

The What, Why, and How of Probabilistic Verification

Part 1: Motivation and Models

Joost-Pieter Katoen



UNIVERSITY OF TWENTE.

CAV Invited Tutorial 2015, San Francisco

Roadmap of This Tutorial

Part 1. Motivation and Models

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

Part 2. Algorithmic Foundations

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

Part 3. Treating Gigantic Markov Models

- Abstraction: Precise, Aggressive, and Compositional

Part 4. Recent Research Developments

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

The Relevance of Probabilities

Markov Models and Properties

Overview

The Relevance of Probabilities

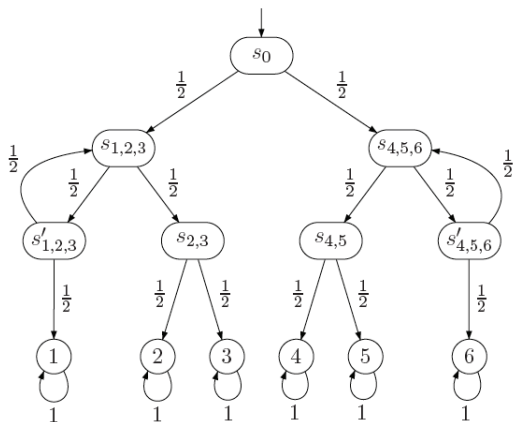
Markov Models and Properties

More Than Five Reasons for Probabilities



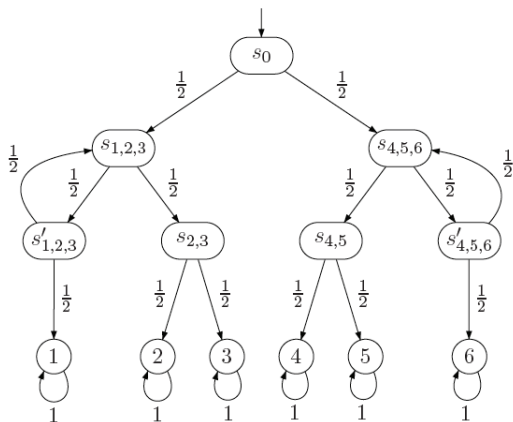
1. Randomised Algorithms
2. Reducing Complexity
3. Probabilistic Programming
4. Reliability
5. Performance
6. Optimization
7. Systems Biology

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 $\leq 2^{-k}$

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]

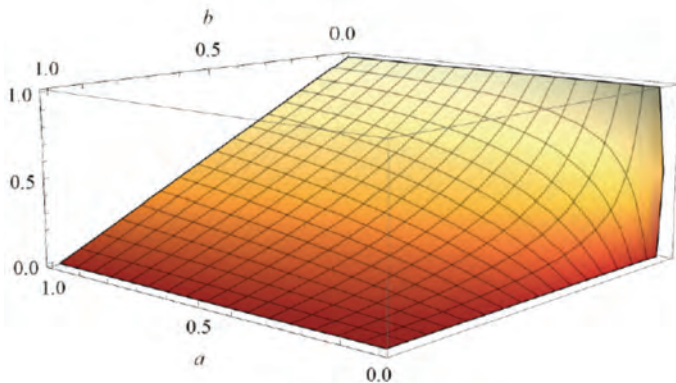
```

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
}

```

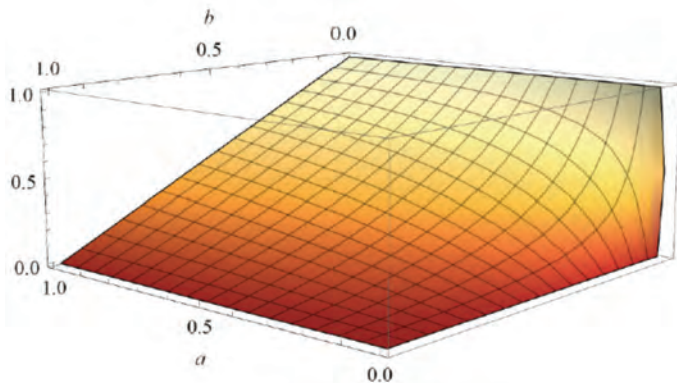
Claim: cowboy A wins the duel with probability at least $\frac{(1-b) \cdot a}{a+b-a \cdot b}$

Survivor Probability



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

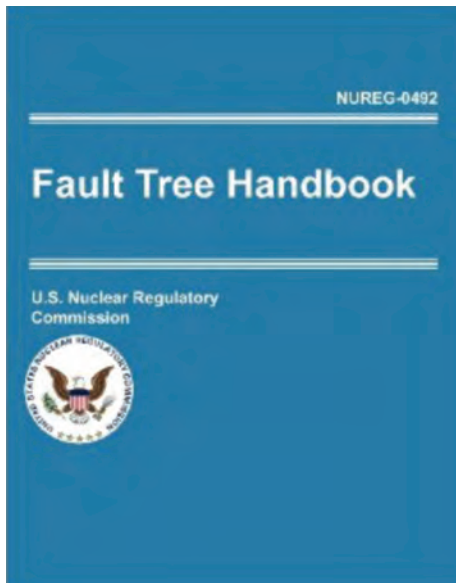
Survivor Probability



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) OR



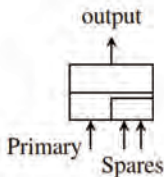
(b) AND



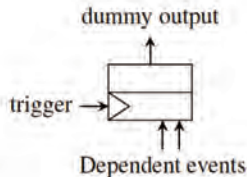
(c) VOTING



(d) PAND

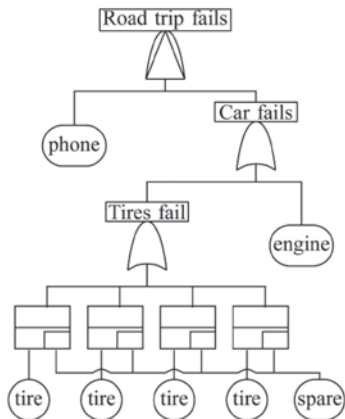


(e) SPARE

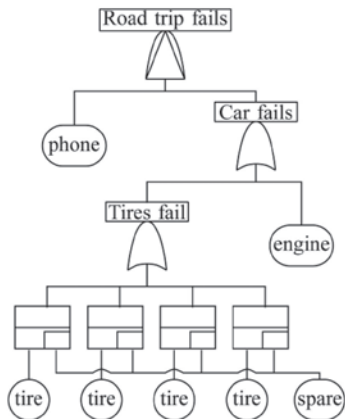


(f) FDEP

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**

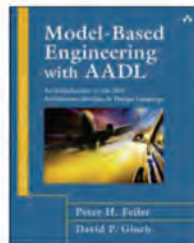
2006 Error Annex 1.0

2009 AADL 2.0

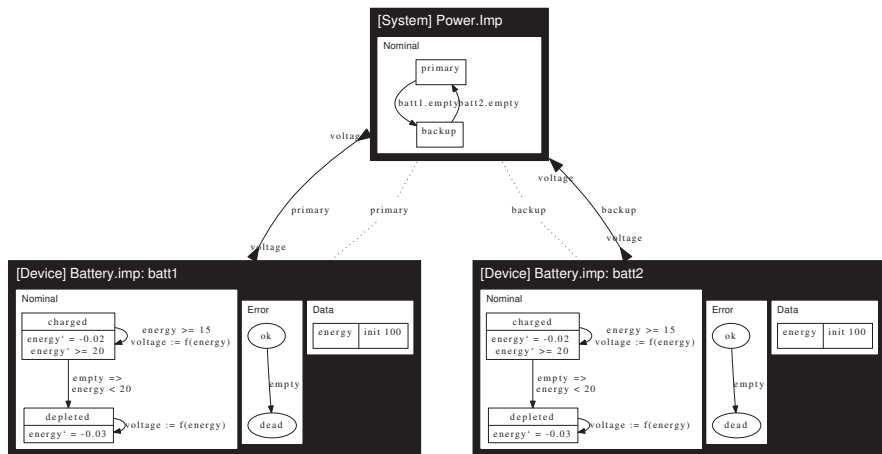
2010 Error Annex 2.0

2014 Error Annex 3.0

[AADL SAE Standard]

www.aadl.info

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;
```

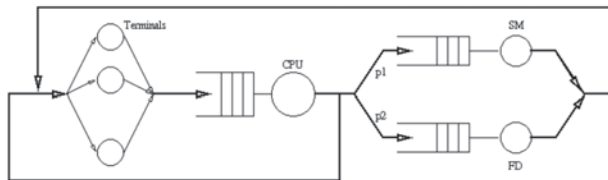
Fault injection

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

Performance: GSPNs

[Ajmone Marsan *et al.*, 1984]

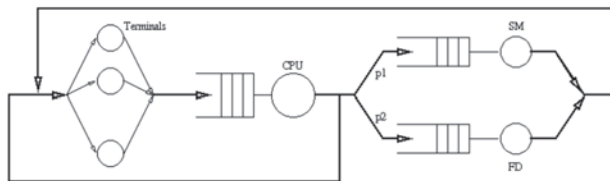
The early days:



Performance: GSPNs

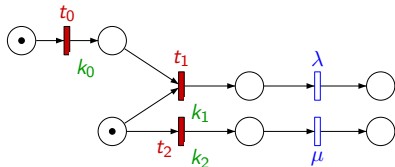
[Ajmone Marsan *et al.*, 1984]

The early days:



More modern times: Petri nets with

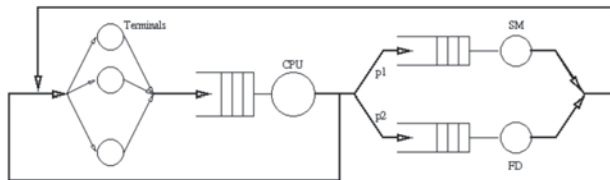
- ▶ Timed transitions
- ▶ Immediate transitions
- ▶ Natural weights



Performance: GSPNs

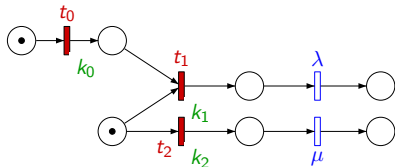
[Ajmone Marsan *et al.*, 1984]

The early days:



More modern times: Petri nets with

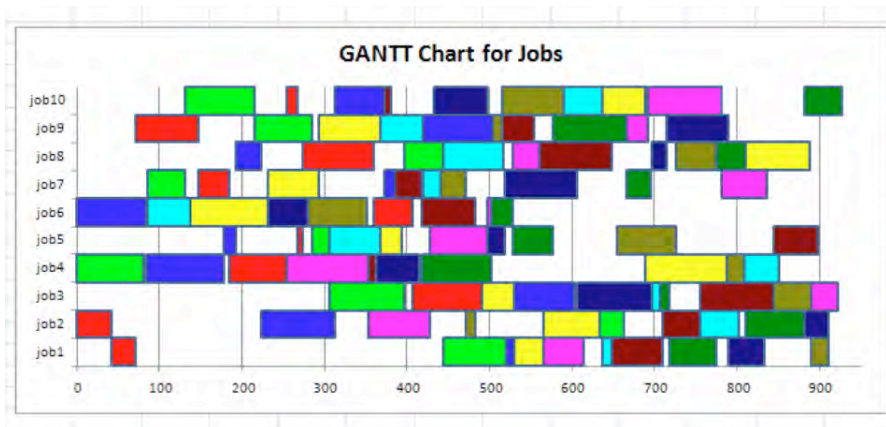
- ▶ Timed transitions
- ▶ Immediate transitions
- ▶ Natural weights



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

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

Stochastic Scheduling



Stochastic Scheduling

JOSÉ NIÑO-MORA

Department of Statistics,

Universidad Carlos III de Madrid, Getafe, Spain

MSC2000: 90B36

Article Outline

[Introduction](#)

[Models](#)

[Scheduling a Batch of Stochastic Jobs](#)

[Multi-Armed Bandits](#)

[Scheduling Queueing Systems](#)

[References](#)

Introduction

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

optimal performance.

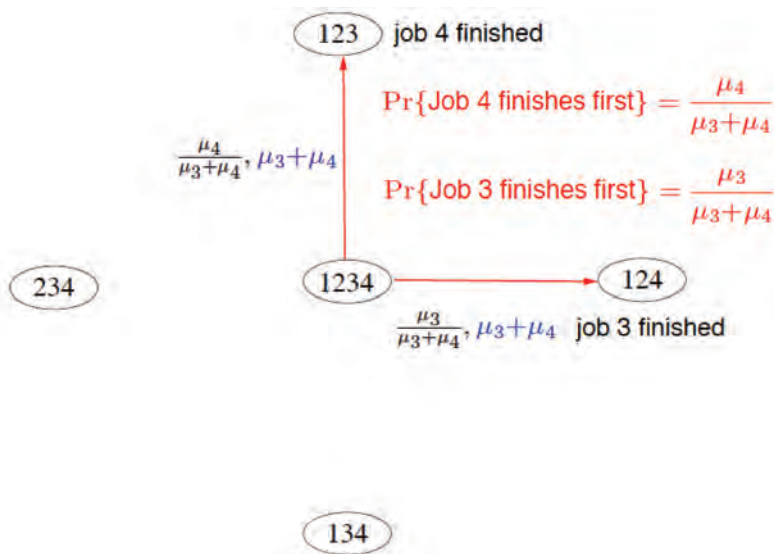
The theory of stochastic scheduling is a goal in the idealized models. Real-world random arrivals or processing times have their probability distributions that vary across several different scheduling policies. Consider the case of a scheduling policy that is based on the order of arrival and processing times. The objective is to be optimized. In practice, scheduling policies are required to be non-preemptive. In other words, once a job cannot make use of the processor, it must wait until its known total duration has elapsed before it is yet finished.

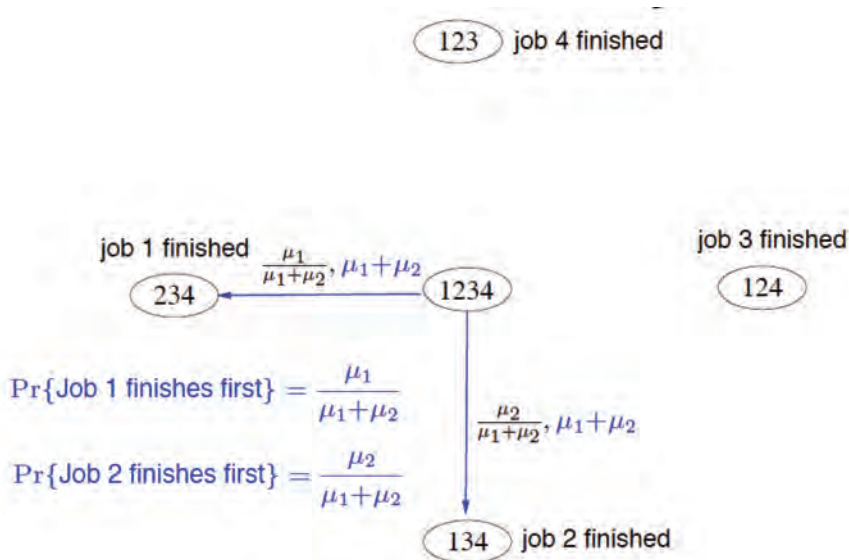
Regarding stochastic scheduling, it seems fair to say that the field is yet available to design optimal policies across a wide range of stochastic scheduling models. The problem can be cast in the framework of a scheduling model that is not straightforward.

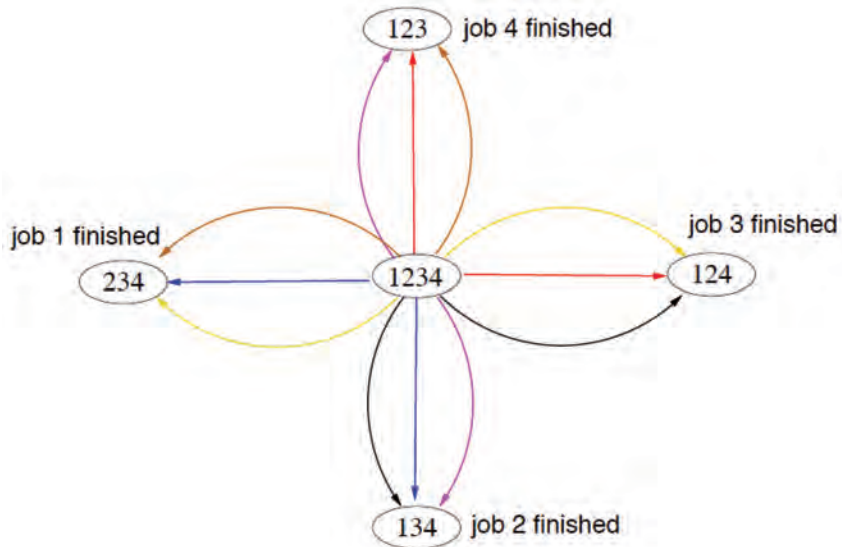
Stochastic Scheduling

- ▶ 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”

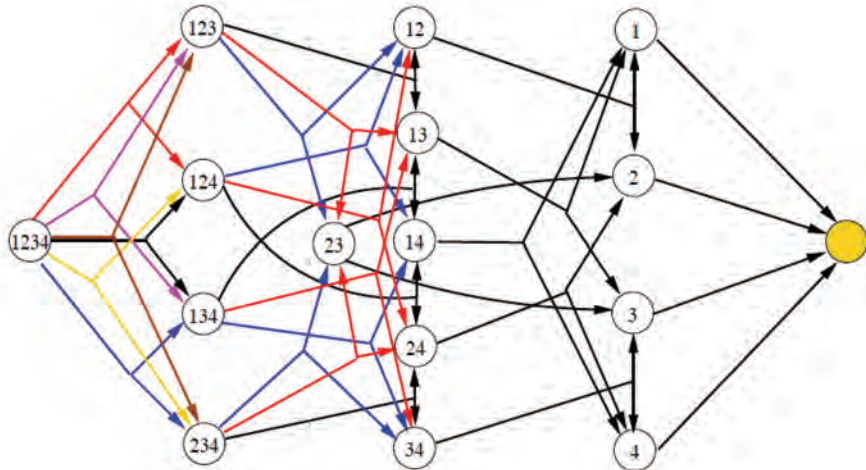
Which policy maximises the probability to finish all jobs on time?

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

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

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

Stochastic Model



Systems Biology



Enzyme-catalysed substrate conversion

Enzyme · Wikipedia, the free encyclopedia - Mozilla Firefox

File Edit View History Bookmarks Tools Help

W 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 reform products.

Kinetics

Main article: *Enzyme kinetics*

Catalytic step

$$E + S \rightleftharpoons ES \longrightarrow E + P$$

Substrate binding

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 rate data used in kinetic analyses are obtained from *enzyme assays*.

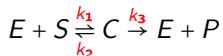
In 1902 **Victor Henri**^[45] proposed a quantitative theory of enzyme kinetics, were not useful because the significance of the hydrogen ion concentration. After **Peter Lauritz Sorensen** had defined the logarithmic pH-scale and introduced buffering in 1909^[46] the German chemist **Leonor Michaelis** and his Canadian colleague **Menten** repeated Henri's experiments and confirmed his equation which is now known as **Henri-Michaelis-Menten kinetics** (sometimes also **Michaelis-Menten kinetics**), developed by **G. E. Briggs** and **J. B. S. Haldane**, who derived kinetic equations 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 enzyme to form an enzyme-substrate complex. This is sometimes called the Michaelis complex. The enzyme then catalyzes the chemical step in the reaction to form the product.

Enzymes can catalyze up to several million reactions per second. For example, the reaction catalyzed

Stochastic Chemical Kinetics

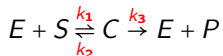
- ▶ Types of reaction described by **stoichiometric equations**:



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

Stochastic Chemical Kinetics

- Types of reaction described by **stoichiometric equations**:

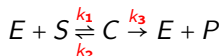


- N different types of molecules that **randomly collide**
where state $X(t) = (x_1, \dots, x_N)$ with $x_i = \#$ molecules of sort i
- Reaction probability** within infinitesimal interval $[t, t+\Delta)$:

$$\alpha_m(\vec{x}) \cdot \Delta = \Pr\{\text{reaction } m \text{ in } [t, t+\Delta) \mid X(t) = \vec{x}\}$$
 where $\alpha_m(\vec{x}) = k_m \cdot \#$ possible combinations of reactant molecules in \vec{x}

Stochastic Chemical Kinetics

- Types of reaction described by **stoichiometric equations**:



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

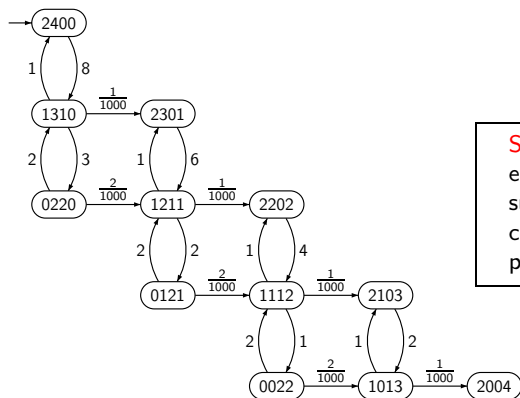
- Reaction probability** within infinitesimal interval $[t, t+\Delta)$:

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

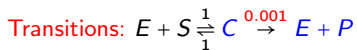
where $\alpha_m(\vec{x}) = k_m \cdot \#$ possible combinations of reactant molecules in \vec{x}

- Process has the **Markov property** and is **time-homogeneous**

Substrate Conversion in the Small



States:	<i>init</i>	<i>goal</i>
enzymes	2	2
substrates	4	0
complex	0	0
products	0	4



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

Overview

The Relevance of Probabilities

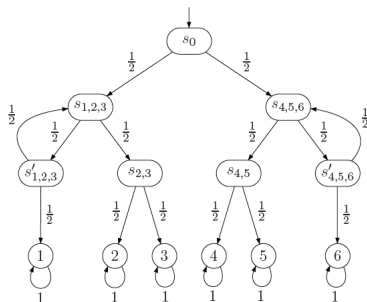
Markov Models and Properties

Common Feature

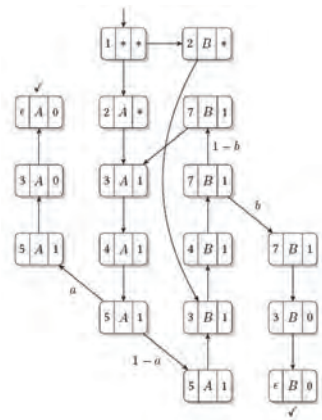
All these applications consider **Markov** models¹

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

Discrete-Time Markov Models

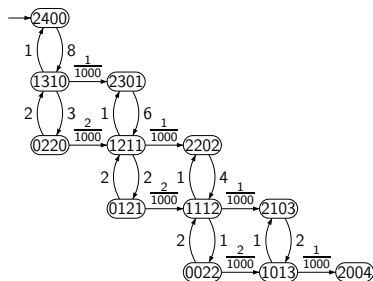


A Markov **chain**
for Knuth-Yao's algorithm

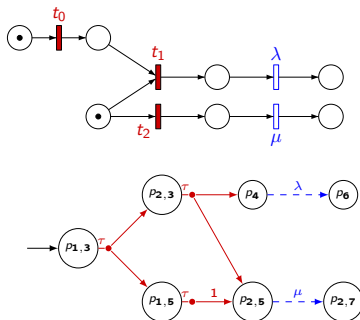


A Markov **decision process**
for the cowboy program

Continuous-Time Markov Models

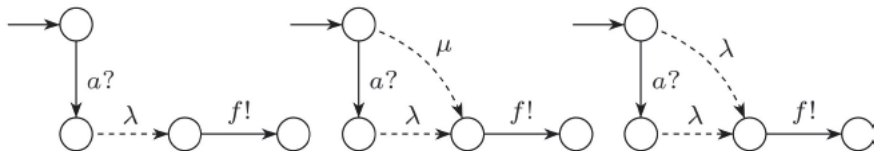


A Markov **chain**
for substrate conversion



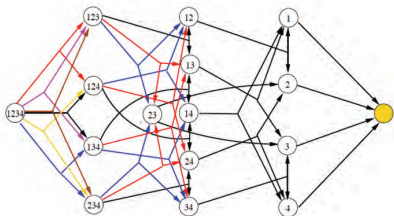
A Markov **decision** process
for the GSPN

Fault Trees are Continuous-Time MDPs

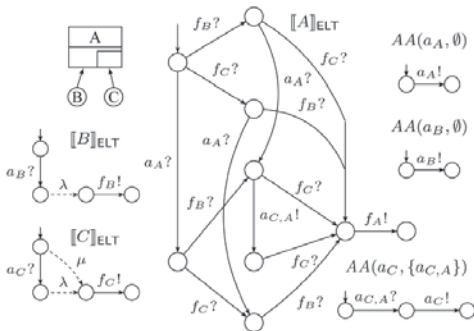


Markov models of a cold, warm and hot basic event
(dormancy factor $\mu = \alpha \cdot \lambda$)

Continuous-Time Markov Models



Markov **decision** process for stochastic scheduling



Markov **decision** process^a for a SPARE gate

^aIn 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