $\Pr(s \models \Diamond G) = \Pr_s\{ \pi \in Paths(s) \mid \pi \models \Diamond G \}$

Characterisation of reachability probabilities

- Let variable $x_s = \Pr(s \models \Diamond G)$ for any state s
  - if G is not reachable from s, then $x_s = 0$
  - if $s \in G$ then $x_s = 1$
- For any state $s \in Pre^*(G) \setminus G$:

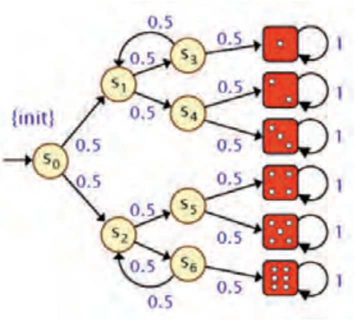
$$x_s = \underbrace{\sum_{t \in S \setminus G} \mathbf{P}(s,t) \cdot x_t}_{\text{reach } G \text{ via } t \in S \setminus G} + \underbrace{\sum_{u \in G} \mathbf{P}(s,u)}_{\text{reach } G \text{ in one step}}$$

Reachability Probabilities: Knuth-Yao's Die



- Consider the event $\Diamond 4$

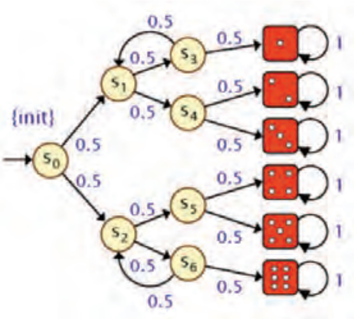
Reachability Probabilities: Knuth-Yao's Die



- Consider the event $\Diamond 4$
- We obtain:

  $x_1 = x_2 = x_3 = x_5 = x_6 = 0$ and $x_4 = 1$

Reachability Probabilities: Knuth-Yao's Die

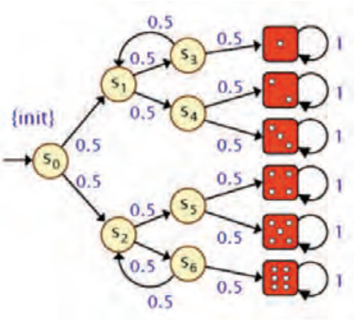


- Consider the event $\Diamond 4$
- We obtain:

$$x_1 = x_2 = x_3 = x_5 = x_6 = 0 \text{ and } x_4 = 1$$

$$x_{s_1} = x_{s_3} = x_{s_4} = 0$$

Reachability Probabilities: Knuth-Yao's Die



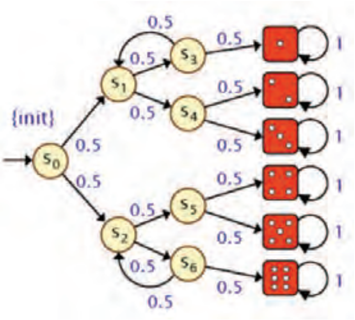
- Consider the event $\Diamond 4$
- We obtain:

$$x_1 = x_2 = x_3 = x_5 = x_6 = 0 \text{ and } x_4 = 1$$

$$x_{s_1} = x_{s_3} = x_{s_4} = 0$$

$$x_{s_0} = \frac{1}{2}x_{s_1} + \frac{1}{2}x_{s_2}$$

Reachability Probabilities: Knuth-Yao's Die



- Consider the event $\Diamond 4$
- We obtain:

$$x_1 = x_2 = x_3 = x_5 = x_6 = 0 \text{ and } x_4 = 1$$

$$x_{s_1} = x_{s_3} = x_{s_4} = 0$$

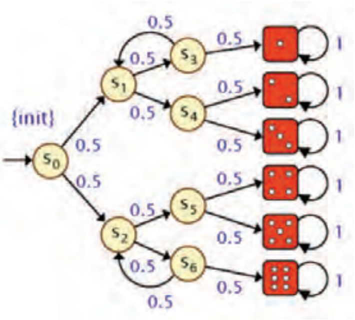
$$x_{s_0} = \frac{1}{2}x_{s_1} + \frac{1}{2}x_{s_2}$$

$$x_{s_2} = \frac{1}{2}x_{s_5} + \frac{1}{2}x_{s_6}$$

$$x_{s_5} = \frac{1}{2}x_5 + \frac{1}{2}x_4$$

$$x_{s_6} = \frac{1}{2}x_{s_2} + \frac{1}{2}x_6$$

Reachability Probabilities: Knuth-Yao's Die



- Consider the event $\Diamond 4$
- We obtain:

$$x_1 = x_2 = x_3 = x_5 = x_6 = 0 \text{ and } x_4 = 1$$

$$x_{s_1} = x_{s_3} = x_{s_4} = 0$$

$$x_{s_0} = \frac{1}{2}x_{s_1} + \frac{1}{2}x_{s_2}$$

$$x_{s_2} = \frac{1}{2}x_{s_5} + \frac{1}{2}x_{s_6}$$

$$x_{s_5} = \frac{1}{2}x_5 + \frac{1}{2}x_4$$

$$x_{s_6} = \frac{1}{2}x_{s_2} + \frac{1}{2}x_6$$

- Gaussian elimination yields:

$$x_{s_5} = \frac{1}{2},\ x_{s_2} = \frac{1}{3},\ x_{s_6} = \frac{1}{6},\ \text{and } \boxed{x_{s_0} = \frac{1}{6}}$$

Reachability Probabilities are Pivotal

- Repeated reachability $\Pr(s \models \Box\Diamond G)$:

  Determine probability to reach a terminal SCCs containing a G-state

Reachability Probabilities are Pivotal

- Repeated reachability $\Pr(s \models \Box \Diamond G)$:

  Determine probability to reach a terminal SCCs containing a G-state

- Probabilistic CTL model checking

  Recursive descent on parse tree using reach-probabilities at nodes

Reachability Probabilities are Pivotal

- Repeated reachability $\Pr(s \models \Box\Diamond G)$:

  Determine probability to reach a terminal SCCs containing a G-state

- Probabilistic CTL model checking

  Recursive descent on parse tree using reach-probabilities at nodes

- LTL formulas $\Pr(s \models \varphi)$:
  1. Transform φ into a deterministic (Rabin) automaton
  2. Take the product of the Markov chain and the automaton
  3. Determine the probability to reach an accepting terminal SCC from s

Reachability Probabilities are Pivotal

- Repeated reachability $\Pr(s \models \Box\Diamond G)$:

  Determine probability to reach a terminal SCCs containing a G-state

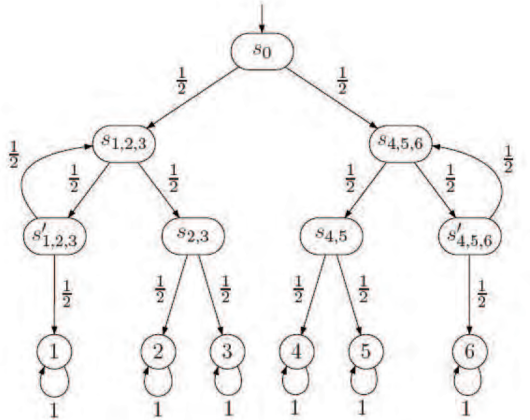
- Probabilistic CTL model checking

  Recursive descent on parse tree using reach-probabilities at nodes

- LTL formulas $\Pr(s \models \varphi)$:
  1. Transform φ into a deterministic (Rabin) automaton
  2. Take the product of the Markov chain and the automaton
  3. Determine the probability to reach an accepting terminal SCC from s

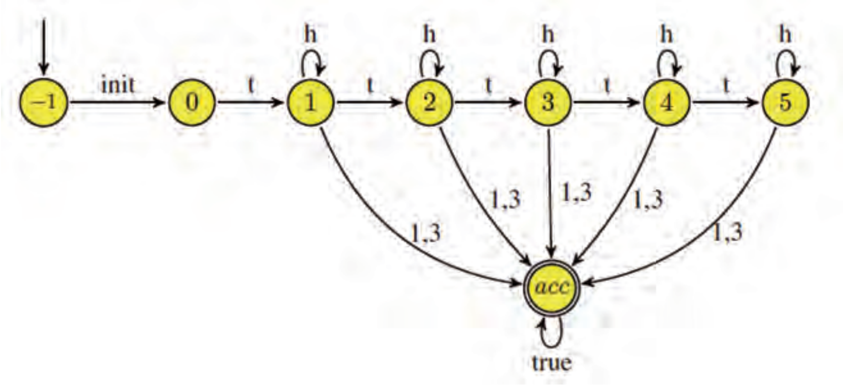
For MDPs, solving linear inequality systems are key.

Randomised Algorithms: Simulating a Die [Knuth & Yao, 1976]



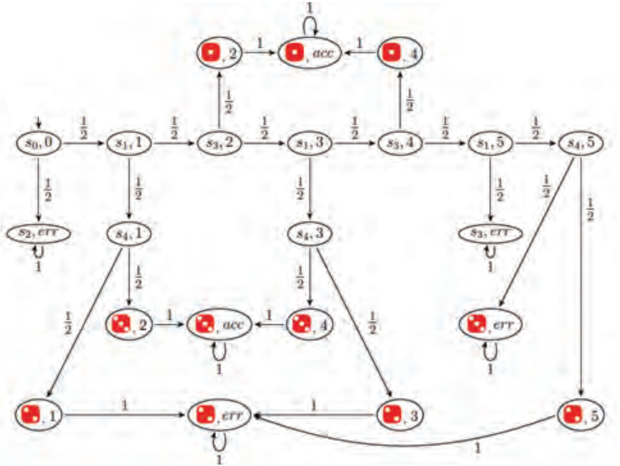
Probability of after initial tails, yield 1 or 3 but with at most five tails in total?

Property of Knuth-Yao's Algorithm



After initial tails, yield 1 or 3 but with at most five times tails in total

Product Markov Chain



Reachability probability of terminal SCC with $(\cdot, q_{acc})$ is $\frac{1}{8} + \frac{1}{8} + \frac{1}{32} + \frac{1}{32} = \frac{5}{16}$.

Probabilistic CTL [Hansson & Jonsson, 1989]

Probabilistic CTL [Hansson & Jonsson, 1989]

- PCTL interpretation is Boolean, i.e., a formula is satisfied or not.

Probabilistic CTL

[Hansson & Jonsson, 1989]

- PCTL interpretation is Boolean, i.e., a formula is satisfied or not.
- For path-formula φ and threshold $\succ p$ with $\succ \in \{ >, \geqslant \}$ and $p \in \mathbb{Q}$:

$$\text{PCTL-formula} \quad [\varphi]_{\succ p} \quad \text{denotes}$$

  all paths satisfying φ occur with probability $\succ p$

- $[\cdot]_{\succ p}$ is probabilistic counterpart of CTL path-quantifiers $\exists$ and $\forall$.

Probabilistic CTL

[Hansson & Jonsson, 1989]

- PCTL interpretation is Boolean, i.e., a formula is satisfied or not.
- For path-formula φ and threshold $\succ p$ with $\succ \in \{ >, \geqslant \}$ and $p \in \mathbb{Q}$:

  PCTL-formula $[\varphi]_{\succ p}$ denotes

  all paths satisfying φ occur with probability $\succ p$
- $[\cdot]_{\succ p}$ is probabilistic counterpart of CTL path-quantifiers $\exists$ and $\forall$.
- Examples: $[\Diamond a]_{>1/2}$, $[\Diamond [\Box a]_{=1}]_{>1/2}$ and $[\Box (\neg a \wedge [\Diamond a]_{>0})]_{>0}$.

Probabilistic CTL

[Hansson & Jonsson, 1989]

- PCTL interpretation is Boolean, i.e., a formula is satisfied or not.
- For path-formula φ and threshold $\unrhd p$ with $\unrhd \in \{ >, \geqslant \}$ and $p \in \mathbb{Q}$:

  PCTL-formula $\quad [\varphi]_{\unrhd p} \quad$ denotes

  all paths satisfying φ occur with probability $\unrhd p$
- $[\cdot]_{\unrhd p}$ is probabilistic counterpart of CTL path-quantifiers $\exists$ and $\forall$.
- Examples: $[\Diamond a]_{>1/2}$, $[\Diamond[\Box a]_{=1}]_{>1/2}$ and $[\Box(\neg a \wedge [\Diamond a]_{>0})]_{>0}$.

PCTL model checking is in P.

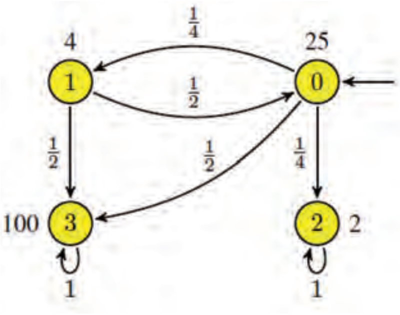
CTL versus Probabilistic CTL

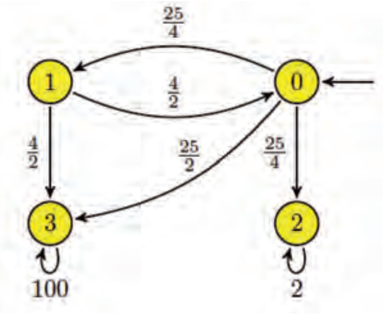
- Qualitative PCTL (only $>0-$ bounds) and CTL are incomparable.
  1. There is no CTL formula that is equivalent to $[\Diamond a]_{=1}$.
  2. There is no PCTL formula that is equivalent to $\forall \Diamond a$.
- These results all rely on countably infinite MCs
  1. For finite MCs, $[\Diamond a]_{=1} \equiv \forall \Diamond a$ under fairness.
  2. For finite MCs, $\Diamond\Box$-modalities are PCTL-definable, but not in CTL.

Random Timing

Continuous-Time Markov Chains

A CTMC is a DTMC with an *exit rate* function $r : S \to \mathbb{R}_{>0}$ where $r(s)$ is the rate of an exponential distribution.





Zenoness

Zeno theorem

In every CTMC, almost surely no Zeno runs occur.

In contrast to timed automata verification, Zeno runs thus pose no problem.

Timed Reachability Probabilities

Problem

Consider a finite CTMC with $s \in S$, $t \in \mathbb{R}_{\geqslant 0}$ and $G \subseteq S$.

Aim: determine $\Pr(s \models \Diamond^{\leqslant t} G)$.

Timed Reachability Probabilities

Problem

Consider a finite CTMC with $s \in S$, $t \in \mathbb{R}_{\geqslant 0}$ and $G \subseteq S$.

Aim: determine $\Pr(s \models \Diamond^{\leqslant t} G)$.

Characterisation of timed reachability probabilities

- Let function $x_s(t) = \Pr(s \models \Diamond^{\leqslant t} G)$ for any state s

Timed Reachability Probabilities

Problem

Consider a finite CTMC with $s \in S$, $t \in \mathbb{R}_{\geqslant 0}$ and $G \subseteq S$.

Aim: determine $\Pr(s \models \Diamond^{\leqslant t} G)$.

Characterisation of timed reachability probabilities

- Let function $x_s(t) = \Pr(s \models \Diamond^{\leqslant t} G)$ for any state s
  - if G is not reachable from s, then $x_s(t) = 0$ for all t

Timed Reachability Probabilities

Problem

Consider a finite CTMC with $s \in S$, $t \in \mathbb{R}_{\geqslant 0}$ and $G \subseteq S$.

Aim: determine $\Pr(s \models \Diamond^{\leqslant t} G)$.

Characterisation of timed reachability probabilities

- Let function $x_s(t) = \Pr(s \models \Diamond^{\leqslant t} G)$ for any state s
  - if G is not reachable from s, then $x_s(t) = 0$ for all t
  - if $s \in G$ then $x_s(t) = 1$ for all t

Timed Reachability Probabilities

Problem

Consider a finite CTMC with $s \in S$, $t \in \mathbb{R}_{\geqslant 0}$ and $G \subseteq S$.

Aim: determine $\Pr(s \models \Diamond^{\leqslant t} G)$.

Characterisation of timed reachability probabilities

- Let function $x_s(t) = \Pr(s \models \Diamond^{\leqslant t} G)$ for any state s
  - if G is not reachable from s, then $x_s(t) = 0$ for all t
  - if $s \in G$ then $x_s(t) = 1$ for all t
- For any state $s \in Pre^*(G) \setminus G$:

Timed Reachability Probabilities

Problem

Consider a finite CTMC with $s \in S$, $t \in \mathbb{R}_{\geqslant 0}$ and $G \subseteq S$.

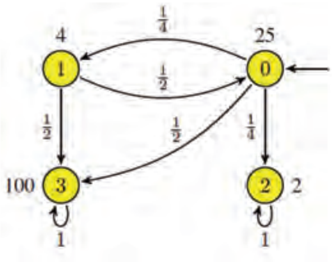
Aim: determine $\Pr(s \models \Diamond^{\leqslant t} G)$.

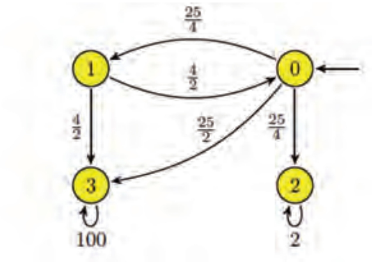
Characterisation of timed reachability probabilities

- Let function $x_s(t) = \Pr(s \models \Diamond^{\leqslant t} G)$ for any state s
  - if G is not reachable from s, then $x_s(t) = 0$ for all t
  - if $s \in G$ then $x_s(t) = 1$ for all t
- For any state $s \in Pre^*(G) \setminus G$:

$$x_s(t) = \int_0^t \sum_{s' \in S} \underbrace{\mathbf{R}(s,s') \cdot e^{-r(s) \cdot x}}_{\substack{\text{probability to move to} \\ \text{state } s' \text{ at time } x}} \cdot \underbrace{x_{s'}(t-x)}_{\substack{\text{prob. to fulfill} \\ \Diamond^{\leqslant t-x} G \text{ from } s'}} \, dx$$

Timed Reachability Probabilities





Integral equations for $\Diamond^{\leqslant 10}\, 2$:

- $x_3(d) = 0$ and $x_2(d) = 1$ for all d
- $x_0(d) = \displaystyle\int_0^d {}^{25}/_{4} \cdot e^{-25 \cdot x} \cdot x_1(d-x) \;+\; {}^{25}/_{4} \cdot e^{-25 \cdot x} \cdot x_2(d-x)\; dx$
- $x_1(d) = \displaystyle\int_0^d {}^{4}/_{2} \cdot e^{-4 \cdot x} \cdot x_0(d-x) \;+\; {}^{4}/_{2} \cdot e^{-4 \cdot x} \cdot x_3(d-x)\; dx$

Timed Reachability Probabilities

Reachability probabilities

Solve a system of linear equations for which many efficient techniques exist.

Timed Reachability Probabilities

Reachability probabilities

Solve a system of linear equations for which many efficient techniques exist.

Timed reachability probabilities

Solve a system of Volterra integral equations.
Non-trivial, inefficient, and has several pitfalls such as numerical stability.

Timed Reachability Probabilities

Reachability probabilities

Solve a system of linear equations for which many efficient techniques exist.

Timed reachability probabilities

Solve a system of Volterra integral equations.
Non-trivial, inefficient, and has several pitfalls such as numerical stability.

Solution

Reduce $\Pr(s \models \Diamond^{\leqslant t} G)$ to computing transient probabilities.

Timed Reachability Probabilities = Transient Probabilities

Aim

Compute $\Pr(s \models \Diamond^{\leqslant t} G)$ in CTMC $\mathcal{C}$. Observe that once a path π reaches G within t time, then the remaining behaviour along π is not important.
$\Rightarrow$ make all states in G absorbing.

Timed Reachability Probabilities = Transient Probabilities

Aim

Compute $\Pr(s \vDash \Diamond^{\leqslant t} G)$ in CTMC $\mathcal{C}$. Observe that once a path π reaches G within t time, then the remaining behaviour along π is not important.
$\Rightarrow$ make all states in G absorbing.

$$\underbrace{\Pr(s \vDash \Diamond^{\leqslant t} G)}_{\text{timed reachability in } \mathcal{C}} = \underbrace{\Pr(s \vDash \Diamond^{= t} G)}_{\text{timed reachability in } \mathcal{C}[G]} = \underbrace{\vec{p}(t) \text{ with } \vec{p}(0) = \mathbf{1}_s}_{\text{transient prob. in } \mathcal{C}[G]} .$$

Timed Reachability Probabilities = Transient Probabilities

Aim

Compute $\Pr(s \vDash \Diamond^{\leqslant t} G)$ in CTMC $\mathcal{C}$. Observe that once a path π reaches G within t time, then the remaining behaviour along π is not important.
$\Rightarrow$ make all states in G absorbing.

$$\underbrace{\Pr(s \vDash \Diamond^{\leqslant t} G)}_{\text{timed reachability in } \mathcal{C}} \quad = \quad \underbrace{\Pr(s \vDash \Diamond^{= t} G)}_{\text{timed reachability in } \mathcal{C}[G]} \quad = \quad \underbrace{\vec{p}(t) \text{ with } \vec{p}(0) = \mathbf{1}_s}_{\text{transient prob. in } \mathcal{C}[G]} .$$

Transient probabilities can be efficiently computed as solutions of linear differential equations.

Computing Transient Probabilities

By solving a linear differential equation system

The transient probability vector $\underline{p}(t) = (p_{s_1}(t), \ldots, p_{s_k}(t))$ satisfies:

$$\underline{p}'(t) \;=\; \underline{p}(t) \cdot (\mathbf{R} - \mathbf{r}) \qquad \text{given} \qquad \underline{p}(0)$$

where $\mathbf{r}$ is the diagonal matrix of vector $\underline{r}$.

[2] 19 dubious ways to compute a matrix exponential [Moler & Van Loan, 1978/2003].

Computing Transient Probabilities

By solving a linear differential equation system

The transient probability vector $\underline{p}(t) = (p_{s_1}(t), \ldots, p_{s_k}(t))$ satisfies:

$$\underline{p}'(t) \;=\; \underline{p}(t) \cdot (\mathbf{R} - \mathbf{r}) \quad \text{given} \quad \underline{p}(0)$$

where $\mathbf{r}$ is the diagonal matrix of vector $\underline{r}$.

Solution using standard knowledge yields: $\underline{p}(t) \;=\; \underline{p}(0) \cdot e^{(\mathbf{R}-\mathbf{r}) \cdot t}$.

[2] 19 dubious ways to compute a matrix exponential [Moler & Van Loan, 1978/2003].

Computing Transient Probabilities

By solving a linear differential equation system

The transient probability vector $\underline{p}(t) = (p_{s_1}(t), \ldots, p_{s_k}(t))$ satisfies:

$$\underline{p}'(t) \;=\; \underline{p}(t) \cdot (\mathbf{R} - \mathbf{r}) \quad \text{given} \quad \underline{p}(0)$$

where $\mathbf{r}$ is the diagonal matrix of vector $\underline{r}$.

Solution using standard knowledge yields: $\underline{p}(t) \;=\; \underline{p}(0) \cdot e^{(\mathbf{R}-\mathbf{r}) \cdot t}$.

Computing the matrix exponential is a challenging numerical problem[2].

[2] 19 dubious ways to compute a matrix exponential [Moler & Van Loan, 1978/2003].

Uniformisation

CTMC $\mathcal{C}$ is uniform if $r(s) = r$ for all $s \in S$ for some $r \in \mathbb{R}_{>0}$.

Uniformisation

CTMC $\mathcal{C}$ is uniform if $r(s) = r$ for all $s \in S$ for some $r \in \mathbb{R}_{>0}$.

Uniformisation [Jensen, 1953] [Gross and Miller, 1984]

Let $r \in \mathbb{R}_{>0}$ such that $r \geqslant max_{s \in S}\ r(s)$.

Uniformisation

CTMC $\mathcal{C}$ is uniform if $r(s) = r$ for all $s \in S$ for some $r \in \mathbb{R}_{>0}$.

Uniformisation [Jensen, 1953] [Gross and Miller, 1984]

Let $r \in \mathbb{R}_{>0}$ such that $r \geqslant max_{s \in S}\ r(s)$. Then $\overline{r}(\mathcal{C})$ is the CTMC $\mathcal{C}$ with two changes: $\overline{r}(s) = r$ for all $s \in S$

Uniformisation

CTMC $\mathcal{C}$ is uniform if $r(s) = r$ for all $s \in S$ for some $r \in \mathbb{R}_{>0}$.

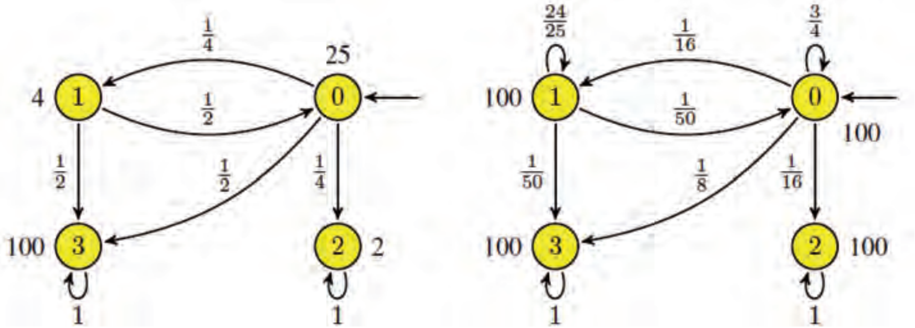
Uniformisation [Jensen, 1953] [Gross and Miller, 1984]

Let $r \in \mathbb{R}_{>0}$ such that $r \geqslant max_{s \in S}\ r(s)$. Then $\overline{r}(\mathcal{C})$ is the CTMC $\mathcal{C}$ with two changes: $\overline{r}(s) = r$ for all $s \in S$, and:

$$\overline{\mathbf{P}}(s,s') = \frac{r(s)}{r} \cdot \mathbf{P}(s,s') \text{ if } s' \neq s \quad \text{and} \quad \overline{\mathbf{P}}(s,s) = \frac{r(s)}{r} \cdot \mathbf{P}(s,s) + 1 - \frac{r(s)}{r}.$$

$\overline{\mathbf{P}}$ is a stochastic matrix and $\overline{r}(\mathcal{C})$ is uniform.

Uniformisation by Example



Uniformisation amounts to normalise the residence time in every CTMC state.

Benefits of Uniformisation

Transient probabilities of a CTMC and its uniformized CTMC coincide.

Thus: $\underbrace{\underline{p}(t) \;=\; \underline{p}(0) \cdot e^{(\mathbf{R}-\mathbf{r}) \cdot t}}_{\text{transient probablity in } \mathcal{C}} \;=$

Benefits of Uniformisation

Transient probabilities of a CTMC and its uniformized CTMC coincide.

Thus: $\underbrace{\underline{p}(t) \;=\; \underline{p}(0) \cdot e^{(\mathbf{R}-\mathbf{r}) \cdot t}}_{\text{transient probablity in } \mathcal{C}} \;=\; \underbrace{\underline{p}(0) \cdot e^{(\overline{\mathbf{R}}-\overline{\mathbf{r}}) \cdot t}}_{\text{transient probablity in } \overline{r}(\mathcal{C})} \;=$

Benefits of Uniformisation

Transient probabilities of a CTMC and its uniformized CTMC coincide.

Thus: $\underbrace{\underline{p}(t) \;=\; \underline{p}(0)\cdot e^{(\mathbf{R}-\mathbf{r})\cdot t}}_{\text{transient probablity in } \mathcal{C}} \;=\; \underbrace{\underline{p}(0)\cdot e^{(\overline{\mathbf{R}}-\overline{\mathbf{r}})\cdot t}}_{\text{transient probablity in } \overline{r}(\mathcal{C})} \;=\; \underline{p}(0)\cdot e^{-r\cdot t}\cdot e^{r\cdot t\cdot \overline{\mathbf{P}}}$

Still a matrix exponential remains. Did we gain anything?
Yes. Since $\overline{\mathbf{P}}$ is stochastic, Taylor-Maclaurin yields $\sum_i \ldots \overline{\mathbf{P}}^i$.

Other Properties on CTMCs

- Expected time objectives

  Can be characterised as solution of set of linear equations

[3] This yields a piecewise deterministic Markov process.

Other Properties on CTMCs

- Expected time objectives

  Can be characterised as solution of set of linear equations

- Long-run average objectives
  1. Determine the limiting distribution in any terminal SCC
  2. Take weighted sum with reachability probabilities terminal SCCs

[3] This yields a piecewise deterministic Markov process.

Other Properties on CTMCs

- Expected time objectives

  Can be characterised as solution of set of linear equations

- Long-run average objectives
  1. Determine the limiting distribution in any terminal SCC
  2. Take weighted sum with reachability probabilities terminal SCCs

- Probabilistic timed CTL model checking

  recursive descent over parse tree

[3] This yields a piecewise deterministic Markov process.

Other Properties on CTMCs

- Expected time objectives

  Can be characterised as solution of set of linear equations

- Long-run average objectives
  1. Determine the limiting distribution in any terminal SCC
  2. Take weighted sum with reachability probabilities terminal SCCs

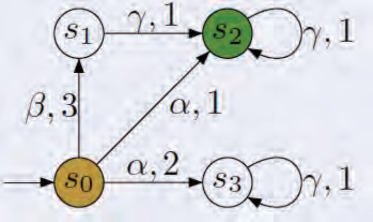
- Probabilistic timed CTL model checking

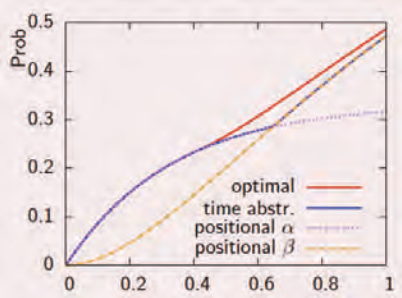
  recursive descent over parse tree

- Deterministic timed automata objectives
  1. Take product of the MC and the Zone automaton of the DTA[3]
  2. Determine the probability to reach an accepting zone

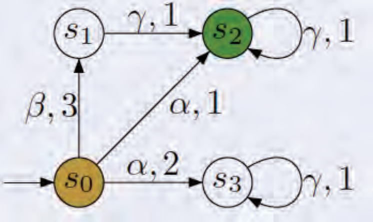
[3] This yields a piecewise deterministic Markov process.

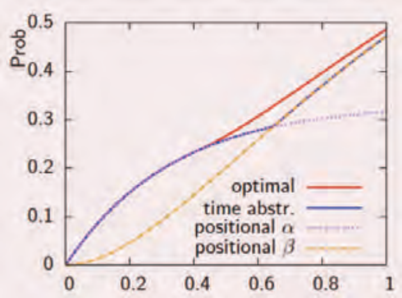
Timed Reachability in CTMDPs is Hard





Timed Reachability in CTMDPs is Hard





- Timed policies are optimal; any time-abstract policy is inferior.
- If long time remains: choose β; if short time remains: choose α.
- Optimal for deadline 1: choose α if $1-t_0 \leqslant \ln 3 - \ln 2$, otherwise β

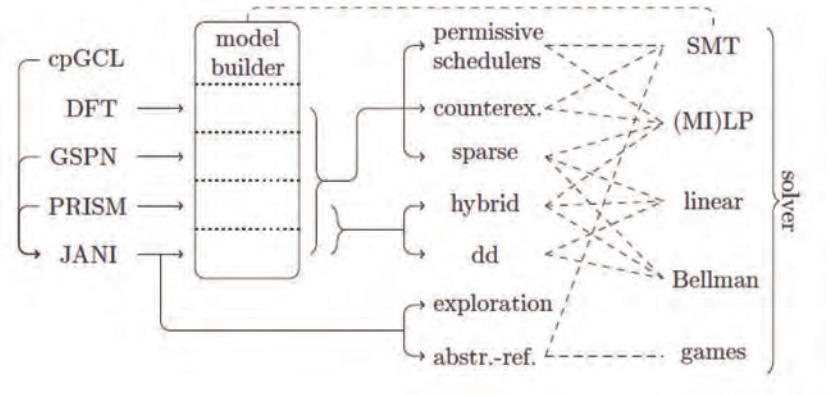
Probabilistic Model Checkers

- PRISM[4] [Kwiatkowska, Parker *et al.*]
- MRMC [Katoen *et al.*]
- iscasMC [Zhang *et al.*]
- iBioSim [Myers *et al.*]
- GreatSPN [Franceschinis *et al.*]
- SMART [Ciardo *et al.*]
- MarCie [Heiner *et al.*]
- PAT [Song Dong *et al.*]
- SToRM [Dehnert, Katoen *et al.*]
- ......

Statistical model checkers: Ymer, Vesta, UppAal-SMC, PlasmaLab, ...

[4] Recipient HVC Award 2016.

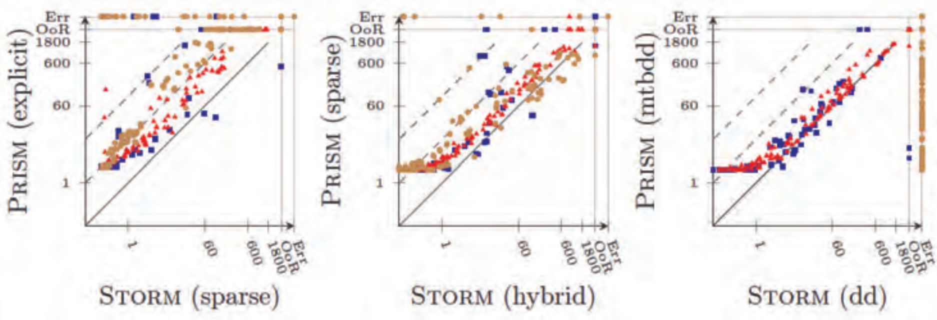
The Probabilistic Model Checker SToRM



Supports Markov chains, CTMCs, MDPs, and CTMDPs

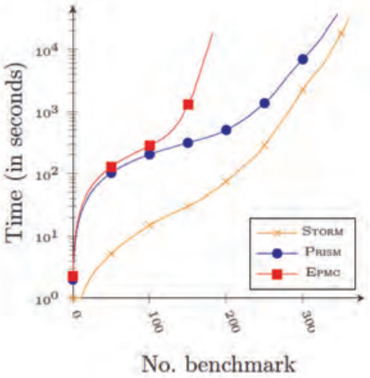
About 100,000 lines of C++ code

Comparison to PRISM



More information at: `stormchecker.org`

Experimental Comparison



Comparing the best engines for all

Overview

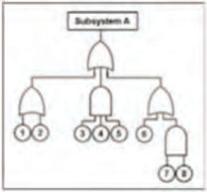
Fault Tree Analysis

fault tree analysis

All Images Books Videos News More ▾ Search tools

About 561,000 results (0.55 seconds)

Fault tree analysis (FTA) is a top down, deductive failure **analysis** in which an undesired state of a system is analyzed using Boolean logic to combine a series of lower-level events.

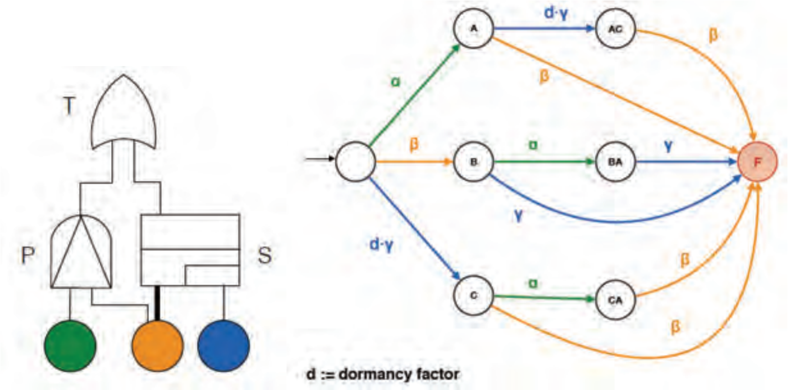


Fault tree analysis - Wikipedia, the free encyclopedia
https://en.wikipedia.org/wiki/**Fault_tree_analysis** Wikipedia ▾

About this result • Feedback

Dynamic Fault Trees

[Dugan *et al.*, 1995]



Markov decision process for a DFT

Experiences with FT model checking

- Logics like PCTL allow for expressing more properties
  - but hide logics as far as possible: use specification patterns or so
- Enable the analysis of a larger class of DFTs
- Model checking mostly substantially faster than FT analysis
- Abstraction aggravates this–for traditional FTA–even further:
  1. compositional minimisation
  2. tailored abstractions for FTs
  3. symmetry reduction and modularisation on FTs
  4. aggressive abstraction on FTs

  yield several orders of magnitude improvements.

Probabilistic Bisimulation

Intuition: transition probabilities for each equivalence class coincide.

Probabilistic Bisimulation

Intuition: transition probabilities for each equivalence class coincide.

Probabilistic bisimulation [Larsen & Skou, 1989]

Consider a DTMC with state space S and equivalence $R \subseteq S \times S$.
R is a probabilistic bisimulation on S if for any $(s, t) \in R$:

$$L(s) = L(t) \quad \text{and} \quad \mathbf{P}(s, C) = \mathbf{P}(t, C) \text{ for each } C \in S/R$$

where $\mathbf{P}(s, C) = \sum_{s' \in C} \mathbf{P}(s, s')$.

Let $\sim$ denote the largest possible probabilistic bisimulation.

Probabilistic Bisimulation

Intuition: transition probabilities for each equivalence class coincide.

Probabilistic bisimulation [Larsen & Skou, 1989]

Consider a DTMC with state space S and equivalence $R \subseteq S \times S$.
R is a probabilistic bisimulation on S if for any $(s,t) \in R$:

$$L(s) = L(t) \quad \text{and} \quad \mathbf{P}(s,C) = \mathbf{P}(t,C) \text{ for each } C \in S/R$$

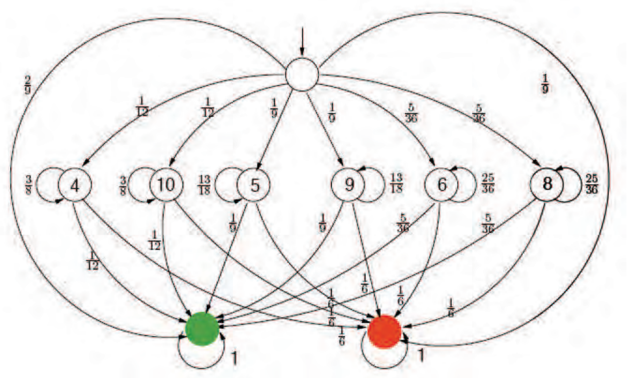
where $\mathbf{P}(s,C) = \sum_{s' \in C} \mathbf{P}(s,s')$.

Let $\sim$ denote the largest possible probabilistic bisimulation.

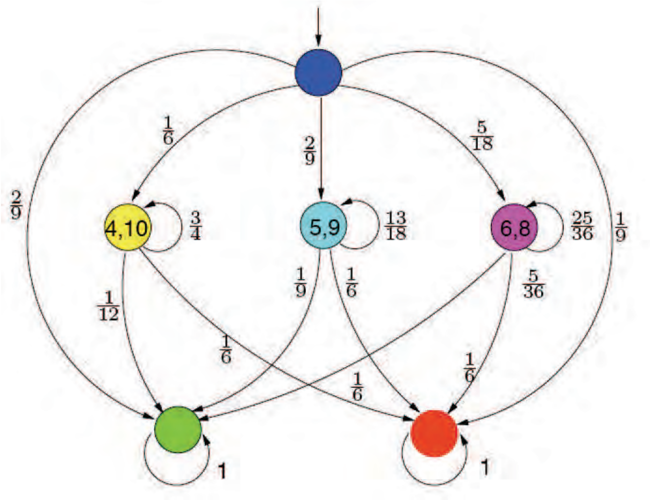
Variants: weak, divergence-sensitive, distribution-based, for CTMC, MDPs, etc.

Craps

- Come-out roll:
  - 7 or 11: win
  - 2, 3, or 12: lose
  - else: roll again
- Next roll(s):
  - 7: lose
  - point: win
  - else: roll again



Craps's Bisimulation Quotient



Properties

Quotienting: using partition-refinement in $\mathcal{O}(|\mathbf{P}| \cdot \log |S|)$

Preservation: all probabilistic CTL*-formulas

Congruence: with respect to parallel composition

$$(\mathcal{M}_1 \sim \mathcal{N}_1 \text{ and } \mathcal{M}_2 \sim \mathcal{N}_2) \quad \text{implies} \quad \mathcal{M}_1 \,||\, \mathcal{M}_2 \sim \mathcal{N}_1 \,||\, \mathcal{N}_2$$

Stuttering: weak variants preserve PCTL* without next-modalities

Savings: potentially exponentially in time and space

Exploiting Compositionality [Hermanns and K., 2000]

- Assume system is given by:

$$\mathcal{M}_1 \,\|\, \ldots \,\|\, \mathcal{M}_i \,\|\, \ldots \,\|\, \mathcal{M}_k$$

with $\mathcal{M}_j$ a Markov automaton and CSP-like composition $\|$

Exploiting Compositionality [Hermanns and K., 2000]

- Assume system is given by:

$$\mathcal{M}_1 \,||\, \ldots \,||\, \mathcal{M}_i \,||\, \ldots \,||\, \mathcal{M}_k$$

with $\mathcal{M}_j$ a Markov automaton and CSP-like composition $||$

- Recall congruence property:

$$(\mathcal{M}_1 \sim \mathcal{N}_1 \text{ and } \mathcal{M}_2 \sim \mathcal{N}_2) \quad \text{implies} \quad \mathcal{M}_1 \,||\, \mathcal{M}_2 \sim \mathcal{N}_1 \,||\, \mathcal{N}_2$$

Exploiting Compositionality [Hermanns and K., 2000]

- Assume system is given by:

$$\mathcal{M}_1 \,\|\, \ldots \,\|\, \mathcal{M}_i \,\|\, \ldots \,\|\, \mathcal{M}_k$$

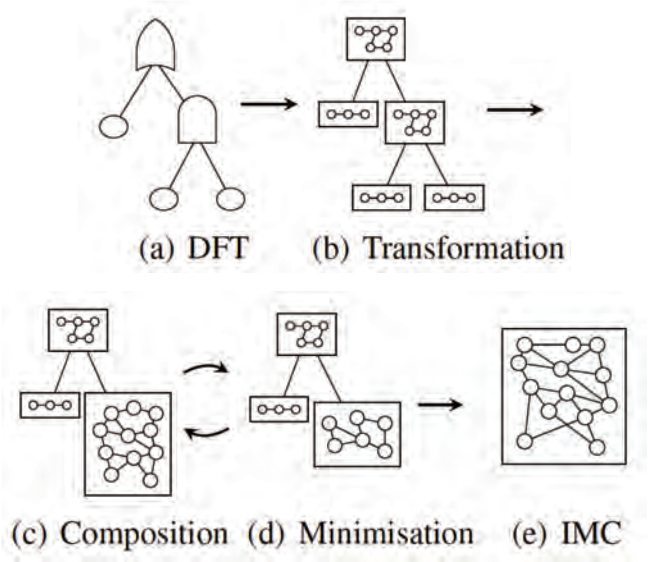
  with $\mathcal{M}_j$ a Markov automaton and CSP-like composition $\|$

- Recall congruence property:

  $(\mathcal{M}_1 \sim \mathcal{N}_1$ and $\mathcal{M}_2 \sim \mathcal{N}_2)$ implies $\mathcal{M}_1 \,\|\, \mathcal{M}_2 \sim \mathcal{N}_1 \,\|\, \mathcal{N}_2$

- Component-wise minimisation
  1. Pick process $\mathcal{M}_i$ and consider its quotient $\mathcal{M}_i/\sim$ under $\sim$
  2. Yielding $\mathcal{M}_1 \,\|\, \ldots \,\|\, \mathcal{M}_i/\sim \,\|\, \ldots \,\|\, \mathcal{M}_k$; repeat 1. and 2.
  3. Once all done, minimise pairs $\mathcal{M}_i/\sim \,\|\, \mathcal{M}_{i+1}/\sim$ etc.

Compositional DFT Minimisation [Crouzen *et al.*, 2010]



Compositional DFT Minimisation [Crouzen *et al.*, 2010]

case study	peak # states	# transitions	unreliability	time (s)
CPS	4113	24608	.00135	490

CPS	133	465	.00135	67

Comparing `Galileo DIFTree` (top) to `DFTCalc` (bottom)

Compositional DFT Minimisation [Crouzen *et al.*, 2010]

case study	peak # states	# transitions	unreliability	time (s)
CPS	4113	24608	.00135	490
CAS	8	10	.65790	1

CPS	133	465	.00135	67
CAS	36	119	.65790	94

Comparing `Galileo DIFTree` (top) to `DFTCalc` (bottom)

Compositional DFT Minimisation [Crouzen *et al.*, 2010]

case study	peak # states	# transitions	unreliability	time (s)
CPS	4113	24608	.00135	490
CAS	8	10	.65790	1
CAS-PH	x	x	x	x
NDPS	x	x	x	x

case study	peak # states	# transitions	unreliability	time (s)
CPS	133	465	.00135	67
CAS	36	119	.65790	94
CAS-PH	40052	265442	.112	231
NDPS	61	169	[.00586, .00598]	266

Comparing `Galileo DIFTree` (top) to `DFTCalc` (bottom)

Compositional DFT Minimisation [Crouzen *et al.*, 2010]

case study	peak # states	# transitions	unreliability	time (s)
CPS	4113	24608	.00135	490
CAS	8	10	.65790	1
CAS-PH	x	x	x	x
NDPS	x	x	x	x
FTTP-4	32757	426826	.01922	13111
FTTP-5	MO	MO	MO	MO

CPS	133	465	.00135	67
CAS	36	119	.65790	94
CAS-PH	40052	265442	.112	231
NDPS	61	169	[.00586, .00598]	266
FTTP-4	1325	13642	.01922	65
FTTP-6	11806565	22147378	.00045	1989

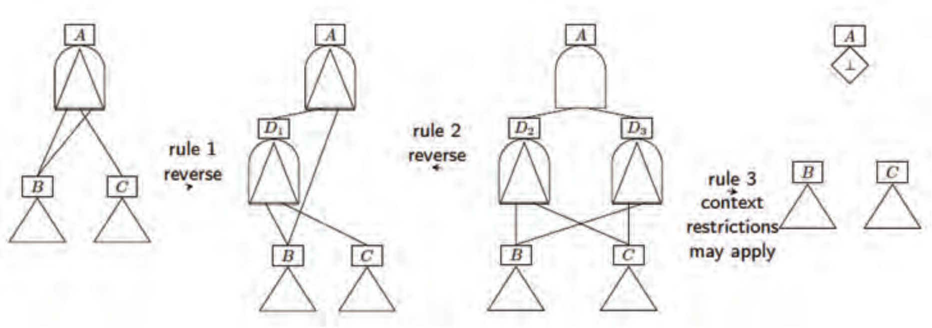
Comparing `Galileo DIFTree` (top) to `DFTCalc` (bottom)

Tailored DFT Abstraction

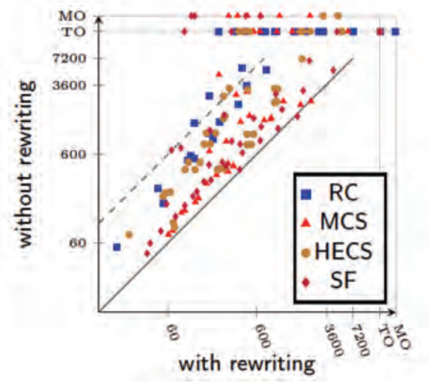
[Junges *et al.*, 2015]

Key idea

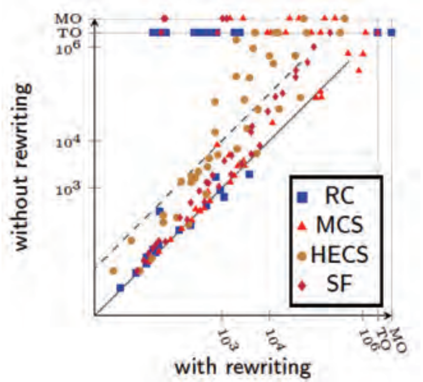
Simplify DFTs by graph rewriting prior to (compositional) state space generation.



Tailored DFT Abstraction



total verification and minimisation time

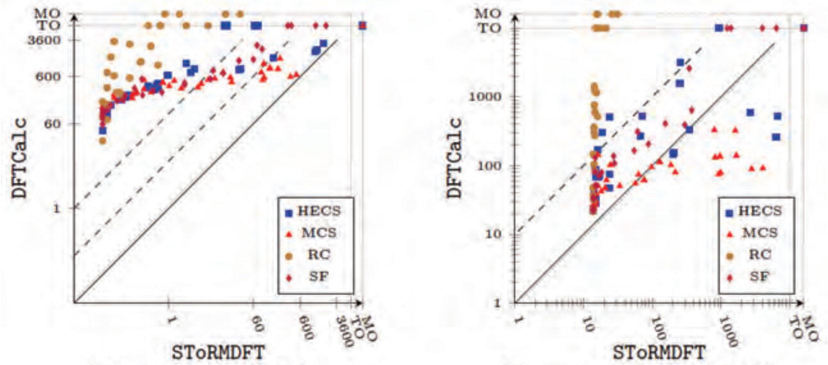


state space size of resulting CTMDP

49 out of 179 case studies could be treated now that could not be treated before

Boosting DFT State Space Generation [Volk *et al.*, 2016]

Apply POR, symmetry reduction, bisimulation, and state bit vectors.



(a) run time (seconds) (b) memory footprint (MB)

⇒ This boosts FT analysis by several orders of magnitude.

More Aggressive Abstraction

- Partition the state space into groups of concrete states
  - allow any partitioning, not just grouping of bisimilar states

More Aggressive Abstraction

- Partition the state space into groups of concrete states
  - allow any partitioning, not just grouping of bisimilar states
- This typically yields over-approximations
  - abstraction yields safe bounds on true measures

More Aggressive Abstraction

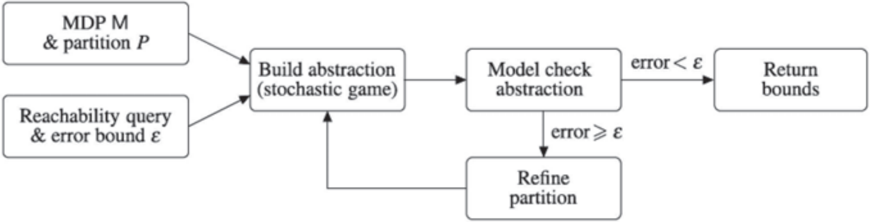
- Partition the state space into groups of concrete states
  - allow any partitioning, not just grouping of bisimilar states
- This typically yields over-approximations
  - abstraction yields safe bounds on true measures
- Correctness relies on simulation relations
  - preserve safety fragments of PCTL

More Aggressive Abstraction

- Partition the state space into groups of concrete states
  - allow any partitioning, not just grouping of bisimilar states
- This typically yields over-approximations
  - abstraction yields safe bounds on true measures
- Correctness relies on simulation relations
  - preserve safety fragments of PCTL
- Various abstract probabilistic models exist
  - two-player SGs, interval MCs, abstract PA, modal models, etc.

Abstraction-Refinement

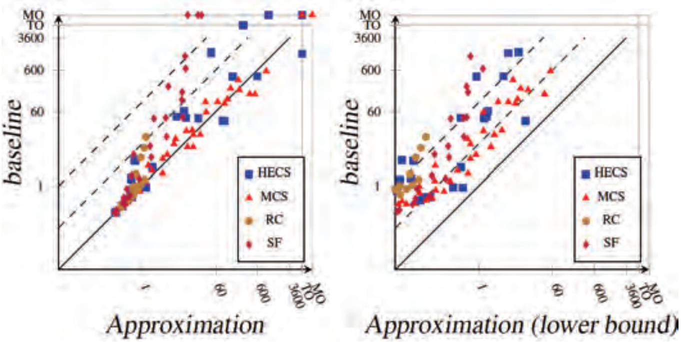
[Kwiatkowska *et al.*, 2010]



Millions of states can be reduced to hundreds of states in a few AR iterations

Abstraction-Refinement for DFTs [Volk *et al.*, 2017]

Partial fault tree analysis, making best/worst-case assumptions.



Abstraction-refinement terminates at 10% precision: $u - \ell < {}^{1}/_{10} \cdot \frac{u-\ell}{2}$

$\Rightarrow$ Scalable and one order of magnitude faster.

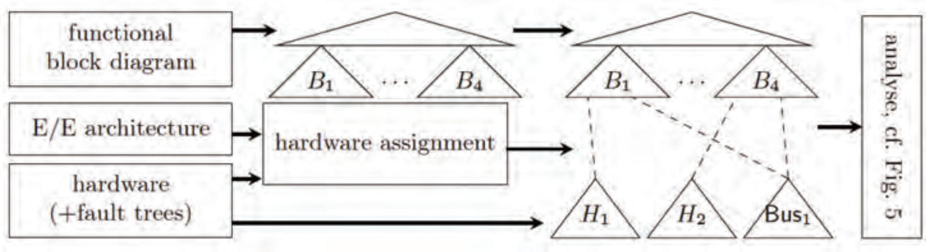
Safety Analysis of Vehicle Guidance [Ghadhab *et al.*, 2017]

Major safety goal: avoid wrong vehicle guidance.

Fail Operational

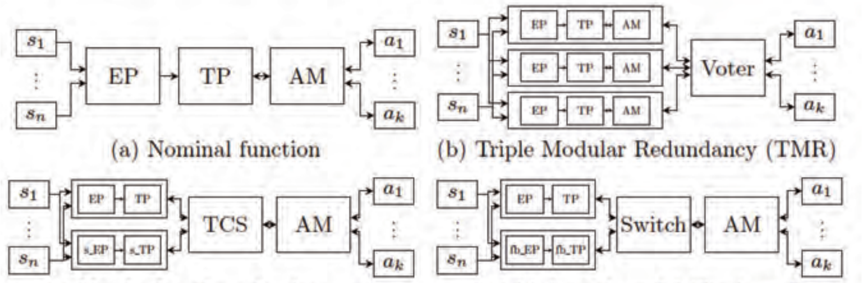
Automotive Safety Integrity Level (ASIL)

Approach



Vehicle guidance ASIL-D: 10^{-8} residual HW failures/hour

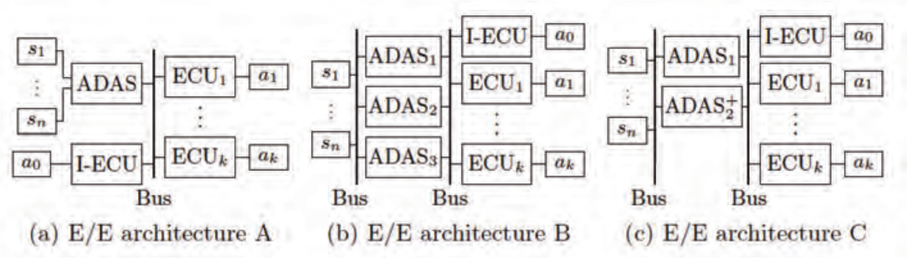
Vehicle Guidance



(a) Nominal function
(b) Triple Modular Redundancy (TMR)
(c) Nominal path and safety path
(d) Main path and fallback path

Fail-operational design patterns for autonomous driving.

Modern Car Architectures



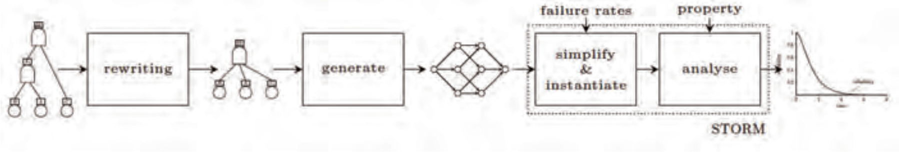
.

DFT Statistics

	Scenario					DFT			CTMC		
	SC	Arch.	Adap.	Sens.	Act.	#BE	#Dyn.	#Elem.	#States	#Trans.	Degrad.
I	SC1	B	—	2/4	4/4	76	25	233	5,377	42,753	—
II	SC2	B	—	2/4	4/4	70	23	211	5,953	50,049	19.35%
III	SC2	C	ADAS+	2/4	4/4	57	19	168	1,153	7,681	16.65%
IV	SC3	C	—	2/4	4/4	57	21	170	385	1,985	12.47%
V	SC2	A	—	2/4	4/4	58	19	185	193	897	0.00%
VI	SC2	B	removed I-ECU	2/4	4/4	65	21	199	1,201	8,241	19.98%
VII	SC2	B	5 ADAS, 2 BUS	2/8	7/7	96	30	266	194,433	2,171,905	19.35%
VIII	SC2	B	8 ADAS, 2 BUS	6/8	7/7	114	36	305	3,945,985	66,225,665	10.90%

One of the largest real-life DFTs in the literature

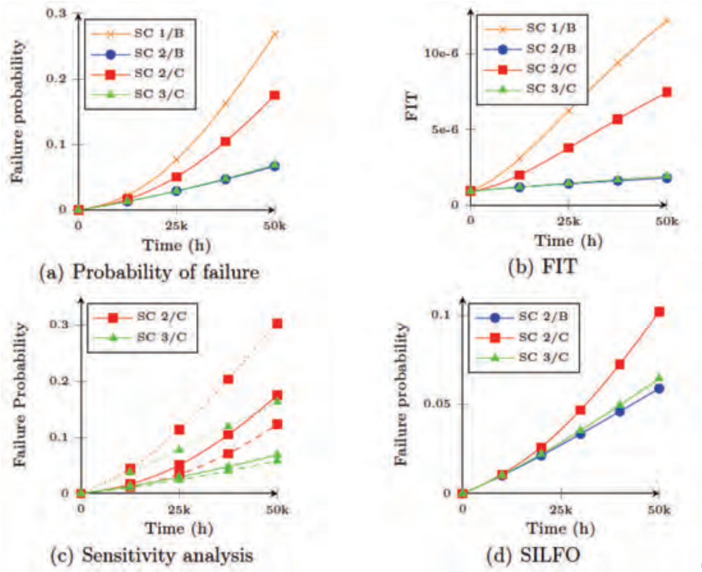
Analysis Approach



Measures

	Measure	Model Checking Queries
System	integrity	$1 - P(\Diamond^{\leq t} \text{ failed})$
	FIT	$\frac{1}{\text{lifetime}} \cdot (1 - P(\Diamond^{\leq \text{lifetime}} \text{ failed}))$
	MTTF	$\mathsf{ET}(\Diamond \text{ failed})$
Degradation	FFA	$1 - P(\Diamond^{\leq t} (\text{failed} \vee \text{degraded}))$
	FWD	$P((\neg\text{degraded}) \, \mathsf{U}^{\leq t} \, (\neg\text{degraded} \wedge \text{failed}))$
	MTDF	$\Sigma_{s \in \text{degraded}} (P(\neg\text{degraded} \, \mathsf{U} \, s) \cdot \mathsf{ET}^s(\Diamond \text{ failed}))$
	MDR	$\text{argmin}_{s \in \text{degraded}} (1 - P^s(\Diamond^{\leq t} \text{ failed}))$
	SILFO	$1 - \left(FWD + \Sigma_{s \in \text{degraded}} (P(\neg\text{degraded} \, \mathsf{U}^{\leq t} \, s) \cdot P^s(\Diamond^{\leq \text{drivecycle}} \text{ failed}))\right)$

Model Checking Results



(a) Probability of failure

(b) FIT

(c) Sensitivity analysis

(d) SILFO

Overview

The Need for Parameter Synthesis

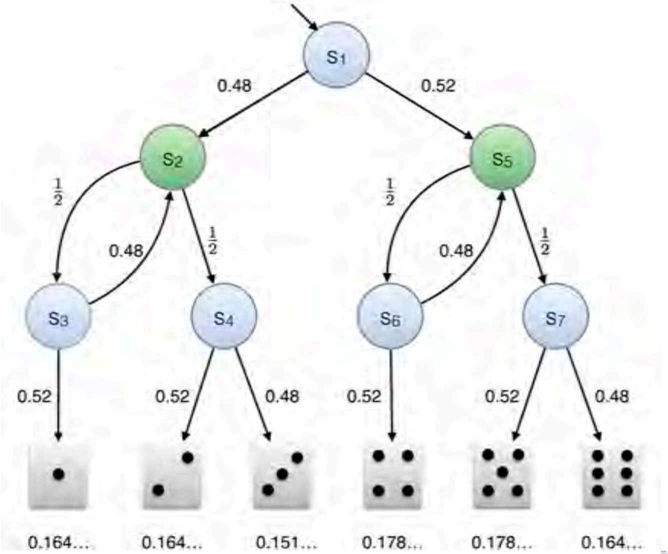
Fact:
Probabilistic model checking is applicable to various areas, e.g.:

- fault trees
- randomised algorithms
- systems biology

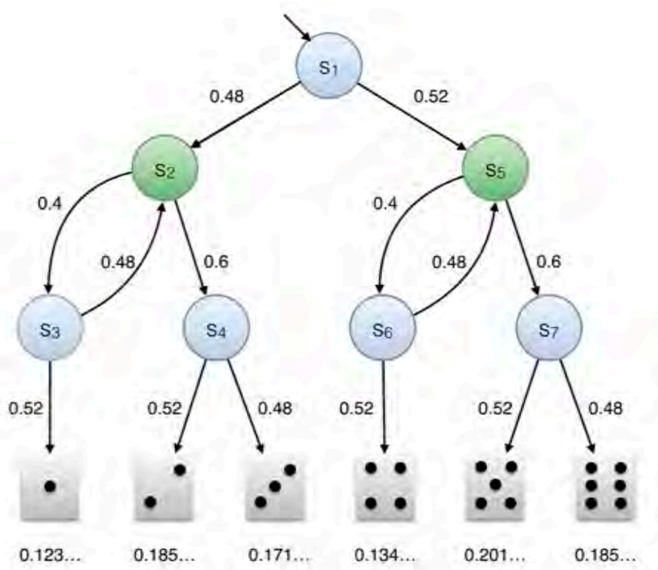
Limitation:
Probabilities need to be known a priori. Is this a valid assumption?
How sensitive are results when transition probabilities fluctuate?

Goal:
Treat parametric models, synthesise "safe" parameter values

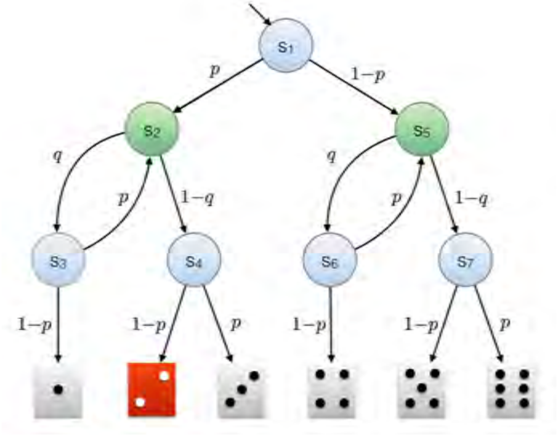
Biased Knuth-Yao's Die



Biased Knuth-Yao's Die

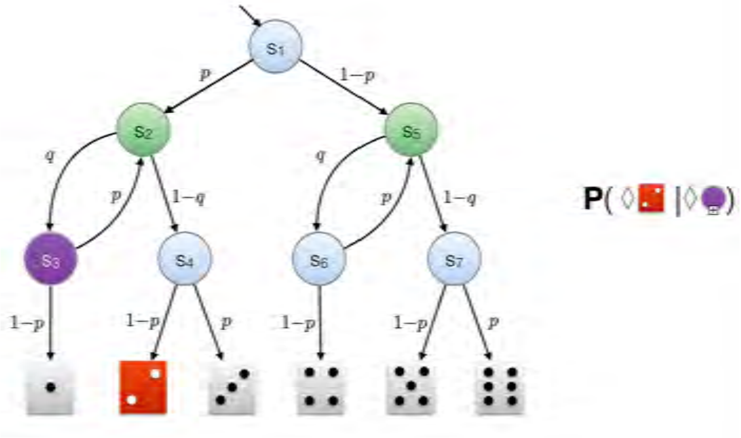


Parametric Knuth-Yao's Die



For which $1/10 \leqslant p \leqslant 9/10$ and $2/5 \leqslant q \leqslant 3/5$ does $\Pr(\Diamond 2) \geqslant 3/20$ hold?

Conditional Probabilities



Parameter Synthesis

Inputs:

1. a (finite) parametric Markov model
2. a property (e.g., reachability, expected reward, conditional reachability)
3. a threshold

Output:

For which parameter values does the pMC satisfy the property with the given threshold?

Parameter Synthesis

Inputs:

1. a (finite) parametric Markov model
2. a property (e.g., reachability, expected reward, conditional reachability)
3. a threshold

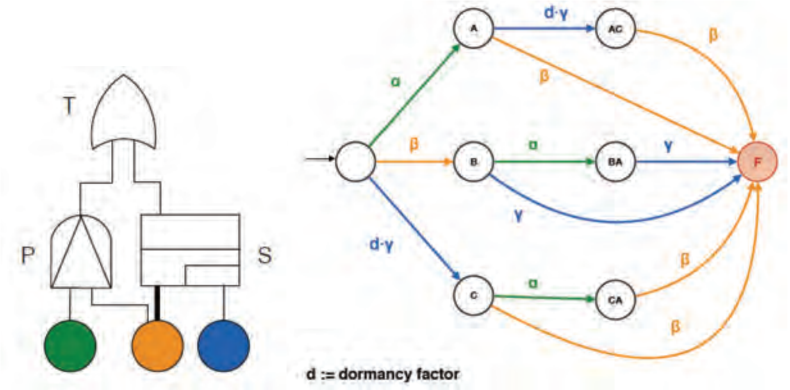
Output:

For which parameter values does the pMC satisfy the property with the given threshold?

Problem instances:

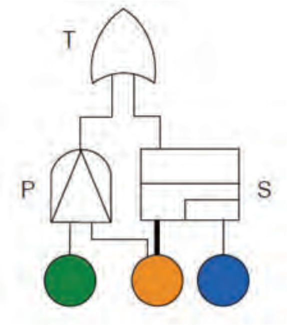
- What is the maximal tolerable message loss ensuring delivery $\geqslant 98\%$?
- ... the tolerable failure rate in a DFT ensuring MTTF $\geqslant 3$ hours?
- ......

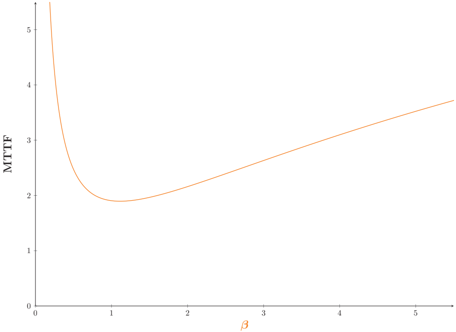
Recall Dynamic Fault Trees



Markov chain process for a DFT

Parametric Fault Trees





Sample parametric DFT and its MTTF

$$\text{MTTF} = \frac{200x^2 + 20x + 201}{x \cdot (20x + 201)} \text{ for } (\alpha, \beta, \gamma, d) = (10, x, 0.1, 0.5)$$

Parameter Synthesis

Aim:
partition the parameter space into safe and unsafe regions

Parameter Synthesis

Aim:

partition the parameter space into safe and unsafe regions

- Region = half-space defined by linear inequalities over the parameters
- A region R for threshold $\leqslant \beta$ is safe if no MC with $v \in R$ exceeds β
- A region R for threshold $\leqslant \beta$ is unsafe if no MC with $v \in R$ is at most β

Parameter Synthesis

Aim:

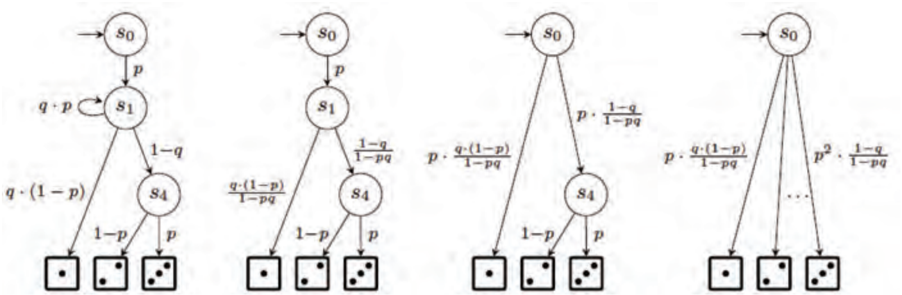
partition the parameter space into safe and unsafe regions

- Region = half-space defined by linear inequalities over the parameters
- A region R for threshold $\leqslant \beta$ is safe if no MC with $v \in R$ exceeds β
- A region R for threshold $\leqslant \beta$ is unsafe if no MC with $v \in R$ is at most β

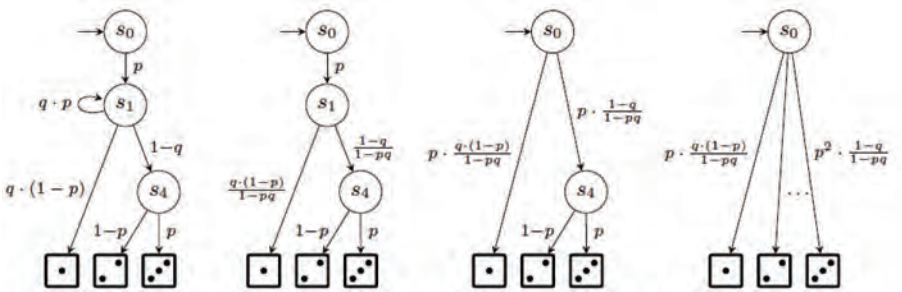
We present two approaches:

1. An exact procedure. How? Using SMT techniques
2. An approximate technique. How? Using parameter lifting

Computing Rational Functions [Daws, 2004]

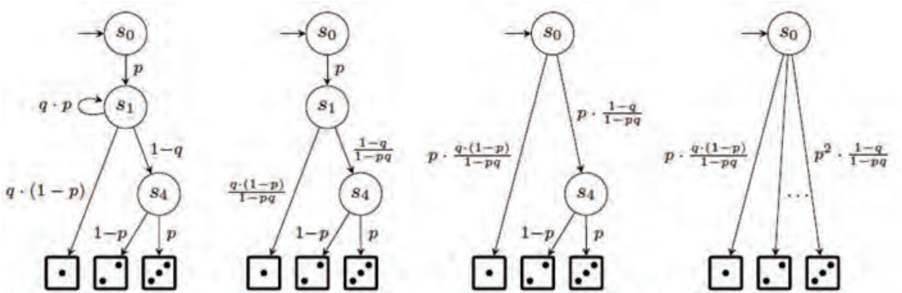


Computing Rational Functions [Daws, 2004]



$$\Pr(s_0 \vDash \Diamond(1 \text{ or } 3)) \leqslant 1/3 \quad \text{iff} \quad p\cdot q\cdot\frac{1-p}{1-p\cdot q} + p^2\cdot\frac{1-q}{1-p\cdot q} \leqslant 1/3$$

Computing Rational Functions [Daws, 2004]



$$\Pr(s_0 \models \Diamond(1 \text{ or } 3)) \leqslant 1/3 \quad \text{iff} \quad p \cdot q \cdot \frac{1-p}{1-p \cdot q} + p^2 \cdot \frac{1-q}{1-p \cdot q} \leqslant 1/3$$

This may yield large high-degree rational functions.

Resulting Rational Functions

Zooming In

```
16113281250000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000*
0000000000000000000000000000*y^8*x^16+70908645645691418683528900014
0000000000000000000000000000*y^16*x^10+117467949929856829214096069333
0000000000000000000000000000000*y^11*x^14+2496820390874591873323652
00000*y^9*x^5+1623069176566709188034320899985527367308573715625381 4
3414378976566791534423828125000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000*y^9*x^
000000000*y^10*x^5+212270382782282965928362847764266275121524939197
942867626605903252792113065719604492187500000000000000000000000000
766624450683593750000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000*y^11*x^5+7224355958514
000000000000000000000000*y^11*x^10+947070457251963420330399669597294
138726376835263049363738601237408229827880859375000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000*y^16*x^8+
```

Zooming In

```
516113281250000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000*
0000000000000000000000000000*y^8*x^16+7090864564569141868352890014
00000000000000000000000000*y^16*x^10+11746794992985682921409606933
000000000000000000000000000000*y^11*x^14+2496820390874591873323652
00000*y^9*x^5+16230691765667091880343208999855273673085737156253814
3414378976566791534423828125000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000*y^9*x^
000000000*y^10*x^5+212270382782282965928362847764266275121524939197
94286762660590325279211306571960449218750000000000000000000000000
7666244506835937500000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000*y^11*x^5+7224355958514
000000000000000000000000*y^11*x^10+947070457251963420330399669597294
138726376835263049363738601237408229827880859375000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000*y^16*x^8+
```

41 states, 138 transitions, 2 parameters:

numerator = 48 terms, denominator = product of 48 (linear) polynomials

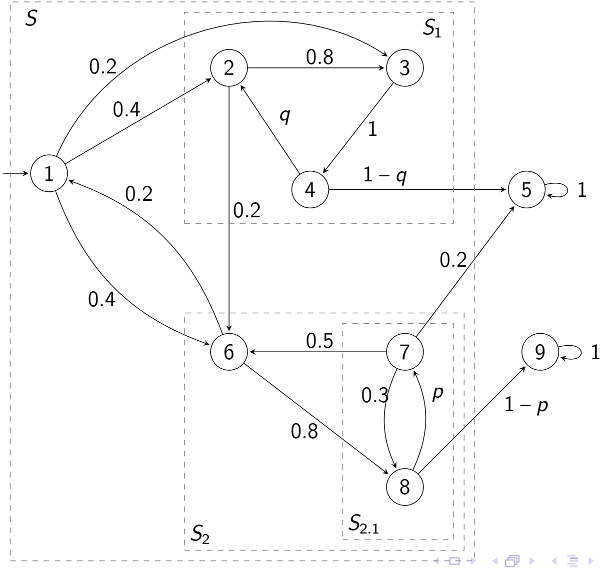
Zooming In

```
516113281250000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000*
000000000000000000000000000000*y^8*x^16+709086456456914186835289001
0000000000000000000000000000*y^16*x^10+11746794992985682921409606933
000000000000000000000000000000*y^11*x^14+2496820390874591873323652
00000*y^9*x^5+16230691765667091880343208999855273673085737156253814
341437897656679153442382812500000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000*y^9*x^
000000000*y^10*x^5+2122703827822829659283628477642662751215249391
942867626605903252792113065719604492187500000000000000000000000000
766624450683593750000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000*y^11*x^5+7224355958514
000000000000000000000000*y^11*x^10+9470704572519634203303996695972
138726376835263049363738601237408229827880859375000000000000000000
00000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000*y^16*x^8+
```

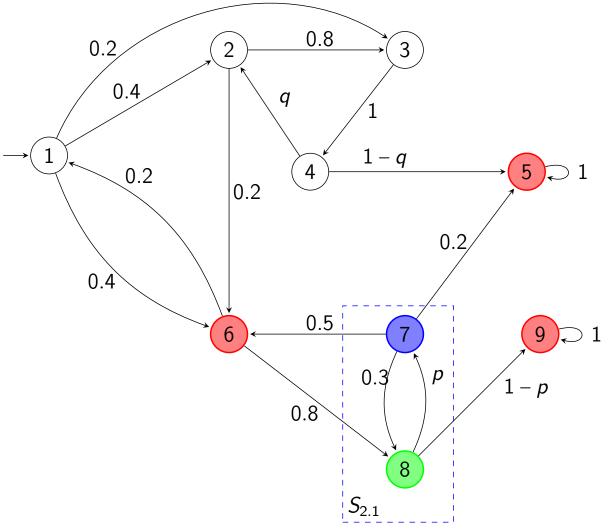
41 states, 138 transitions, 2 parameters:
numerator = 48 terms, denominator = product of 48 (linear) polynomials

⇒ Use bisimulation, SCC-decomposition and efficient gcd-computation

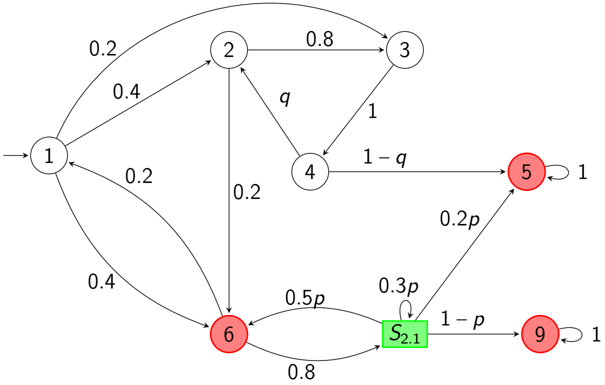
Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



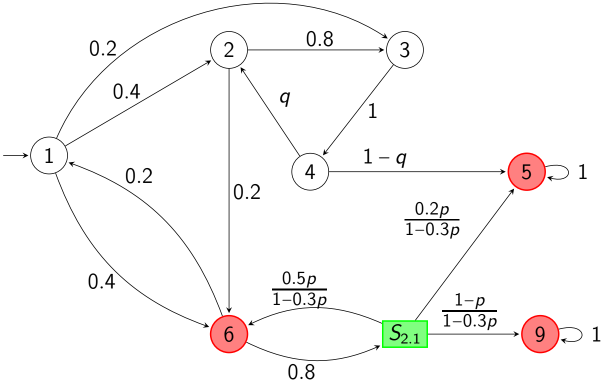
Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



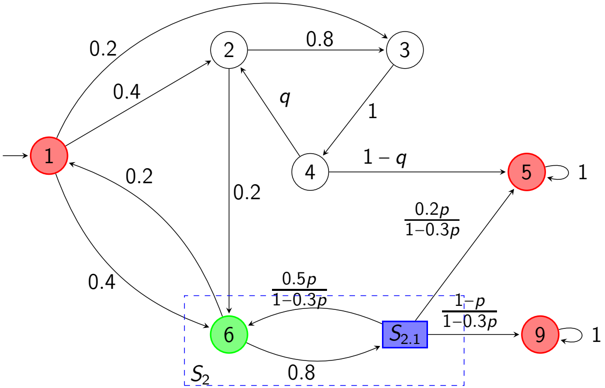
Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



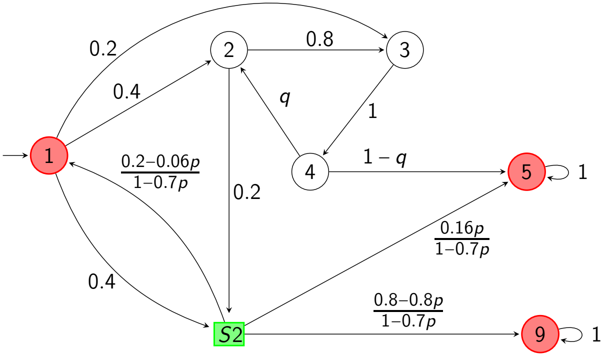
Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



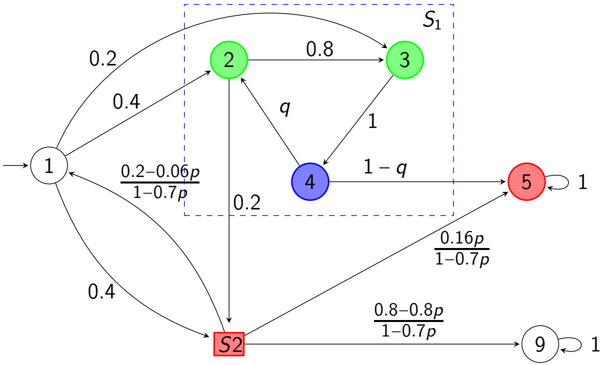
Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



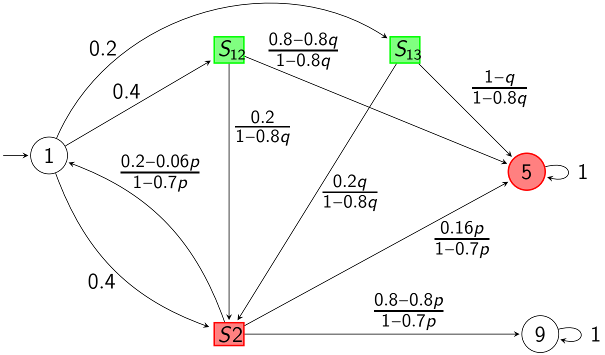
Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



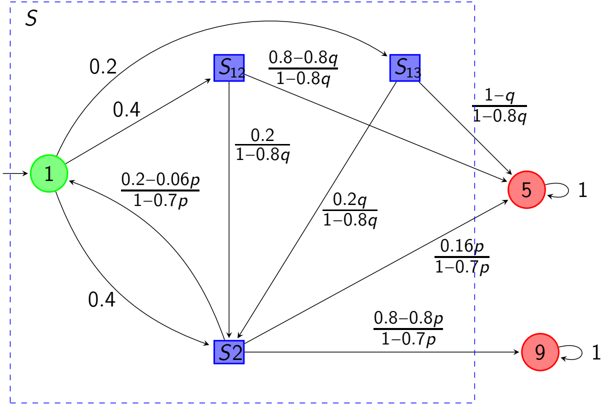
Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



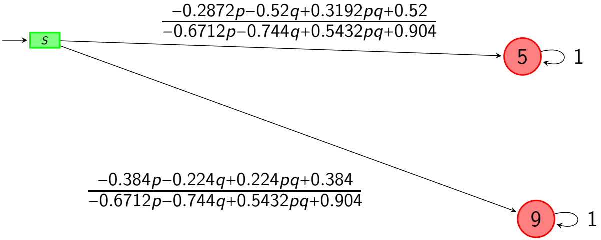
Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



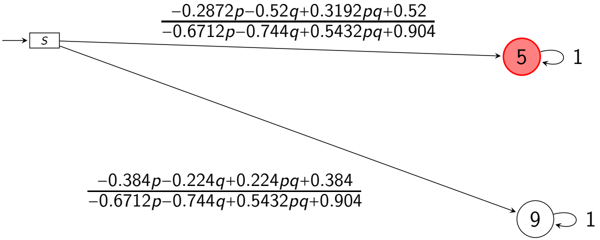
Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



Hierarchical SCC Decomposition [Jansen *et al.*, 2014]



For which (combinations of) values for p and q is the probability of reaching 5 smaller than $c \in [0, 1]$?
$\Rightarrow$ Evaluate rational function.

Exploiting SMT

Goal: partition parameter space in regions R that are either safe or unsafe

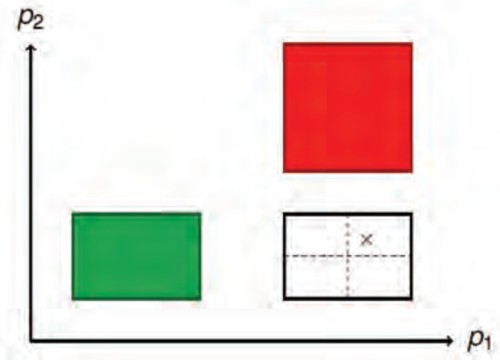
Idea: generate region candidates R and ask SMT solver[5] for counterexample

[5] Over non-linear real arithmetic using `Z3` or `SMT-RAT`.

Exploiting SMT

Goal: partition parameter space in regions R that are either safe or unsafe

Idea: generate region candidates R and ask SMT solver[5] for counterexample



UNSAT ⇒ R safe / unsafe

SAT ⇒ resample & refine

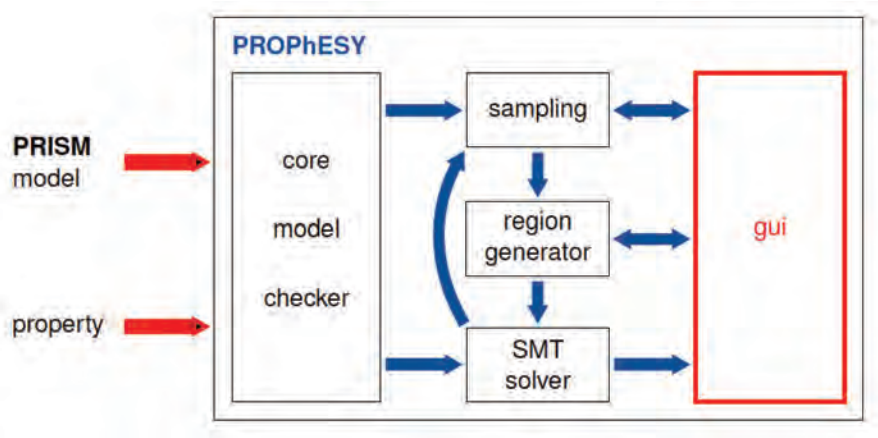
timeout ⇒ smaller regions

strategies

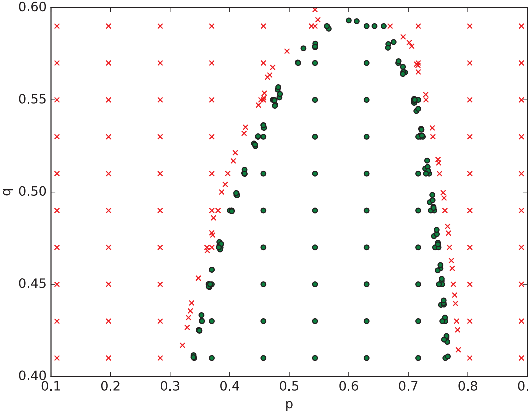
- cubes
- growing rectangles
- halfspaces

[5] Over non-linear real arithmetic using Z3 or SMT-RAT.

CEGAR-Like Parameter Synthesis

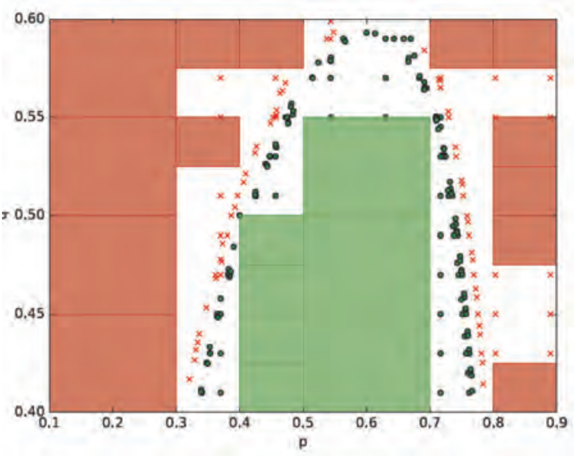


CEGAR-Like Parameter Synthesis



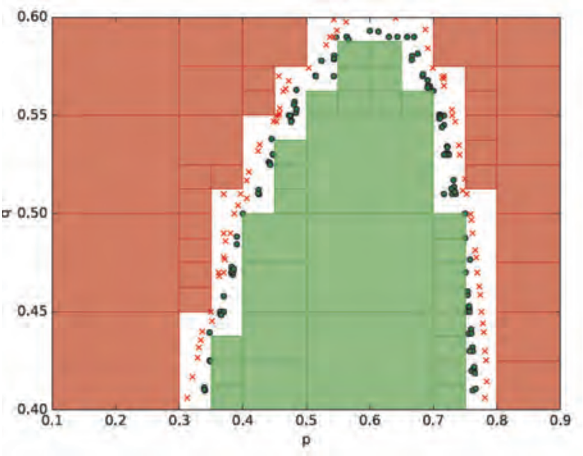
For which $1/10 \leqslant p \leqslant 9/10$ and $2/5 \leqslant q \leqslant 3/5$ does $\Pr(\Diamond 2) \geqslant 3/20$ hold?

CEGAR-Like Parameter Synthesis



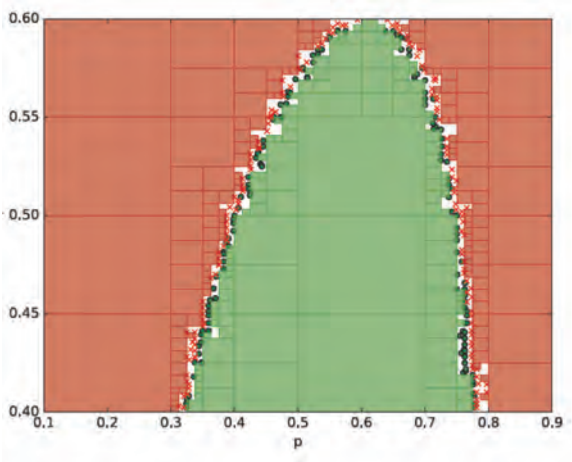
For which $1/10 \leqslant p \leqslant 9/10$ and $2/5 \leqslant q \leqslant 3/5$ does $\Pr(\Diamond 2) \geqslant 3/20$ hold?

CEGAR-Like Parameter Synthesis



For which $1/10 \leqslant p \leqslant 9/10$ and $2/5 \leqslant q \leqslant 3/5$ does $\Pr(\Diamond 2) \geqslant 3/20$ hold?

CEGAR-Like Parameter Synthesis



For which $1/10 \leqslant p \leqslant 9/10$ and $2/5 \leqslant q \leqslant 3/5$ does $\Pr(\Diamond 2) \geqslant 3/20$ hold?

Experimental Results

[Dehnert *et al.*, 2015]

competitors

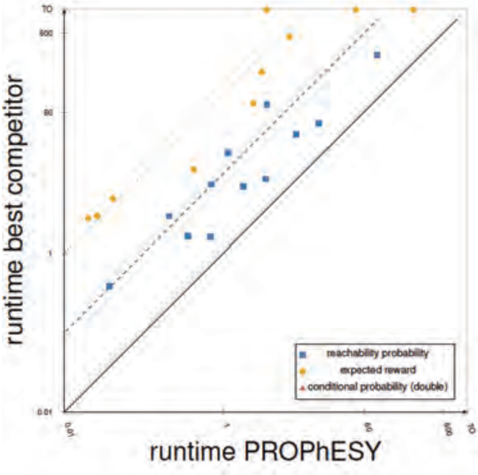
- PARAM [Hahn et al., 2010]
- PRISM [Parker et al., 2011]

models

- Bounded retransmission protocol
- NAND multiplexing
- Zeroconf, Crowds protocol
- 10^4 to $7.5 \cdot 10^6$ states

experiments:

- best set-up for each tool
- log-scale x- and y-axis



runner-up in the CAV 2015 artefact evaluation

Experimental Results

[Dehnert *et al.*, 2015]

competitors

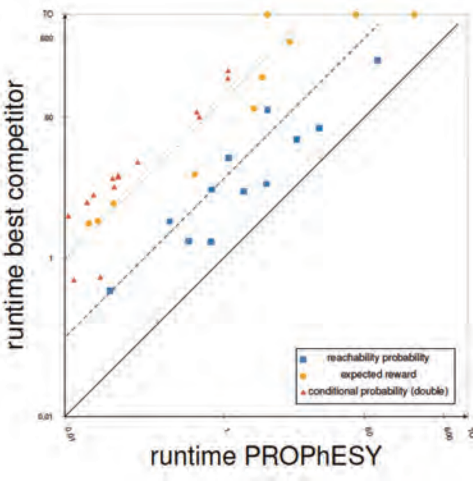
- PARAM [Hahn et al., 2010]
- PRISM [Parker et al., 2011]
- prototype [Baier et al., 2014]

models

- Bounded retransmission protocol
- NAND multiplexing
- Zeroconf, Crowds protocol
- 10^4 to $7.5 \cdot 10^6$ states

experiments:

- best set-up for each tool
- log-scale x- and y-axis



runner-up in the CAV 2015 artefact evaluation

Parameter Synthesis using SMT

Pros:

- Exact results: rational function is an exact symbolic object
- Drastic improvements over existing tools PARAM and PRISM
- User-friendly representation

Parameter Synthesis using SMT

Pros:

- Exact results: rational function is an exact symbolic object
- Drastic improvements over existing tools — `PARAM` and `PRISM`
- User-friendly representation

Cons:

- Rational function requires many gcd-computations — > 4 parameters?
- SMT performance unpredictable — heuristics hard

Parameter Synthesis using SMT

Pros:

- Exact results: rational function is an exact symbolic object
- Drastic improvements over existing tools PARAM and PRISM
- User-friendly representation

Cons:

- Rational function requires many gcd-computations > 4 parameters?
- SMT performance unpredictable heuristics hard

Can we do better by sacrificing exactness? Yes.

Approximate Parameter Synthesis [Quatmann *et al*,, 2016]

Let transition probabilities be linear in each variable.

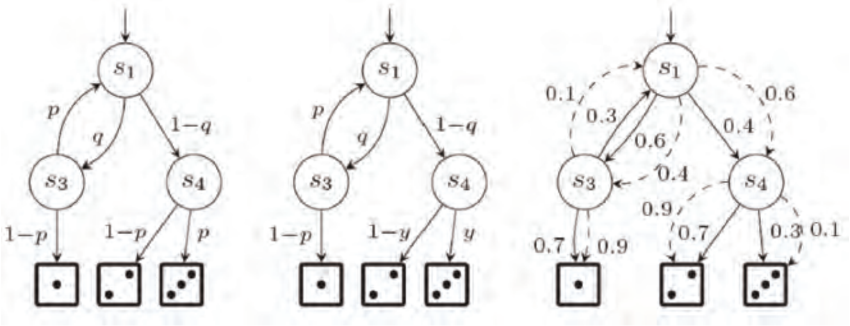
That is, transition functions f are multi-affine multivariate polynomials of form:

$$f = \sum a_i \cdot \left(\prod_{x \in V} x \right) \text{ with } a_i \in \mathbb{Q}$$

Examples: $3x \cdot y + 4y \cdot z$, $1 - x$, $x \cdot y \cdot z$ etc.

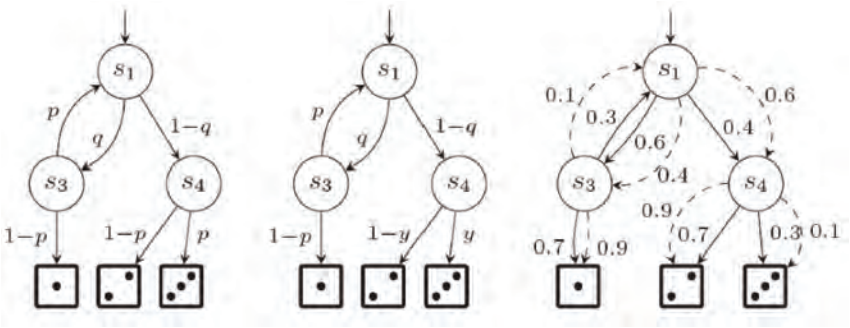
Approximate Parameter Synthesis [Quatmann *et al*,, 2016]

Approximate Parameter Synthesis [Quatmann *et al*,, 2016]



Two-phase approach: first remove dependencies, then substitute extremal values

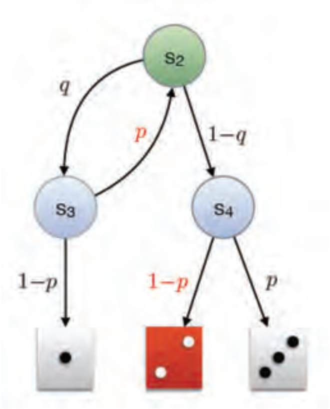
Approximate Parameter Synthesis [Quatmann *et al*,, 2016]

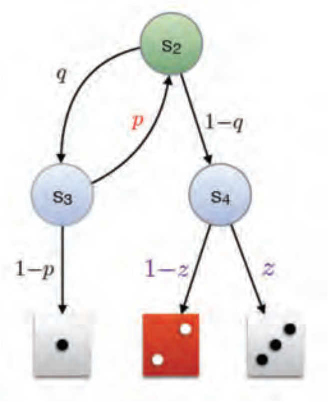


Two-phase approach: first remove dependencies, then substitute extremal values

Also applicable to parametric MDPs.

Phase 1: Relaxation





Parameter dependencies are removed; $\Pr(\Diamond 2) = (1 - z) \cdot \frac{1-q}{1-p \cdot q}$

$\Rightarrow$ each state is equipped with its own parameter

Phase 1: Relaxation

Correctness:

- Relaxed regions contain more valuations than original regions

⇒ Relaxation yields over-approximations

⇒ Relaxation preserves upper-bounds on reachability probs

Phase 1: Relaxation

Correctness:

- Relaxed regions contain more valuations than original regions

$\Rightarrow$ Relaxation yields over-approximations

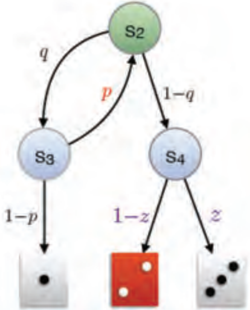
$\Rightarrow$ Relaxation preserves upper-bounds on reachability probs

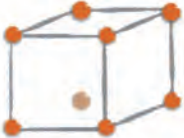
Complexity of parameter synthesis :

- Relaxation increases the number of parameters
- Extremal values of the state parameters attain maximal probabilities

$\Rightarrow$ Valuations for maximal probabilities are easier to find

Phase 2: Substitution

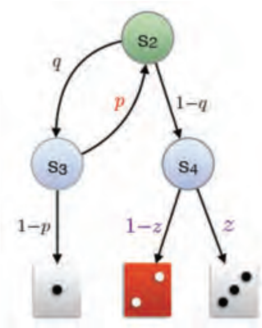


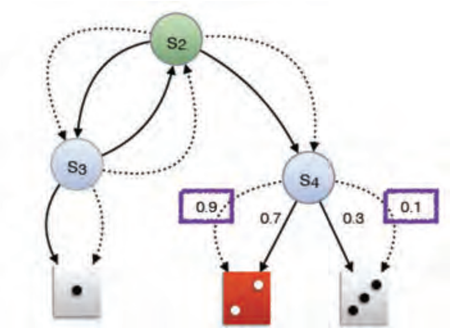


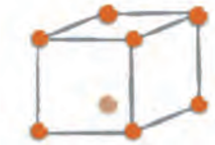
$$p, z \in [0.1, 0.3] \qquad p, z \in \{0.1, 0.3\}$$
$$q \in [0.4, 0.6] \qquad q \in \{0.4, 0.6\}$$

Local parameters per state $\Rightarrow$ extremal values at states suffice

Phase 2: Substitution



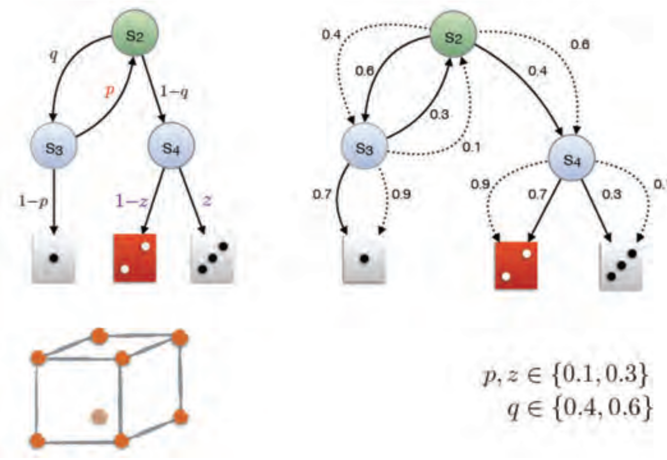




$$p, z \in \{\boxed{0.1}, 0.3\}$$
$$q \in \{0.4, 0.6\}$$

Local parameters per state ⇒ extremal values at states suffice

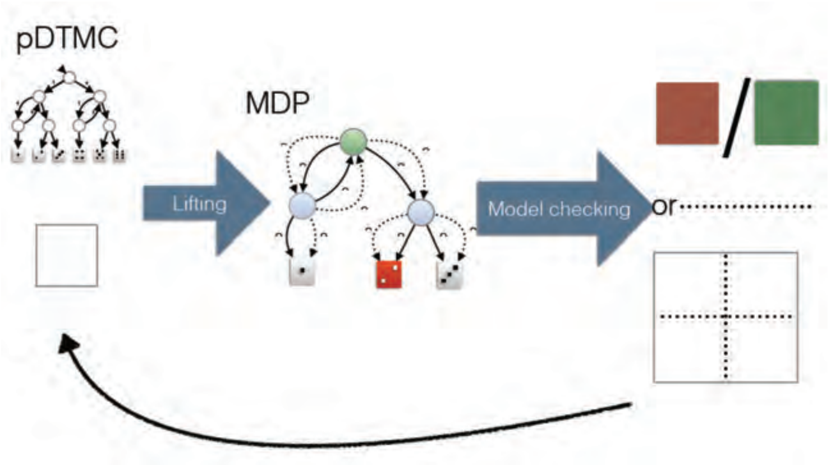
Phase 2: Substitution



This results in a Markov decision process.

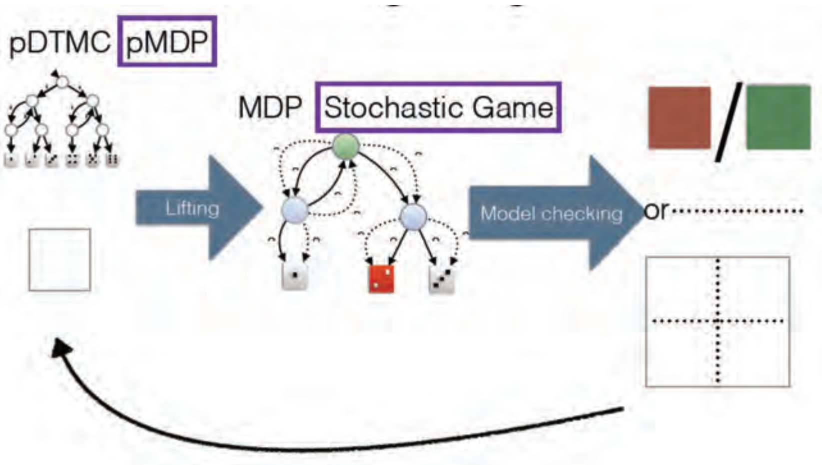
Its extremal reachability probabilities provide bounds for parametric MC.

Parameter Synthesis



Until $\approx 95\%$ of the parameter space is covered

Parameter Synthesis



Until 95% of the parameter space is covered

Coverage

		φ	n	# states	# trans	% p	t	safe	unsafe	neither	unkn
pMC	brp	$\mathbb{E}$	2	20 744	27 651	48%	**51**	14.9%	79.2%	**5.8%**	**0.2%**
		$\mathbb{E}$	4	20 744	27 651	48%	**71**	7.5%	51.0%	**40.6%**	**0.8%**
	crowds	$\mathbb{P}$	2	104 512	246 082	19%	**44**	54.4%	41.1%	**4.2%**	**0.3%**
	nand	$\mathbb{P}$	2	35 112	52 647	47%	**21**	21.4%	68.5%	**6.9%**	**3.2%**
pMDP	brp	$\mathbb{P}$	2	40 721	55 143	50%	**153**	6.6%	90.4%	**3.0%**	**0.0%**
	cons	$\mathbb{P}$	4	22 656	75 232	41%	**357**	2.6%	87.0%	**10.4%**	**0.0%**
	sav	$\mathbb{P}$	4	379	1 127	50%	**2**	44.0%	15.4%	**35.4%**	**5.3%**
	zconf	$\mathbb{P}$	2	88 858	203 550	40%	**186**	16.6%	77.3%	**5.6%**	**0.5%**

Parameter space $R = [10^{-5}, 1{-}10^{-5}]^n$ until 95% coverage for n parameters for 625 equally-sized regions without region refinement

single core, 2.0 GHz, 30GB RAM, TO = one hour

Parametric Markov Chain Benchmarks

						PLA			PRISM
benchmark	instance	φ	#pars	#states	#trans	#regions	direct	bisim	best
brp	(256,5)	$\mathbb{P}$	2	19 720	26 627	37	**6**	14	TO
	(4096,5)	$\mathbb{P}$	2	315 400	425 987	13	**233**	TO	TO
	(256,5)	$\mathbb{E}$	2	20 744	27 651	195	**8**	15	TO
	(4096,5)	$\mathbb{E}$	2	331 784	442 371	195	502	**417**	TO
	(16,5)	$\mathbb{E}$	4	1 304	1 731	1 251 220	2 764	**1 597**	TO
	(32,5)	$\mathbb{E}$	4	2 600	3 459	1 031 893	TO	**2 722**	TO
	(256,5)	$\mathbb{E}$	4	20 744	27 651	–	TO	TO	TO
crowds	(10,5)	$\mathbb{P}$	2	104 512	246 082	123	17	**6**	2038
	(15,7)	$\mathbb{P}$	2	8 364 409	25 108 729	116	1 880	**518**	TO
	(20,7)	$\mathbb{P}$	2	45 421 597	164 432 797	119	TO	**2 935**	TO
nand	(10,5)	$\mathbb{P}$	2	35 112	52 647	469	**22**	30	TO
	(25,5)	$\mathbb{P}$	2	865 592	1 347 047	360	**735**	2 061	TO

coverage of 95%; refinement into four equally-sized regions

SMT approach needs >one hour on all instances.

Parametric MDP Benchmarks

						PLA			PRISM
benchmark	instance	φ	#pars	#states	#trans	#regions	direct	bisim	best
brp	(256,5)	$\mathbb{P}$	2	40 721	55 143	37	**35**	3 359	TO
	(4096,5)	$\mathbb{P}$	2	647 441	876 903	13	**3 424**	TO	TO
consensus	(2,2)	$\mathbb{P}$	2	272	492	119	< **1**	< **1**	31
	(2,32)	$\mathbb{P}$	2	4 112	7 692	108	**113**	141	TO
	(4,2)	$\mathbb{P}$	4	22 656	75 232	6 125	**1 866**	2 022	TO
	(4,4)	$\mathbb{P}$	4	43 136	144 352	–	TO	TO	TO
sav	(6,2,2)	$\mathbb{P}$	2	379	1 127	162	< **1**	< **1**	TO
	(100,10,10)	$\mathbb{P}$	2	1 307 395	6 474 535	37	**1 612**	TO	TO
	(6,2,2)	$\mathbb{P}$	4	379	1 127	621 175	944	**917**	TO
	(10,3,3)	$\mathbb{P}$	4	1 850	6 561		TO	TO	TO
zeroconf	(2)	$\mathbb{P}$	2	88 858	203 550	186	**86**	1 295	TO
	(5)	$\mathbb{P}$	2	494 930	1 133 781	403	**2 400**	TO	TO

coverage of 95%

Summary So Far

SMT-based approach:

- Exact
- Requires rational functions
- Fickle SMT performance
- $\approx 10^6$ states, 2 parameters
- Restricted to Markov chains
- CEGAR-like refinement

Summary So Far

SMT-based approach:

- Exact
- Requires rational functions
- Fickle SMT performance
- $\approx 10^6$ states, 2 parameters
- Restricted to Markov chains
- CEGAR-like refinement

Parameter lifting approach:

- Approximative
- Off-the-shelf model checking
- No SMT, no rational functions
- $\approx 10^7$ states, 4–5 parameters
- Applicable to MDPs and games
- CEGAR-like refinement

Multiple Objectives

Multiple Objectives

Inputs:

1. a (finite) parametric MDP $\mathcal{M}$ over $V = \{ x_1, \ldots, x_n \}$
   with signomial parameter functions $c \cdot x_1^{a_1} \cdot \ldots \cdot x_n^{a_n}$ for $c \in \mathbb{R}$
2. multiple objectives $\varphi_1, \ldots, \varphi_m$ (reachability, expected reward)
3. objective function f over V: $\sum_{k=1}^{N} c_k \cdot x_1^{a_{1k}} \cdot \ldots \cdot x_n^{a_{nk}}$ for $c_k \in \mathbb{R}$

Multiple Objectives

Inputs:

1. a (finite) parametric MDP $\mathcal{M}$ over $V = \{ x_1, \ldots, x_n \}$
   with signomial parameter functions $c \cdot x_1^{a_1} \cdot \ldots \cdot x_n^{a_n}$ for $c \in \mathbb{R}$
2. multiple objectives $\varphi_1, \ldots, \varphi_m$ (reachability, expected reward)
3. objective function f over V: $\sum_{k=1}^{N} c_k \cdot x_1^{a_{1k}} \cdot \ldots \cdot x_n^{a_{nk}}$ for $c_k \in \mathbb{R}$

Output:

A (randomised) policy σ and valuation u such that:

$$\underbrace{\mathcal{M}^\sigma[u] \models \varphi_1 \wedge \ldots \wedge \varphi_m}_{\text{"feasibility"}} \quad \text{and} \quad \underbrace{\text{the objective } f \text{ is minimised}}_{\text{"optimality"}}$$

Multiple Objectives

Inputs:

1. a (finite) parametric MDP $\mathcal{M}$ over $V = \{ x_1, \ldots, x_n \}$
   with signomial parameter functions $c \cdot x_1^{a_1} \cdot \ldots \cdot x_n^{a_n}$ for $c \in \mathbb{R}$
2. multiple objectives $\varphi_1, \ldots, \varphi_m$ (reachability, expected reward)
3. objective function f over V: $\sum_{k=1}^{N} c_k \cdot x_1^{a_{1k}} \cdot \ldots \cdot x_n^{a_{nk}}$ for $c_k \in \mathbb{R}$

Output:

A (randomised) policy σ and valuation u such that:

$$\underbrace{\mathcal{M}^{\sigma}[u] \vDash \varphi_1 \wedge \ldots \wedge \varphi_m}_{\text{“feasibility”}} \quad \text{and} \quad \underbrace{\text{the objective } f \text{ is minimised}}_{\text{“optimality”}}$$

multi-objective MDP: use LP [Etessami *et al.*, 2008]

multi-objective parametric MDP: use special type NLP [Cubuktepe *et al.*, 2017]

NLP for Two Objectives

Objectives: minimise f, reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to: $p_{s_I} \leqslant p$ reachability objective

$c_{s_I} \leqslant c$ expected reward objective

NLP for Two Objectives

Objectives: minimise f, reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to:	$p_{s_I} \leqslant p$	reachability objective
	$c_{s_I} \leqslant c$	expected reward objective
$\forall s:$	$\sum_{\alpha \in Act(s)} \sigma^{s,\alpha} = 1$	randomised scheduler
$\forall s, \alpha:$	$0 \leqslant \sigma^{s,\alpha} \leqslant 1$	

NLP for Two Objectives

Objectives: minimise f, reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to: $p_{s_I} \leqslant p$ — reachability objective

$c_{s_I} \leqslant c$ — expected reward objective

$\forall s: \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} = 1$ — randomised scheduler

$\forall s, \alpha: 0 \leqslant \sigma^{s,\alpha} \leqslant 1$

$\forall s, \alpha: \sum_{t \in S} \mathcal{P}(s, \alpha, t) = 1$ — probabilistic choice

$\forall s, t, \alpha: 0 \leqslant \mathcal{P}(s, \alpha, t) \leqslant 1$

NLP for Two Objectives

Objectives: minimise f, reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to: $p_{s_I} \leqslant p$ — reachability objective

$c_{s_I} \leqslant c$ — expected reward objective

$\forall s: \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} = 1$ — randomised scheduler

$\forall s, \alpha: 0 \leqslant \sigma^{s,\alpha} \leqslant 1$

$\forall s, \alpha: \sum_{t \in S} \mathcal{P}(s, \alpha, t) = 1$ — probabilistic choice

$\forall s, t, \alpha: 0 \leqslant \mathcal{P}(s, \alpha, t) \leqslant 1$

$\forall s \in T: p_s = 1$ — reach prob of T

$\forall s \notin T: p_s = \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{t \in S} \mathcal{P}(s, \alpha, t) \cdot p_t$ — transition probabilities

NLP for Two Objectives

Objectives: minimise f, reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to:

	$p_{s_I} \leqslant p$	reachability objective
	$c_{s_I} \leqslant c$	expected reward objective
$\forall s:$	$\sum_{\alpha \in Act(s)} \sigma^{s,\alpha} = 1$	randomised scheduler
$\forall s, \alpha:$	$0 \leqslant \sigma^{s,\alpha} \leqslant 1$	
$\forall s, \alpha:$	$\sum_{t \in S} \mathcal{P}(s, \alpha, t) = 1$	probabilistic choice
$\forall s, t, \alpha:$	$0 \leqslant \mathcal{P}(s, \alpha, t) \leqslant 1$	
$\forall s \in T:$	$p_s = 1$	reach prob of T
$\forall s \notin T:$	$p_s = \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{t \in S} \mathcal{P}(s, \alpha, t) \cdot p_t$	transition probabilities
$\forall s \in G:$	$c_s = 0$	expected cost of G
$\forall s \notin G:$	$c_s = \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \left(c(s, \alpha) + \sum_{t \in S} \mathcal{P}(s, \alpha, t) \cdot c_t \right)$	expected costs

Theorem: This NLP is sound and complete. But solving NLPs is exponential.

Can We Do Better?

Yes.

1. Get a feasible solution in polynomial time[6]. How? Geometric programming.
2. Get local optimum. How? Sequential convex programming.

Solutions are approximations that can be arbitrarily close.

[6] Approximation of arbitrarily precise results by interior point methods with barriers

Geometric Programming

Objective: minimise $f :: \sum_{k=1}^{N} c_k \cdot x_1^{a_{1k}} \cdot \ldots \cdot x_n^{a_{nk}}$ for $c_k \in \mathbb{R}_{\geqslant 0}$

Subject to:

$$\begin{array}{rlll} \forall i \in [1..m]: & g_i & \leqslant 1 & \text{posynomial } g_i \\ \forall j \in [1..\ell]: & h_j & = 1 & \text{monomial } h_j \end{array}$$

Division transformation: $f \leqslant h$ if and only if $\frac{f}{h} \leqslant 1$

Relaxation: $f = h$ implies $f \leqslant h$ if and only if $\frac{f}{h} \leqslant 1$

Convexification

$$\forall s \in S \setminus T. \quad p_s = \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{s' \in S} \mathcal{P}(s,\alpha,s') \cdot p_{s'}$$

$$\Downarrow \text{ division transformation}$$

upper bound on actual probability

$$\forall s \in S \setminus T. \quad p_s \geq \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{s' \in S} \mathcal{P}(s,\alpha,s') \cdot p_{s'}$$

$$\Downarrow \text{ relaxation}$$

still signomials (not convex)

$$\forall s \in S \setminus T. \quad \frac{\sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \cdot \sum_{s' \in S} \mathcal{P}(s,\alpha,s') \cdot p_{s'}}{p_s} \leq 1$$

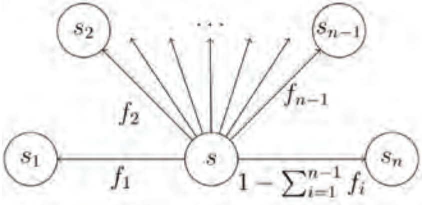
Lifting

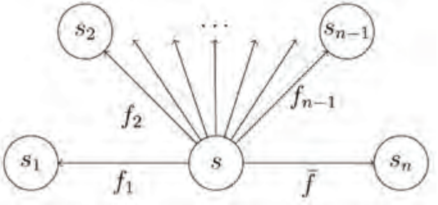
$$\mathcal{P}(s,\alpha,\bar{s}) = 1 - \sum_{s'\in S\setminus\{\bar{s}\}} \mathcal{P}(s,\alpha,s') \Longrightarrow \bar{\mathcal{P}}(s,\alpha,\bar{s}) = \bar{p}_{s,\alpha,\bar{s}} \in L$$

posynomial

signomial

lifting variable





GP for Two Objectives

Objectives: reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to:

$$\frac{p_{s_I}}{p} \leqslant 1 \qquad \text{reachability}$$

$$\frac{c_{s_I}}{c} \leqslant 1 \qquad \text{expected reward}$$

GP for Two Objectives

Objectives: reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to:

$$\frac{p_{s_I}}{p} \leqslant 1 \qquad \text{reachability}$$

$$\frac{c_{s_I}}{c} \leqslant 1 \qquad \text{expected reward}$$

$$\forall s: \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \leqslant 1 \qquad \text{randomised scheduler}$$

$$\forall s, \alpha: \sigma^{s,\alpha} \leqslant 1$$

GP for Two Objectives

Objectives: reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to:

$$\frac{p_{s_I}}{p} \leqslant 1 \qquad \text{reachability}$$

$$\frac{c_{s_I}}{c} \leqslant 1 \qquad \text{expected reward}$$

$$\forall s: \quad \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \leqslant 1 \qquad \text{randomised scheduler}$$

$$\forall s, \alpha: \quad \sigma^{s,\alpha} \leqslant 1$$

$$\forall s, \alpha: \quad \sum_{t \in S} \overline{\mathcal{P}}(s, \alpha, t) \leqslant 1 \qquad \text{probabilistic choice}$$

$$\forall s, t, \alpha: \quad \overline{\mathcal{P}}(s, \alpha, t) \leqslant 1$$

GP for Two Objectives

Objectives: reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to:

$$\frac{p_{s_I}}{p} \leqslant 1 \qquad \text{reachability}$$

$$\frac{c_{s_I}}{c} \leqslant 1 \qquad \text{expected reward}$$

$$\forall s: \sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \leqslant 1 \qquad \text{randomised scheduler}$$

$$\forall s, \alpha: \sigma^{s,\alpha} \leqslant 1$$

$$\forall s, \alpha: \sum_{t \in S} \overline{\mathcal{P}}(s, \alpha, t) \leqslant 1 \qquad \text{probabilistic choice}$$

$$\forall s, t, \alpha: \overline{\mathcal{P}}(s, \alpha, t) \leqslant 1$$

$$\forall s \in T: p_s = 1 \qquad \text{reach prob of } T$$

$$\forall s \notin T: \frac{\sum_{\alpha} \sigma^{s,\alpha} \cdot \sum_{t \in S} \overline{\mathcal{P}}(s, \alpha, t) \cdot p_t}{p_s} \leqslant 1 \qquad \text{transition probabilities}$$

GP for Two Objectives

Objectives: reach T with probability $\leqslant p$, expected cost to reach $G \leqslant c$

Subject to:

	$\frac{p_{s_I}}{p} \leqslant 1$	reachability
	$\frac{c_{s_I}}{c} \leqslant 1$	expected reward
$\forall s:$	$\sum_{\alpha \in Act(s)} \sigma^{s,\alpha} \leqslant 1$	randomised scheduler
$\forall s, \alpha:$	$\sigma^{s,\alpha} \leqslant 1$	
$\forall s, \alpha:$	$\sum_{t \in S} \overline{\mathcal{P}}(s,\alpha,t) \leqslant 1$	probabilistic choice
$\forall s, t, \alpha:$	$\overline{\mathcal{P}}(s,\alpha,t) \leqslant 1$	
$\forall s \in T:$	$p_s = 1$	reach prob of T
$\forall s \notin T:$	$\frac{\sum_\alpha \sigma^{s,\alpha} \cdot \sum_{t \in S} \overline{\mathcal{P}}(s,\alpha,t) \cdot p_t}{p_s} \leqslant 1$	transition probabilities
$\forall s \notin G:$	$\frac{\sum_\alpha \sigma^{s,\alpha} \cdot \left(c(s,\alpha) + \sum_{t \in S} \overline{\mathcal{P}}(s,\alpha,t) \cdot c_t\right)}{c_s} \leqslant 1$	expected costs

Correctness

Use the objective function F now[7]

$$\text{Minimise} \quad \sum_{p \in V} \frac{1}{p} + \sum_{p \in L} \frac{1}{\overline{p}} + \sum_{s,\alpha} \frac{1}{\sigma^{s,\alpha}}$$

yields that all variables p, $\overline{p}$ and $\sigma^{s,\alpha}$ are maximised.

Theorem: The GP with objective function F yields a feasible solution.

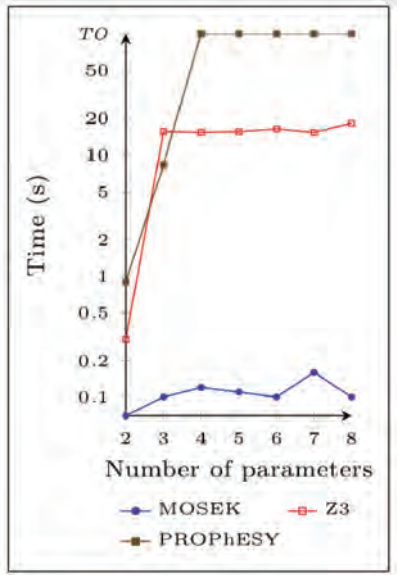
Solving this GP can be done in polynomial time.

[7] Note: the original objective function f is dropped.

Experimental Results

Benchmark	#states	#par	specs	MOSEK (s)		Z3
BRP (pMC)	5382	2	EC, $\mathbb{P}$, $*$	23.17	(6.48)	–
	112646	2	EC, $\mathbb{P}$, $*$	3541.59	(463.74)	–
	112646	4	EC, $\mathbb{P}$, $*$	4173.33	(568.79)	–
	5382	2	EC, $\mathbb{P}$	3.61		904.11
	112646	2	EC, $\mathbb{P}$	479.08		*TO*
NAND (pMC)	4122	2	EC, $\mathbb{P}$, $*$	14.67	(2.51)	–
	35122	2	EC, $\mathbb{P}$, $*$	1182.41	(95.19)	–
	4122	2	EC, $\mathbb{P}$	1.25		1.14
	35122	2	EC, $\mathbb{P}$	106.40		11.49
BRP (pMDP)	5466	2	EC, $\mathbb{P}$, $*$	31.04	(8.11)	–
	112846	2	EC, $\mathbb{P}$, $*$	4319.16	(512.20)	–
	5466	2	EC, $\mathbb{P}$	4.93		1174.20
	112846	2	EC, $\mathbb{P}$	711.50		*TO*
CONS (pMDP)	4112	2	EC, $\mathbb{P}$, $*$	102.93	(1.14)	–
	65552	2	EC, $\mathbb{P}$, $*$	*TO*		–
	4112	2	EC, $\mathbb{P}$	6.13		*TO*
	65552	2	EC, $\mathbb{P}$	1361.96		*TO*

Experimental Results



Epilogue

SMT-based approach:

- Exact
- Requires rational functions
- Fickle SMT performance
- $\approx 10^6$ states, 2 parameters
- Restricted to Markov chains
- CEGAR-like refinement

Epilogue

SMT-based approach:

- Exact
- Requires rational functions
- Fickle SMT performance
- $\approx 10^6$ states, 2 parameters
- Restricted to Markov chains
- CEGAR-like refinement

Parameter lifting approach:

- Approximative
- Off-the-shelf model checking
- No SMT, no rational functions
- $\approx 10^7$ states, 4–5 parameters
- Applicable to MDPs and games
- CEGAR-like refinement

Epilogue

SMT-based approach:

- Exact
- Requires rational functions
- Fickle SMT performance
- $\approx 10^6$ states, 2 parameters
- Restricted to Markov chains
- CEGAR-like refinement

Geometric programming approach:

- Numerical approximation
- Multiple objectives
- $\approx 10^5$ states, 10 parameters
- Applicable to MDPs
- Possibility of richer objectives

Epilogue

SMT-based approach:

- Exact
- Requires rational functions
- Fickle SMT performance
- $\approx 10^6$ states, 2 parameters
- Restricted to Markov chains
- CEGAR-like refinement

Geometric programming approach:

- Numerical approximation
- Multiple objectives
- $\approx 10^5$ states, 10 parameters
- Applicable to MDPs
- Possibility of richer objectives

Significant progress in the last couple of years.

Epilogue

SMT-based approach:

- Exact
- Requires rational functions
- Fickle SMT performance
- $\approx 10^6$ states, 2 parameters
- Restricted to Markov chains
- CEGAR-like refinement

Geometric programming approach:

- Numerical approximation
- Multiple objectives
- $\approx 10^5$ states, 10 parameters
- Applicable to MDPs
- Possibility of richer objectives

Significant progress in the last couple of years.

More info: QEST'14, CAV'15, ATVA'16, TACAS'17 and

`http://moves.rwth-aachen.de/research/tools/prophesy/`

Overview

Conclusion

Probabilistic Model Checking . . .

- . . . . . . is a mature automated technique
- . . . . . . focuses on quantitative measures
- . . . . . . has a broad range of applications
- . . . . . . such as performance analysis
- . . . . . . and reliability engineering

more information: `http://www.stormchecker.org`

Conclusion

Probabilistic Model Checking ...

- ...... is a mature automated technique
- ...... focuses on quantitative measures
- ...... has a broad range of applications
- ...... such as performance analysis
- ...... and reliability engineering

Current Research

- tight game-based abstractions
- parameter synthesis

more information: `http://www.stormchecker.org`

Conclusion

Probabilistic Model Checking . . .

- . . . . . . is a mature automated technique
- . . . . . . focuses on quantitative measures
- . . . . . . has a broad range of applications
- . . . . . . such as performance analysis
- . . . . . . and reliability engineering

Current Research

- tight game-based abstractions
- parameter synthesis

Big Thanks! To all my co-authors and co-workers

more information: `http://www.stormchecker.org`

Further reading

- C. Baier and JPK.
  Principles of Model Checking. MIT Press, 2008.
- JPK.
  The probabilistic model checking landscape. IEEE LICS, 2016.
- A. Abate, JPK, J. Lygeros, M. Prandini
  Approximate model checking of stochastic hybrid systems.
  Eur. J. on Control, 2010.
- C. Baier, B. Haverkort, H. Hermanns, JPK.
  Performance analysis and model checking join forces. Comm. of the ACM, 2010.
- M. Volk, S. Junges, JPK.
  Fast dynamic fault tree analysis by model checking.
  IEEE Trans. on Industrial Informatics, 2017.
- I. Tkachev, A. Mereacre, JPK, A. Abate.
  Quantitative model-checking of controlled discrete-time Markov processes.
  Information and Computation, 2017.