The What, Why, and How of Probabilistic Verification

Part 1: Motivation and Models

Joost-Pieter Katoen

RWTH AACHEN UNIVERSITY

UNIVERSITY OF TWENTE.

CAV Invited Tutorial 2015, San Francisco

Roadmap of This Tutorial

Part 1. Motivation and Models

- More Than 5 Reasons for Probabilistic Analysis
- Elementary Models and Properties

Part 2. Algorithmic Foundations

- Reachability and Beyond in Discrete Markov Models
- Timed Reachability in Continuous Markov Models

Part 3. Treating Gigantic Markov Models

- Abstraction: Precise, Aggressive, and Compositional

Part 4. Recent Research Developments

- Parameter Synthesis and Model Repair
- Counterexample Generation
- Probabilistic Programming

Overview

The Relevance of Probabilities

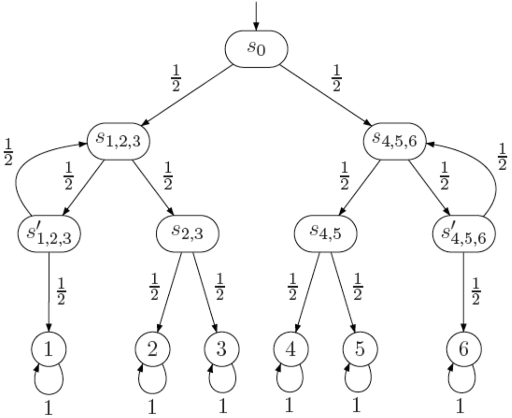
Markov Models and Properties

Overview

More Than Five Reasons for Probabilities

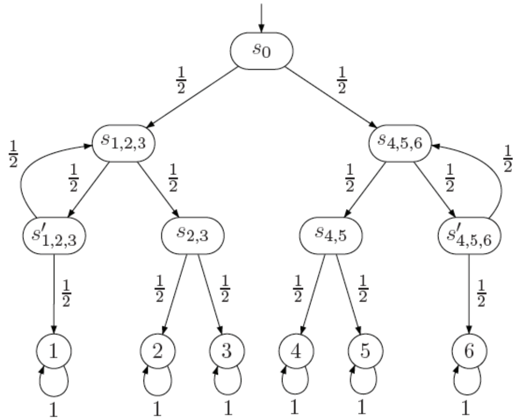
1. Randomised Algorithms
2. Reducing Complexity
3. Probabilistic Programming
4. Reliability
5. Performance
6. Optimization
7. Systems Biology

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



Heads = "go left"; tails = "go right".

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



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

Distributed Computing

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

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

Distributed Computing

FLP impossibility result [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.

Reducing Complexity: Matrix Multiplication [Freivald, 1977]

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

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

Reducing Complexity: Matrix Multiplication [Freivald, 1977]

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

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

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

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

Reducing Complexity: Matrix Multiplication [Freivald, 1977]

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

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

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

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

Randomised:

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

Reducing Complexity: Matrix Multiplication [Freivald, 1977]

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

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

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

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

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

Complexity: in $\mathcal{O}(k \cdot N^2)$, with false positive with probability $\leqslant 2^{-k}$

Probabilistic Programming

2013, DARPA launched a 48M (US dollar) program on

"Probabilistic Programming (PP) for Advanced Machine Learning (ML)"

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

Probabilistic Programming: Once Upon a Time ......

Duelling Cowboys

[McIver and Morgan, 2005]

```
int cowboyDuel(float a, b) { // 0 < a < 1, 0 < b < 1
   int t := A [] t := B; // decide cowboy for first shooting
       turn
   bool c := true;
   while (c) {
     if (t = A) {
         (c := false [a] t := B); // A shoots B with prob. a
     } else {
         (c := false [b] t := A); // B shoots A with prob. b
     }
   }
return t; // the survivor
}
```

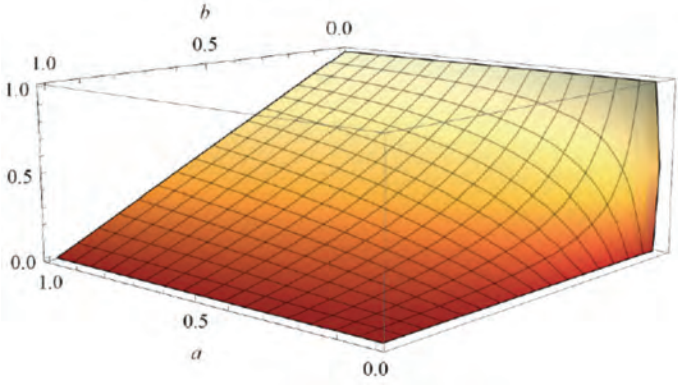
Duelling Cowboys

[McIver and Morgan, 2005]

```
int cowboyDuel(float a, b) { // 0 < a < 1, 0 < b < 1
   int t := A [] t := B; // decide cowboy for first shooting
       turn
   bool c := true;
   while (c) {
     if (t = A) {
         (c := false [a] t := B); // A shoots B with prob. a
     } else {
         (c := false [b] t := A); // B shoots A with prob. b
     }
   }
return t; // the survivor
}
```

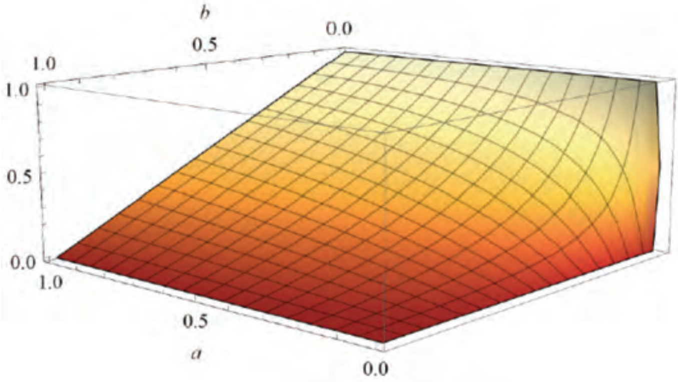
Claim: cowboy A wins the duel with probability at least $\frac{(1-b)\cdot a}{a+b-a\cdot b}$

Survivor Probability



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

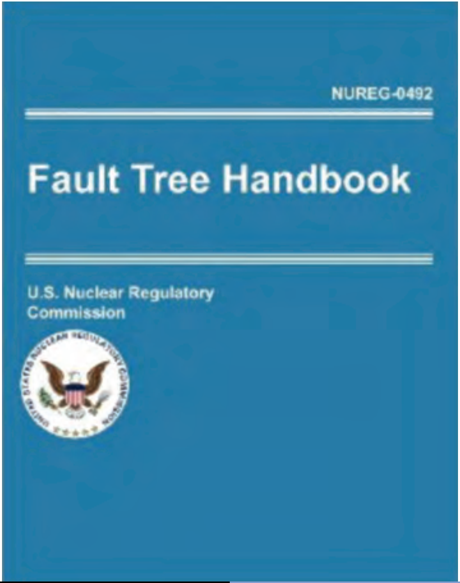
Survivor Probability



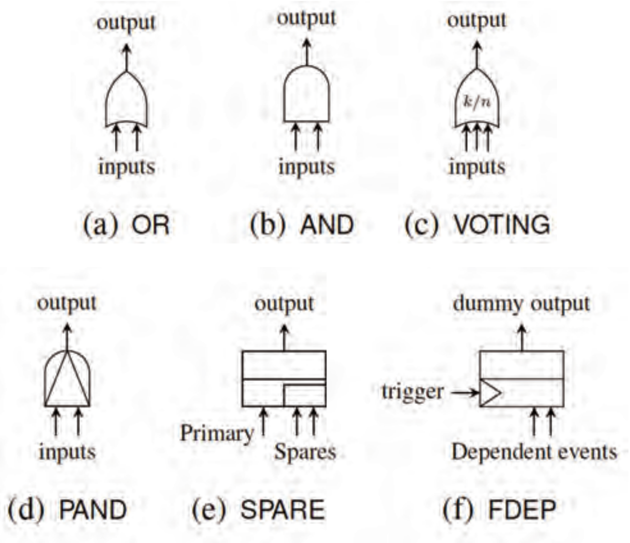
Claim: cowboy A wins the duel with probability at least $\frac{(1-b)\cdot a}{a+b-a\cdot b}$

Usage: security, machine learning, approximate computing . . . . . .

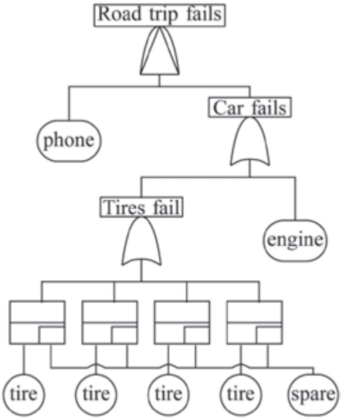
Reliability Engineering



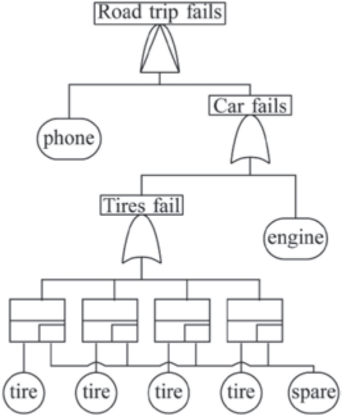
Reliability: (Dynamic) Fault Trees [Dugan *et al.*, 1990]



A Fault Tree Example



A Fault Tree Example

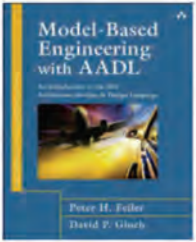


(D)FTs: one of —if not the— most prominent models for risk analysis

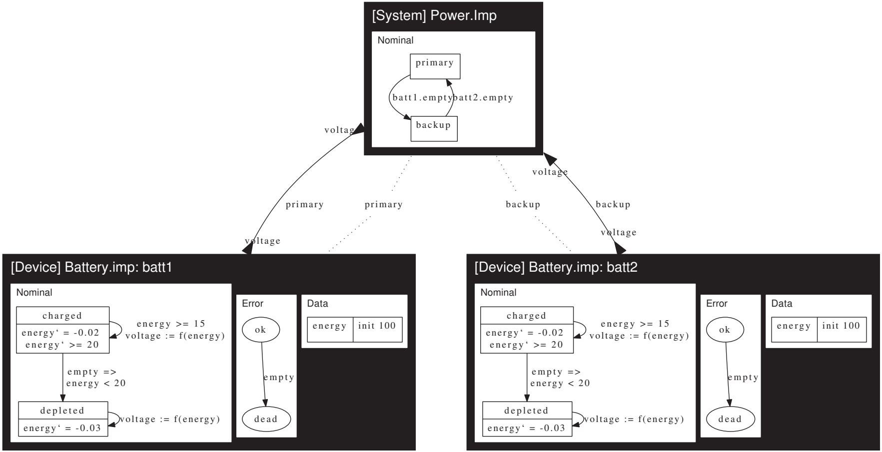
Aims: quantify system reliability and availability, MTTF, ...

Reliability: Architectural Languages [Feiler *et al.*, 2010]

1989 MetaH Language
1998 ......
2004 Architecture & Analysis Design Language 1.0 [AADL SAE Standard]
2006 Error Annex 1.0
2009 AADL 2.0 `www.aadl.info`
2010 Error Annex 2.0
2014 Error Annex 3.0



Reliability: Architectural Languages [Feiler *et al.*, 2010]



Reliability: Architectural Languages [Feiler *et al.*, 2010]

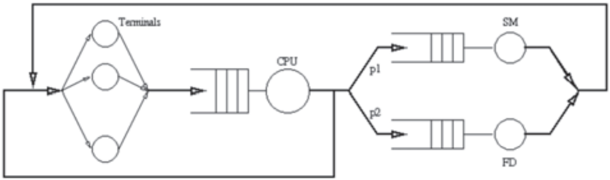
```
error model BatteryFailure
  features
    ok: initial state;
    dead: error state;
    batteryDied: out error propagation;
end BatteryFailure;

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

Fault injection

In error state `dead`, `voltage:=0`

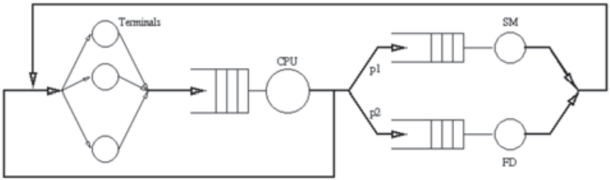
Performance: GSPNs [Ajmone Marsan *et al.*, 1984]

The early days:



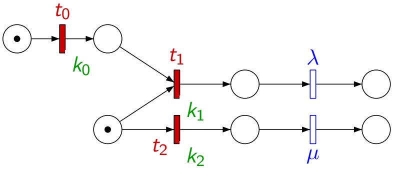
Performance: GSPNs [Ajmone Marsan *et al.*, 1984]

The early days:



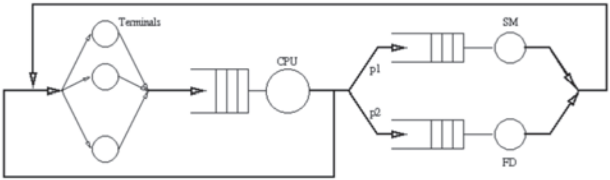
More modern times: Petri nets with

- Timed transitions
- Immediate transitions
- Natural weights



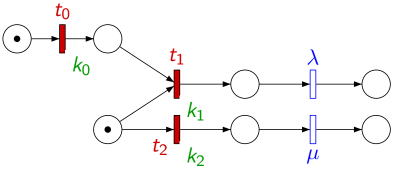
Performance: GSPNs [Ajmone Marsan *et al.*, 1984]

The early days:



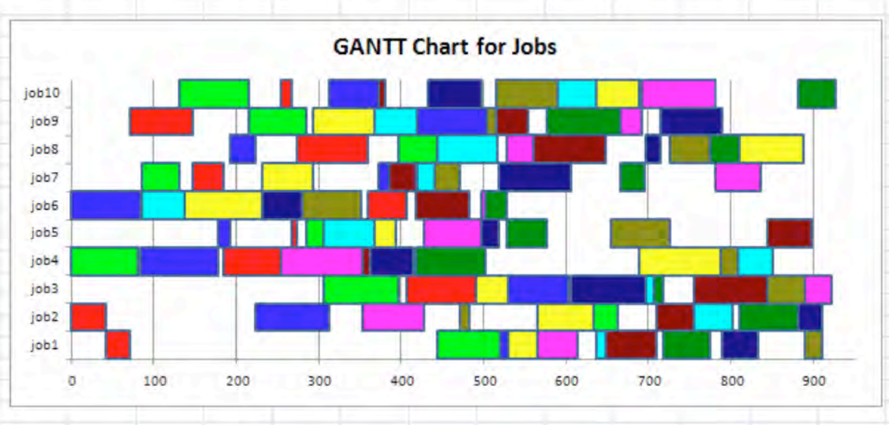
More modern times: Petri nets with

- Timed transitions
- Immediate transitions
- Natural weights



Aims: quantify arrivals, waiting times, QoS, soft deadlines, ...
GSPNs: very—if not the most—popular in performance modeling

Stochastic Scheduling



Encyclopedia of Optimization 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 co
terarrival and proces
arrangement of servic
jective to be optimiz
are required to be no
cannot make use of fu
known total duration
yet finished.
Regarding solutio
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

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

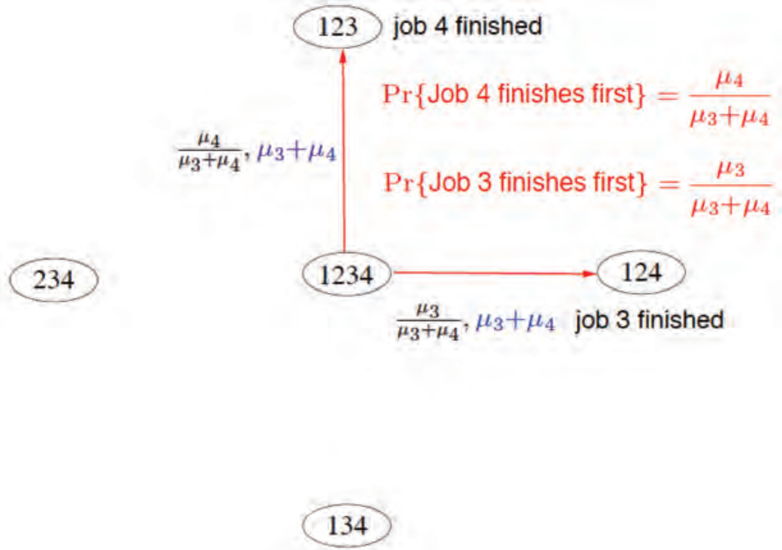
- Objective = minimal expected makespan—finishing time of last job

- SEPT policy yields minimal expected makespan (Bruno et al., JACM 1981)

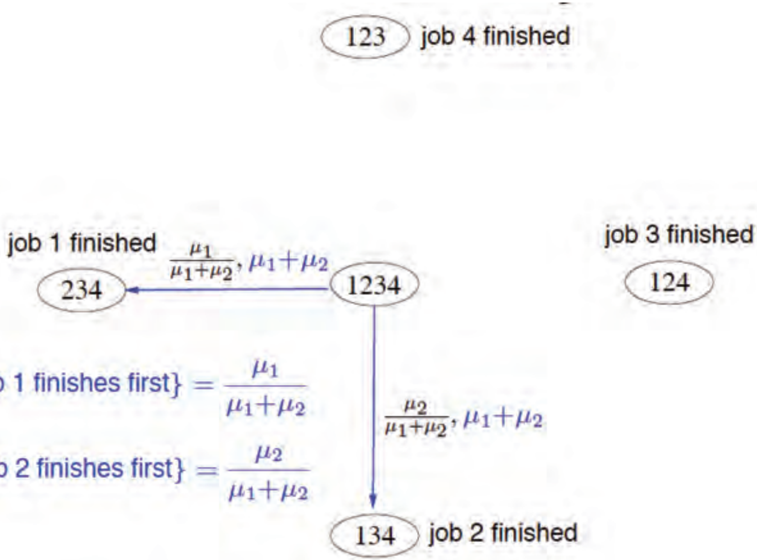
  "it is hard to calculate these expected values"

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

Stochastic Scheduling ($N = 4$; $M = 2$)



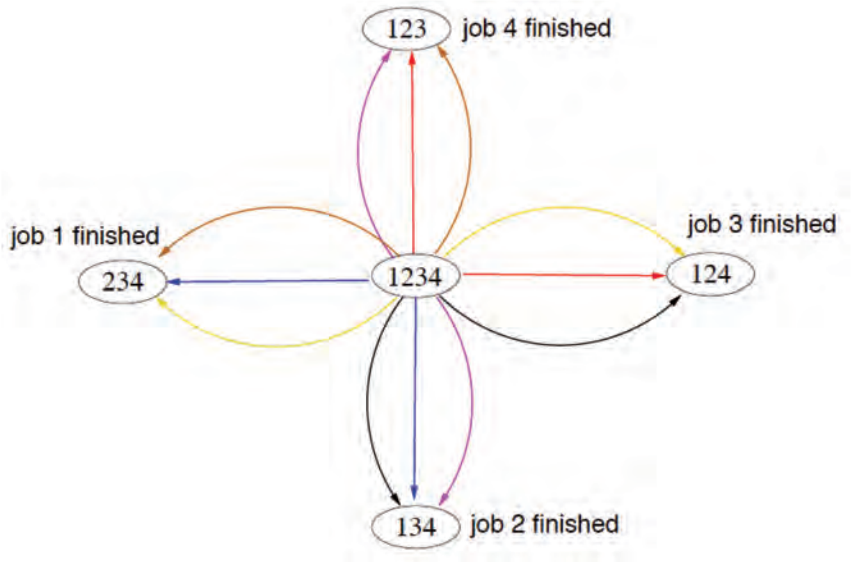
Stochastic Scheduling ($N = 4; M = 2$)



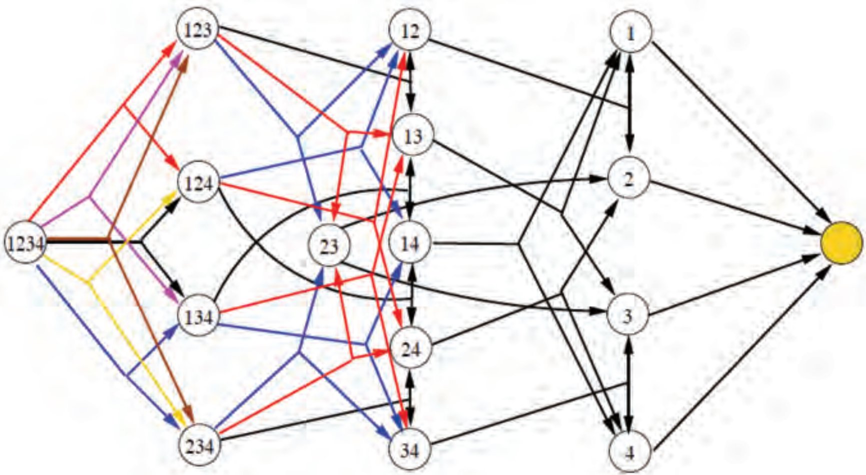
$$\Pr\{\text{Job 1 finishes first}\} = \frac{\mu_1}{\mu_1+\mu_2}$$

$$\Pr\{\text{Job 2 finishes first}\} = \frac{\mu_2}{\mu_1+\mu_2}$$

Stochastic Scheduling $(N = 4; M = 2)$



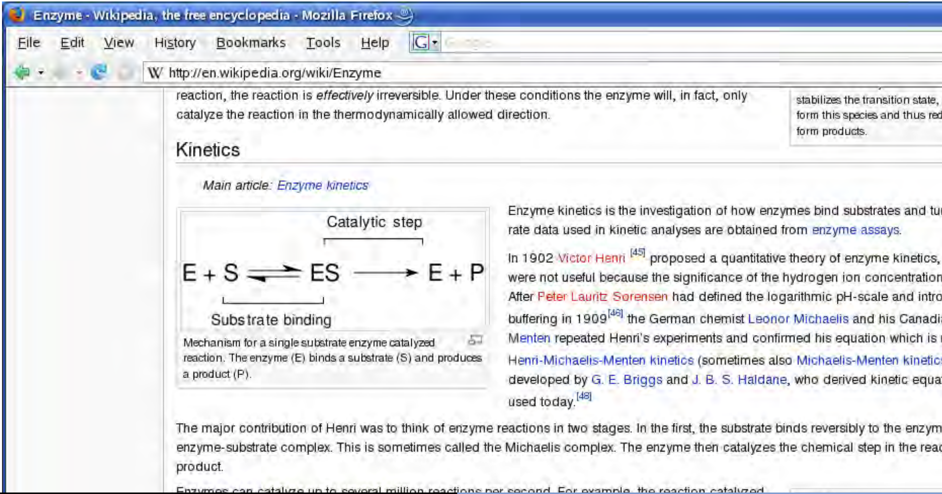
Stochastic Model



Systems Biology

Systems Biology

Enzyme-catalysed substrate conversion



Stochastic Chemical Kinetics

- Types of reaction described by stochiometric equations:

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

- 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

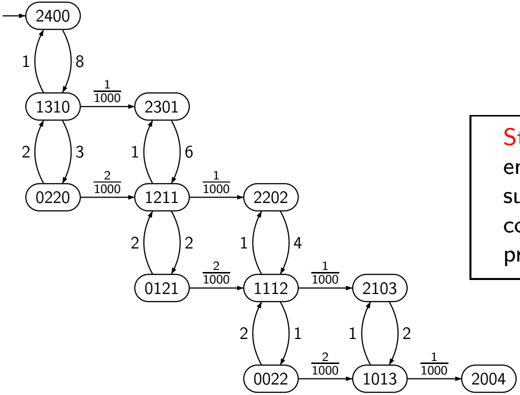
- Reaction probability within infinitesimal interval $[t, t{+}\Delta)$:

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

  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 $x_C > 0$

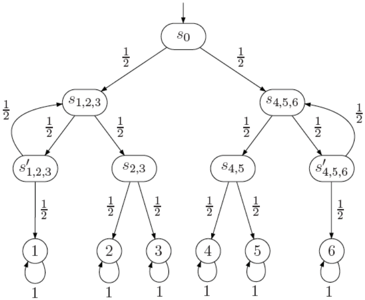
Overview

Common Feature

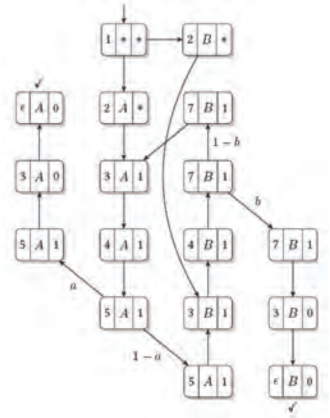
All these applications consider Markov models[1]

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

Discrete-Time Markov Models

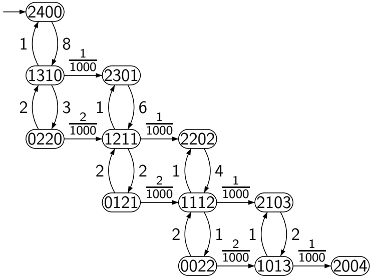


A Markov chain
for Knuth-Yao's algorithm

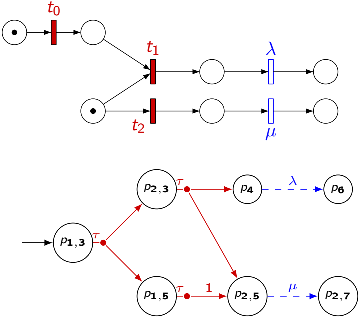


A Markov decision process
for the cowboy program

Continuous-Time Markov Models

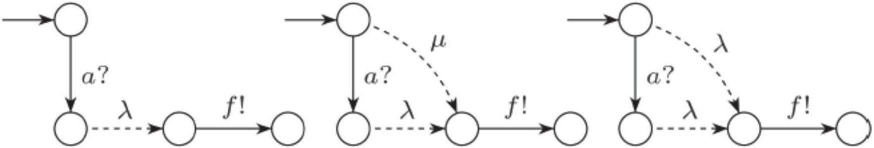


A Markov chain
for substrate conversion



A Markov decision process
for the GSPN

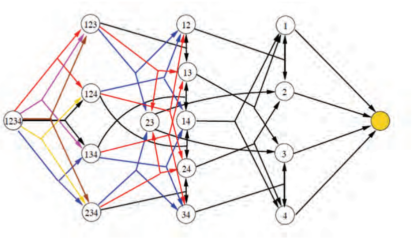
Fault Trees are Continuous-Time MDPs



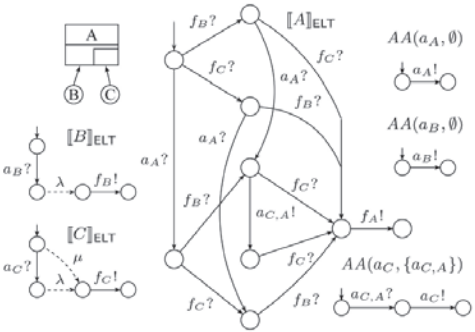
Markov models of a cold, warm and hot basic event

(dormancy factor $\mu = \alpha \cdot \lambda$)

Continuous-Time Markov Models



Markov decision process for stochastic scheduling



Markov decision process[a] for a SPARE gate

---

[a] In fact, an interactive Markov chain.

Markov Models

	Nondeterminism no	Nondeterminism yes
Discrete time	discrete-time Markov chain (DTMC)	Markov decision process (MDP)
Continuous time	CTMC	CTMDP

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

Properties

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

Properties

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

Core problem: computing (timed) reachability probabilities