Performance Analysis by Model Checking

Joost-Pieter Katoen

Department of Electrical Engineering, Mathematics and Informatics
University of Twente

joint work with Christel Baier, Boudewijn Haverkort and Holger Hermanns

Infinite Systems and Verification of Quantitative Properties, March 6, 2003

Overview

- Introduction and motivation
- The discrete-time setting
- The continuous-time setting
- Incorporating costs
- Epilogue

What is *model-based* performance evaluation?

- *Aim*:

  answer questions about the performance and dependability of systems, e.g.:

  - how many clients can this file server adequately support?
  - how large should the buffers be to guarantee a packet loss ratio of at most 10^{-6}?
  - how many jobs can be processed before a failure occurs?

- *Approach:*

  attempts to forecast system performance using (abstract) models

Model specification techniques

- Performance models are specified using *probabilistic* extensions of:
  - formal description languages, such as SDL, Petri nets or process algebras
- Advantages:
  - compact notation, compositionality, basis for formal reasoning
  - integrated formalism for qualitative and quantitative modeling
- If present, time is specified stochastically ("soft" real time)
- The underlying stochastic process is typically a *Markov chain*
  - discrete-time Markov chain (DTMC)
  - continuous-time Markov chain (CTMC)
  - generalised semi-Markov chain (GSMC)
  - . . . . . . and versions thereof with nondeterminism (e.g., MDC)

Why model-checking Markov chains?

- *Uniform framework* for qualitative and quantitative properties
- Unambiguous and precise *specification* of performance measures:

  $\Rightarrow$ using appropriate extensions of temporal logic (PCTL, CSL)

  $\Rightarrow$ lucid and compact specification of complex measures
- *Automated* verification of performance requirements:

  $\Rightarrow$ automated and uniform means to analyse these measures

  $\Rightarrow$ automated measure-driven tailoring of model

  $\Rightarrow$ specialized algorithms are hidden from the (performance) engineer

Finite *discrete-time* Markov chains

A labeled DTMC $\mathcal{D}$ is a triple $(S, \mathbf{P}, L)$ with:

- S, a finite discrete state space
- $\mathbf{P} : S \times S \rightarrow [0, 1]$, a probability matrix with $\sum_{s'} \mathbf{P}(s, s') \in \{ 0, 1 \}$
  - $\mathbf{P}(s, s')$ is the discrete probability to jump from s to s' in one step
  - s is called absorbing if $\mathbf{P}(s, s') = 0$ for all s'
- $L : S \rightarrow 2^{AP}$, the labelling function
  - $L(s)$ is the set of atomic propositions that are valid in s

$\Rightarrow$ a DTMC is thus a Kripke structure with probabilistic transitions

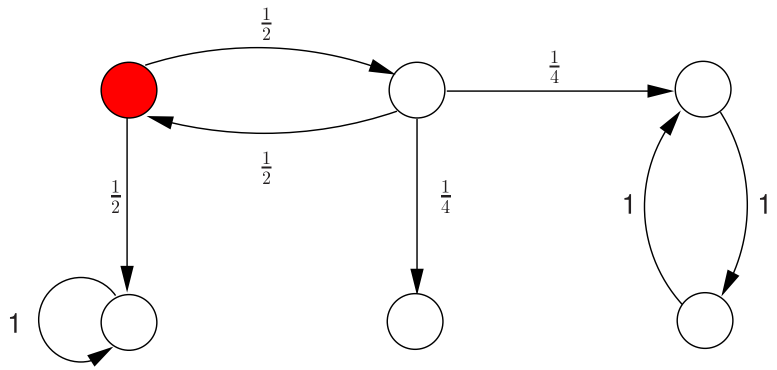
Elementary probabilities for DTMCs

- *Long-run* probability vector $\underline{\pi} = (\cdots, \pi_i, \cdots)$
  - where π_i is the probability to be in state s_i on the long run
  - π_i is mostly *in*dependent from the starting distribution
  - $\underline{\pi}$ is computed from a system of linear equations:

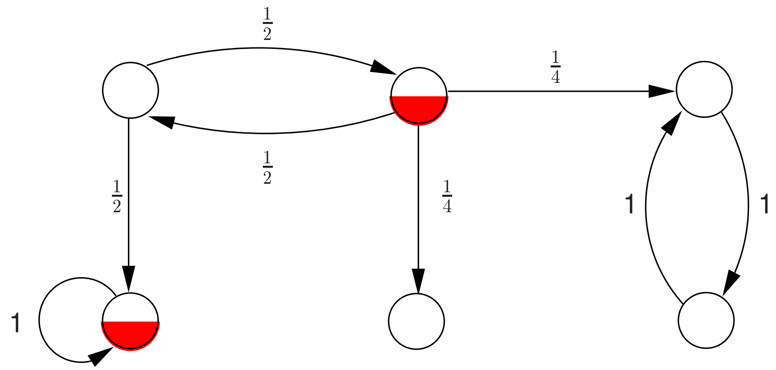
$$\underline{\pi} \cdot \mathbf{P} = \underline{\pi}, \quad \sum_i \pi_i = 1 \;\; \text{(normalisation)}$$

- *Transient* probability vector $\underline{\pi}(n) = (\cdots, \pi_i(n), \cdots)$ for natural n
  - where $\pi_i(n)$ is the probability to be in state s_i after n steps (given $\underline{\pi}(0)$)
  - $\underline{\pi}(n)$ is computed iteratively: $\underline{\pi}(n) = \underline{\pi}(n-1) \cdot \mathbf{P}, \quad$ given $\underline{\pi}(0)$

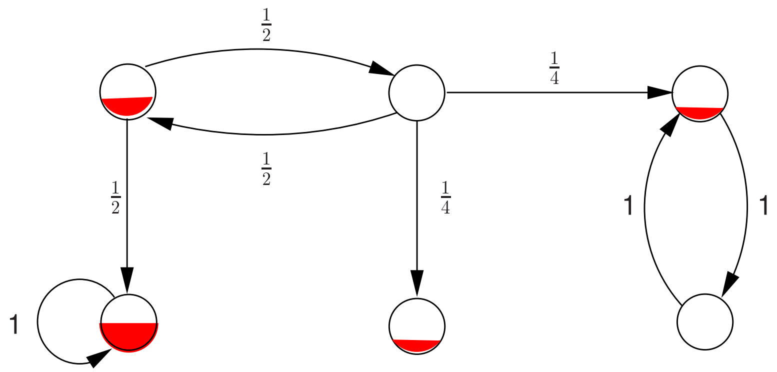
Evolution of an example DTMC



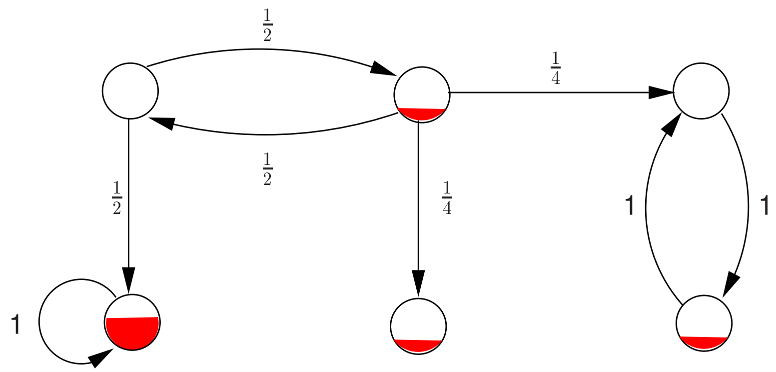
zero-th epoch



first epoch

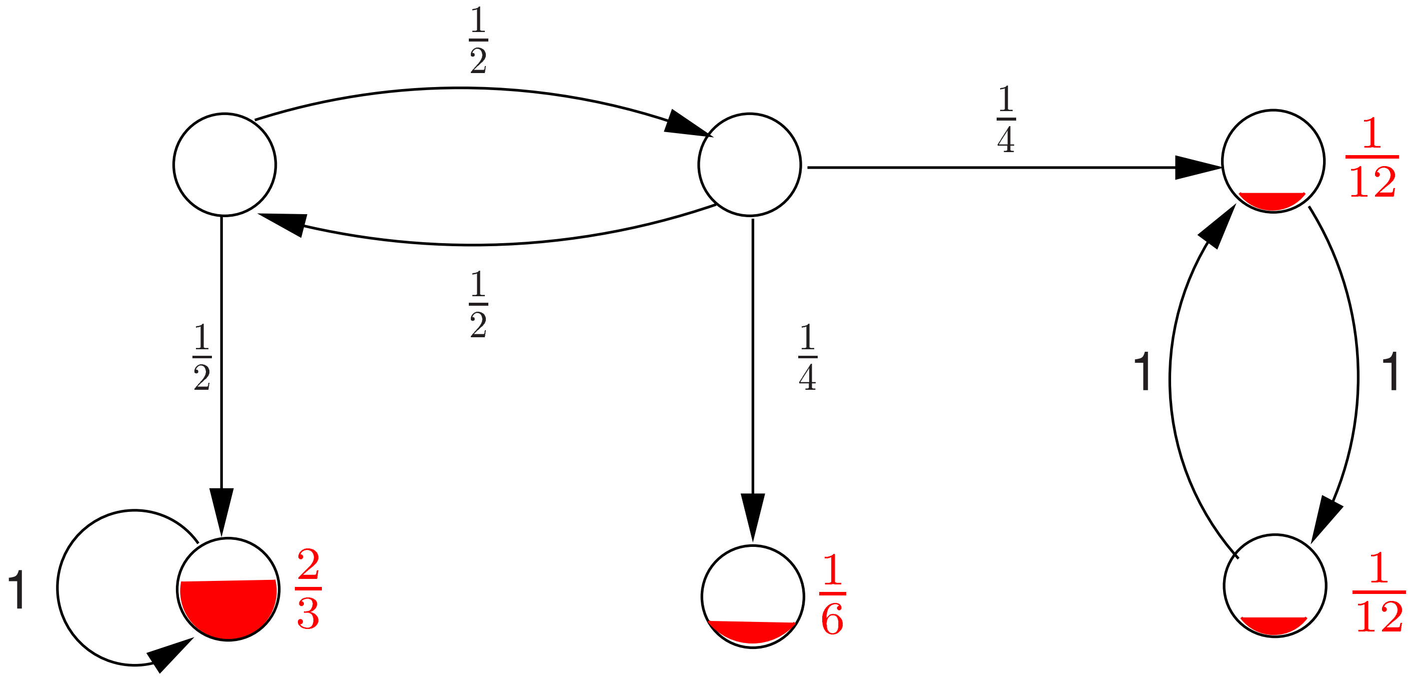


second epoch



third epoch

On the long run



Probabilistic computation tree logic (PCTL)

State-formulas $\Phi \ ::= \ a \ \big| \ \neg\,\Phi \ \big| \ \Phi \ \vee \ \Phi \ \big| \ \mathcal{P}_{\trianglelefteq p}(\varphi)$

with probability p and comparison operator $\trianglelefteq$

$\mathcal{P}_{\trianglelefteq p}(\varphi)$ probability that paths fulfill φ is $\trianglelefteq p$

$\exists\varphi =_{fair} \mathcal{P}_{>0}(\varphi)$ and $\forall\varphi =_{fair} \mathcal{P}_{\geqslant 1}(\varphi)$

Path-formulas $\varphi \ ::= \ \mathsf{X}\,\Phi \ \big| \ \Phi\,\mathsf{U}^{\leqslant k}\,\Phi$ with integer k or $k = \infty$

$\mathsf{X}\,\Phi$ next state fulfills Φ

$\Phi\,\mathsf{U}^{\leqslant k}\,\Psi$ Φ holds along the path until Ψ holds within k steps

CTL operator U is a special case

Example performance requirements

- The probability of not going down in the next state is at least 0.95:

$$\mathcal{P}_{\geqslant 0.95}(\mathsf{X}\, \neg \textit{down})$$

- The probability the system goes down within 5 steps exceeds 0.01:

$$\mathcal{P}_{>0.01}(\Diamond^{\leqslant 5}\, \textit{down})$$

- The probability of the system going down within 5 steps after continuously operating with at least 2 processors is at most 0.01:

$$\mathcal{P}_{\leqslant 0.01}((\textit{up}_2 \,\vee\, \textit{up}_3)\, \mathsf{U}^{\leqslant 5}\, \textit{down})$$

Semantics of PCTL: state-formulas

$\mathcal{D}, s \models \Phi$ if and only if formula Φ holds in state s of DTMC $\mathcal{D}$

Relation $\models$ is defined by:

$s \models a$	iff	$a \in L(s)$
$s \models \neg\, \Phi$	iff	not $(s \models \Phi)$
$s \models \Phi \,\vee\, \Psi$	iff	$(s \models \Phi)$ or $(s \models \Psi)$
$s \models \mathcal{P}_{\trianglelefteq p}(\varphi)$	iff	$\Pr\{\, \sigma \in \mathit{Paths}(s) \mid \sigma \models \varphi \,\} \trianglelefteq p$

$\Pr\{\ldots\}$ is measurable by a standard (i.e., cone) Borel space construction

Semantics of PCTL: path-formulas

A *path* in DTMC $\mathcal{D}$ is an infinite sequence $s_0\, s_1 \ldots$ with $\mathbf{P}(s_i, s_{i+1}) > 0$

Semantics of path-formulas is defined by:

$$\sigma \models \mathsf{X}\,\Phi \qquad \text{iff } \sigma[1] \models \Phi$$

$$\sigma \models \Phi\, \mathsf{U}^{\leqslant k}\, \Psi \qquad \text{iff } \exists\, 0 \leqslant j \leqslant k.\, \sigma[j] \models \Psi \;\wedge\; (\forall\, 0 \leqslant i < j.\, \sigma[i] \models \Phi)$$

where $\sigma[i]$ denotes the $(i{+}1)$-th state in the path σ

Abbreviations:

$$\mathcal{P}_{\trianglelefteq p}(\Diamond^{\leqslant k}\,\Phi) = \mathcal{P}_{\trianglelefteq p}(\mathsf{true}\,\mathsf{U}^{\leqslant k}\Phi) \quad \text{and} \quad \mathcal{P}_{\trianglelefteq p}(\Diamond\,\Phi) = \mathcal{P}_{\trianglelefteq p}(\mathsf{true}\,\mathsf{U}\,\Phi)$$

$$\mathcal{P}_{\geqslant p}(\Box^{\leqslant k}\,\Phi) = \mathcal{P}_{\leqslant 1-p}(\Diamond^{\leqslant k}\,\neg\Phi) \quad \text{and} \quad \mathcal{P}_{\geqslant p}(\Box\,\Phi) = \mathcal{P}_{\leqslant 1-p}(\Diamond\,\neg\Phi)$$

Model-checking PCTL

- checking whether a state s in a DTMC satisfies a PCTL formula is performed in the same way as for CTL:
  - compute recursively the set $Sat(\Phi)$ of states that satisfy Φ
  - check whether state s belongs to $Sat(\Phi)$

  $\Rightarrow$ bottom-up traversal of the parse tree of Φ

- for the non-stochastic part: as for CTL

- how to compute $Sat(\Phi)$ for the probabilistic operators?

Model-checking probabilistic next and until

- Alternative formulation: $s \in Sat(\mathcal{P}_{\trianglelefteq p}(\varphi))$ if and only if $Prob(s, \varphi) \trianglelefteq p$
- Next: $Prob(s, \mathsf{X}\, \Phi)$ equals $\sum_{s' \in Sat(\Phi)} \mathbf{P}(s, s')$
- Until: probability $Prob(s, \Phi\, \mathsf{U}^{\leqslant k}\, \Psi)$ is the least solution of:
  - 1 if $s \in Sat(\Psi)$
  - for $k > 0$ and $s \in Sat(\Phi) - Sat(\Psi)$:

$$\sum_{s' \in S} \mathbf{P}(s, s') \cdot Prob(s', \Phi\, \mathsf{U}^{\leqslant k-1}\, \Psi)$$

  - 0 otherwise
- More efficient algorithms are used for $\mathcal{P}_{>0}(\Phi\, \mathsf{U}^{\leqslant k}\, \Psi)$ and $\mathcal{P}_{\geqslant 1}(\Phi\, \mathsf{U}^{\leqslant k}\, \Psi)$

Reduction to transient analysis

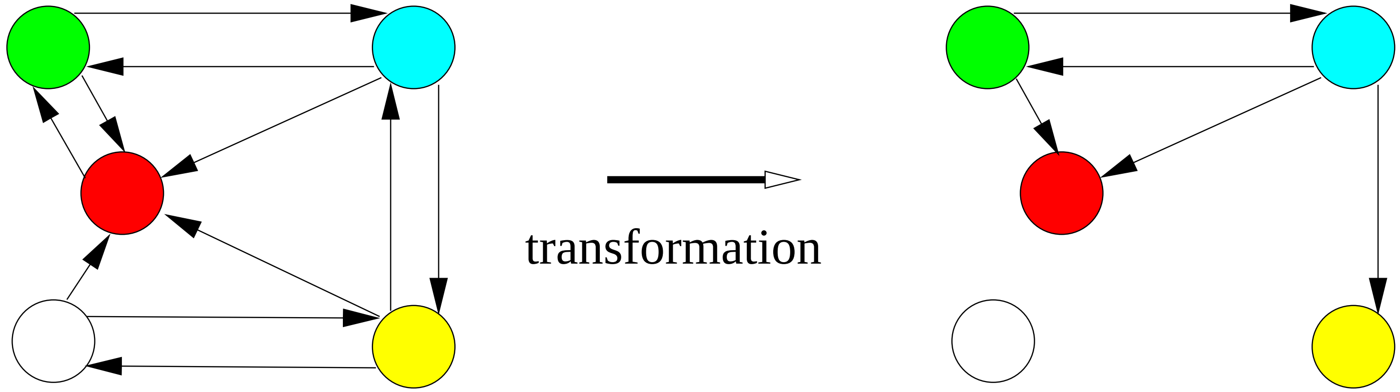
- For an arbitrary DTMC $\mathcal{D}$ and property $\varphi = \Phi \, \mathsf{U}^{\leqslant k} \, \Psi$ we have:
  - φ is fulfilled once a Ψ-state is reached in at most k steps along a Φ-path
  - φ is violated once a $\neg \, (\Phi \, \vee \, \Psi)$-state is visited within k steps
- Approach: *transform* the DTMC $\mathcal{D}$ as follows
  - make all Ψ-states absorbing
  - make all $\neg \, (\Phi \, \vee \, \Psi)$-states absorbing
  - check $\Diamond^{=k} \, \Psi$ in the obtained DTMC $\mathcal{D}'$
  - . . . but . . .
  - this amounts to a standard transient analysis in $\mathcal{D}'$: $\displaystyle\sum_{s \in Sat(\Psi)} \pi_s(k)$

$\Rightarrow$ Checking probabilistic until amounts to matrix-vector multiplication

Checking probabilistic conditional reachability

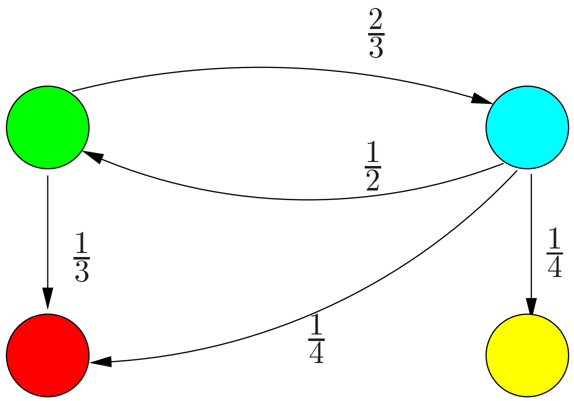
PCTL-formula: $\mathcal{P}_{\geqslant 0.4}((\mathit{up}_3 \vee \mathit{up}_2) \, \mathsf{U}^{\leqslant 3} \, \mathit{down})$

the transformation of the DTMC under consideration is as follows:

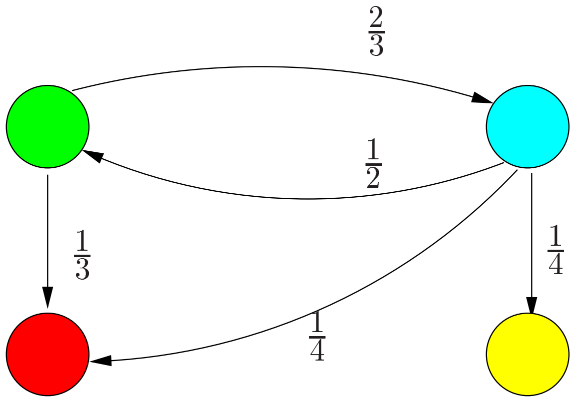


and then the recursive computation starts.........

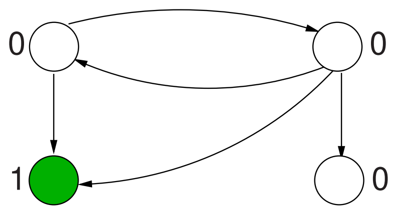
Checking $\mathcal{P}_{\geqslant 0.4}((\textit{green} \vee \textit{blue}) \, \mathsf{U}^{\leqslant 3} \, \textit{red})$



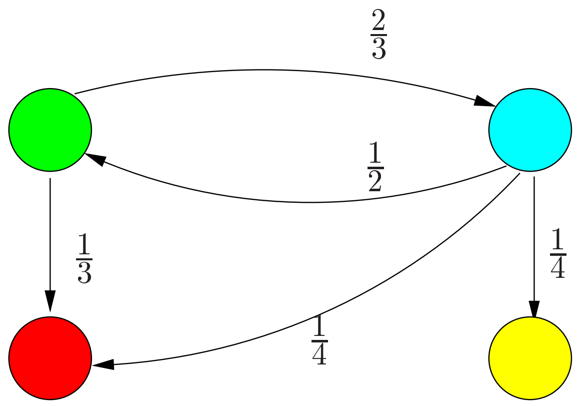
Checking $\mathcal{P}_{\geqslant 0.4}((\textit{green} \vee \textit{blue})\, \mathsf{U}^{\leqslant 3}\, \textit{red})$



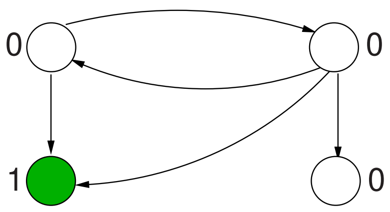
first iteration



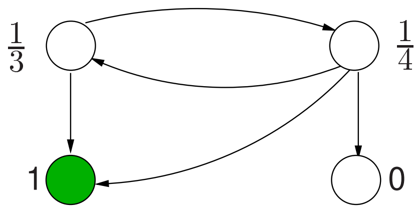
Checking $\mathcal{P}_{\geqslant 0.4}((\mathit{green} \vee \mathit{blue}) \, \mathsf{U}^{\leqslant 3} \, \mathit{red})$



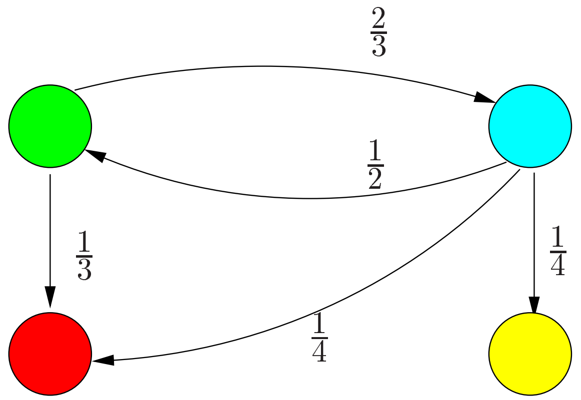
first iteration



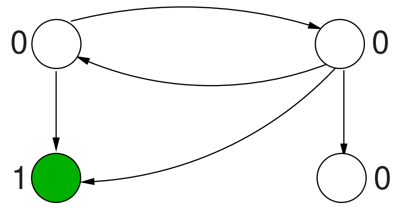
second iteration



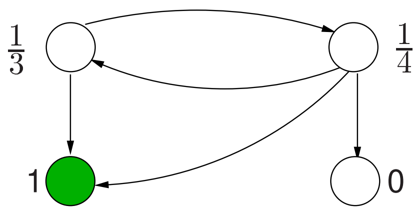
Checking $\mathcal{P}_{\geqslant 0.4}((\mathit{green} \vee \mathit{blue})\, \mathsf{U}^{\leqslant 3}\, \mathit{red})$



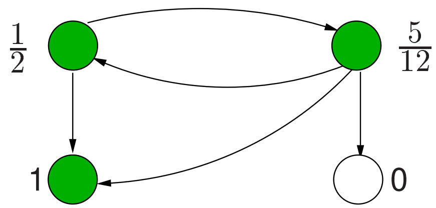
first iteration



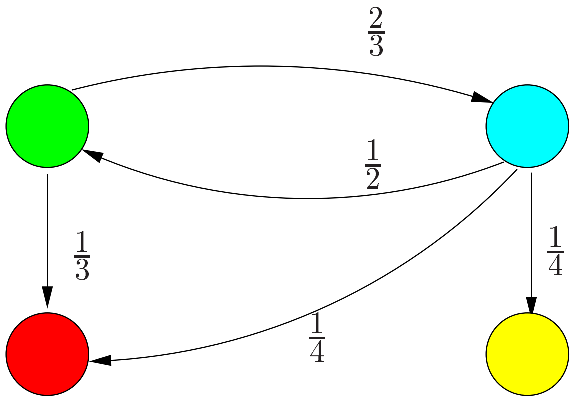
second iteration



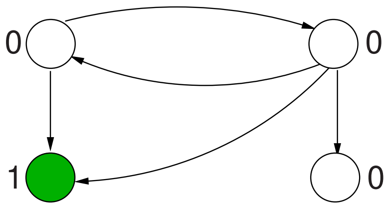
third iteration



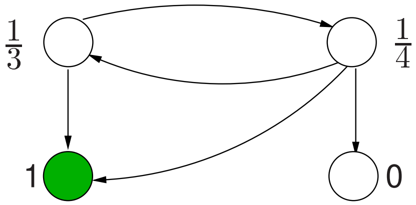
Checking $\mathcal{P}_{\geqslant 0.4}((\mathit{green} \vee \mathit{blue})\ \mathsf{U}^{\leqslant 3}\ \mathit{red})$



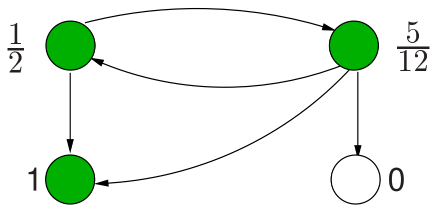
first iteration



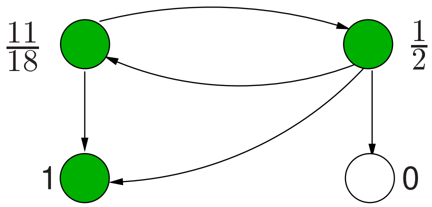
second iteration



third iteration



fourth iteration



Probabilistic bisimulation

- a *probabilistic bisimulation* on DTMC $\mathcal{D} = (S, \mathbf{P}, L)$ is an equivalence R on S such that whenever $(s, s') \in R$ then

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

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

- $\mathcal{D}/R$ can be computed using partition refinement: $\mathcal{O}(|\mathbf{P}| \cdot \log |S|)$

$\boxed{\textit{PCTL-equivalence coincides with probabilistic bisimulation}}$

Summary of model-checking PCTL

- Propositional logic: model checking à la CTL
- Probabilistic next: matrix-vector multiplication
- Probabilistic (bounded) until:
  - graph analysis to obtain states with zero probability
  - solving a linear equation system, or matrix-vector multiplication
- Worst case time-complexity:

$$\mathcal{O}(|\Phi| \cdot k_{max} \cdot M \cdot N)$$

  with $|\Phi|$ the length of Φ, M number of transitions, N the number of states, and k_{max} the largest bound ($\neq \infty$) of an until in Φ
- Tools: TPWB, VOPP, PROBVERUS, E $\vdash$ MC^2 , PRISM

Finite *continuous-time* Markov chains

A labeled CTMC $\mathcal{M}$ is a triple $(S, \mathbf{R}, L)$ with:

- *finite* discrete state space S
- the transition labels are *rates* (in $\mathbf{R}$) of the exponential distribution:

$$F_X(t) = 1 - e^{-\lambda \cdot t}, \quad t \geqslant 0; \qquad E[X] = \frac{1}{\lambda}$$

  - combinations can approximate general distributions arbitrarily closely
  - maximises “entropy” if just the mean is known
  - adequate for many phenomena (life-times, inter-arrivals, ...)

- states are labelled with sets of *atomic propositions* (notation: $L(s)$)

$\Rightarrow$ a CTMC is thus a Kripke structure with probabilistic timing information

The rate matrix $\mathbf{R}$

- With probability $1 - e^{-\mathbf{R}(s,s') \cdot t}$ transition $s \to s'$ *may* be triggered within time t

- The probability to move from non-absorbing s to s' within time t is:

$$\underbrace{\frac{\mathbf{R}(s,s')}{E(s)}}_{\substack{\text{probability} \\ s \,\to\, s' \text{ wins "race"}}} \cdot \underbrace{\left(1 - e^{-E(s) \cdot t}\right)}_{\substack{\text{probability to} \\ \text{leave } s \text{ within } t}}$$

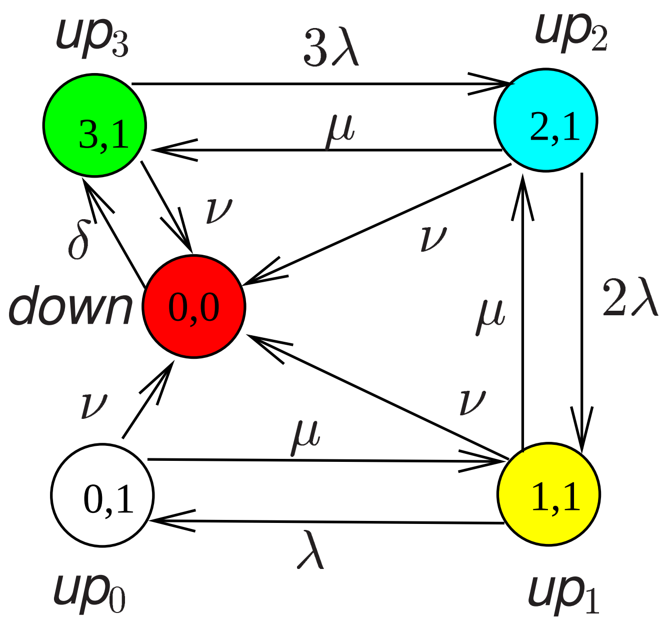
- $(S, \mathbf{P}, L)$ is the *embedded* DTMC of the CTMC $(S, \mathbf{R}, L)$
  - where $\mathbf{P}(s,s') = \frac{\mathbf{R}(s,s')}{E(s)}$ for non-absorbing s, and 0 otherwise.

Modelling techniques for CTMCs

- stochastic Petri nets [Molloy 1977]
- Markovian queueing networks [Kleinrock 1975]
- stochastic automata networks [Plateau 1985]
- stochastic activity networks [Meyer & Sanders 1985]
- stochastic process algebra [Herzog *et al.* 1993]
- probabilistic input/output automata [Smolka *et al.* 1994]

and many variants thereof

Modelling a TMR system as a CTMC



- processor failure rate is λ fph; its repair rate is μ rph
- voter failure rate is ν fph; its repair rate is δ rph
- rate matrix: e.g., $\mathbf{R}((3,1),(2,1)) = 3\lambda$
- exit rates: e.g., $\mathbf{E}(3,1) = 3\lambda+\nu$
- probability matrix: e.g.,

$$\mathbf{P}((3,1),(2,1)) = \frac{3\lambda}{3\lambda+\nu}$$

Elementary probabilities for CTMCs

- *Steady-state* probability vector $\underline{\pi} = (\cdots, \pi_i, \cdots)$
  - where π_i is the probability to be in state s_i on the long run
  - π_i is mostly *in*dependent from the starting distribution
  - $\underline{\pi}$ is computed from a system of linear equations:

$$\underline{\pi} \cdot \mathbf{Q} = \underline{0} \ \ \text{(flow balance)}, \qquad \sum_i \pi_i = 1 \ \ \text{(normalisation)}$$

  where generator matrix $\mathbf{Q} = \mathbf{R} - \mathit{diag}(E)$

- *Transient* probability vector $\underline{\pi}(t) = (\cdots, \pi_i(t), \cdots)$ for real t
  - where $\pi_i(t)$ is the probability to be in state s_i at time t (given $\underline{\pi}(0)$)
  - computed from a system of linear differential equations:

$$\underline{\pi}'(t) = \underline{\pi}(t) \cdot \mathbf{Q}, \quad \text{given } \underline{\pi}(0)$$

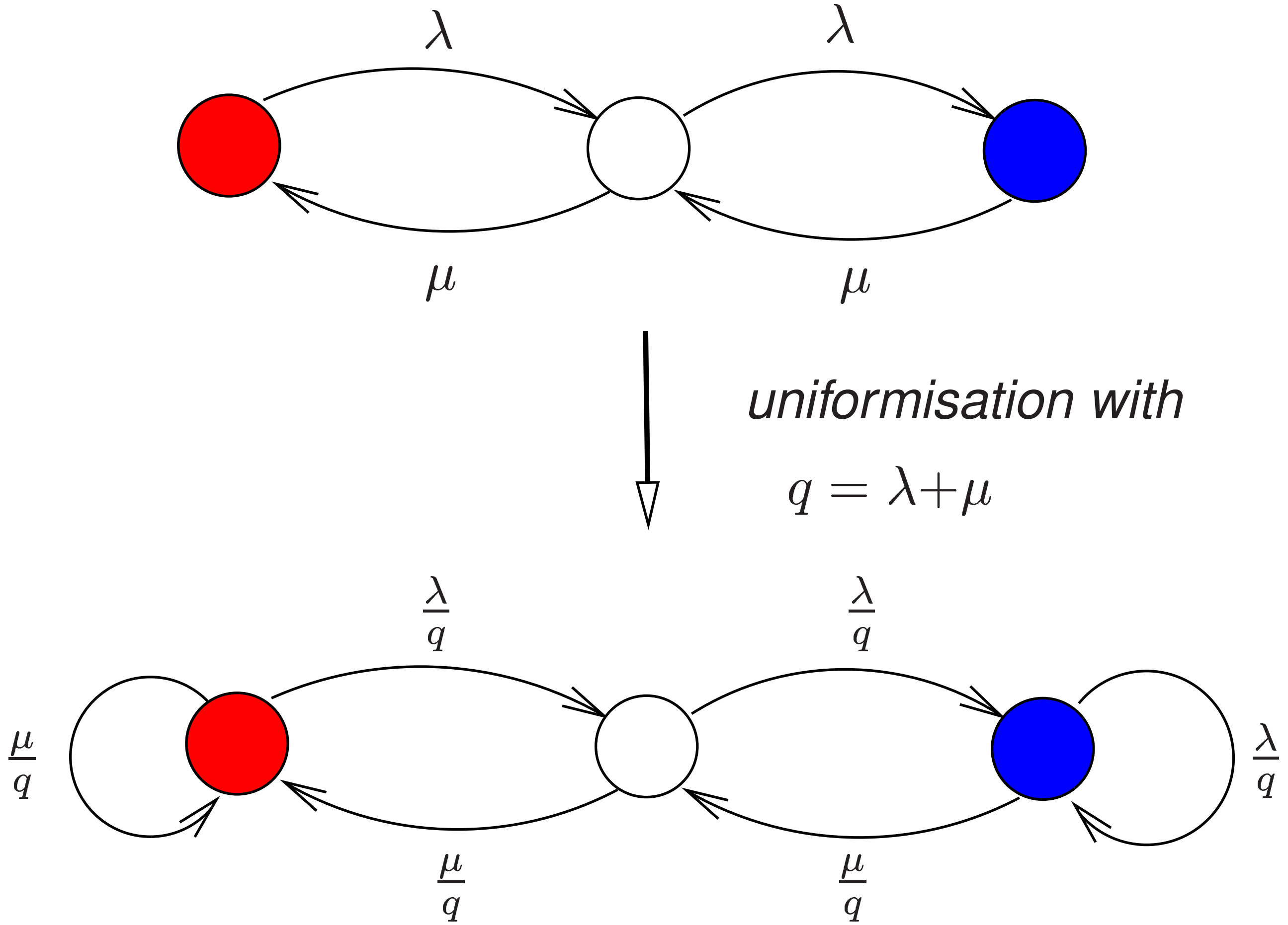
  - specialised solution method: *uniformisation*

Uniformisation

- Solution to $\underline{\pi}'(t) = \underline{\pi}(t) \cdot \mathbf{Q}$ is: $\underline{\pi}(t) = \underline{\pi}(0) \cdot e^{\mathbf{Q}t} = \underline{\pi}(0) \cdot \sum_{i=0}^{\infty} \frac{(\mathbf{Q}t)^i}{i!} \quad (*)$
- Main problems: infinite summation + numerical instability due to
  - $\mathbf{Q}^i$ becomes non-sparse with positive and negative entries
- Solution: transform CTMC $(S, \mathbf{R}, L)$ into DTMC $(S, \mathbf{U}, L)$ with
  - $\mathbf{U} := \mathbf{I} + \frac{\mathbf{Q}}{q}$ with $q > \max_i \{ E(s_i) \}$
- We can now rewrite $(*)$ as: $\underline{\pi}(t) = \underline{\pi}(0) \cdot \sum_{i=0}^{\infty} \underbrace{e^{-qt} \frac{(qt)^i}{i!}}_{\text{Poisson prob.}} \mathbf{U}^i$
  - the infinite summation can be truncated *a priori* for a given error bound

$\Rightarrow$ boils down to a transient analysis of a DTMC

Uniformisation example



Lumpability (bisimulation)

- A *bisimulation* (lumping) on $\mathcal{M} = (S, \mathbf{R}, L)$ is an equivalence R on S such that whenever $(s, s') \in R$ then

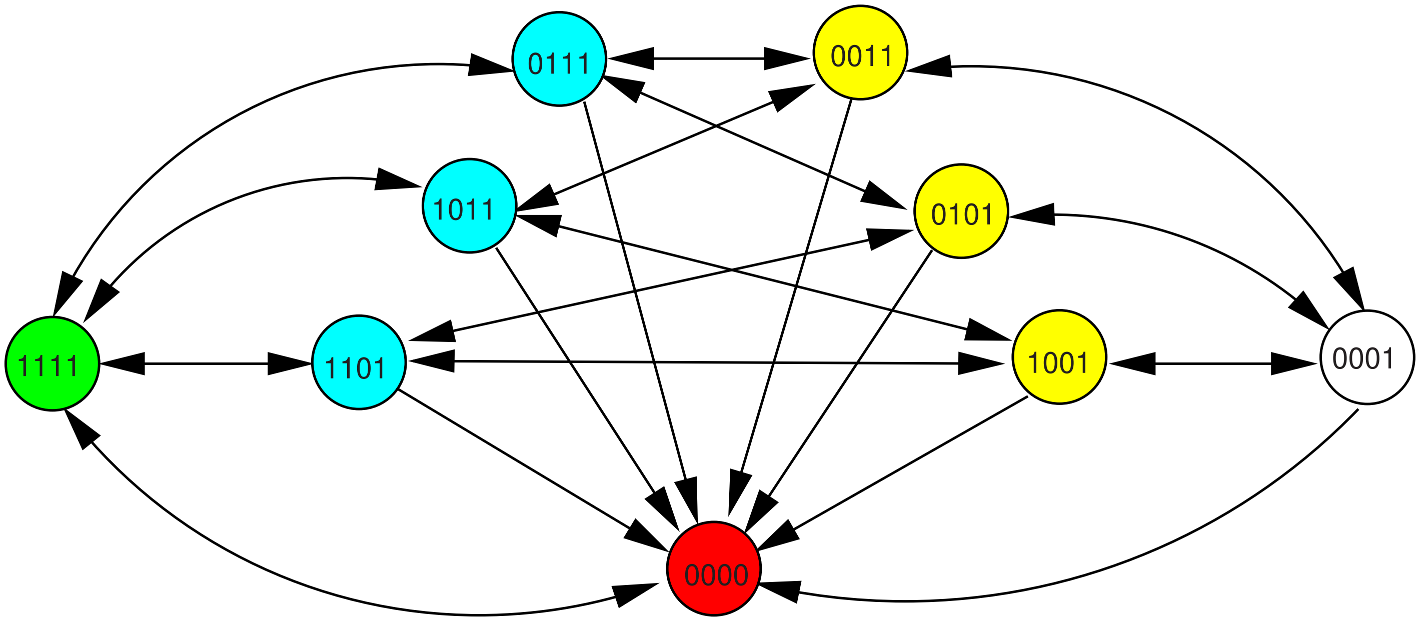
$$L(s) = L(s') \text{ and } \mathbf{R}(s, C) = \mathbf{R}(s', C) \text{ for all } C \in S/R,$$

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

- *Lumping preserves steady-state and transient characteristics.* Suppose $Z \subset S$ is lumped to a single state z, then:

$$\pi_z^{\text{new}}[t] = \sum_{s \in Z} \pi_s^{\text{old}}[t]$$

“Unlumping” states in the TMR model



$$\pi^{\mathcal{M}/R}(s_{2,1},t) = \pi^{\mathcal{M}}(0111,t) + \pi^{\mathcal{M}}(1011,t) + \pi^{\mathcal{M}}(1101,t)$$

Continuous Stochastic Logic

State-*formulas*	$\Phi \ ::= \ a \ \Big| \ \neg\,\Phi \ \Big| \ \Phi \ \vee \ \Phi \ \Big| \ \mathcal{S}_{\trianglelefteq p}(\Phi) \ \Big| \ \mathcal{P}_{\trianglelefteq p}(\varphi)$ with probability p and comparison operator $\trianglelefteq$	
$\mathcal{S}_{\trianglelefteq p}(\Phi)$	probability that Φ holds in steady state is $\trianglelefteq p$	
$\mathcal{P}_{\trianglelefteq p}(\varphi)$	probability that paths fulfill φ is $\trianglelefteq p$	

$\exists\varphi =_{fair} \mathcal{P}_{>0}(\varphi)$ and $\forall\varphi =_{fair} \mathcal{P}_{\geqslant 1}(\varphi)$

Path-*formulas*	$\varphi \ ::= \ \mathsf{X}^{I}\,\Phi \ \Big| \ \Phi\,\mathsf{U}^{I}\,\Phi$	with interval I
$\mathsf{X}^{I}\,\Phi$	next state is reached at time $t \in I$ and fulfills Φ	
$\Phi\,\mathsf{U}^{I}\,\Psi$	Φ holds along the path until Ψ holds at time $t \in I$	

CTL operators X and U are special cases

Complex measures-of-interest

- in equilibrium, with probability at most 0.9 the probability that the system will go down within 10 time units is at most 0.01:

$$\mathcal{S}_{\leqslant 0.9}(\mathcal{P}_{\geqslant 0.01}(\Diamond^{[0,10]} down))$$

- with probability at least 0.5 a state will be reached between 10 and 20 time-units (without going down), which guarantees the system to be operational with 2 or 3 processors when the system is in equilibrium:

$$\mathcal{P}_{\geqslant 0.5}(\neg down \, \mathcal{U}^{[10,20]} \, \mathcal{S}_{\geqslant 0.8}(\mathit{up}_3 \vee \mathit{up}_2))$$

$\Rightarrow$ “specification patterns” (to be done) ease the use of the logic

Semantics of CSL: state-formulas

$\mathcal{M}, s \models \Phi$ if and only if formula Φ holds in state s of CTMC $\mathcal{M}$

Relation $\models$ is defined by:

$s \models a$	iff	$a \in L(s)$
$s \models \neg\, \Phi$	iff	not $(s \models \Phi)$
$s \models \Phi \;\vee\; \Psi$	iff	$(s \models \Phi)$ or $(s \models \Psi)$
$s \models \mathcal{S}_{\bowtie p}(\Phi)$	iff	$\lim_{t \to \infty} \Pr\{\, \sigma \in \mathit{Paths}(s) \mid \sigma@t \models \Phi \,\} \trianglelefteq p$
$s \models \mathcal{P}_{\trianglelefteq p}(\varphi)$	iff	$\Pr\{\, \sigma \in \mathit{Paths}(s) \mid \sigma \models \varphi \,\} \trianglelefteq p$

$\Pr\{\ldots\}$ is measurable by a (i.e., cone) Borel space construction on paths in a CTMC

Semantics of CSL: path-formulas

A *path* in CTMC $\mathcal{M}$ is an infinite alternating sequence

$$s_0\, t_0\, s_1\, t_1 \ldots \quad \text{with} \quad \mathbf{R}(s_i, s_{i+1}) > 0 \quad \text{and} \quad t_i > 0$$

Semantics of path-formulas is defined by:

$\sigma \models \mathsf{X}^{I}\, \Phi$ iff $\sigma[1] \models \Phi$ and $t_0 \in I$

$\sigma \models \Phi\, \mathsf{U}^{I}\, \Psi$ iff $\exists t \in I.\ (\sigma@t \models \Psi \ \wedge\ (\forall t' \in [0, t).\, \sigma@t' \models \Phi))$

where $\sigma@t$ denotes the state in the path σ at time t

Model-checking CSL

- checking whether a state s in a CTMC satisfies a CSL formula is performed in the same way as for CTL:
  - compute recursively the set $Sat(\Phi)$ of states that satisfy Φ
  - check whether state s belongs to $Sat(\Phi)$

  $\Rightarrow$ bottom-up traversal of the parse tree of Φ

- for the non-stochastic part: as for CTL

- how to compute $Sat(\Phi)$ for the stochastic operators?

Model-checking the steady-state operator

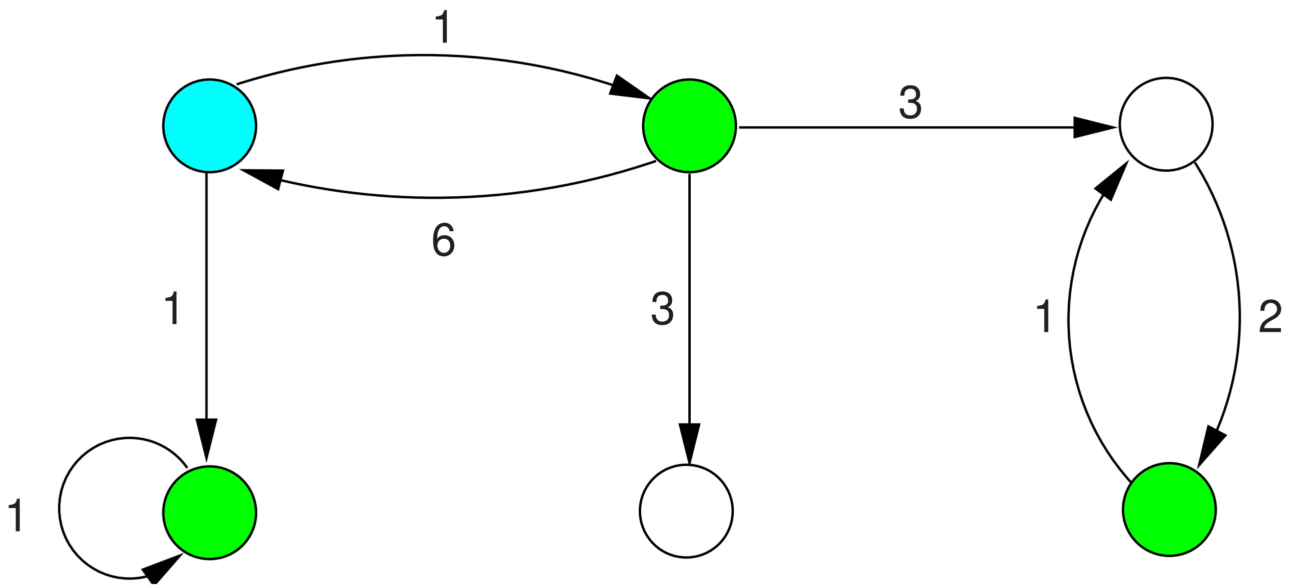
- for an ergodic (i.e., strongly-connected) CTMC:

$$s \in Sat(\mathcal{S}_{\trianglelefteq p}(\Phi)) \text{ iff } \sum_{s' \in Sat(\Phi)} \pi_{s'} \trianglelefteq p$$

$\Longrightarrow$ this boils down to a standard steady-state analysis

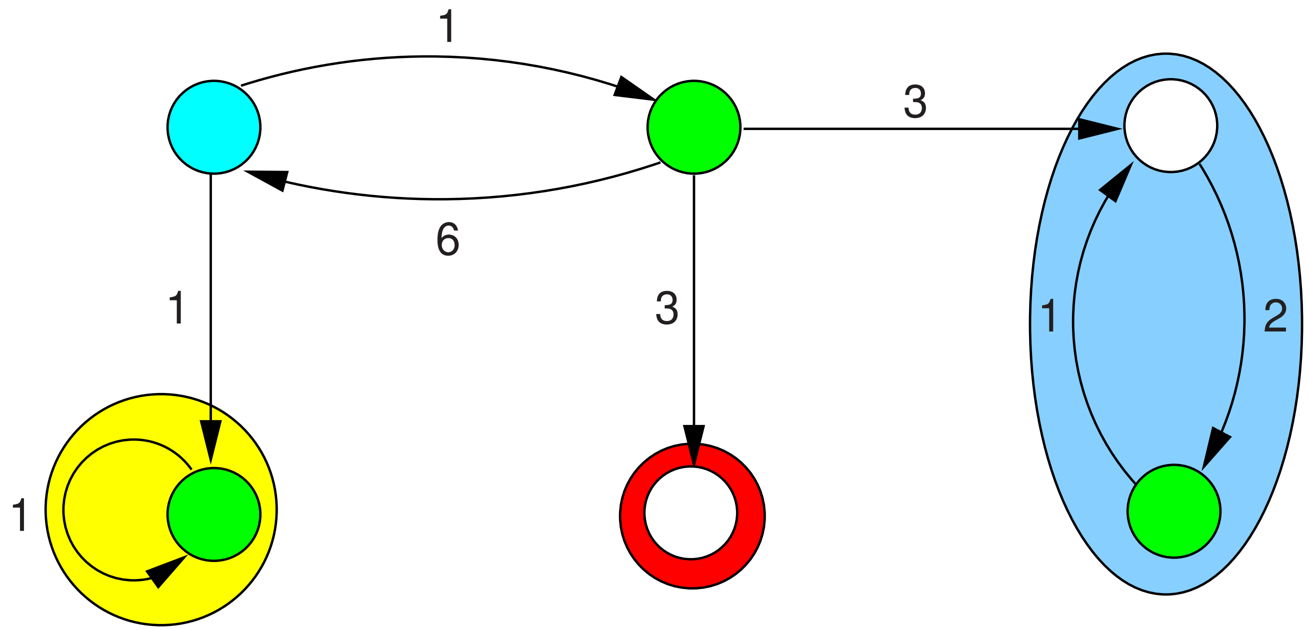
- for an arbitrary CTMC:
  - determine the *bottom* strongly-connected components (BSCCs)
  - for BSCC B determine the steady-state probability of a Φ-state
  - compute the probability to reach BSCC B from state s
  - check whether $\displaystyle\sum_{B} \left(\Pr\{ \text{reach } B \text{ from } s \} \cdot \sum_{s' \in B \cap Sat(\Phi)} \pi^{B}_{s'} \right) \trianglelefteq p$

Verifying steady-state properties: an example



determine the bottom strongly-connected components

Verifying steady-state properties: an example



$$s \models \mathcal{S}_{>0.75}(\textit{green}) \quad \text{iff} \quad Prob(s, \Diamond at_{yellow}) \cdot \pi^{yellow}(\textit{green}) + Prob(s, \Diamond at_{blue}) \cdot \pi^{blue}(\textit{green}) > 0.75$$

Characterising path measures

- $s \in Sat(\mathcal{P}_{\trianglelefteq p}(\varphi))$ iff $Prob(s, \varphi) \trianglelefteq p$
- main problem: how to compute this probability for $\Phi \, \mathsf{U}^I \, \Psi$-formulas?
- for $I = [0, t]$, probability $Prob(s, \Phi \, \mathsf{U}^{[0,t]} \Psi)$ is the least solution of:
  - 1 if $s \in Sat(\Psi)$
  - 0 if $s \notin Sat(\Psi) \cup Sat(\Phi)$
  - otherwise:

$$\int_0^t \sum_{s' \in S} \underbrace{\mathbf{R}(s, s') \cdot e^{-\mathbf{E}(s) \cdot x}}_{\text{probability to move to state } s' \text{ at time } x} \cdot \underbrace{Prob(s', \Phi \, \mathsf{U}^{[0,t-x]} \, \Psi)}_{\text{probability to fulfill } \Phi \, \mathsf{U} \, \Psi \text{ before time } t-x \text{ from } s'} \, dx$$

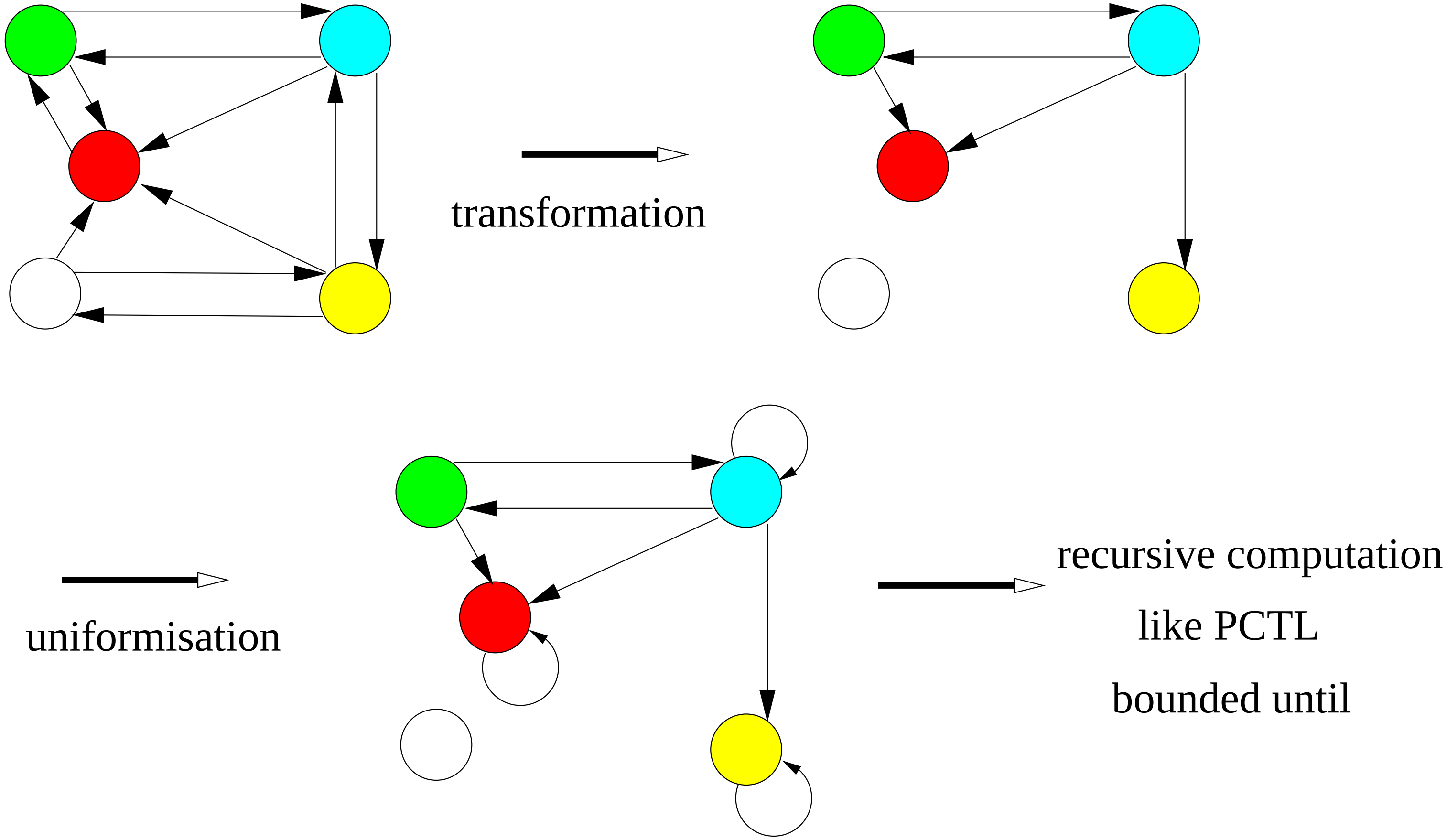
Verifying time-bounded until

- for an arbitrary CTMC $\mathcal{M}$ and property $\varphi = \Phi \, \mathsf{U}^{[0,t]} \, \Psi$ we have:
  - φ is fulfilled once a Ψ-state is reached before t along a Φ-path
  - φ is violated once a $\neg \, (\Phi \, \vee \, \Psi)$-state is visited before t

- approach: *transform* the CTMC $\mathcal{M}$ as follows
  - make all Ψ-states absorbing
  - make all $\neg \, (\Phi \, \vee \, \Psi)$-states absorbing
  - check $\Diamond^{[t,t]} \, \Psi$ in the obtained CTMC $\mathcal{M}'$
  - ... but ...
  - this amounts to a standard transient analysis in $\mathcal{M}'$: $\sum_{s \in Sat(\Psi)} \pi_s(t)$

- cases with $[t', t]$ for $t' > 0$ can be reduced to this case

Example: TMR with $\mathcal{P}_{\unlhd p}((\mathit{up}_3 \vee \mathit{up}_2)\, \mathsf{U}^{[0,3]}\, \mathit{down})$



Verifying time-bounded until (II)

CTMC $\mathcal{M}$ and property $\varphi = \Phi \, \mathsf{U}^{[t,t']} \, \Psi$ with $0 < t \leqslant t'$

- for any path σ that fulfills φ we have
  - Φ holds continuously up to time t, and
  - the suffix of path σ that starts at time t fulfills $\Phi \, \mathsf{U}^{[0,t'-t]} \, \Psi$

- approach: split the problem into two subproblems:
  - check $\Box^{[0,t]} \, \Phi$
    ... but ... this reduces to a transient analysis for t, yielding $\underline{\pi}(t)$
  - check $\Phi \, \mathsf{U}^{[0,t'-t]} \, \Psi$ with starting distribution $\underline{\pi}(t)$
    ... but ... this is the problem we have seen before

Lumpability revisited

- A bisimulation on CTMC $\mathcal{M} = (S, \mathbf{R}, L)$ is an equivalence R on S such that whenever $(s, s') \in R$ then

$$L(s) = L(s') \text{ and } \mathbf{R}(s, C) = \mathbf{R}(s', C) \text{ for all } C \in S/R$$

- Possible to parameterise this with a set of CSL-formulae

- $\mathcal{M}/R$ can be computed using partition refinement: $\mathcal{O}(|\mathbf{R}| \cdot \log |S|)$

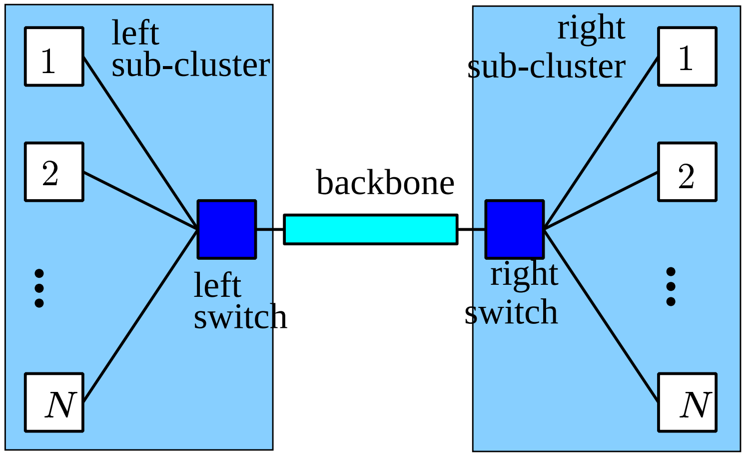
CSL-equivalence coincides with lumping

Summary of model checking CSL

- Propositional logic: model checking à la CTL
- Steady-state operator: graph analysis + linear system(s) of equations
- Time-bounded next: matrix-vector multiplication
- Time-bounded until: model transformation and uniformisation
- Worst case time-complexity: $\mathcal{O}(|\Phi| \cdot (M \cdot q \cdot t_{max} + N^{2.81}))$

  with $|\Phi|$ the length of Φ, M number of transitions, N the number of states, q the uniformisation rate, and t_{max} the largest time bound in Φ
- tools: E $\vdash$ MC^2 , PRISM, APNN TOOLBOX, DANAMICS

Application: workstation cluster

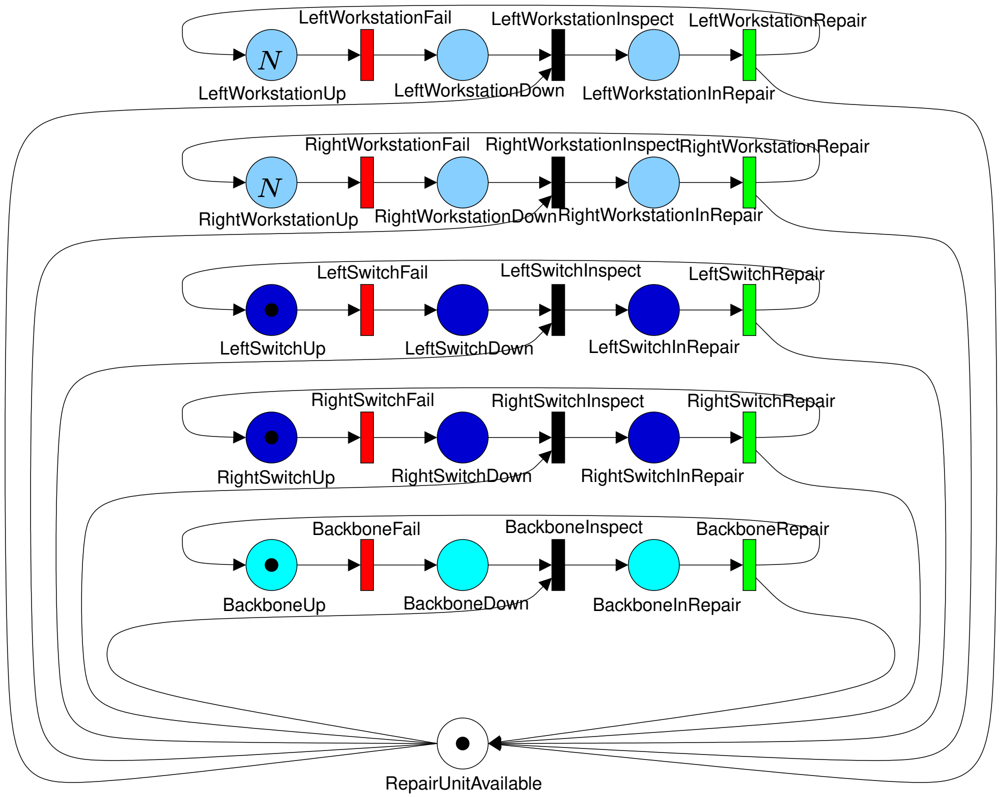


- all components are failure-prone
- single repair-unit
- mobile switching centre

minimum QoS: at least $k < N$ workstations are operational and connected via operational switches

premium QoS: at least N workstations are operational and connected via operational switches

A stochastic Petri net model



Average durations:

BackboneFail	5000h
LeftSwitchFail	4000h
RightSwitchFail	4000h
RightWSFail	500h
LeftWSFail	500h
BackboneRepair	8h
RightSwitchRepair	4h
LeftSwitchRepair	4h
RightWSRepair	0.5h
LeftWSRepair	0.5h

Abbreviations of basic formulas

$$\begin{aligned}
\mathit{LeftOp}_i &= (\#\text{LeftWorkstationUp} \geqslant i) \wedge (\#\text{LeftSwitchUp} > 0) \\
\mathit{RightOp}_i &= (\#\text{RightWorkstationUp} \geqslant i) \wedge (\#\text{RightSwitchUp} > 0) \\
\mathit{Conn} &= \quad (\#\text{LeftSwitchUp} > 0) \wedge (\#\text{BackboneUp} > 0) \\
&\quad \wedge (\#\text{RightSwitchUp} > 0) \\
\mathit{Op}_i &= (\#\text{LeftWorkstationUp} + \#\text{RightWorkstationUp} \geqslant i) \wedge \mathit{Conn} \\
\mathit{Min} &= \mathit{LeftOp}_k \vee \mathit{RightOp}_k \vee \mathit{Op}_k \\
\mathit{Prem} &= \mathit{LeftOp}_N \vee \mathit{RightOp}_N \vee \mathit{Op}_N
\end{aligned}$$

Note that *Min* and *Prem* characterize the desired QoS levels

Dependability measures of interest

- on the long run, 70% of the time premium QoS: $\mathcal{S}_{\geqslant 0.70}(\textit{Prem})$
- the system will always be able to offer premium QoS at some point in the future: $\mathcal{P}_{\geqslant 1}(\Diamond \textit{Prem})$
- the probability that QoS drops below minimum quality within 85 time units is less than 10%: $\textit{Min} \;\Rightarrow\; \mathcal{P}_{<0.10}(\Diamond^{[0,85]} \neg \textit{Min})$
- the probability to turn from minimum QoS to premium QoS without violating the minimum QoS constraint on the way is more than 99%:

$$\textit{Min} \wedge \neg \textit{Prem} \Rightarrow \mathcal{P}_{\geqslant 0.99}(\textit{Min} \;\mathsf{U}\; \textit{Prem})$$

Model generation

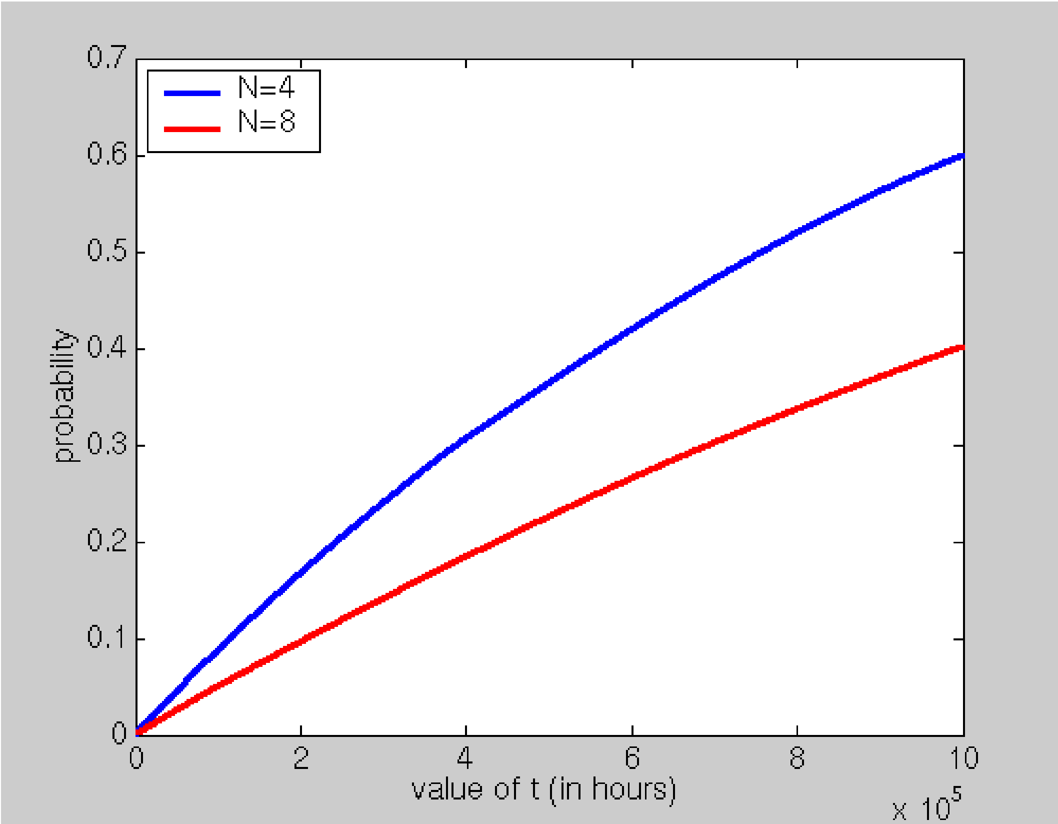
N	# *states*	# *non-zero rates*	*memory (in Kb)*	*construction time (in s)*
4	820	3,616	51.9	0.14
8	2,772	12,832	183	0.15
16	10,132	48,160	683	0.20
32	38,676	186,400	2,637	0.5
64	151,060	733,216	10,362	1.65
128	0.6 x 10^6	2.9 x 10^6	41,076	6.20
256	2.3 x 10^6	1.2 x 10^7	162,582	23.43
512	9.5 x 10^6	4.6 x 10^7	649,900	100.96

440 MHz SUN Ultra-10, 0.5 Gb RAM, relative error $< 10^{-6}$

Verification times

N	$\mathcal{S}_{\geqslant 0.70}(\mathit{Prem})$			$\mathit{Min} \Rightarrow \mathcal{P}_{<0.10}(\Diamond^{[0,85]} \neg \mathit{Min})$		
	time per iteration (in s)	# iterations	total time (in s)	time per iteration (in s)	# iterations	total time (in s)
4	0.00031	64	0.02	0.00011	3,971	0.44
8	0.00085	82	0.07	0.00053	3,972	2.11
16	0.0035	141	0.49	0.0023	3,975	9.14
32	0.014	260	3.64	0.0088	3,980	35.0
64	0.14	540	75.6	0.072	3,992	287.4
128	0.39	1,276	497.6	0.3	4,017	1205
256		3,634		1.02	4,064	4145
512		30,476		12.41	4,159	51613

Experimental results



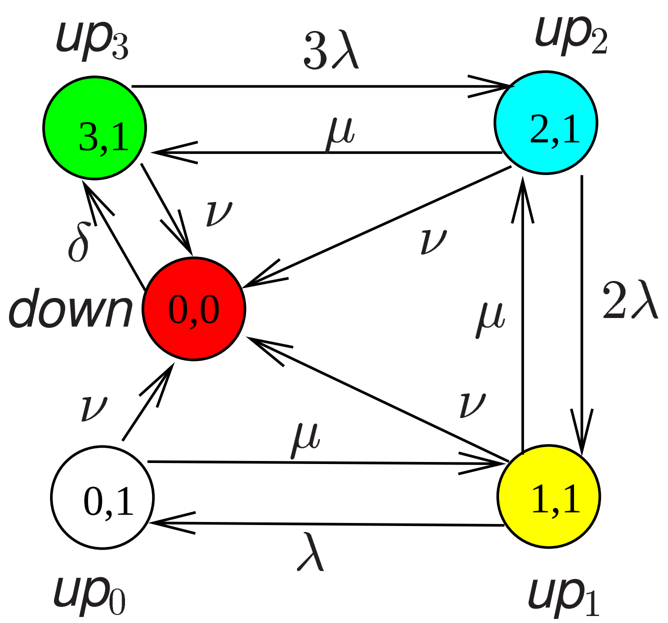
probability that QoS drops below minimum within t time units

Performance $+$ Dependability $=$ Performability

- interplay of performance *and* dependability is important, e.g.,
  - what is the performance of a communication network if connections break?
  - what is the efficiency of a production line if machines fail?
- popular model for performability: Markov reward model (MRM)
- MRM: CTMC equipped with a reward structure ρ

  that assigns to each state s a (positive real) rate reward $\rho(s)$
- when staying t time units in state s, a reward $\rho(s) \cdot t$ is earned
- impulse rewards associated with transitions not considered here

An MRM of the TMR system



Example reward assignments:

Availability:

$$\rho(0,0) = \rho(0,1) = 0$$

$$\rho(i,1) = 1 \text{ for } i \geqslant 1$$

System capacity:

$$\widehat{\rho}(0,0) = 0$$

$$\widehat{\rho}(i,1) = k \times i$$

where k jobs are processed per processor per time unit

A logic for performability measures: CSRL

Basic idea: decorate the path operators X and U with constraints on the time spent (as before) *and* on the reward accumulated

Path-formulas	$\varphi \ ::= \ \mathsf{X}^{I}_{J}\,\Phi \ \Big\vert \ \Phi\,\mathsf{U}^{I}_{J}\,\Phi$	with intervals I, J
$\mathsf{X}^{I}_{J}\,\Phi$	the next state fulfills Φ and is reached at time $t \in I$ while earning a cumulative reward $r \in J$ up to t	
$\Phi\,\mathsf{U}^{I}_{J}\,\Psi$	Φ holds along the path until Ψ holds at time $t \in I$ while earning a cumulative reward $r \in J$ up to t	

state-formulas remain unchanged and X^{I}, X, U^{I} and U are special cases

Examples of performability measures

- the probability of processing at least 50 jobs before the system fails exceeds 0.8:

$$\mathcal{P}_{>0.8}(\neg \mathit{down}\, \mathsf{U}_{[50,\infty)}\, \mathit{down})$$

- with probability at least 0.95 the expected number of jobs in process at time instant 60 is at most 2:

$$\mathcal{P}_{\geqslant 0.95}(\Diamond^{[60,60]}_{[0,2]} \neg \mathit{down})$$

- with probability at least 0.98 at least 7 jobs have been processed before the first processor fails within 30 time units:

$$\mathcal{P}_{\geqslant 0.98}(\mathit{up}_3\, \mathsf{U}^{[0,30]}_{[7,\infty)}\, \mathit{up}_2)$$

Duality

Idea: regard progress of time as earning of reward and vice versa

- transform MRM $\mathcal{M} = (S, \mathbf{R}, L, \rho)$ into $\mathcal{M}^{-1} = (S, \mathbf{R}', L, \rho')$ with
  - $\mathbf{R}'(s, s') = \mathbf{R}(s, s')/\rho(s)$ rescale transition rates by rewards
  - $\rho'(s) = 1/\rho(s)$ invert the reward structure

  $\Rightarrow$ transformation restricted to non-zero rewards

- transform state-formula Φ into Φ^{-1} by
  - $\mathrm{X}_J^I \rightarrow \mathrm{X}_I^J$ and $\mathrm{U}_J^I \rightarrow \mathrm{U}_I^J$ swap reward and time bounds

- Duality theorem: $Sat^{\mathcal{M}}(\Phi) = Sat^{\mathcal{M}^{-1}}(\Phi^{-1})$

$\Rightarrow$ $\boxed{\text{verifying } \mathrm{U}_J \text{ (in } \mathcal{M}) \text{ is identical to model-checking } \mathrm{U}^J \text{ (in } \mathcal{M}^{-1})}$

Overview of CSRL model checking

- fragment without rewards: procedures à la CSL
- reward-bounded until: duality transformation + uniformisation
- time- and reward-bounded until:

method	space	time
approximation	$\mathcal{O}(N^2 \cdot k^2)$	$\mathcal{O}(k_\varepsilon \cdot (N^2 \cdot k^2)$
discretisation	$\mathcal{O}(N \cdot \frac{r}{d})$	$\mathcal{O}(N \cdot t \cdot \lvert t - r \rvert \cdot d^{-2})$
occupation times	$\mathcal{O}(k_\varepsilon^2 \cdot N^3)$	$\mathcal{O}(k_\varepsilon^3 \cdot N^3)$

N:	# states	t:	time bound	r:	cost bound
k:	cost bound discretisation	d:	stepsize	k_ε:	uniformisation step count

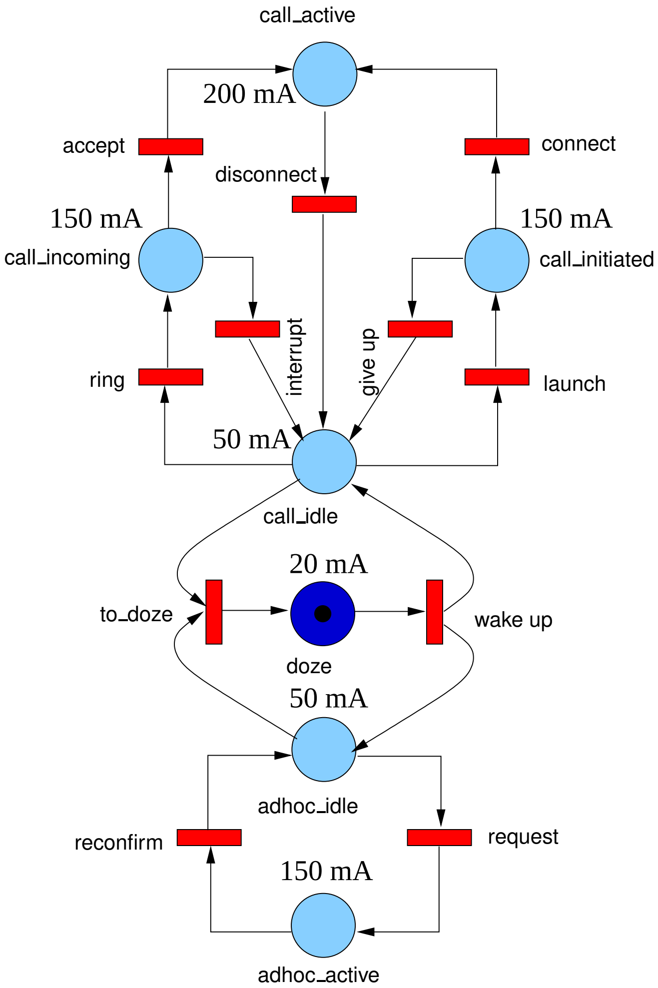
Application: mobile ad-hoc network

- single battery-powered mobile phone with ad-hoc traffic

  such as the IEEE 802.11 standard or Hyperlan

- two types of traffic: ad-hoc traffic and ordinary calls
  - offer transmission capabilities for data transfer between third parties (altruism)
  - normal call traffic
- rate rewards are used to model power consumption
  - in *doze* mode (20 mA), calls can neither be made nor received
  - active calls are assumed to consume 200 mA
  - ad-hoc traffic and call handling takes 120 mA; idle mode costs 50 mA
  - total battery capacity is 750 mAh; reward equals one mA

A stochastic reward Petri net model



transition	mean time (in min)	rate (per h)
accept	20	180
connect	10	360
disconnect	4	15
doze	5	12
give up	1	60
interrupt	1	60
launch	80	0.75
reconfirm	4	15
request	10	6
ring	80	0.75
wake up	16	3.75

Performability measures of interest

- The probability to receive a call within 24 hours exceeds 0.5:
  $\mathcal{P}_{>0.5}(\Diamond^{\leqslant 24}\ \textit{call\_incoming})$

- The probability to receive a call while having consumed at most 80% power exceeds 0.5: $\mathcal{P}_{>0.5}(\Diamond_{\leqslant 600}\ \textit{call\_incoming})$

- The probability to launch a call before having consumed at most 80% power within 24 hours – while using the phone only for ad-hoc transfer beforehand – exceeds 0.5:

$$\mathcal{P}_{>0.5}((\textit{call\_Idle} \vee \textit{doze})\ \ \mathcal{U}^{\leqslant 24}_{\leqslant 600}\ \ \textit{call\_initiated})$$

Experimental results

occupation time distribution			
ε	k_ε	*numerical value*	*time (in s)*
10^{-2}	519	0.49068833	83.0
10^{-4}	551	0.49536172	94.8
10^{-6}	574	0.49540351	103.1
10^{-8}	594	0.49540399	110.8
discretization			
step size	*numerical value*	*relative error*	*time (in s)*
1/32	0.49566676	0.05 %	26.7
1/64	0.49553603	0.03 %	107.6
1/128	0.49547017	0.01 %	431.9
1/256	0.49543712	$<$ 0.01 %	1712
Erlangian approximation			
# phases	*numerical value*	*relative error*	*time (in s)*
16	0.49177955	0.73 %	0.01
64	0.49457832	0.17 %	0.05
256	0.49520304	0.04 %	0.5
1024	0.49535410	0.01 %	21.34

Very large state spaces

Techniques to handle very large state spaces:

- aggregation techniques that preserve the validity of formulas
  - lumping equivalent CTMCs satisfy the same CSL-formulas
  - more aggressive aggregation techniques useful for weak preservation
- exploit compositionality using Kronecker representations
  - model checking can be done using “standard” Kronecker operations
- use symbolic data structures such as binary decision diagrams
  - model checking can be done using “standard” MTBDD operations
- distributed model checking (CTL) using a workstation cluster

workstation cluster with 1024 stations and 37 x 10^6 states treated

Epilogue

- logic-based specification languages for performance, dependability and performability properties
- automated verification techniques for these properties
- application examples using stochastic Petri nets
- in more general terms:

cross-fertilization of model checking
and performance/dependability evaluation techniques