Model Checking:
One Can Do Much More Than You Think!

Joost-Pieter Katoen

Software Modeling and Verification Group

RWTH Aachen University, Germany

Schloss Dagstuhl, Meeting Research Training Groups, June 17, 2014

The quest for correctness

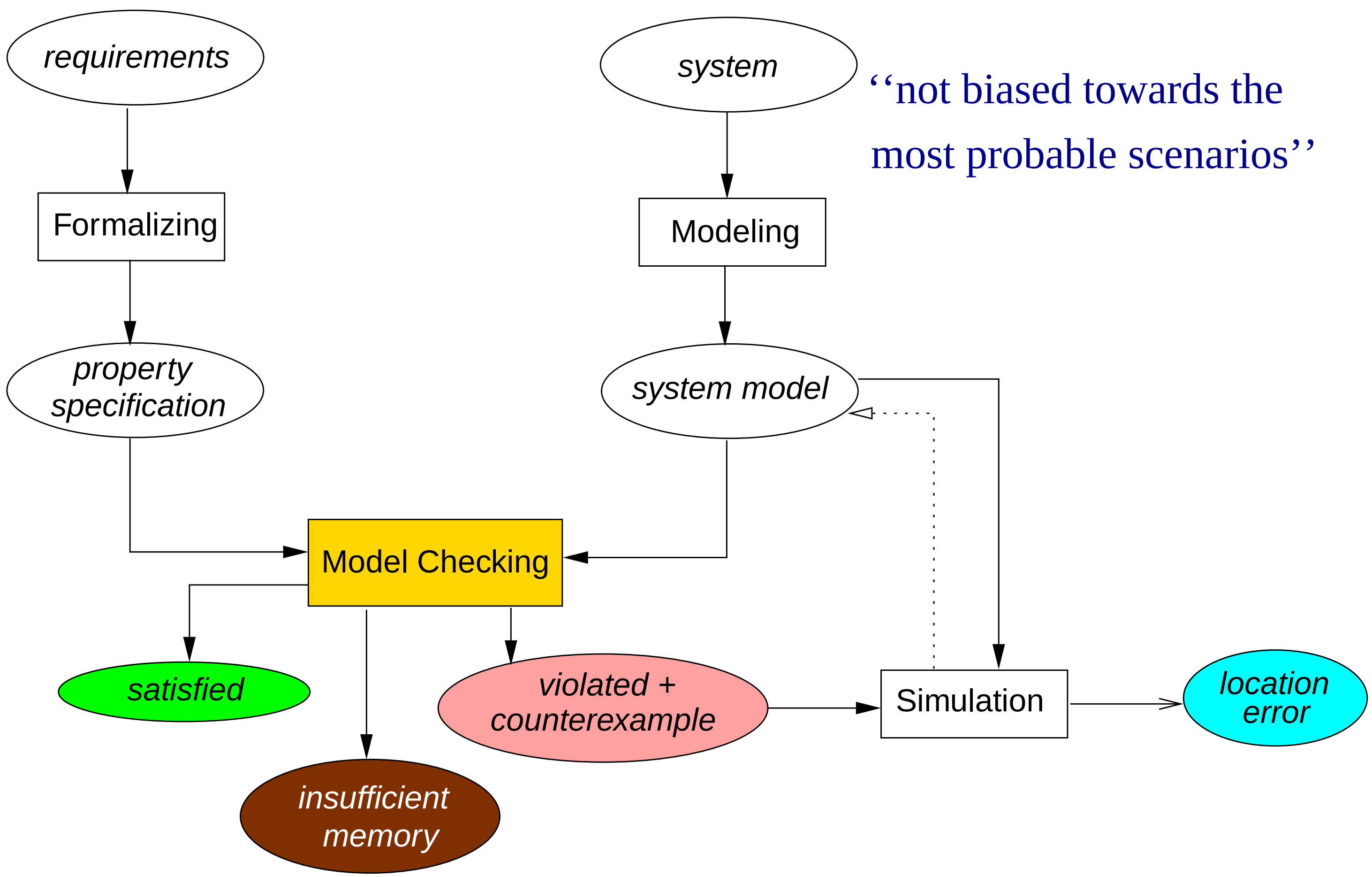
It is fair to state, that in this digital era correct systems for information processing are more valuable than gold.

Henk Barendregt (1996)

The search for ensuring correctness

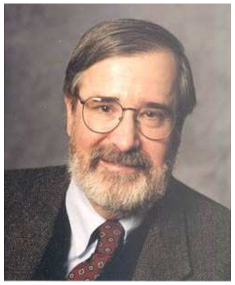
- Mathematical approach towards program correctness (Turing, 1949)
- Syntax-based technique for sequential programs (Hoare, 1969)
  - for a given input, does a computer program generate the correct output?
  - based on compositional proof rules expressed in predicate logic
- Syntax-based technique for concurrent programs (Pnueli, 1977)
  - can handle properties referring to situations during the computation
  - based on proof rules expressed in temporal logic
- Automated verification of concurrent programs (Emerson & Clarke, 1981)
  - model-based instead of proof-rule based approach
  - does the concurrent program satisfy a given (logical) property?

Model checking overview



Paris Kanellakis Theory and Practice Award 1998


Randal Bryant


Edmund Clarke

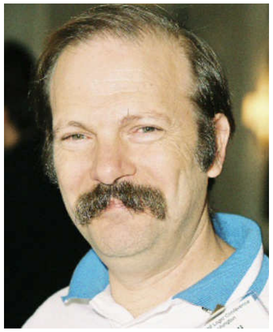

E. Allen Emerson


Ken McMillan

For their invention of "symbolic model checking," a method of formally checking system designs, which is widely used in the computer hardware industry and starts to show significant promise also in software verification and other areas.

Some other winners: Rivest et al., Paige and Tarjan, Buchberger, . . .

Gödel Prize 2000



Moshe Vardi

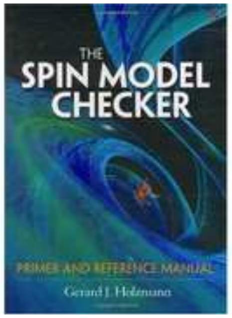
Pierre Wolper

"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

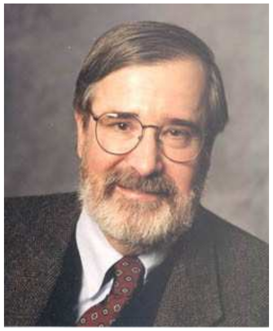


SPIN book

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.

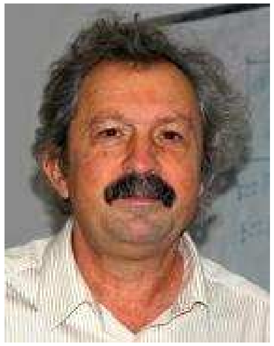
Some other winners: TeX, Postscript, UNIX, TCP/IP, Java, Smalltalk, . . .

ACM Turing Award 2007



Edmund Clarke

E. Allen Emerson



Joseph Sifakis

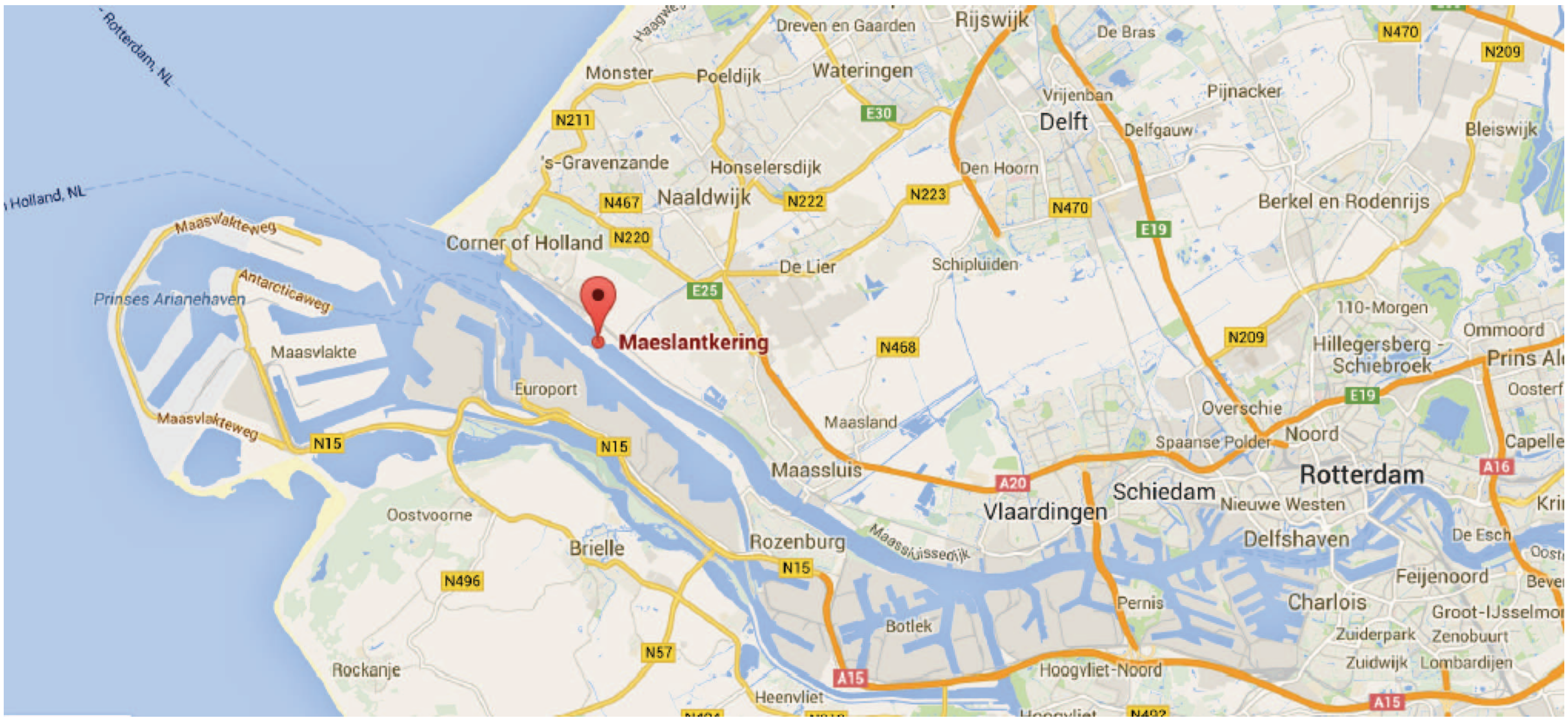
"For their role in developing Model-Checking into a highly effective verification technology, widely adopted in the hardware and software industries."

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

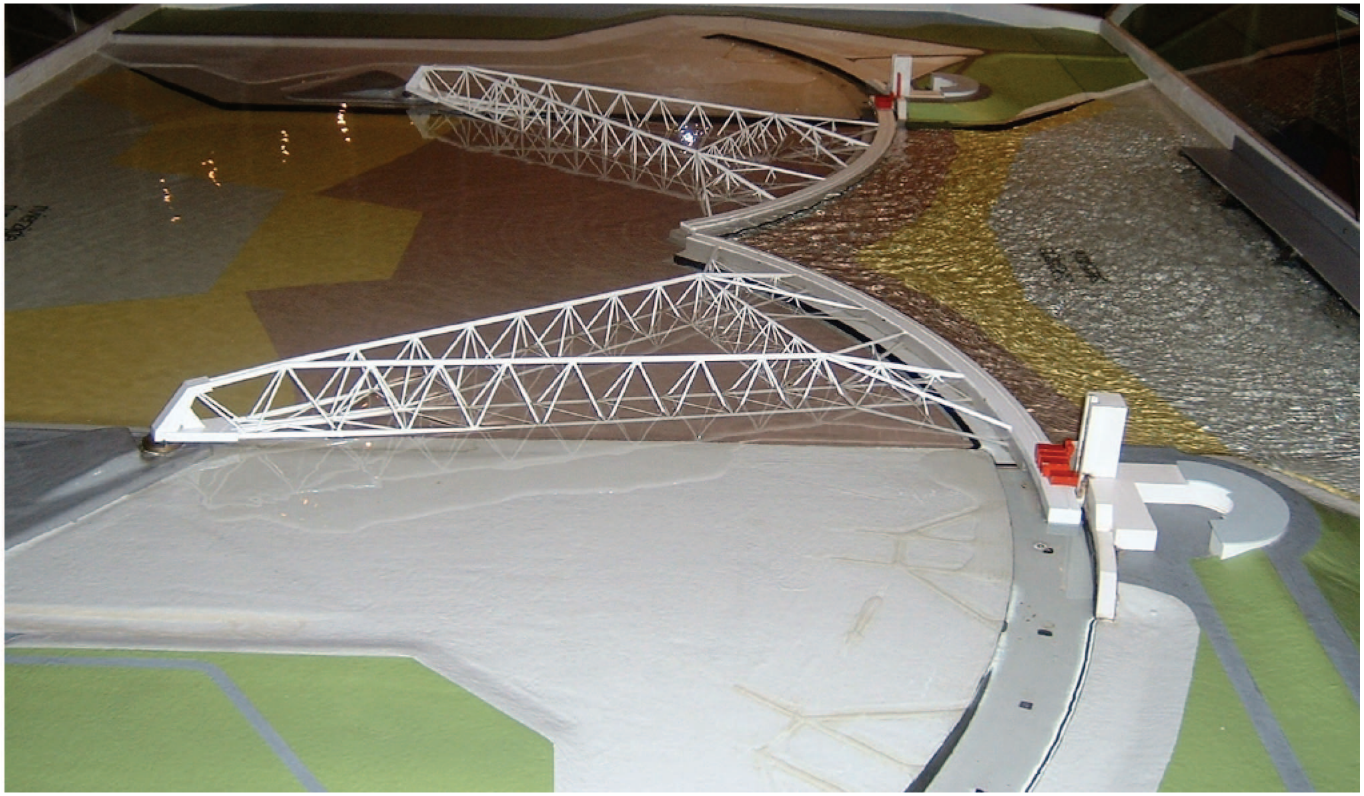
Striking practical examples

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

Storm surge barrier Maeslantkering

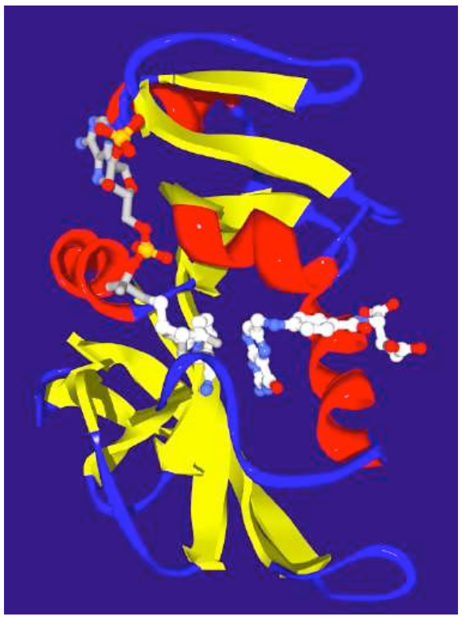


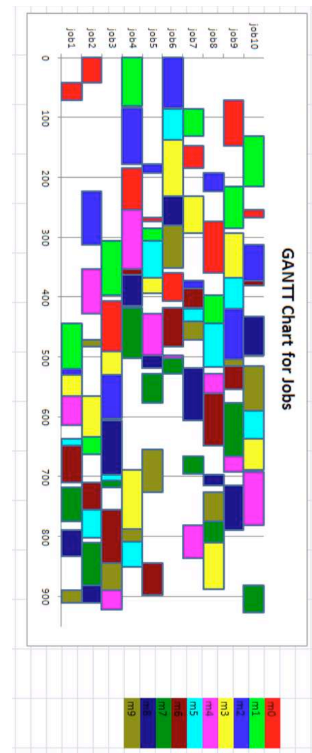
Storm surge barrier Maeslantkering



Storm surge barrier Maeslantkering

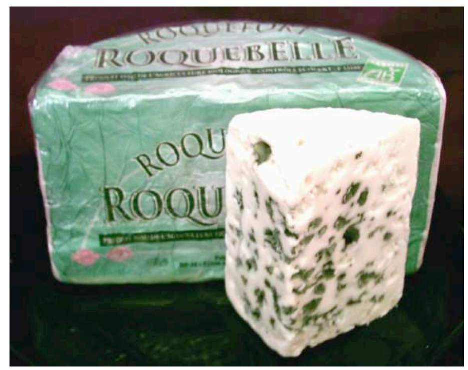
Towards ubiquitous model checking?!





Systems biology

Enzymes are omnipresent



Enzyme-catalysed substrate conversion

Enzyme - Wikipedia, the free encyclopedia - Mozilla Firefox

File Edit View History Bookmarks Tools Help

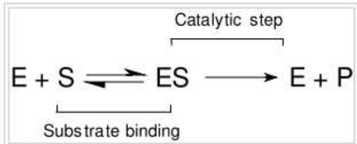
http://en.wikipedia.org/wiki/Enzyme

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, reducing the energy needed to form this species and thus reducing the energy required to form products.

Kinetics

Main article: Enzyme kinetics

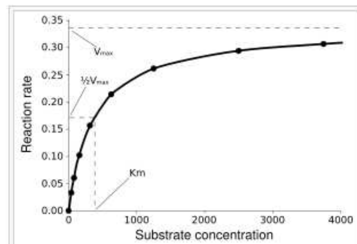


Mechanism for a single substrate enzyme catalyzed reaction. The enzyme (E) binds a substrate (S) and produces a product (P).

Enzyme kinetics is the investigation of how enzymes bind substrates and turn them into products. The rate data used in kinetic analyses are obtained from enzyme assays.

In 1902 Victor Henri [45] proposed a quantitative theory of enzyme kinetics, but his experimental data were not useful because the significance of the hydrogen ion concentration was not yet appreciated. After Peter Lauritz Sørensen had defined the logarithmic pH-scale and introduced the concept of buffering in 1909[46] the German chemist Leonor Michaelis and his Canadian postdoc Maud Leonora Menten repeated Henri's experiments and confirmed his equation which is referred to as Henri-Michaelis-Menten kinetics (sometimes also Michaelis-Menten kinetics).[47] Their work was further developed by G. E. Briggs and J. B. S. Haldane, who derived kinetic equations that are still widely 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, forming the enzyme-substrate complex. This is sometimes called the Michaelis complex. The enzyme then catalyzes the chemical step in the reaction and releases the product.



Saturation curve for an enzyme reaction showing the relation between the substrate concentration (S) and rate (v).

Enzymes can catalyze up to several million reactions per second. For example, the reaction catalyzed by orotidine 5'-phosphate decarboxylase will consume half of its substrate in 78 million years if no enzyme is present. However, when the decarboxylase is added, the same process takes just 25 milliseconds.[49] Enzyme rates depend on solution conditions and substrate concentration. Conditions that denature the protein abolish enzyme activity, such as high temperatures, extremes of pH or high salt concentrations, while raising substrate concentration tends to increase activity. To find the maximum speed of an enzymatic reaction, the substrate concentration is increased until a constant rate of product formation is seen. This is shown in the saturation curve on the right. Saturation happens because, as substrate concentration increases, more and more of the free enzyme is converted into the substrate-bound ES form. At the maximum velocity (V_{max}) of the enzyme, all the enzyme active sites are bound to substrate, and the amount of ES complex is the same as the total amount of enzyme. However, V_{max} is only one kinetic constant of enzymes. The amount of substrate needed to achieve a given rate of reaction is also important. This is given by the Michaelis-Menten constant (K_m), which is the substrate concentration required for an enzyme to reach one-half its maximum velocity. Each enzyme has a characteristic K_m for a given substrate, and this can show how tight the binding of the substrate is to the enzyme. Another useful constant is

Done

Stochastic chemical kinetics

- Types of reaction described by stochiometric equations:

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

- N different types of molecules that randomly collide

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

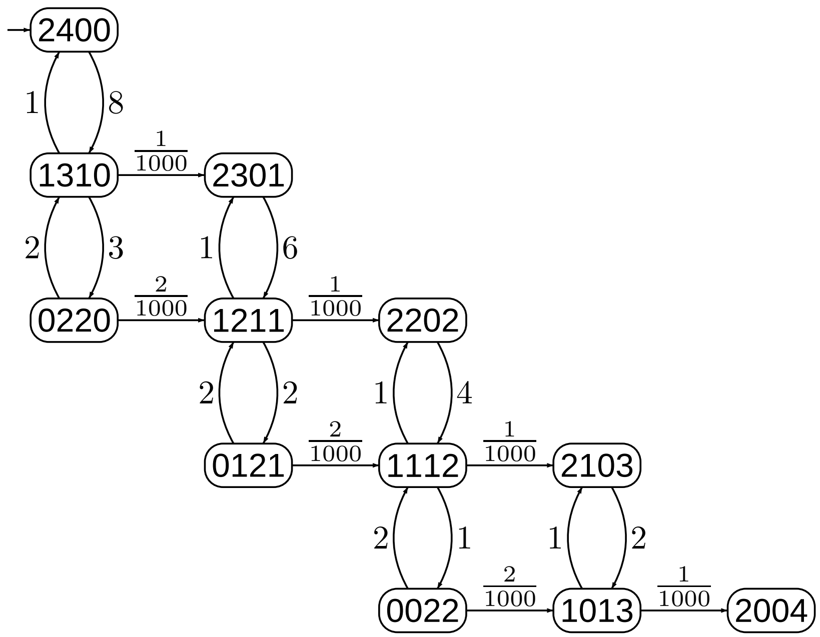
- Reaction probability within infinitesimal interval $[t, t{+}\Delta)$:

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

  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

Modeling as a continuous-time Markov chain



States:	*init*	*goal*
enzymes	2	2
substrates	4	0
complex	0	0
products	0	4

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

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

Solving the Markov chain

1. Use the chemical master equation for $p(\vec{x},t) = \Pr\{X(t) = \vec{x}\}$:

$$\dot{p}(\vec{x},t) = \underbrace{\sum_{\text{reaction } m} \alpha_m(\vec{y})\cdot p(\vec{y},t)}_{\text{prob. to reach } \vec{x} \text{ from another state}} - \underbrace{\sum_{\text{reaction } m} \alpha_m(\vec{x})\cdot p(\vec{x},t)}_{\text{prob. to leave state } \vec{x}}$$

$\Rightarrow$ Limitations: curse of dimensionality

Solving the Markov chain

1. Use the chemical master equation for $p(\vec{x}, t) = \Pr\{X(t) = \vec{x}\}$:

$$\dot{p}(\vec{x}, t) \;=\; \sum_{\text{reaction } m} \alpha_m \cdot (\vec{y}) \cdot p(\vec{y}, t) - \alpha_m(\vec{x}) \cdot p(\vec{x}, t)$$

$\Rightarrow$ Limitations: curse of dimensionality, stiffness

2. Apply Monte carlo simulation

   generate random runs, and estimate expectations/variances of populations

$\Rightarrow$ Limitations: impractical for estimating distributions

- $\approx 20 \cdot 10^6$ runs needed for precision $\epsilon = 10^{-5}$

Solving the Markov chain

1. Use the chemical master equation for $p(\vec{x}, t) = \Pr\{X(t) = \vec{x}\}$:

$$\dot{p}(\vec{x}, t) \;=\; \sum_{\text{reaction } m} \alpha_m \cdot (\vec{x}) \cdot p(\vec{x}, t) - \alpha_m(\vec{x}) \cdot p(\vec{x}, t)$$

$\Rightarrow$ Limitations: curse of dimensionality, stiffness

2. Apply Monte carlo simulation

   generate random runs, and estimate expectations/variances of populations

$\Rightarrow$ Limitations: impractical for estimating distributions

- $\approx 20 \cdot 10^6$ runs needed for precision $\epsilon = 10^{-5}$

our solution: apply model checking

Reachability probabilities (Baier, Katoen & Hermanns 1999)

The reachability problem:

Input: a continuous-time Markov chain, a target state, and deadline $d \in \mathbb{R}$

Output: an ϵ-approximation of the probability to reach the target in d time

is efficiently computable.

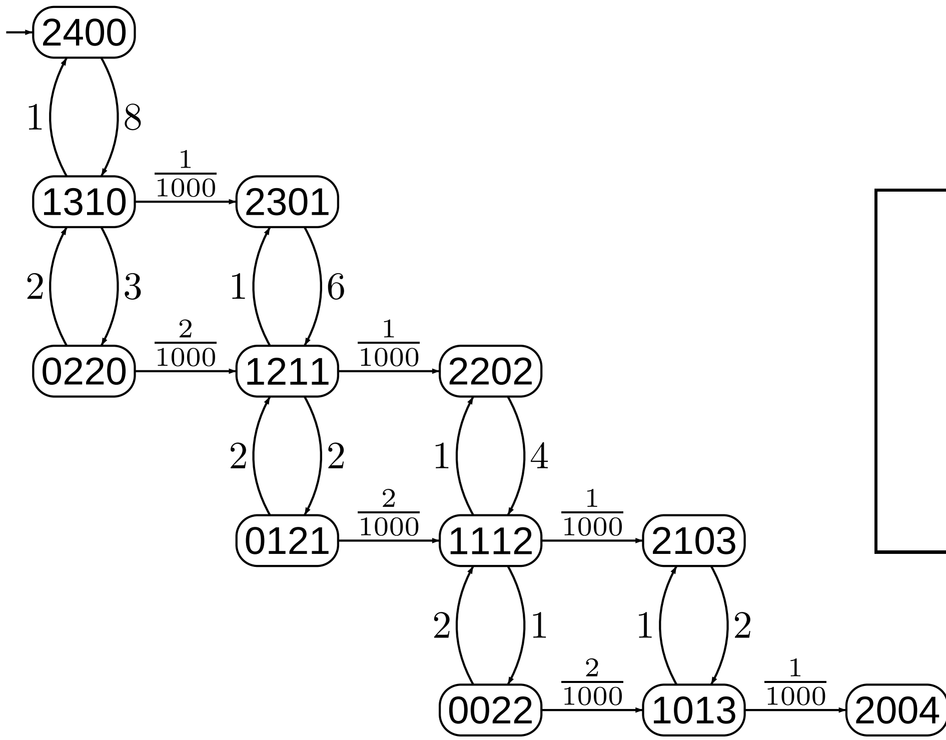
Reachability probabilities

- State s, set G of goal states, and deadline d
- $\Pr(s \models \Diamond^{\leqslant d} G)$ is the least solution of:
  - 1 if $s \in G$
  - otherwise:

$$\int_0^d E(s) \cdot e^{-E(s)\cdot x} \cdot \sum_{s' \in S} \mathbf{P}(s, s') \cdot \Pr(s' \models \Diamond^{\leqslant d-x} G) \, dx$$

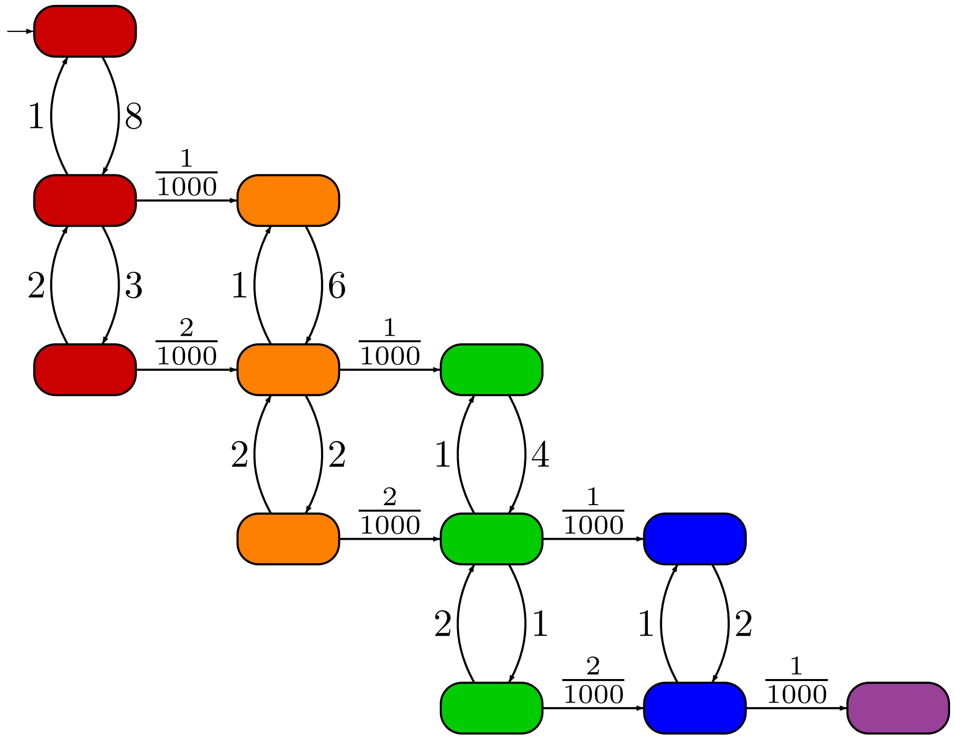
- Reduction to well-studied problem allows stable, efficient computation

Recall the Markov chain model



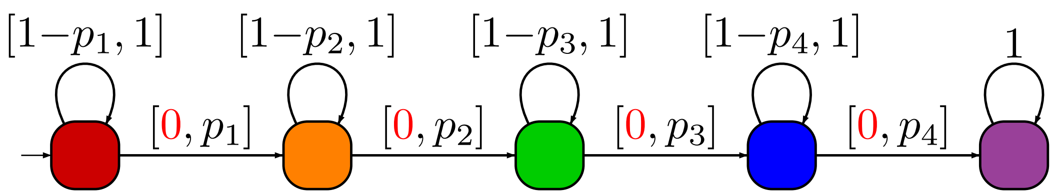
- Verification takes days
- $\approx 6{\cdot}10^7$ iterations needed
- Mainly due to stiffness
- Solution: abstraction

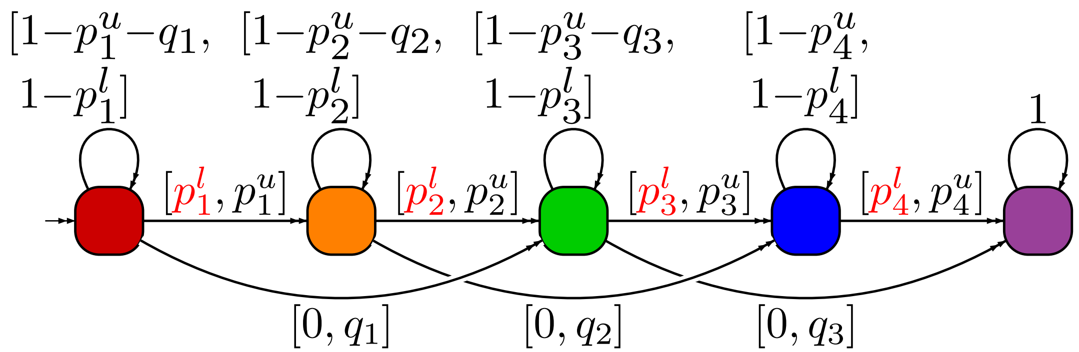
Abstraction



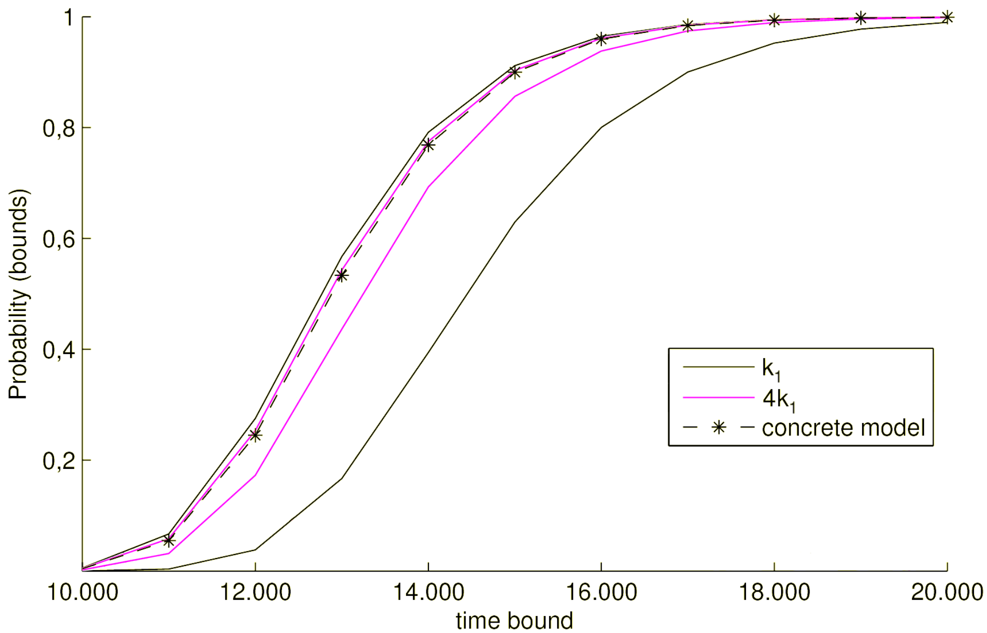
rule of thumb: group sets of “fast” connected states

Improving lower bounds





Model checking results



$\lvert\mathcal{A}\rvert$	$\lvert S\rvert$	time
50	861	$0m\ \ 5s$
300	6111	$37m\ 36s$
500	10311	$70m\ 39s$
1000	20811	$144m\ 49s$
1500	31311	$214m\ \ 2s$
2000	41811	$322m\ 50s$

probability of only having products in deadline t (200 substrates, 20 enzymes)

results using Markov Chain Model Checker `www.mrmc-tool.org`

Model checking results

Grid abstraction versus tree-analysis techniques (error bound 10^{-6}):

		grid abstraction								uniformization	
	diff	grid 12	grid 16	grid 20	grid 24	grid 28	grid 32	grid 36	grid 40	trunc	≈ states
	2.5	0.0224	0.001	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	10^{-6}	185	10^{129}
t	7.5	0.3117	0.0580	0.0062	0.0004	10^{-5}	10^{-6}	10^{-6}	10^{-6}	270	10^{188}
	15	0.4054	0.1345	0.0376	0.0086	0.0015	0.0002	$2 \cdot 10^{-5}$	$3 \cdot 10^{-6}$	398	10^{278}
states		6188	20349	53130	118755	237336	435894	749398	1221759		
distributions		28666	96901	256796	579151	1164206	2146761	3701296	6047091		
time (h:mm:ss)		0:00:26	0:01:33	0:04:15	0:09:50	0:20:14	0:38:13	1:07:57	2:06:04		

⇒ Abstraction yields same accuracy by 1.2 million states as 10^{278} ones!

⇒ First time that tree-based QBDs of this size have been analyzed

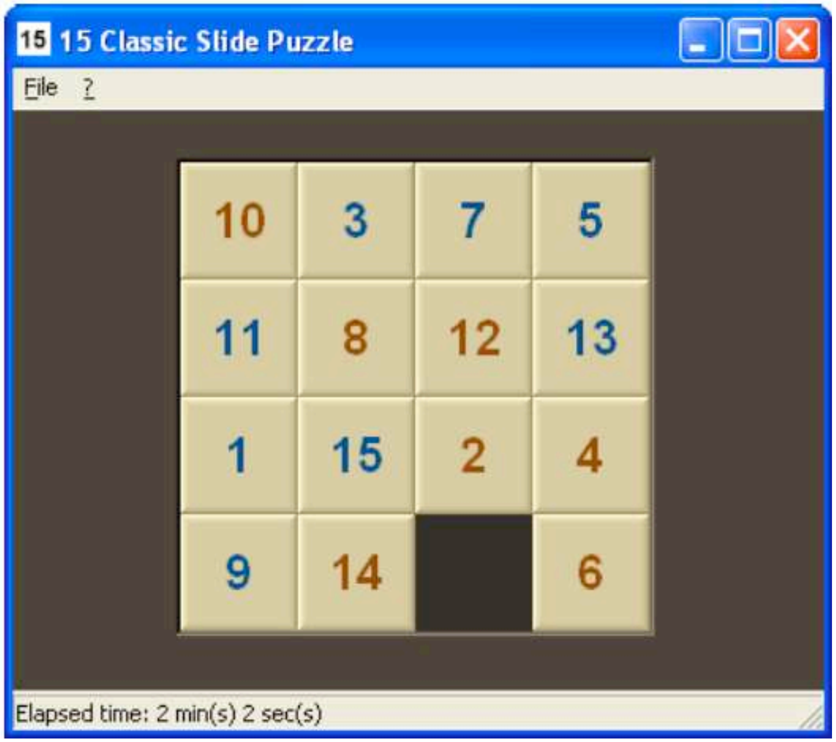
Batteries

Counterexamples are indispensable

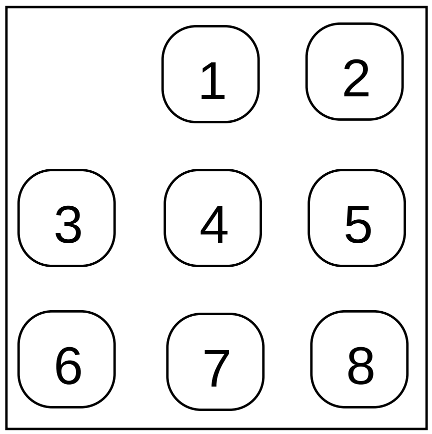
"It is impossible to overestimate the importance of counterexamples.
The counterexamples are invaluable in debugging complex systems.
Some people use model checking just for this feature."

Ed Clarke, 25 Years of Model Checking, FLOC 2008

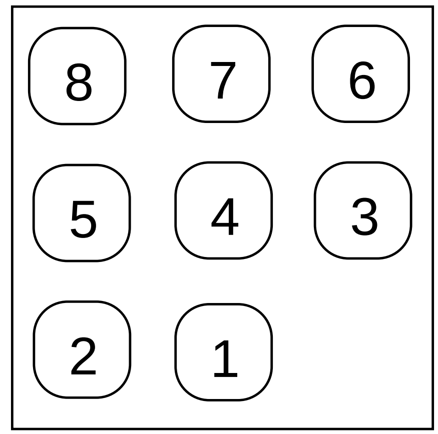
The 15-puzzle (Chapman, 1874)



Scheduling by model checking



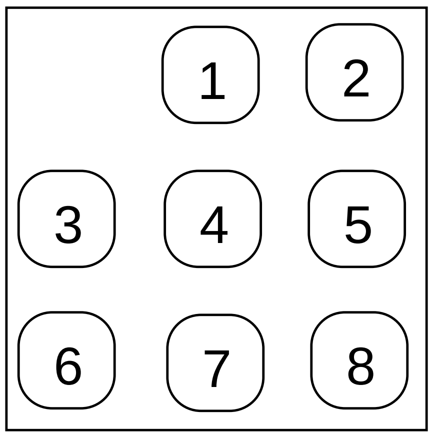
initial configuration



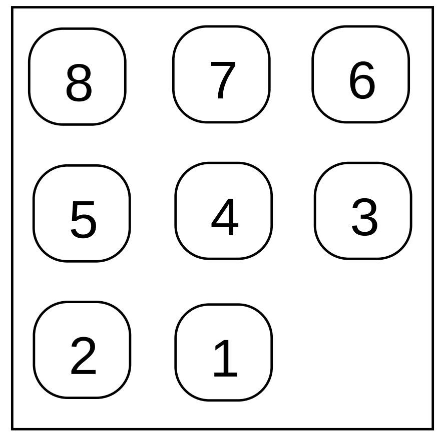
final configuration

there are about $4 \cdot (N \cdot K)!$ possible moves in an $N \times K$ puzzle

Scheduling by model checking



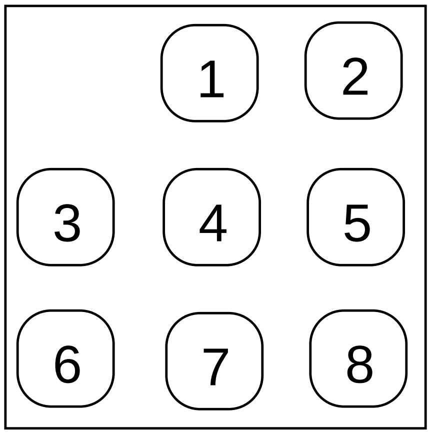
initial configuration



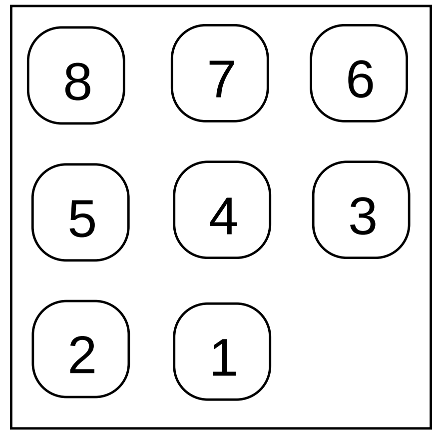
final configuration

check the property “the final configuration is not reachable”

Scheduling by model checking



initial configuration



final configuration

counterexample: r r d d l l u u r r d d l l u u r r d d l l u u r r d d r

Batteries

- Batteries are essential for mobile devices


laptop computers


phones and PDAs


sensor networks


military equipment

Batteries

- Batteries are essential for mobile devices

- Battery capacity is limited

~ 15.4 kJ

~ 1171 kJ

Batteries

- Batteries are essential for mobile devices

- Battery capacity is limited

~ 15.4 kJ

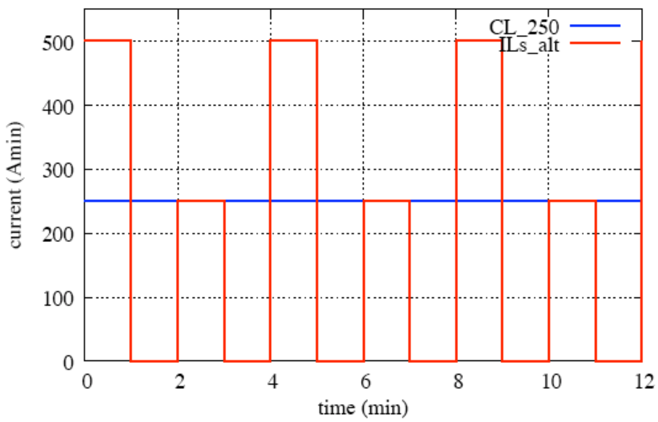
~ 1171 kJ

- Battery lifetime determines system uptime and depends on

  battery capacity, level of discharge current, usage profile

Multi-battery scheduling problem

Usage profile:

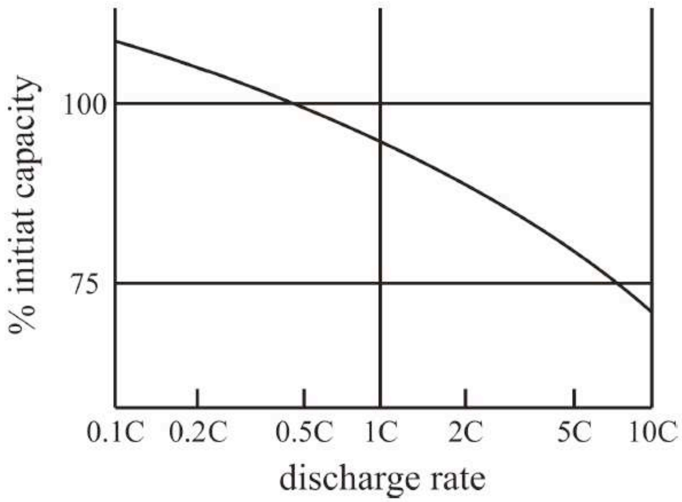


Multiple batteries:

which battery to use to maximise system lifetime?

Non-linear battery effects

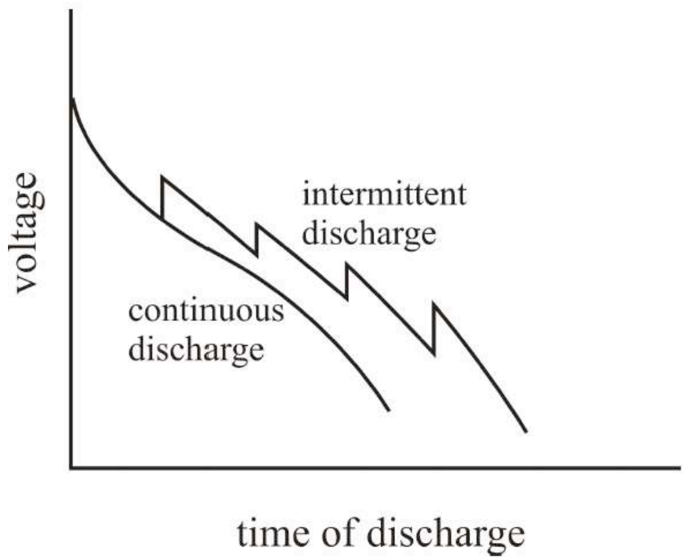
Rate capacity effect:



capacity drops for high discharge currents

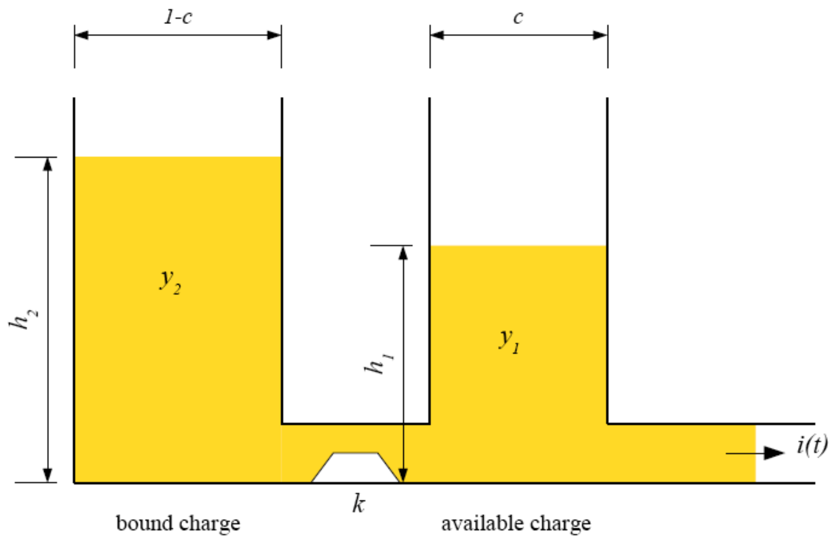
relative capacity to 0,5 C = total discharge in 2 hours

Recovery effect:



battery regains capacity in idle periods

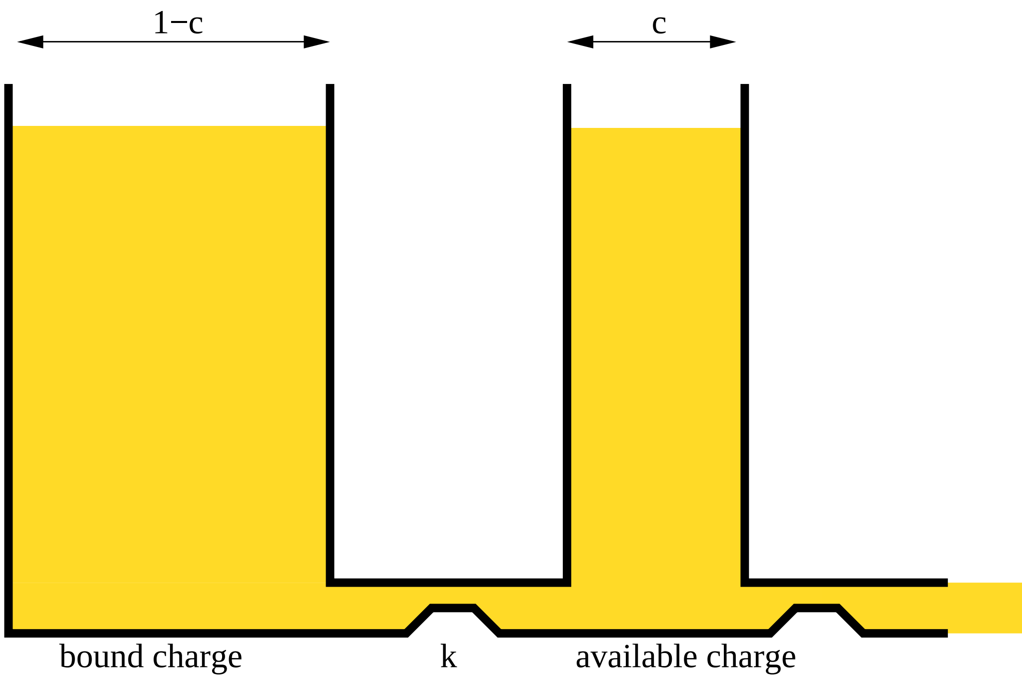
Kinetic battery model (Manwell & McGowan 1993)



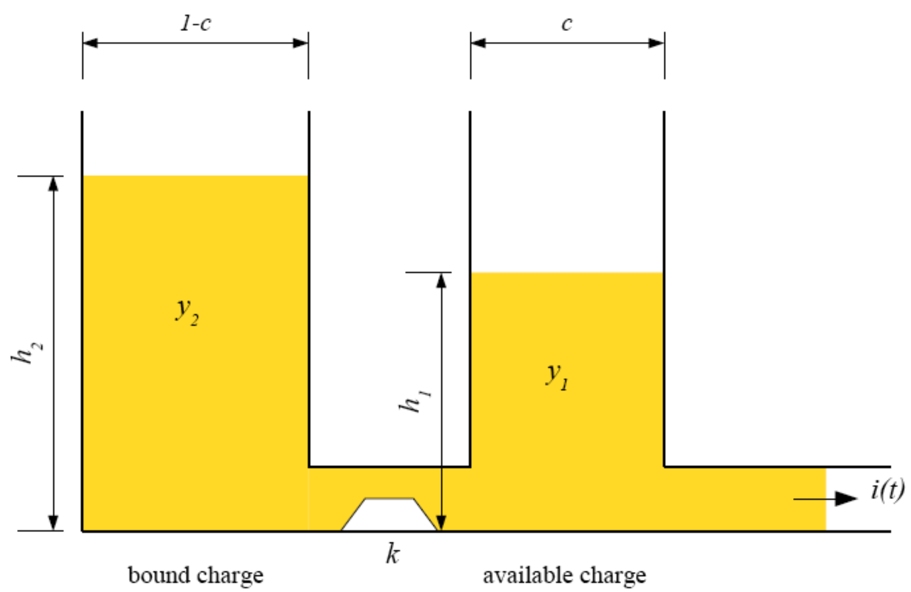
- lead-acid batteries
- charge distributed over 2 wells
- battery empty = $y_1 = 0$

- recovery effect
  - when idle, charge flows between wells
  - rate depends on $h_2 - h_1$ and "resistance" k
- rate capacity effect
  - higher discharge leaves less time to recover

Kinetic battery model



Kinetic battery model

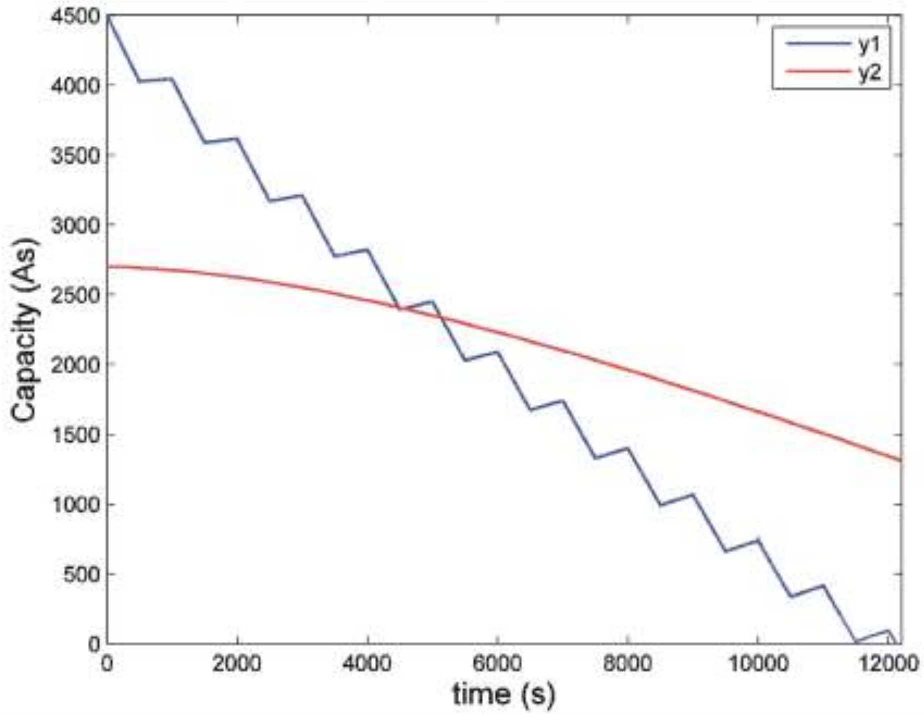


$$h_1(t) = \frac{y_1(t)}{c}$$
$$h_2(t) = \frac{y_2(t)}{1-c}$$

$$\dot{y}_1(t) = -i(t) + k\cdot(h_2(t) - h_1(t))$$
$$\dot{y}_2(t) = -k\cdot(h_2(t) - h_1(t))$$

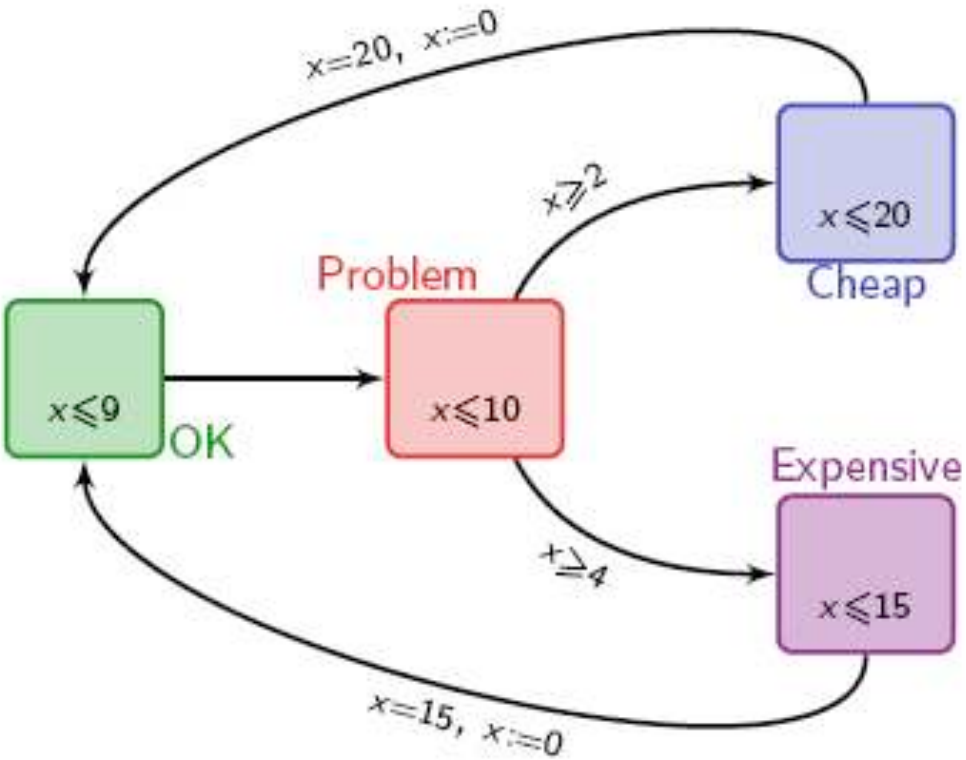
$$y_1(0) = c\cdot C$$
$$y_2(0) = (1-c)\cdot C$$

Discharge curves

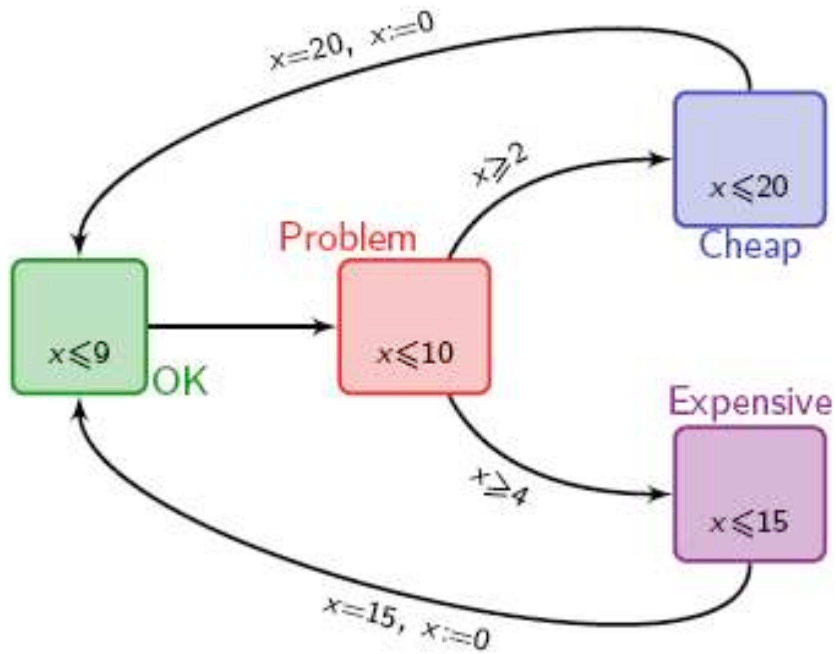


$C = 7500$ As, $k = 4, 5 \cdot 10^{-5}$, $c = 0.625$, $f = 0.001$ Hz, 500 s on, 500 s off

Timed automata

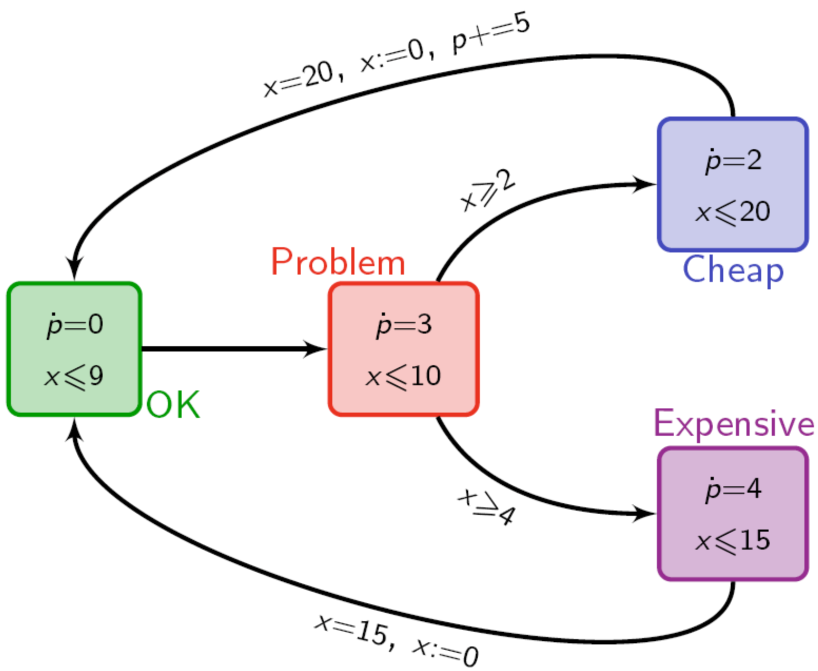


Example run



$$\begin{array}{c} \mathit{OK} \\ 0 \end{array} \stackrel{1.7}{\Longrightarrow} \begin{array}{c} \mathit{OK} \\ 1.7 \end{array} \longrightarrow \begin{array}{c} \mathit{Problem} \\ 1.7 \end{array} \stackrel{7.0}{\Longrightarrow} \begin{array}{c} \mathit{Problem} \\ 8.7 \end{array} \longrightarrow \begin{array}{c} \mathit{Cheap} \\ 8.7 \end{array} \stackrel{11.3}{\Longrightarrow} \begin{array}{c} \mathit{Cheap} \\ 20.0 \end{array} \longrightarrow \begin{array}{c} \mathit{OK} \\ 0 \end{array} \ldots$$

Priced timed automata



OK		*OK*		*Problem*		*Problem*		*Cheap*		*Cheap*		*OK*	
0	$\stackrel{1.7}{\Longrightarrow}$	1.7	$\rightarrow$	1.7	$\stackrel{7.0}{\Longrightarrow}$	8.7	$\rightarrow$	8.7	$\stackrel{11.3}{\Longrightarrow}$	20.0	$\rightarrow$	0	...
0		0		0		21.0		21.0		43.6		48.6	

Minimum-cost reachability (Behrmann *et al.*, Alur *et al.*, 2001)

The minimum-cost reachability problem:

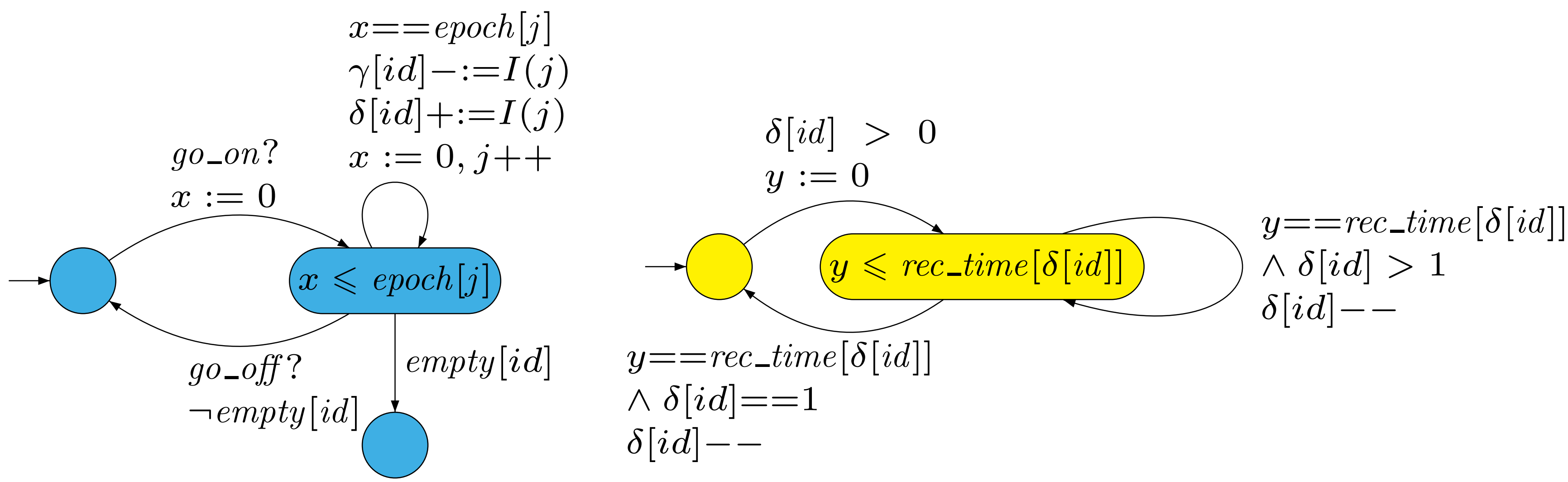
Input: a priced timed automaton and a target state

Output: the minimum cost of runs from the initial state to the target

is effectively computable.

a by-product of the model checking is a schedule that yields the minimal cost

Priced timed automata model



PTA model: $\underbrace{(DC_1 \,||\, RC_1)}_{\text{battery 1}} \;||\; \ldots\ldots \;||\; \underbrace{(DC_n \,||\, RC_n)}_{\text{battery n}} \;||$ Load || Scheduler

Objective: minimise the bound charge levels (of all batteries) once all batteries are empty

Model checking results

test load	sequential lifetime (min)	round robin lifetime (min)	best-of-two lifetime (min)	optimal lifetime (min)
CL_250	9.12	11.60	11.60	12.04
CL_500	4.10	4.53	4.53	4.58
CL_alt	5.48	6.10	6.12	6.48
ILs_250	22.80	38.96	38.96	40.80
ILs_500	8.60	10.48	10.48	10.48
ILs_alt	12.38	12.82	16.30	16.91
ILs_r1	12.80	16.26	16.26	20.52
ILs_r2	12.24	14.50	14.50	14.54
ILℓ_250	45.84	76.00	76.00	78.96
ILℓ_500	12.94	15.96	15.96	18.68

results using Uppaal Cora `www.uppaal.com`

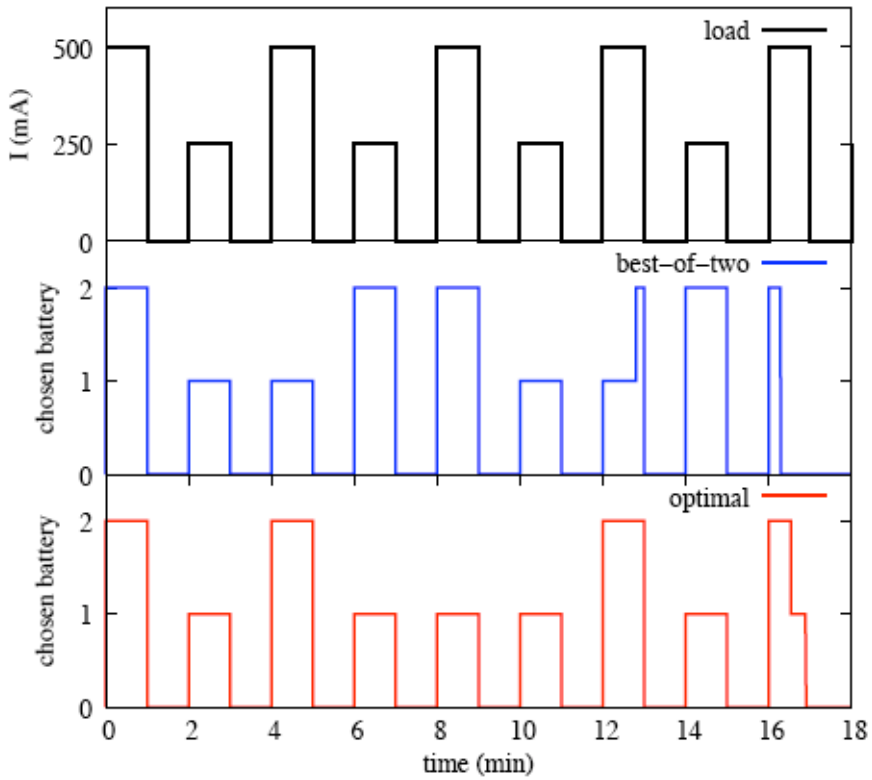
The recovery effect

test load	sequential lifetime (min)	round robin lifetime (min)	best-of-two lifetime (min)	optimal lifetime (min)
CL_250	9.12	11.60	11.60	12.04
CL_500	4.10	4.53	4.53	4.58
CL_alt	5.48	6.10	6.12	6.48
ILs_250	22.80	38.96	38.96	40.80
ILs_500	8.60	10.48	10.48	10.48
ILs_alt	12.38	12.82	16.30	16.91
ILs_r1	12.80	16.26	16.26	20.52
ILs_r2	12.24	14.50	14.50	14.54
$IL\ell$_250	45.84	76.00	76.00	78.96
$IL\ell$_500	12.94	15.96	15.96	18.68

Smart scheduling pays off!

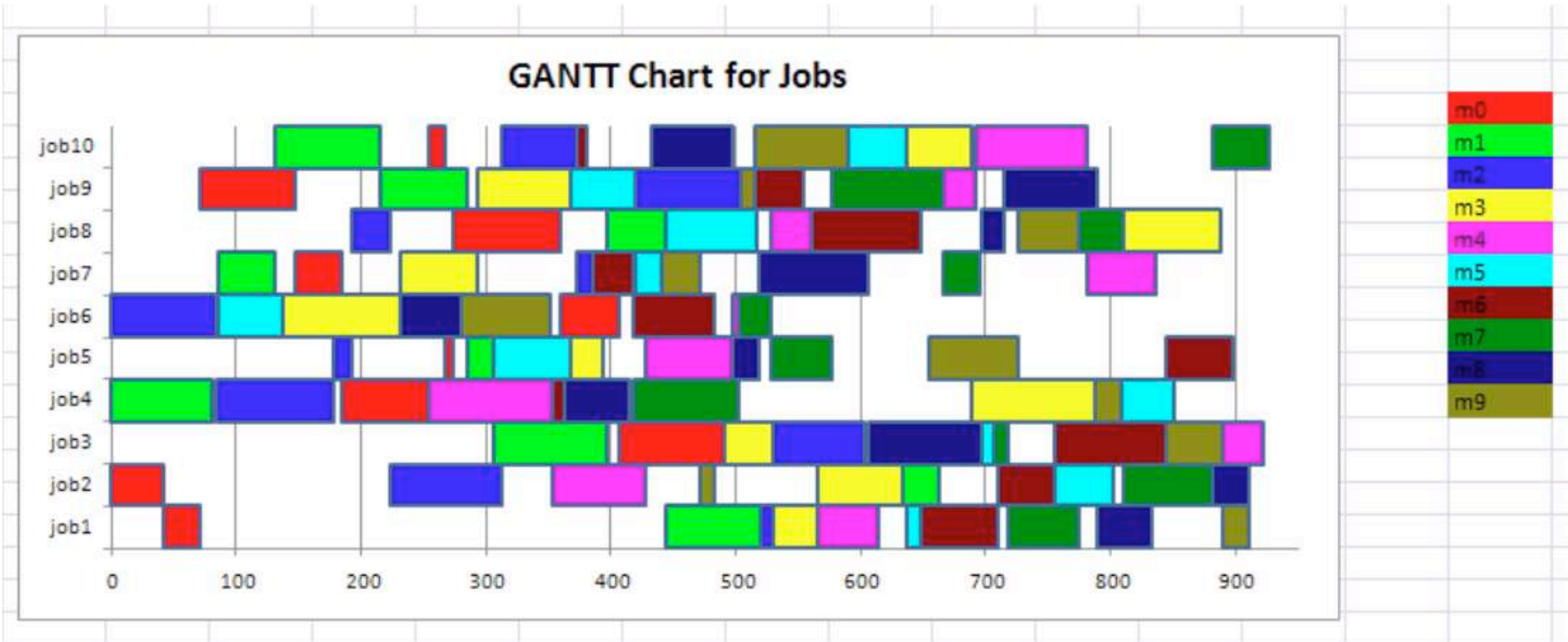
test load	sequential lifetime (min)	round robin lifetime (min)	best-of-two lifetime (min)	optimal lifetime (min)
CL_250	9.12	11.60	11.60	12.04
CL_500	4.10	4.53	4.53	4.58
CL_alt	5.48	6.10	6.12	6.48
ILs_250	22.80	38.96	38.96	40.80
ILs_500	8.60	10.48	10.48	10.48
ILs_alt	12.38	12.82	16.30	16.91
ILs_r1	12.80	16.26	16.26	20.52
ILs_r2	12.24	14.50	14.50	14.54
ILℓ_250	45.84	76.00	76.00	78.96
ILℓ_500	12.94	15.96	15.96	18.68

The schedules



note that best-of-two requires charge observability

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, . . .

Stochastic scheduling

- Job processing times are subject to random variability 

  machine breakdowns and repairs, job parameters, . . .

- Performance measured as minimization of:
  - expected flow time—total time of all jobs staying in the system
  - expected makespan—finishing time of last job

  significantly depends on the scheduling policy

Stochastic scheduling

- Job processing times are subject to random variability 

  machine breakdowns and repairs, job parameters, . . .

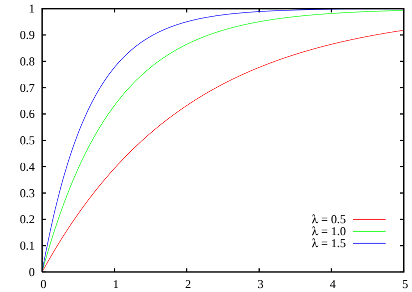
- Performance measured as minimization of:
  - expected flow time—total time of all jobs staying in the system
  - expected makespan—finishing time of last job

  significantly depends on the scheduling policy

Which policies are optimal, realizable, and how to determine them?

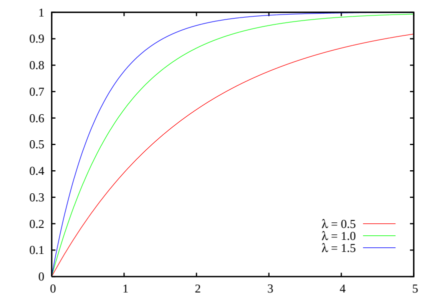
The problem

- N independent jobs with a random duration
  - exponentially distributed durations with mean $\frac{1}{\mu_i}$
- M identical machines for processing jobs



The problem

- N independent jobs with a random duration
  - exponentially distributed durations with mean $\frac{1}{\mu_i}$

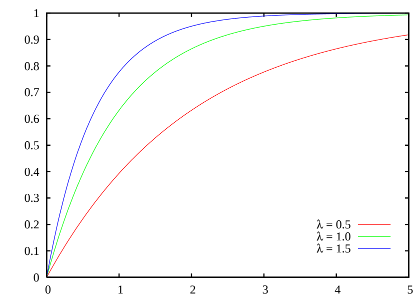


- M identical machines for processing jobs

- Jobs may be preempted at decision epochs
  - due to memoryless property, job durations are unaffected

The problem

- N independent jobs with a random duration
  - exponentially distributed durations with mean $\frac{1}{\mu_i}$



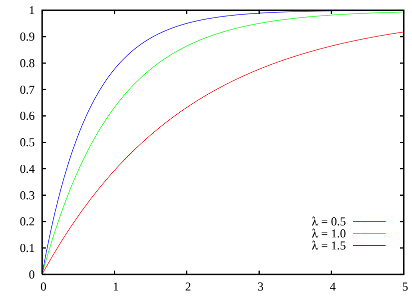
- M identical machines for processing jobs

- Jobs may be preempted at decision epochs
  - due to memoryless property, job durations are unaffected

- Well-known facts: (Bruno et al., JACM 1981)
  - SEPT policy yields minimal flow time
  - LEPT policy yields minimal expected makespan
  - "it is hard to calculate these expected values"

The problem

- N independent jobs with a random duration
  - exponentially distributed durations mean $\frac{1}{\mu_i}$



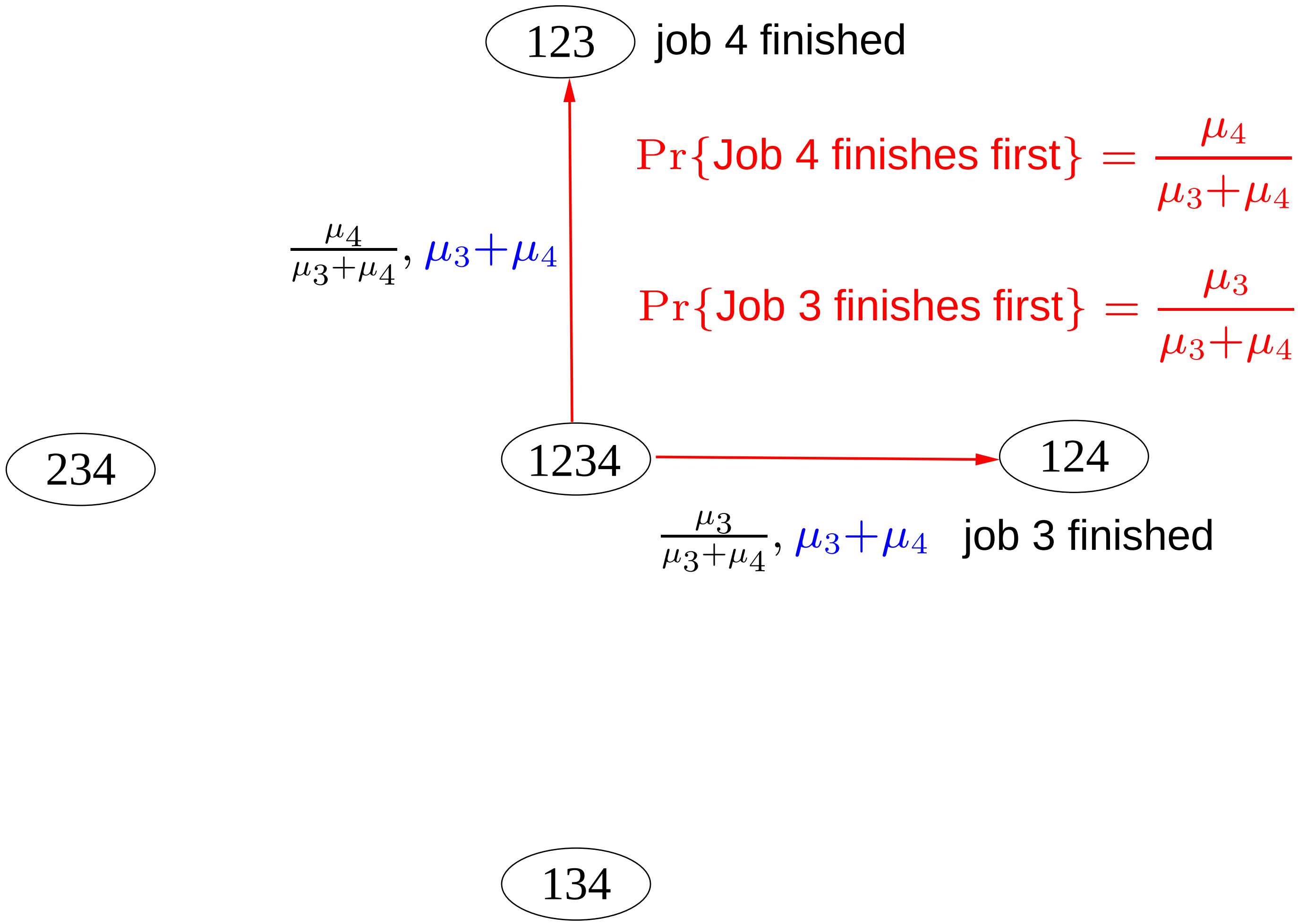
- M identical machines for processing jobs

- Jobs may be preempted at decision epochs
  - due to memoryless property, job durations are unaffected

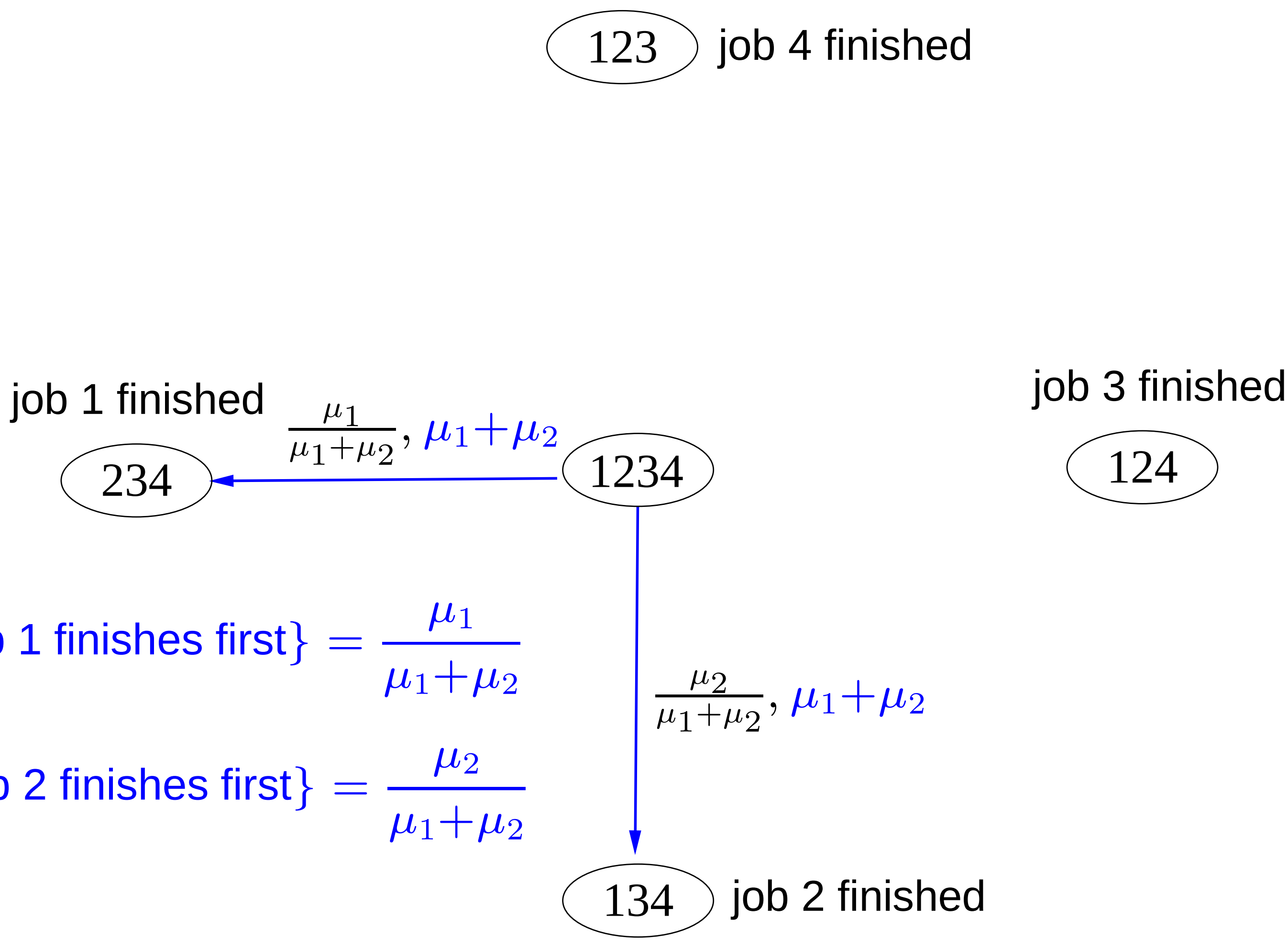
- Well-known facts: (Bruno et al., JACM 1981)
  - SEPT policy yields minimal flow time
  - LEPT policy yields minimal expected makespan

Which policy is optimal to maximise the probability to finish all jobs on time?

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



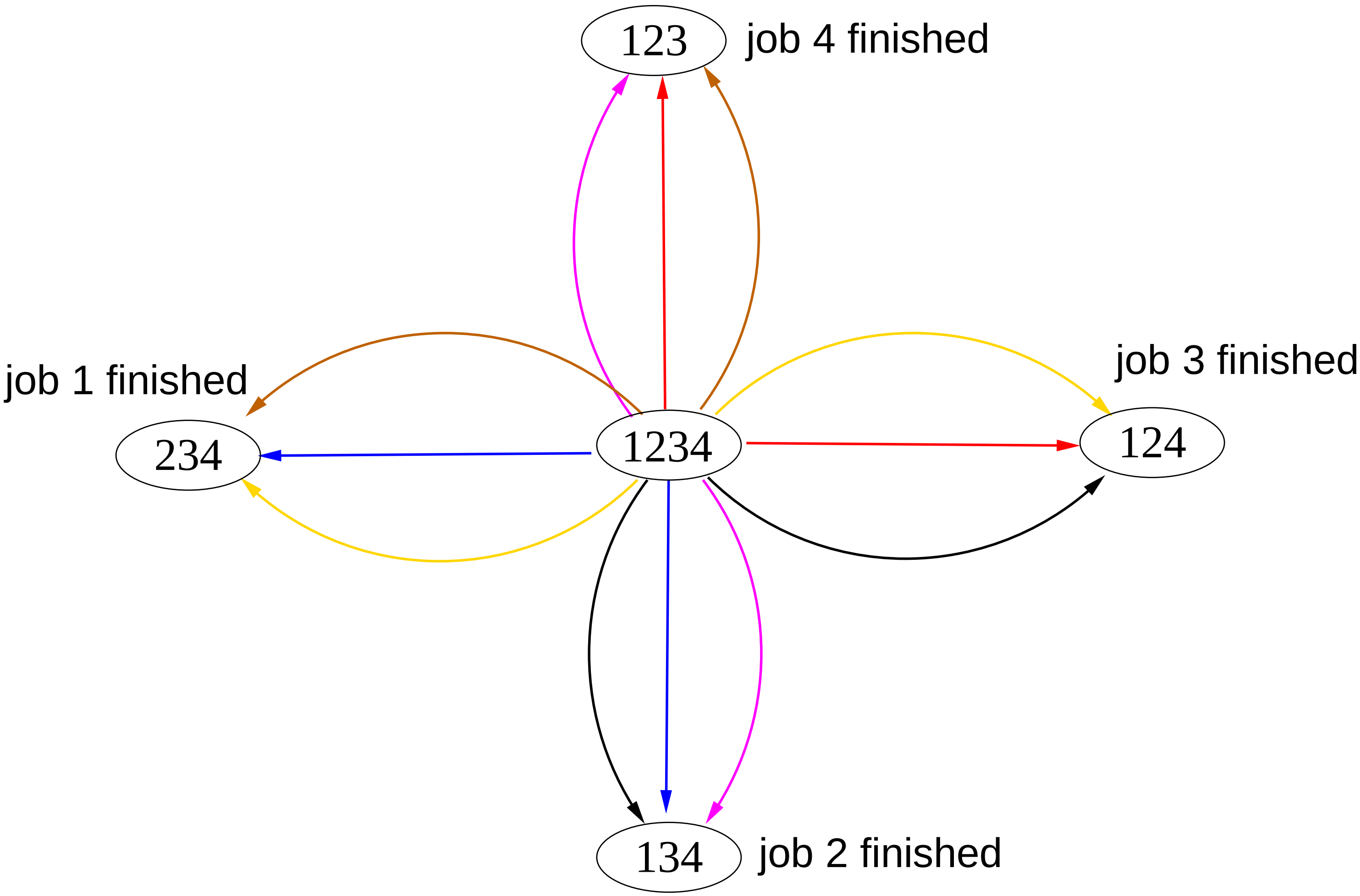
Stochastic scheduling



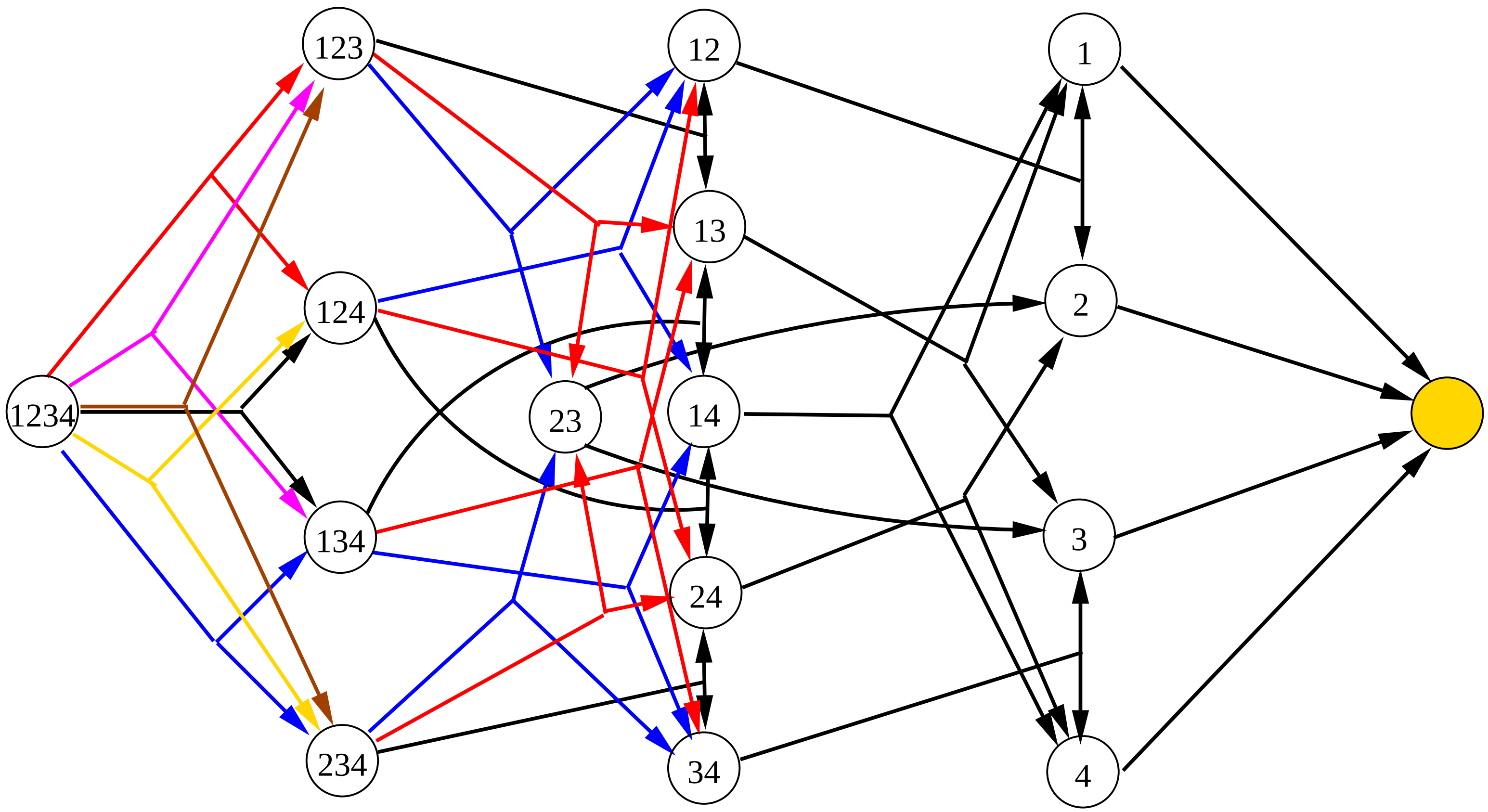
$$\Pr\{\text{Job 1 finishes first}\} = \frac{\mu_1}{\mu_1+\mu_2}$$

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

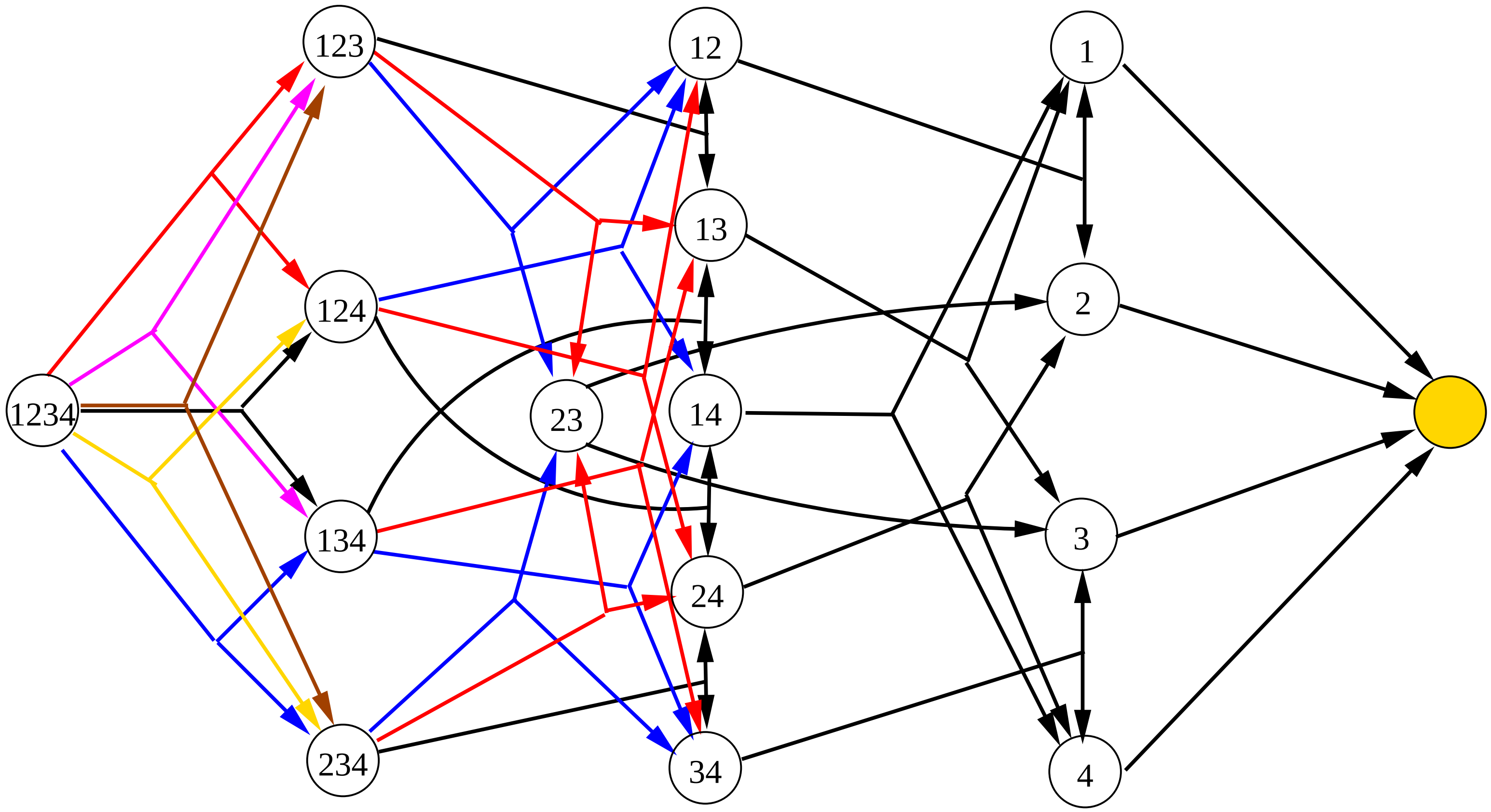
Stochastic scheduling



Stochastic scheduling



Stochastic scheduling



Which policy is optimal to maximize the probability to finish all jobs on time?

Maximal reachability probabilities (Neuhäußer & Zhang 2010)

Under mild conditons, the maximal reachability problem:

Input: a continuous-time Markov decision process, a target state and deadline d

Output: an ϵ-optimal policy maximizing the probability to reach the target within d

is effectively computable.

a by-product of the model checking yields the maximal probability

main technical issue: optimal policies are time-dependent

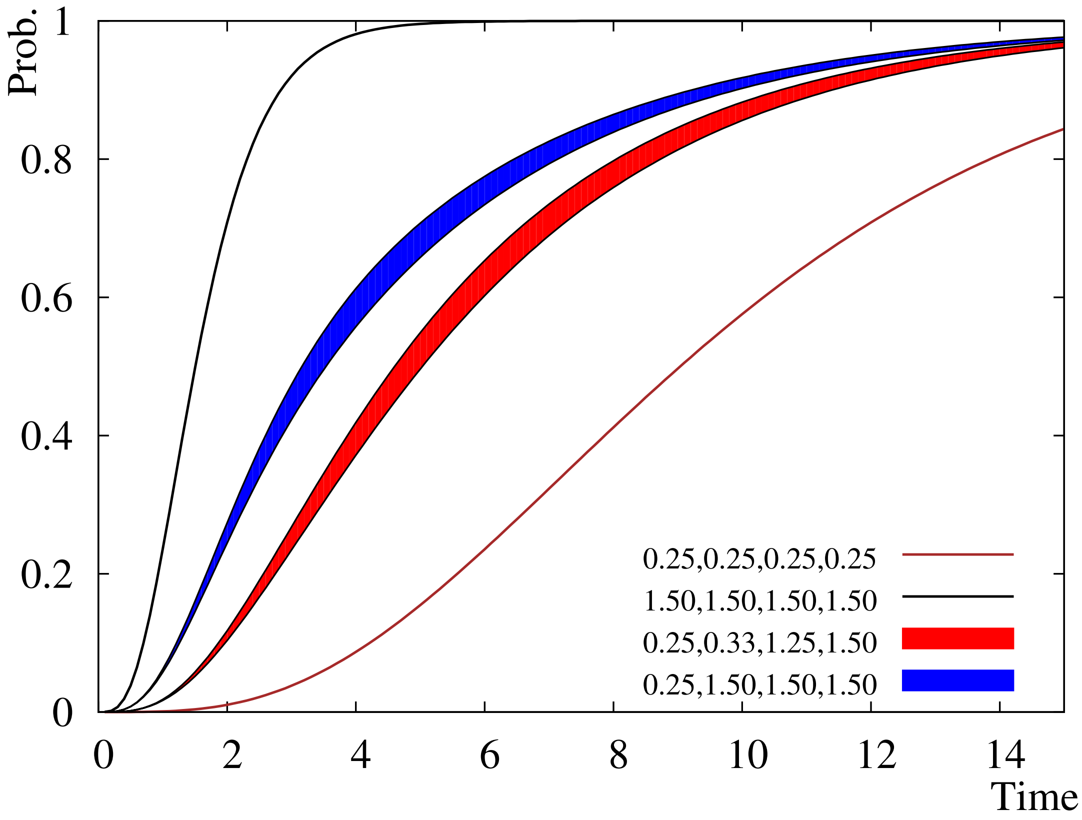
Maximal reachability probabilities

- State s, set G of goal states, and deadline t
- $\Pr^{\max}(s \models \Diamond^{\leqslant t} G)$ is the least solution of:
  - 1 if $s \in G$
  - otherwise:

$$\int_0^t E(s) \cdot e^{-E(s)\cdot x} \cdot \max_{\alpha} \sum_{s' \in S} \mathbf{P}(s, \alpha, s') \cdot \Pr(s' \models \Diamond^{\leqslant t-x} G) \, dx$$

- Discretization to well-studied MDP problem allows stable computation

Model checking results



the optimal policy is shortest expected processing time (SEPT) first

Epilogue

Three non-standard model-checking examples:

- Enzyme kinetics
- Optimal scheduling of multi-battery systems
- Stochastic scheduling

Benefits:

$\Rightarrow$ Alternative to existing, established techniques

$\Rightarrow$ Offers new solutions to open problems

Epilogue

Three non-standard model-checking examples:

- Enzyme kinetics
- Optimal scheduling of multi-battery systems
- Stochastic scheduling

Benefits:

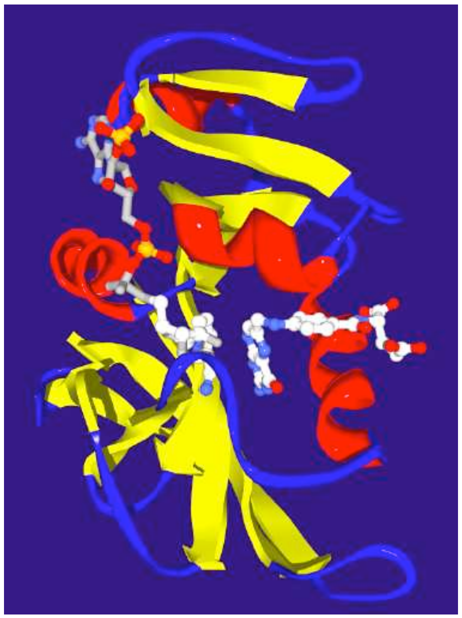
$\Rightarrow$ Alternative to existing, established techniques

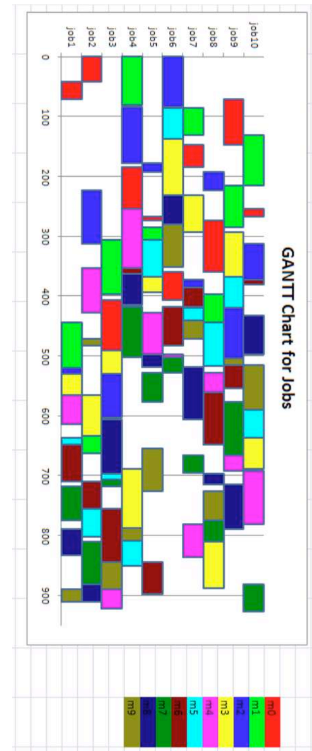
$\Rightarrow$ Offers new solutions to open problems

Clear trend

- Need for **quantitative** model checking

Thank you





You can do much more with model checking than you think!

Further reading

- CTMC model checker MRMC
  - [Katoen *et al.*, Journal on Performance Evaluation 2011]
- Abstraction: biology example
  - [Katoen, Klink, Leucker & Wolf, CONCUR 2008]
- Abstraction: queuing network example
  - [Klink, Remke Katoen & Haverkort, Journal on Performance Evaluation 2012]
- Battery scheduling
  - [Jongerden *et al.*, IEEE Transactions on Industrial Informatics 2010]
- Stochastic scheduling
  - [Zhang and Neuhäusser, TACAS 2010]