Probabilistic Model Checking

Marta Kwiatkowska

THE UNIVERSITY
OF BIRMINGHAM

Model Checking Summer School SFM-02:MC

`http://www.cs.bham.ac.uk/~mzk`

Bertinoro, September 2002

Overview of Tutorial

- Motivation
  - Probability basics
  - Dice example

- Discrete Markovian models
  - The logic PCTL
  - Model checking for PCTL
  - Randomised leader election example

- Continuous time Markov chains
  - The logic CSL
  - Model checking for CSL
  - Wireless cell example

- Probabilistic timed automata
  - Model checking for probabilistic timed automata
  - IEEE 1394 root contention protocol example

Motivation

- **Probability used widely:**
  - as a symmetry breaker in distributed algorithms
    "leader election is eventually resolved with probability 1"
  - to model failures
    "the chance of shutdown occurring is at most 1%"
  - to model soft deadlines, QoS
    "the chance of a frame being delivered within 5ms is at least 89%"
  - to model system performance
    "in the long run, mean waiting time in a lift queue is 30sec"

- **Need software tools for**
  - building models of probabilistic systems
  - model checking such systems
    - functional correctness, performance/QoS prediction, etc

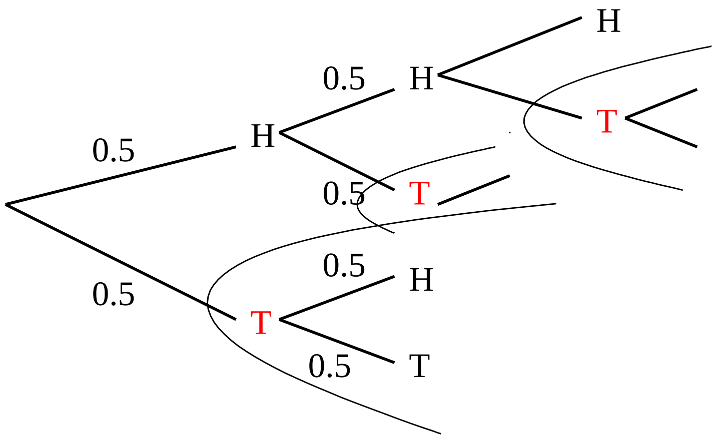
- **For example see probabilistic model checkers:**
  - PRISM (Birmingham), `http://www.cs.bham.ac.uk/~dxp/prism/`
  - ETMCC (Erlangen-Twente), `http://www7.informatik.uni-erlangen.de/etmcc/`
  - TwoTowers (Bologna), `http://www.sti.uniurb.it/bernardo/twotowers/`
  - RAPTURE, `http://www.irisa.fr/prive/bjeannet/prob/prob.html`

Probability

- We need an experiment:
  - sample set Ω, its set of possible outcomes
  - event, a subset of Ω
  - can form events: $A \cup B$, $A \cap B$, $\neg A$
- Examples
  - Toss a coin:
    - Sample set: {H,T}
    - Events: "result is H", "result is T"
  - Toss two coins:
    - Sample set: {(H,H),(H,T),(T,H),(T,T)}
    - Events: "the first is H", "at least one H"
  - Toss a coin ∞-often:
    - Sample set: infinite sequences, e.g. HTTHTTT....)
    - Events: "the first is H", "eventually T"
- Probability assigned to sample points:
  - Pr["the first is H"] $= \frac{1}{2}$
  - Pr["at least one H"] $= \frac{3}{4}$
  - Pr["eventually T"] $=$?

Probability space

- First define $(\Omega, \mathcal{F})$ sample space where $\mathcal{F}$ is σ-field of subsets (closed under countable unions and intersections, and complement)



  - Build sample space from basic cylinders (cones)
    - $\mathcal{B}$(H), all sequences beginning with H, probability $\frac{1}{2}$
    - $\mathcal{B}$(HT), all sequences beginning with HT, probability $\frac{1}{2} \cdot \frac{1}{2}$, etc

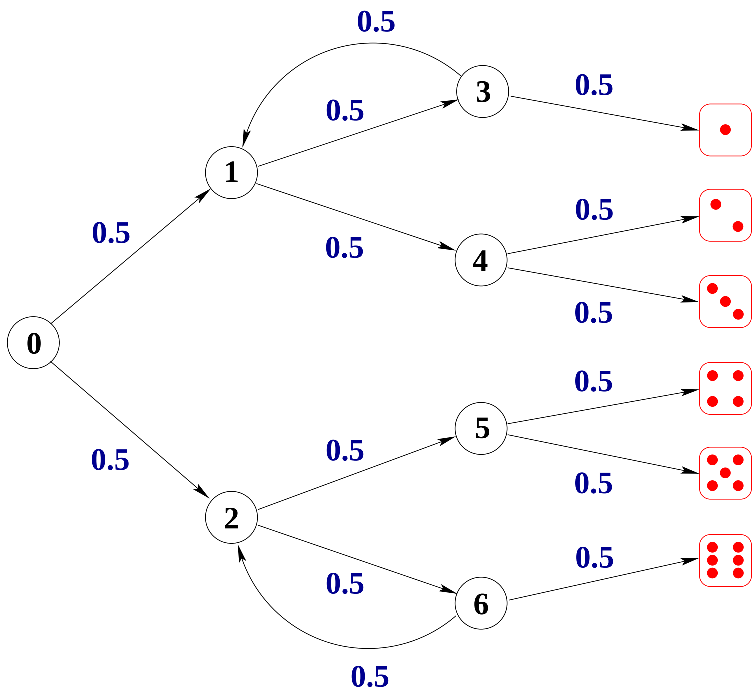
    by closing w.r.t. countable unions and complement.

- $(\Omega, \mathcal{F}, \mathrm{Pr})$ is probability space if Pr is probability measure, i.e.
  - $0 \leq \mathrm{Pr}[A] \leq 1$ for all $A \in \mathcal{F}$; $\quad \mathrm{Pr}[\emptyset] = 0, \ \ \mathrm{Pr}[\Omega] = 1$
  - $\mathrm{Pr}[\bigcup_{k=1}^{\infty} A_k] = \Sigma_{k=1}^{\infty} \mathrm{Pr}[A_k]$, A_k disjoint

- Now, since cones for T, HT, HHT, H...HT, etc disjoint
  - $\mathrm{Pr}[\text{“eventually T”}] = \frac{1}{2} + \frac{1}{4} + \frac{1}{8} + ... = 1$

Dice example [Knuth]

- Can one model a dice using only fair coins?



  - Take upper branch when H, lower branch when T
  - Continue until the value of die is decided

- What is the probability of obtaining 6?
  - Pr["eventually 6"] $= (\frac{1}{2})^3 + (\frac{1}{2})^5 + (\frac{1}{2})^7 + \ldots = \frac{1}{6}$
    since disjoint cones are THH, THTHH, THTHTHH, ...

Dice example [Knuth] continued...

- Derive recursive linear equations
  - let x_i denote the probabilities for circled states, $i = 1, 2, ..6$
  - set the probability for dice value 6 state to 1, all other values to 0
  - write down flow equations (omitting variables that reduce to 0)

$$
\begin{aligned}
x_0 &= \tfrac{1}{2} x_2 \\
x_2 &= \tfrac{1}{2} x_6 \\
x_6 &= \tfrac{1}{2} x_2 + \tfrac{1}{2} \cdot 1
\end{aligned}
$$

- Solving the equations yields unique solution

$$
\begin{aligned}
x_0 &= \tfrac{1}{6} \\
x_2 &= \tfrac{1}{3} \\
x_6 &= \tfrac{2}{3}
\end{aligned}
$$

- Hence the probability of obtaining 6 in start state
  - $\Pr[\text{“eventually 6”}] = \frac{1}{6}$

- See PRISM webpage for the dice model and model checking info
  - `http://www.cs.bham.ac.uk/~dxp/prism/`

Probabilistic model checking

- The problem
  - Given a probabilistic model, e.g. randomised leader election containing

```
coin := 0.5: tail + 0.5: heads
```

  - and a (probabilistic) specification
    "leader election eventually resolved with probability 1"
  - establish if the specification holds, or return the probability

- The models
  - probabilistic choice only: discrete time Markov chains (DTMCs)
  - probabilistic & nondeterministic choice: Markov decision processes (MDPs)
  - real time & probabilistic choice: continuous time Markov chains (CTMCs)
  - dense time, probabilistic choice and nondeterminism: probabilistic timed automata (PTAs)

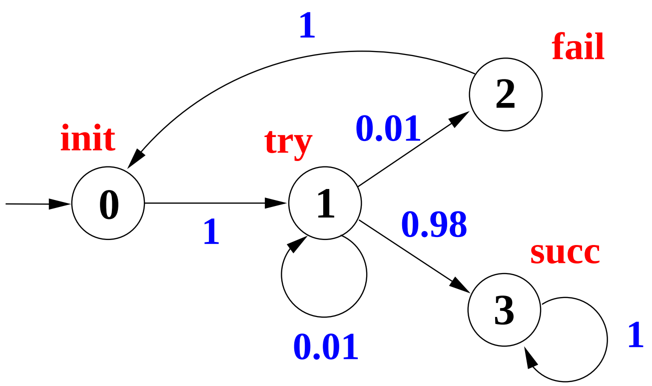
- Specification languages
  - Probabilistic variants of temporal logics
  - Qualitative/quantitative analysis

Overview of Tutorial

- Motivation
  - Probability basics
  - Dice example
- Discrete Markovian models
  - The logic PCTL
  - Model checking for PCTL
  - Randomised leader election example
- Continuous time Markov chains
  - The logic CSL
  - Model checking for CSL
  - Wireless cell example
- Probabilistic timed automata
  - Model checking for probabilistic timed automata
  - IEEE 1394 root contention protocol example

Discrete Time Markov Chains

- Features
  - states and time discrete; only probabilistic choice in each state



- Formally, Discrete Time Markov Chain (DTMCs) is a tuple $M = (S, \mathrm{P}, L)$, where
  - set of states S
  - transition probability matrix $\mathrm{P} : S \times S \longrightarrow [0, 1]$, such that $\sum_{s'} \mathrm{P}(s, s') = 1$ with $\mathrm{P}(s, s')$ probability of making transition from state s to s'
  - $L : S \longrightarrow 2^{AP}$ state labelling with atomic prop.

- Probability space based on paths:
  - unfold DTMCs into infinite paths from a given state
  - take set of infinite paths extending of a finite path as basic events
  - for finite paths accumulate probab., e.g. probability of path 0113 is $1 \cdot 0.01 \cdot 0.98$

Probability measure formally

- **Event space $(Paths_{\text{inf}}, \mathcal{F})$**
  - Sample **outcomes** are (finite/infinite) **paths** π from some **start state**:

$$s_0 s_1 s_2 \cdots s_j \cdots$$

    such that $\mathrm{P}(s_j, s_{j+1}) > 0$ for all $j \geq 0$.
  - **Basic events** are basic cylinders $\mathcal{B}(\rho)$

$$\mathcal{B}(\rho) = \{\pi \text{ infinite } \mid \rho \text{ is a prefix of } \pi\}$$

    where $\rho = s_0 s_1 s_2 s_3 \cdots s_k$ is a **finite path**
  - **Events** are elements of the the **smallest σ-algebra** $\mathcal{F}$ containing $\mathcal{B}(\rho)$

- **Probability measure**
  - for basic cylinders

$$\Pr(\mathcal{B}(\rho)) = \mathrm{P}(s, s_1) \cdot \mathrm{P}(s_1, s_2) \cdots \mathrm{P}(s_{k-1}, s_k)$$

  - Pr extends uniquely to the σ-algebra $\mathcal{F}(s)$

- **Probability space is $(Paths_{\text{inf}}, \mathcal{F}, \Pr)$, usually parameterised by start state**

The logic PCTL

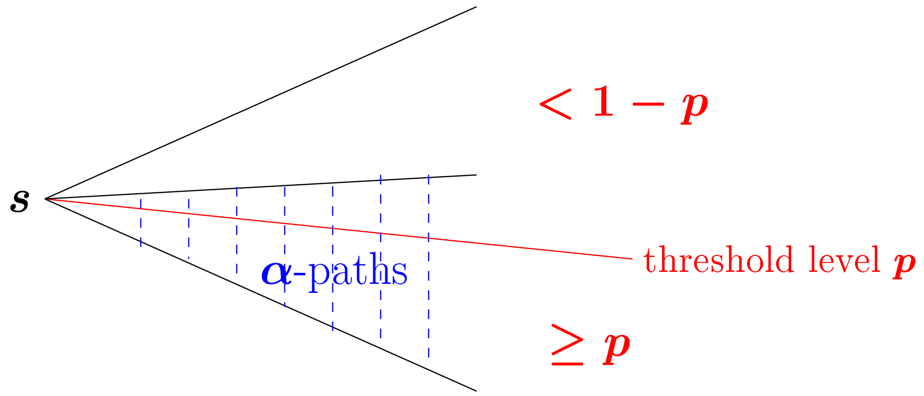
Based on CTL + probabilistic operator $\mathcal{P}_{\bowtie p}(\cdot)$ [Hansson & Jonsson 1994].

- The syntax of state and path formulas of PCTL is:

$$\begin{array}{rcl} \phi & ::= & \textsf{true} \mid a \mid \phi_1 \wedge \phi_2 \mid \neg\phi \mid \mathcal{P}_{\bowtie p}(\alpha) \\ \alpha & ::= & X\phi \mid \phi_1 \; U \; \phi_2 \end{array}$$

  where $p \in [0,1]$ is a probability bound or threshold, and $\bowtie \in \{\leq, <, \geq, >\}$.

- Meaning of probabilistic operator $\mathcal{P}_{\bowtie p}(\cdot)$:
  - applies to path formulas α
  - involves computing probability measure Pr of the set of α-paths
  - yields a (Boolean) state formula



PCTL over Discrete Markov Chains

- Semantics of state formulas ϕ:

$$
\begin{array}{llll}
M, s & \models & \mathsf{true} & \text{for all } s \\
M, s & \models & a & \text{iff } a \in L(s) \\
M, s & \models & \neg\phi & \text{iff } M, s \not\models \phi \\
M, s & \models & \phi_1 \wedge \phi_2 & \text{iff } M, s \models \phi_1 \text{ and } M, s \models \phi_2 \\
 & & & \\
M, s & \models & \mathcal{P}_{\bowtie p}(\alpha) & \text{iff } \Pr\{\pi \mid M, \pi \models \alpha, \pi \text{ starts in } s\} \bowtie p
\end{array}
$$

- Semantics of path formulas α:

$$
\begin{array}{llll}
M, \pi & \models & X\phi & \text{iff } \pi = s_0 s_1 s_2 \cdots \text{ and } M, s_1 \models \phi \\
M, \pi & \models & \phi_1 U \phi_2 & \text{iff } \pi = s_0 s_1 s_2 \cdots \text{ and } \exists k \text{ s.t.} \\
 & & & \quad M, s_k \models \phi_2 \text{ and } \forall j < k . M, s_j \models \phi_1 \\
 & & & \quad \text{(i.e. } \phi_1 \text{ holds until } \phi_2 \text{ holds and } \phi_2 \text{ must become } \mathsf{true}\text{)}
\end{array}
$$

More on PCTL

- Boolean operators only allowed for state formulas
  (otherwise conditional probabilities needed)

- Note on negation:
  $s \models \mathcal{P}_{\geq p}(\alpha)$ iff $s \models \neg\mathcal{P}_{<1-p}(\alpha)$
  $s \models \mathcal{P}_{>p}(\alpha)$ iff $s \models \neg\mathcal{P}_{\leq 1-p}(\alpha)$

- Can add bounded until $\phi_1 \ U^{\leq k} \ \phi_2$, k counts steps
  $$M, \pi \models \phi_1 U^{\leq k} \phi_2 \text{ iff } \pi = s_0 s_1 s_2 \cdots \text{ and } \exists l \leq k \text{ s.t. } M, s_l \models \phi_2 \text{ and } \forall j < l. M, s_j \models \phi_1$$

- Derived operators:
  - $\vee$, $\rightarrow$, etc
  - $\Diamond\phi \stackrel{\text{def}}{=} \textsf{true} \ U \ \phi$, $\quad \Box\phi \stackrel{\text{def}}{=} \neg\Diamond\neg\phi$

- CTL path quantifiers $\forall$, $\exists$ omitted since:
  - no effect on DTMCs
  - can derive analogues using:

    $\mathcal{P}_{>0}(\Diamond\phi)$ (cf $\exists\Diamond\phi$)
    $\mathcal{P}_{\geq 1}(\Diamond\phi)$ (cf $\forall\Diamond\phi$)

Some examples

- $\mathcal{P}_{>0}(X\,fail)$: with positive probability *fail* occurs in next state

  $s_0 \not\models \mathcal{P}_{>0}(X\,error)$

  $s_1 \models \mathcal{P}_{>0}(X\,error)$

- $\mathcal{P}_{\geq 0.98}(try\,U\,succ)$: *try* until *succ* with probability 0.98 or greater

  Sum up probabilities over disjoint cones:

  $s_1 s_3 \ldots$

  $s_1 s_1 s_3 \ldots$

  $s_1 s_1 s_1 s_3 \ldots$

  $\ldots$

  $(s_1)^n s_3 s_3 \ldots$

$$\sum_{n=0}^{\infty} 0.01^n \cdot 0.98 \;=\; 0.989898\ldots$$

  Thus $s_1 \models \mathcal{P}_{\geq 0.98}(try\,U\,succ)$

- Note that
  - $s_1 \models \mathcal{P}_{\geq 1}(\Diamond succ)$ (since probability measure of $s_1 s_1 s_1 \ldots$ is zero)
  - but for CTL formula $s_1 \not\models \forall \Diamond succ$ (since $s_1 s_1 s_1 \ldots \not\models \Diamond succ$)

Model Checking for PCTL

The model checking algorithm proceeds as for CTL by induction on ϕ.

Input:	DTMC $M = (S, \mathrm{P}, L)$, ϕ a PCTL formula
Output:	$Sat(\phi) = \{s \in S \mid M, s \models \phi\}$
case ϕ of	
true:	return S
a:	return $\{s \in S \mid a \in L(s)\}$
$\phi_1 \wedge \phi_2$:	return $Sat(\phi_1) \cap Sat(\phi_2)$
$\neg\psi$:	return $S \setminus Sat(\psi)$
$\mathcal{P}_{\bowtie p}(\alpha)$:	for each s $x_s := \Pr\{\pi \mid M, \pi \models \alpha, \pi \text{ starts in } s\}$ return $\{s \in S \mid x_s \bowtie p\}$

How to Calculate the Probabilities

- Represent
  - DTMC as its matrix P
  - $Sat(\phi)$ as a column vector $\mathrm{b}^{\phi} : S \longrightarrow \{0,1\}$ given by

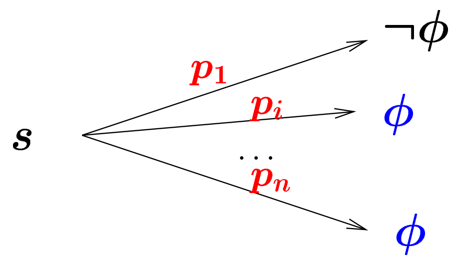
$$\mathrm{b}^{\phi}_{s} = \begin{cases} 1 & \text{if } M, s \models \phi \\ 0 & \text{otherwise} \end{cases}$$

- For the example,
  - the matrix P and the vector b^{fail} for $Sat(fail)$ are:

$$\mathrm{P} \;=\; \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0.01 & 0.01 & 0.98 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \qquad \mathrm{b}^{fail} \;=\; \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix}$$

Model Checking Next State

- Computing probability of paths satisfying $X\phi$:
  for each $s \in S$,
  sum up the probabilities of transitions to ϕ-states



$$x_s = \sum_{t \in Sat(\phi)} \mathrm{P}(s,t)$$

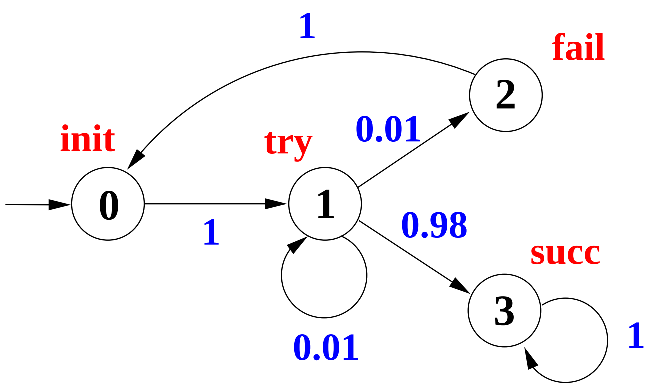
- Equivalently, express as matrix-by-vector multiply

$$\mathrm{x} = \mathrm{P} \cdot \mathrm{b}^{\phi}$$

where
P is the matrix and
b^{ϕ} is the column vector for $Sat(\phi)$.

Linear equations

- Consider the path formula $\mathcal{P}_{\geq 0.98}(try\,U\,succ)$:



- Derive recursive linear equations
  - let x_s denote the probabilities, for $s \in S$
  - write down flow equations:

$$\begin{aligned} x_0 &= 0 \\ x_1 &= 0.01 \cdot x_1 + 0.01 \cdot x_2 + 0.98 \cdot x_3 \\ x_2 &= 0 \\ x_3 &= 1 \end{aligned}$$

- Solving the equations yields unique solution

$$x_0 = 0,\ x_1 = \frac{98}{99},\ x_2 = 0,\ x_3 = 1$$

Equations for PCTL Until

- Consider $\mathcal{P}_{\bowtie p}(\phi_1 \, U \, \phi_2)$

  Calculate:

$$
\begin{aligned}
S^{YES} &:= Sat(\phi_2) \\
S^{>0} &:= Reach(\phi_1, \phi_2) \text{ (states from which one can reach} \\
&\quad\ \text{a } \phi_2\text{-state via } \phi_1\text{-states)} \\
S^{NO} &:= S \backslash S^{>0} \\
S^{?} &:= S \backslash (S^{NO} \cup S^{YES})
\end{aligned}
$$

  and define

$$
x_s = \begin{cases} 0 & \text{if } s \in S^{NO} \\ 1 & \text{if } s \in S^{YES} \\ \sum_{t \in S} \mathrm{P}(s,t) \cdot x_t & \text{if } s \in S^{?} \end{cases}
$$

- Obtain a set of linear equations in $|S|$ unknowns, simplify to a set of equations in $|S^?|$ unknowns:

$$
x_s = \sum_{t \in S^?} \mathrm{P}(s,t) \cdot x_t + \sum_{t \in S^{YES}} \mathrm{P}(s,t)
$$

Solving Equations for PCTL Until

- Equivalently, in matrix form:

$$\mathrm{x} = \mathrm{A} \cdot \mathrm{x} + \mathrm{P} \cdot \mathrm{b}$$

where
A is the submatrix for $S^?$ and
b a column vector for S^{YES}.

- Solve through direct method:
  - Gaussian elimination
  - LU decomposition
  - etc

- or via iteration:

$$\begin{aligned} x_s^0 &= 0 \\ x_s^{n+1} &= \textstyle\sum_{t \in S^?} \mathrm{P}(s,t) \cdot x_t^n + \sum_{t \in S^{YES}} \mathrm{P}(s,t) \end{aligned}$$

Example: linear equations

- Consider $try\ U\ succ$
- We calculate:

  $S^{YES} = \{s_3\}$
  $S^{>0} = \{s_1, s_3\}$
  $S^{NO} = \{s_0, s_2\}$
  $S^{?} = \{s_1\}$

- Obtain the linear equations:

$$\begin{aligned} x_0 &= 0 \\ x_1 &= 0.01 \cdot x_1 + 0.98 \cdot x_3 \\ x_2 &= 0 \\ x_3 &= 1 \end{aligned}$$

- which yield the solution:

$$\mathrm{x} = (0, \frac{98}{99}, 0, 1)$$

- Hence

$$Sat(\mathcal{P}_{\geq 0.98}(try\ U\ succ)) = \{s_1, s_3\}, \quad Sat(\mathcal{P}_{\geq 0.99}(try\ U\ succ)) = \{s_3\}$$

Some Special Cases

- Probabilistic reachability
  - reaching a target set T of states with probability $\bowtie p$
  - expressible as

$$\mathcal{P}_{\bowtie p}(\Diamond a_T)$$

  where a_T is an atomic proposition true at s iff $s \in T$

- Probabilistic invariance
  - remaining in set I of states with probability $\bowtie p$
  - expressible as

$$\mathcal{P}_{\bowtie p}(\Box a_I)$$

  where a_I is an atomic proposition true at s iff $s \in I$

More Special Cases

- Qualitative properties
  - are expressible as

$$\mathcal{P}_{\geq 1}(\alpha)$$

    for α a path formula
  - reachability analysis of the underlying graph suffices
  - can improve the model checking algorithm by precomputing:

$$S^{YES} := \text{ set of all states satisfying } \alpha \text{ with probability } 1$$

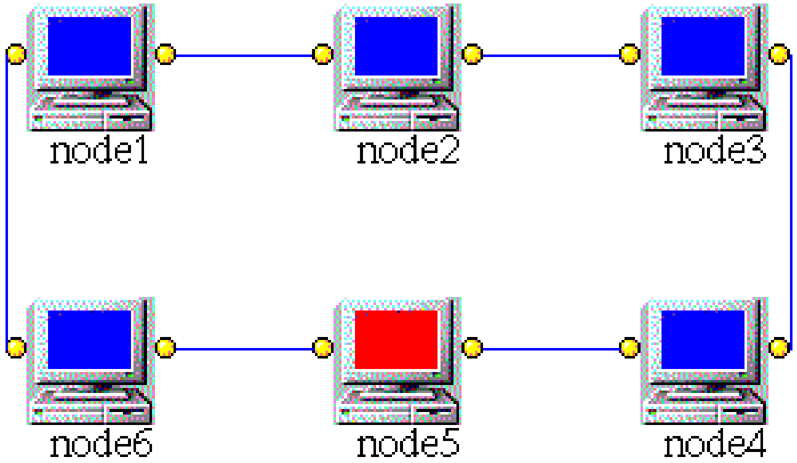
- Likewise

$$\mathcal{P}_{>0}(\alpha)$$

  can be established by reachability analysis

DTMC Example: Leader Election

- The algorithm [Itai & Rodeh]
  - choose a random number from $\{1 \dots K\}$ as id
  - synchronously pass all ids around the ring
  - if there is a unique id, then elect the process with the maximum unique id as leader, otherwise repeat



- Properties
  - "with probability 1 a leader is elected"

$$\mathcal{P}_{\geq 1}(\Diamond\, leader\_elected)$$

  - "the expected number of rounds to elect a leader is less than 4" (expressible using PCTL extension with costs/rewards

Leader Election Results

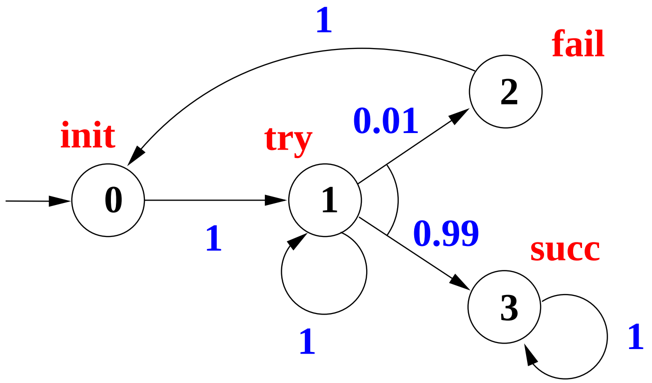
- "With probability 1 a leader is elected"
  - holds for all states

- Expected number of rounds to elect a leader
  - N – number of processes
  - K – random number chosen from $\{1 \ldots K\}$

N:	K:	#States	Expected no. rounds
4	2	55	2
4	4	782	1.18
4	8	12,302	1.04
4	16	196,622	1.01
6	2	329	5.33
6	4	20,524	1.19
6	6	233,340	1.05
8	2	1,803	16
8	4	458,847	1.31

Markov Decision Processes

- Features
  - states and time discrete; probabilistic choice and nondeterminism
  - asynchronous parallel composition of DTMCs



- Formally Markov Decision Process (MDP) is $M = (S, Steps, L)$ where
  - S is a set of states
  - $Steps(s) \subseteq \mu(S)$ for each $s \in S$, $\mu(S)$ denotes probability distributions on S
  - $L : S \longrightarrow 2^{AP}$ labelling with atomic prop.

- Can model distributed computaion but... discrete time only

- Probability and events:
  - unfold MDP into infinite paths from a given state, form events as before
  - probability space undefined if nondeterminism present

Paths and adversaries

- The execution of an MDP
  - alternation of non-deterministic and probabilistic choices
  - non-deterministic choice resolved using adversaries (similar to non-randomized policies in MDPs).
- Paths π of M are (finite or infinite) sequences of states

$$s_0 \xrightarrow{\mu_0} s_1 \xrightarrow{\mu_1} s_2 \xrightarrow{\mu_2} \cdots$$

  where $s_i \in S$, $\mu_i \in Steps(s_i)$ and $\mu_i(s_{i+1}) > 0$.

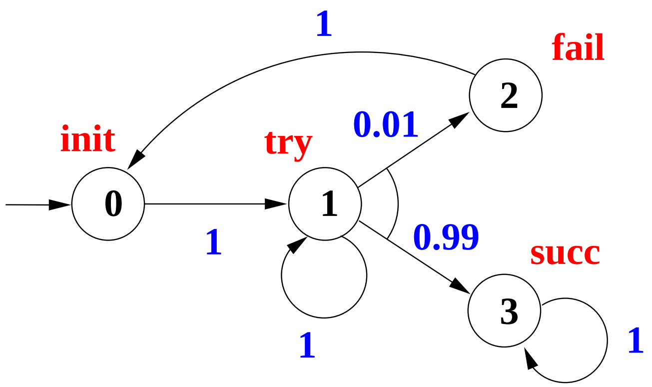
- An adversary A, for start state s
  - maps every finite path ρ onto a distribution (the step to be executed) $A(\rho)$ such that

$$A(\rho) \in Steps(last(\rho))$$

  - induces probability measure $\Pr^A$ on the σ-algebra $\mathcal{F}^A$ of infinite paths corresponding to A.

Paths and adversaries: back to example

- Unfolding the MDP



- Adversary must decide at state 1:
  - whether to loop back to 1 with probability 1
  - or whether to move to 3 with probability 0.99, or 2 with probability 0.01

- Adversary is simple if it makes the same choice independent of history

PCTL over Markov Decision Processes

As for DTMCs + quantification over adversaries [Bianco&de Alfaro 1995].

- The syntax is the same as PCTL over DTMCs.
- Semantics
  - parameterised by Adv, a class of adversaries,
  - for example, all or all fair adversaries.
- The meaning of probabilistic operator:

$$M, s \models_{Adv} \mathcal{P}_{\bowtie p}(\alpha) \text{ iff } \Pr^{A}\{\pi \mid M, \pi \models_{Adv} \alpha, \pi \text{ starts in } s\} \bowtie p$$
$$\text{for all adversaries } A \in Adv$$

i.e. under any scheduling policy of non-deterministic choices.

- The other operators are as for DTMCs.

Model checking for PCTL over MDPs

Assume Adv is the class of all adversaries.

- The algorithm
  - proceeds by induction on the structure of ϕ as for DTMCs
  - except for probability calculation
- Probability calculation for $\bowtie = \geq, >$
  - for path formula α, calculate minimum probability for all $s \in S$:

$$p_s^{min}(\alpha) \; = \; \textsf{inf}\{p_s^A(\alpha) \mid A \in Adv\}$$

where

$$p_s^A(\alpha) = \Pr{}^A\{\pi \mid M, \pi \models_{Adv} \alpha, \pi \text{ starts in } s\}$$

- Establish validity of the probability bound

$$Sat(\mathcal{P}_{\bowtie p}(\alpha)) \; = \; \{s \in S \mid p_s^{min}(\alpha) \bowtie p\}$$

- $\bowtie = \leq, <$ is dual, using max not min

Model checking Next State over MDPs

- Note
  - there are infinitely many adversaries even if S finite
  - suffices to consider simple adversaries (stationary policies in MDP)
- Thus
  - the minimum probability for $X\phi$ is

$$p_s^{min}(X\phi) \;:=\; \mathsf{min}\{\sum_{t\in Sat(\phi)} \mu(t) \mid \mu \in Steps(s)\}$$

  - can be computed through matrix-by-vector multiplication plus the min function
- For
  - the maximum probability for $X\phi$ calculate

$$p_s^{max}(X\phi) \;:=\; \mathsf{max}\{\sum_{t\in Sat(\phi)} \mu(t) \mid \mu \in Steps(s)\}$$

  - can be computed through matrix-by-vector multiplication plus the max function

Model checking Until over MDPs (min probability)

- First calculate
  - S^{YES}, S^{NO} and $S^?$ as for DTMCs
  - the minimum probability vector rewrites to:

$$p_s^{min} = \begin{cases} 0 & \text{if } s \in S^{NO} \\ 1 & \text{if } s \in S^{YES} \\ \min_{\mu \in Steps(s)} \left[\sum_{t \in S^?} \mu(t) \cdot p_t^{min} + \sum_{t \in S^{YES}} \mu(t) \right] & \text{if } s \in S^? \end{cases}$$

- Next, either
  - evaluate p_s^{min} iteratively for $s \in S^?$,

  or
  - solve the linear optimization problem:

Maximise:

$\sum_{s \in S^?} x_s$

Subject to the constraints:

$x_s \leq \sum_{t \in S^?} \mu(t) \cdot x_t + \sum_{t \in S^{YES}} \mu(t)$

for all $s \in S^?$ and all $\mu \in Steps(s)$

(Think of balancing flow, replacing the equality with inequality.)

Model checking Until over MDPs (max probability)

- For the dual comparison operators, first calculate
  - S^{YES}, S^{NO} and $S^{?}$ as before
  - the maximum probability vector is:

$$p_s^{max} = \begin{cases} 0 & \text{if } s \in S^{NO} \\ 1 & \text{if } s \in S^{YES} \\ \max_{\mu \in Steps(s)} \left[\sum_{t \in S^?} \mu(t) \cdot p_t^{max} + \sum_{t \in S^{YES}} \mu(t) \right] & \text{if } s \in S^{?} \end{cases}$$

- Next, either
  - evaluate p_s^{max} iteratively for $s \in S^{?}$,

  or
  - solve the linear optimization problem:

Minimise:

$\sum_{s \in S^?} x_s$

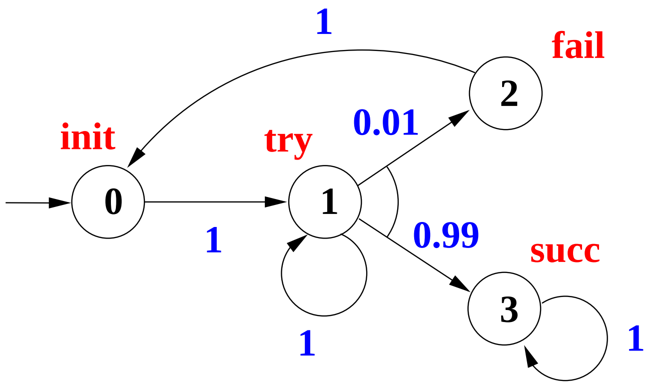
Subject to the constraints:

$x_s \geq \sum_{t \in S^?} \mu(t) \cdot x_t + \sum_{t \in S^{YES}} \mu(t)$

for all $s \in S^?$ and all $\mu \in Steps(s)$

Example: PCTL Until model checking

- Consider $\mathcal{P}_{\geq 0.99}(try\, U\, succ)$, without fairness:



- Calculate
  - $S^{YES} = \{s_3\}$
  - $S^{>0} = \{s_3\}$
  - $S^{NO} = \{s_0, s_1, s_2\}$
  - $S^{?} = \emptyset$

- Hence
  - $x_0 = 0, x_1 = 0, x_2 = 0, x_3 = 1$
  - thus $s_1 \not\models \mathcal{P}_{\geq 0.99}(try\, U\, succ)$
  - since there exists adversary (which loops back to s_1) for which the probability is less than the threshold 0.99

PCTL Until model checking (with fairness)

Assume Adv is the class of all fair adversaries.

- The computation of p_s^{max} remains unchanged
  - remains valid for quantification over a subset of adversaries
- The value of p_s^{min}
  - can be higher for the fair adversaries
  - can reduce to calculation of p_s^{max} by translating to dual problem [Baier 1998]
- Calculate for $\phi_1 \, U \, \phi_2$:
  - S^+ = set of states for which $p_s^{max} > 0$
  - $S^\sharp = S^+ \setminus Sat(\phi_2)$
  - $p_s^{max}(a^\sharp \, U \, \neg a^+)$ where $a^\sharp, a^+$ new atomic propositions for $S^\sharp, S^+$
- Then $p_s^A(\phi_1 \, U \, \phi_2) = 1 - p_s^A(a^\sharp \, U \, \neg a^+)$ for fair adversary A
- Back to the example... obtain
  - $(1, 0.01, 1, 0)$, thus subtracting from 1 gives $(0, 0.98, 0, 1)$
  - which yields $s_1 \models \mathcal{P}_{\geq 0.99}(try \, U \, succ)$ if fairness is assumed

Overview of Tutorial

- Motivation
  - Probability basics
  - Dice example

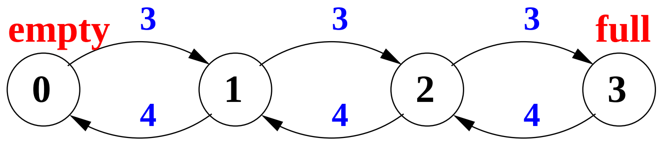
- Discrete Markovian models
  - The logic PCTL
  - Model checking for PCTL
  - Randomised leader election example

- Continuous time Markov chains
  - The logic CSL
  - Model checking for CSL
  - Wireless cell example

- Probabilistic timed automata
  - Model checking for probabilistic timed automata
  - IEEE 1394 root contention protocol example

Continuous Time Markov Chains

- Features
  - discrete states, real time, but no nondeterminism
  - exponentially distributed random delays (race condition)



- Formally, a Continuous Time Markov Chain (CTMC) is a tuple (S, R, L) where
  - S is set of states
  - $\mathrm{R} : S \times S \longrightarrow \mathcal{R}^{\geq 0}$ is the transition rate matrix, where probability of moving from s to s' within $t > 0$ time is $1 - e^{-\mathrm{R}(s,s')\cdot t}$
  - $L : S \longrightarrow 2^{AP}$ state labelling with atomic prop.

- Race condition
  - if > 1 state s' with $\mathrm{R}(s, s') > 0$ then probability $\mathrm{P}(s, s') = \mathrm{R}(s, s')/\Sigma_{s'' \in S}\mathrm{R}(s, s'')$
  - if no outgoing transitions from s, then $\mathrm{P}(s, s') = 1$ if $s = s'$ and 0 if $s \neq s'$

- Conventional CTMC analysis
  - based on steady-state (limiting) probability, as $t \to \infty$
  - here also path probabilities

CTMC Model Checking

- Steady-state probabilities
  - $Pr_s(s')$ - probability of being in state s' in the long run, having started in state s
- Transient probabilities
  - $Pr_s(s', t)$ - probability of being in state s' at time instant t, having started in state s
- Model checking approach
  - Path-based as well as state-based properties
  - Paths are (finite/infinite) sequences:

$$s_0 t_0 s_1 t_1 s_2 \cdots s_j t_j \cdots$$

    such that $\mathrm{R}(s_j, s_{j+1}) > 0$, $t_j \in \mathcal{R}^{\geq 0}$ denotes residence time, for $j \geq 0$
- Probability space constructed similarly to that for DTMCs:
  - take as outcomes set of paths (from state s_0)
  - $\mathcal{F}$ is the least σ-algebra containing cylinder sets $C(s_0, I_0, s_1, I_1, ..., I_n, s_n)$ where I_j are intervals, with path $s'_0 t_0 s'_1 t_1 ... \in C(s_0, I_0, s_1, I_1, ..., I_n, s_n)$ if $s_i = s'_i$ for $i \leq n$ and $t_i \in I_i$ for $i < n$
  - $Prob(A)$ - probability of the set A$\in \mathcal{F}$ of paths

The logic CSL

Based on CTL & PCTL [Aziz et al. 1996 & Baier et al. 1999]

- CSL Syntax:

$$\Phi ::= \mathit{true} \mid a \mid \neg\Phi \mid \Phi \wedge \Phi \mid \mathcal{S}_{\sim p}(\Phi) \mid \mathcal{P}_{\sim p}(\Phi\, \mathcal{U}^{\leq t}\, \Phi)$$

  where $a \in AP$, $\sim\ \in \{<, \leq, \geq, >\}$, $p \in [0, 1]$ and $t \in \mathcal{R}^{\geq 0}$

- Steady-state operator: $\mathcal{S}_{\sim p}(\Phi)$
  - the probability of being in a state satisfying Φ in the steady-state (long run) is $\sim p$

- Bounded until operator: $\mathcal{P}_{\sim p}(\Phi\, \mathcal{U}^{\leq t}\, \Psi)$
  - the probability of reaching a state satisfying Ψ, passing only through states satisfying Φ, within t time units is $\sim p$

- Derived operators:
  - $\mathcal{P}_{\sim p}(\Diamond^{\leq t}\, \Phi) \equiv \mathcal{P}_{\sim p}(\mathit{true}\, \mathcal{U}^{\leq t}\, \Phi)$
    - the probability of reaching a state satisfying Φ within t time units is $\sim p$

- Extensions
  - Other path formulas: $\Phi\, \mathcal{U}\, \Psi$, $X\, \Phi$
  - Other time constraints: $\Phi\, \mathcal{U}^{\mathcal{I}}\, \Psi$ for interval $\mathcal{I} \subseteq \mathcal{R}^{\geq 0}$

CSL Examples

- $\mathcal{S}_{\geq 0.80}(qos)$
  - The probability that an acceptable level of quality of service is achieved in the long run is ≥ 0.80
- $\mathcal{P}_{\leq 0.5}(\Diamond^{\leq 2}\ full)$
  - The probability of the queue becoming full within 2 hours is ≤ 0.5
- $\mathcal{P}_{\geq 0.98}(\neg repair\ \mathcal{U}^{\leq 200}\ success)$
  - The probability of the system terminating successfully within 200 hours, without any repairs being needed, is ≥ 0.98

CSL Semantics

- CSL formula Φ
  - evaluated over states of a CTMC

- $s \models \mathit{true}$
  $s \models a$
  $s \models \neg\Phi$
  $s \models \Phi \wedge \Phi$
  - as for CTL

- $s \models \mathcal{S}_{\sim p}(\Phi)$
  - iff $\sum_{s' \models \Phi} \mathit{Pr}_s(s') \sim p$

- $s \models \mathcal{P}_{\sim p}(\Phi \, \mathcal{U}^{\leq t} \, \Psi)$
  - iff $\mathit{Prob}_s(\Phi \, \mathcal{U}^{\leq t} \, \Psi) \sim p$

CSL Model Checking

- **Model checking problem:**
  - For CTMC $(S, \mathbf{R}, L)$ and CSL formula Φ, calculate the set $\mathsf{Sat}(\Phi)$ of states satisfying Φ

- **Simple operators** $(\mathit{true}, a, \neg, \wedge)$
  - As for CTL/PCTL/...

- $s \models \mathcal{S}_{\sim p}(\Phi)$?
  - Requires computation of steady-state probabilities $Pr_s(s')$ for all states s'
  - Reduces to graph algorithm and solution of linear equation system

- $s \models \mathcal{P}_{\sim p}(\Phi\, \mathcal{U}^{\leq t}\, \Psi)$?
  - Requires computation of probability $Prob_s(\Phi\, \mathcal{U}^{\leq t}\, \Psi)$
  - Solve Volterra integral equation system

    or
  - Reduce to computation of transient probabilities $Pr_s(s', t)$ for all states s' on a modified CTMC

Model Checking $\mathcal{U}^{\leq t}$

- $s \models \mathcal{P}_{\sim p}(\Phi \, \mathcal{U}^{\leq t} \, \Psi)$?
- Reduction to transient analysis
  - Let Q be the generator matrix of CTMC (S, R, L), i.e.
    - $\mathrm{Q}(s, s') = \mathrm{R}(s, s')$ if $s \neq s'$ and $\mathrm{Q}(s, s') = -\Sigma_{s'' \neq s'} \mathrm{R}(s, s')$ if $s = s'$
  - Transform CTMC by removing all outgoing transitions from states satisfying Ψ or $\neg\Phi$

$$(S, Q) \qquad \rightarrow \qquad (S, Q')$$

$$Prob_s(\Phi \, \mathcal{U}^{\leq t} \, \Psi) \;=\; \sum_{s' \models \Psi} Pr_s(s', t)$$

- Transient probabilities $Pr_s(s', t)$
  - computed using uniformisation
  - more efficient and stable, iterative computation based on matrix-vector

Uniformisation

- Computation of transient probabilities at time t
  - $Pr_s(s', t)$ for all states s', i.e. $\underline{Pr_s(t)}$

- Construct *uniformised* DTMC from CTMC
  - Let $P = I + Q/q$ where $q = max_s|Q(s,s)|$

$$
\begin{aligned}
\underline{Pr_s(t)} &= \underline{Pr_s(0)} \cdot e^{q(P-I)t} \\
&= \underline{Pr_s(0)} \cdot \sum_{i=0}^{\infty} (e^{-qt} \cdot \tfrac{(qt)^i}{i!} \cdot P^i) \\
&= \underline{Pr_s(0)} \cdot \sum_{i=0}^{\infty} (\gamma_{qt,i} \cdot P^i)
\end{aligned}
$$

  where $\gamma_{qt,i}$ is the ith Poisson probability with parameter qt

  - weighted sum of P^i gives probability of moving from any state to any other state in t time units
  - P is a stochastic matrix so summation is stable

  - Truncate infinite summation using Fox-Glynn method

Model Checking $\mathcal{U}^{\leq t}$

- Observe:

$$\begin{aligned} \mathit{Prob}_s(\Phi\,\mathcal{U}^{\leq t}\,\Psi) &= \sum_{s' \models \Psi} \mathit{Pr}_s(s', t) \\ &= \underline{\mathit{Pr}_s(t)} \cdot \underline{\psi} \\ &= \underline{\mathit{Pr}_s(0)} \cdot \sum_{i=0}^{\infty} (\gamma_{qt,i} \cdot P^i) \cdot \underline{\psi} \end{aligned}$$

where $\underline{\psi}$ is a 0-1 column vector: $\psi(s) = 1$ iff $s \models \Psi$

- Can now avoid explicit computation of matrix powers P^i:
  - $P^i \cdot \underline{\psi} = (P \cdots (P \cdot (P \cdot \underline{\psi})))$

- Computing $\mathit{Prob}_s(\Phi\,\mathcal{U}^{\leq t}\,\Psi)$ for all states s
  - Computation proceeds backwards not forwards, for all states at once
  - Compute $\mathit{Prob}_s(\Phi\,\mathcal{U}^{\leq t}\,\Psi)$ directly, bypassing calculation of $\mathit{Pr}_s(s', t)$

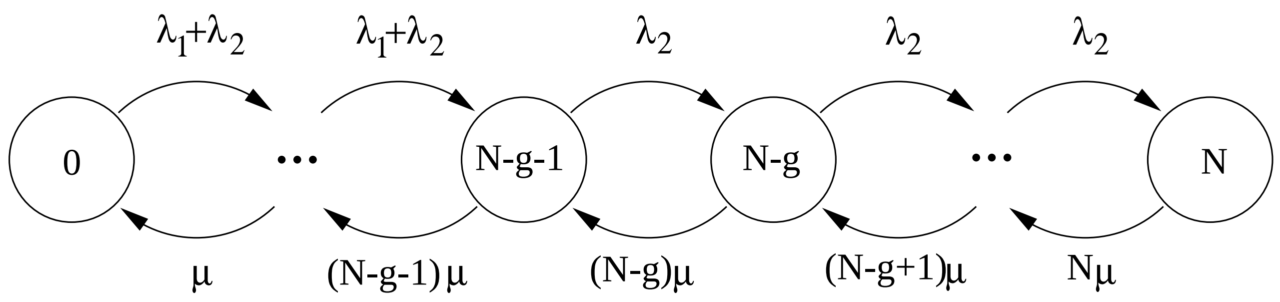
$$\underline{\mathit{Prob}(\Phi\,\mathcal{U}^{\leq t}\,\Psi)} = \sum_{i=0}^{\infty} (\gamma_{qt,i} \cdot P^i) \cdot \underline{\psi}$$

  - Bound summation with Fox-Glynn, as before
  - $\mathcal{O}(n)$ improvement in time efficiency [Katoen et al. 2001]

CTMC Example: Wireless Cell

- Single Cell in a Wireless Communication Network [Haring et al.]
- Assumptions
  - N – number of channels in the cell
  - Guarded channel scheme (GCS): a fixed number g, of (guarded) channels, reserved exclusively for hand off calls
  - μ - departure rate of calls
  - λ_1 - arrival rate of new calls
  - λ_2 - arrival rate of handoff calls

- The model
  - CTMC
  - 20% of the total number of channels are guarded
  - Number of states: $N+1$



Wireless Cell

- The properties

$(n < N) \rightarrow \mathcal{P}_{<0.10}(\Diamond^{[0,100]}\ n = N)$

If a guarded channel is free, then the chance that a hand off call can be dropped within 100 time units is less than 10%

$(n < N - g) \rightarrow \mathcal{P}_{<0.20}(\Diamond^{[0,50]}\ n \leq N - g)$

If a non-guarded channel is free, then the chance that any calls can be dropped within 50 time units is less than 20%.

$(n = N) \rightarrow \mathcal{P}_{>0.99}(\Diamond^{[0,0.5]}\ n < N - g)$

If no channels are free, then within 0.5 time units a new call can be accepted withprobability greater than 0.99

$\mathcal{S}_{\geq 0.9}(n < N - g)$

In the long run, a new call can be accepted with probability greater than or equal to 0.9

Wireless Cell

- Model checking results for
  $(n < N) \rightarrow \mathcal{P}_{<0.10}(\Diamond^{[0,100]}\ n = N)$

N:	Result:	iterations:	time per iteration (in sec)		
			MTBDD	Sparse	Hybrid
200	`false`	30	0.0003	0.00004	0.0002
300	`true`	30	0.0004	0.00005	0.0003
400	`true`	30	0.0004	0.00007	0.0003
500	`true`	30	0.0005	0.00009	0.0003

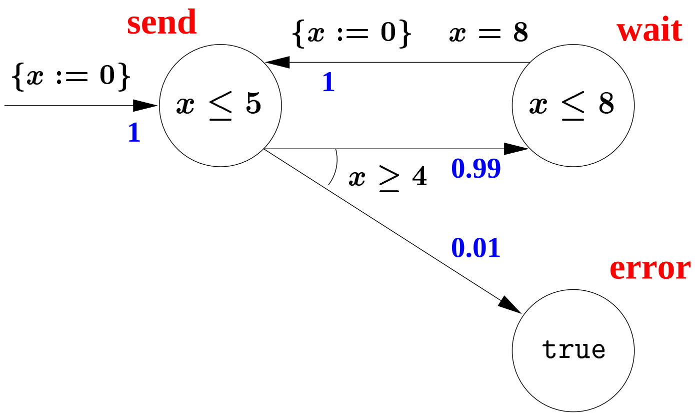
- Extensions
  - Network of cells under different topologies

Overview of Tutorial

- Motivation
  - Probability basics
  - Dice example
- Discrete Markovian models
  - The logic PCTL
  - Model checking for PCTL
  - Randomised leader election example
- Continuous time Markov chains
  - The logic CSL
  - Model checking for CSL
  - Wireless cell example
- Probabilistic timed automata
  - Model checking for probabilistic timed automata
  - IEEE 1394 root contention protocol example

Probabilistic Timed Automata

- Features
  - clock constraints e.g. $y \geq x + 1$, dense real time
  - probabilistic transitions, nondeterministic selection



- Can be reduced to
  - Markov Decision Process over regions/zones of clock valuations, as opposed to symbolic region/zone graph

- Typical specification (logic PTCTL = TCTL + PCTL)
  - Under any scheduling, probability of reaching ϕ-state within time t is $\leq \lambda$ (also $<, >, \geq \lambda$)

- But... very complex to deal with!

Model checking for PTAs

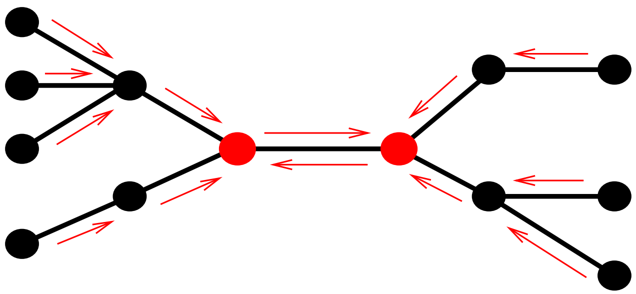
- The approach
  - find symbolic states encoding clock valuations
  - build MDP over symbolic states
  - model check via linear optimisation

- Region-based model construction
  - construct MDP over regions
  - translate PTCTL to PCTL
  - apply model checking as for MDPs
  - problem: high complexity, exponential (in number of clocks, size of constant)

- Forwards reachability, zone-based
  - iterate post from initial state
  - yields maximum reachability probability
  - problems:
    - lose on-the-fly property, need to construct LP problem
    - probability could be higher than that in PTA

- Backwards reachability, zone-based
  - iterate pre from target set
  - yields exact maximum reachability probability
  - problem: conjuction and pre needed, increased complexity
  - combine with forwards?

PTA Example: FireWire



- In network initialisation phase,
  - nodes/devices do not have ids; network acyclic, but not a tree
  - choose leader (root) to perform tree identify

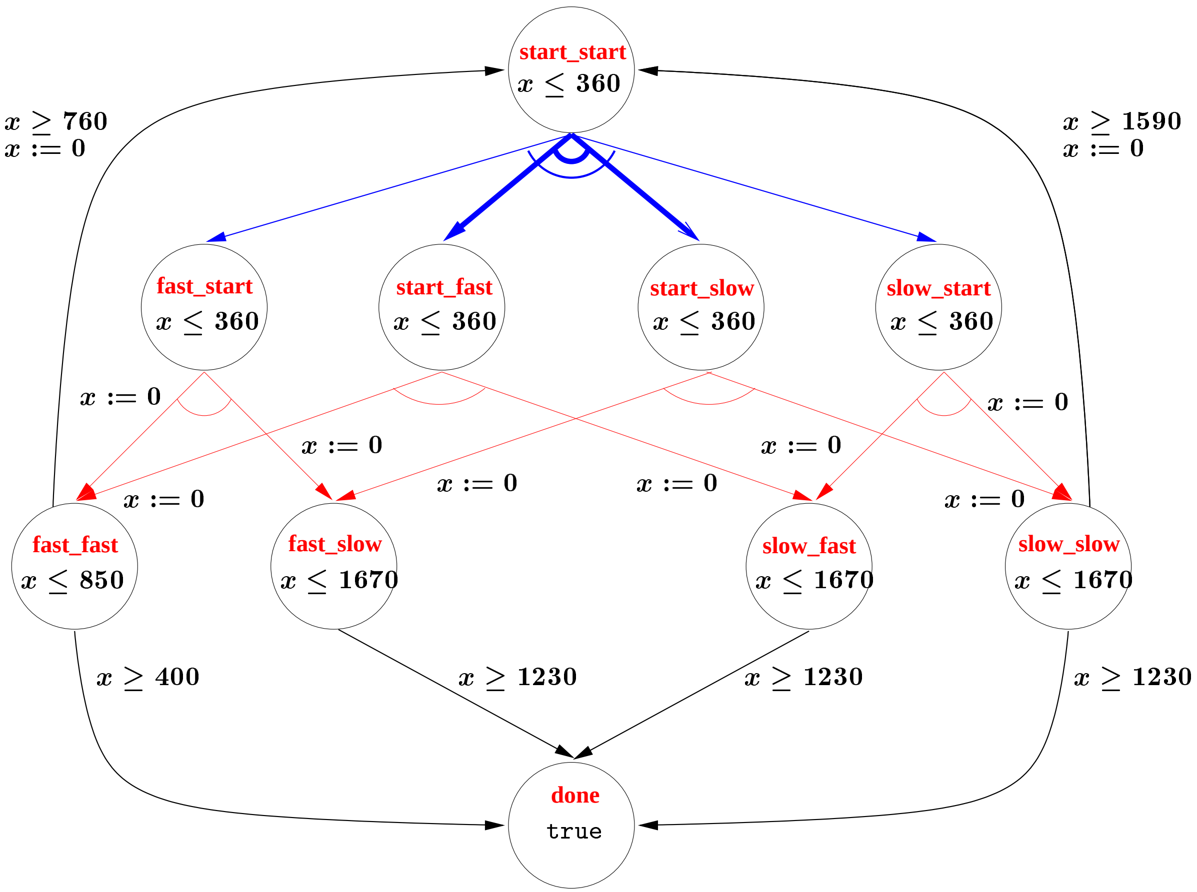
- Randomised algorithm with real time:
  - flip coin, choose short/long waiting time
  - signal transmission delay

- Signals:
  - ***Parent_Notify***: will you be my parent?
  - ***Child_Notify***: acknowledgement of ***Parent_Notify***
  - ***Idle***

- Root contention: if node receives
  - ***Parent_Notify*** while sending ***Parent_Notify***: root contention protocol restarts
  - ***Parent_Notify*** while sending ***Idle***: root contention resolved, node becomes leader and sends ***Child_Notify***

FireWire Root Contention

- Model based on abstract model [Stoelinga & Vaandrager 1999]
  - proved via series of manual refinements
- Here
  - augment with probabilistic choice
  - verify probabilistic deadlines
  - timings from IEEE 1394a standard



FireWire Root Contention

- Probabilistic deadline
  - "Under any scheduling, the probability of electing a leader before the deadline is exceeded is $\geq \lambda$", where $\lambda \in [0, 1]$ is a probability threshold
  - paraphrased: "the minimum probability of electing a leader before the deadline is $\geq \lambda$"
- Recall: forward reachability for PTAs
  - computes the maximum probability
  - use duality between max.prob.reachability and min.prob.invariance for probabilistic deadlines
- To verify probabilistic deadline:
  - augment model with location deadline
  - add clock y to measures global time
  - transform the model
  - model check the dual property

Root Contention Results

- **Probabilistic deadlines**
  - **with forwards reachability, up to deadline 6,000**

deadline (ns)	# symbolic states	probability
2000	48	0
3000	72	0.5
4000	137	0.625
5000	183	0.78125
6000	234	0.851562

  - **with integer semantics, up to 1,000,000 and ∞**
  - **with KRONOS and PRISM, up to 100,000**
- **Expected time to election**
  - **can also calculate in the worst case (=3,650ns)**

Root Contention Results

- Did you know???
  - If you use a biased coin...

fast	*slow*	D=6,000	D=10,000
0.45	0.55	0.830027	0.968230
0.5	0.5	0.851562	0.974731
0.55	0.45	0.864616	0.977771

- This observation
  - was conjectured by Marielle Stoelinga
  - verified with KRONOS/PRISM [Daws et al. 2002]
  - see `www.cs.bham.ac.uk/~dxp/prism/`

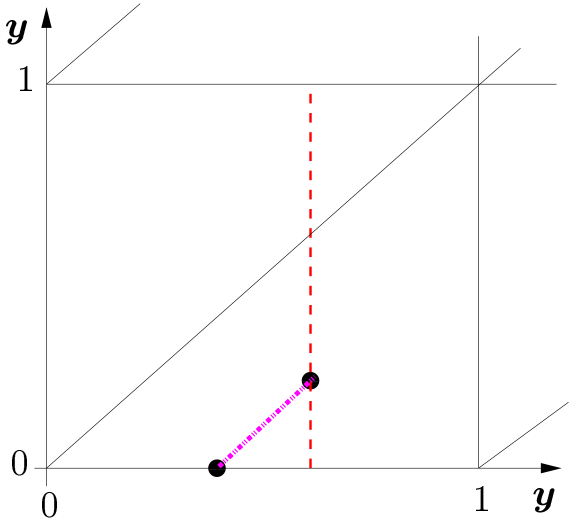
- May need checking again
  - if further new timings are adopted by the standard

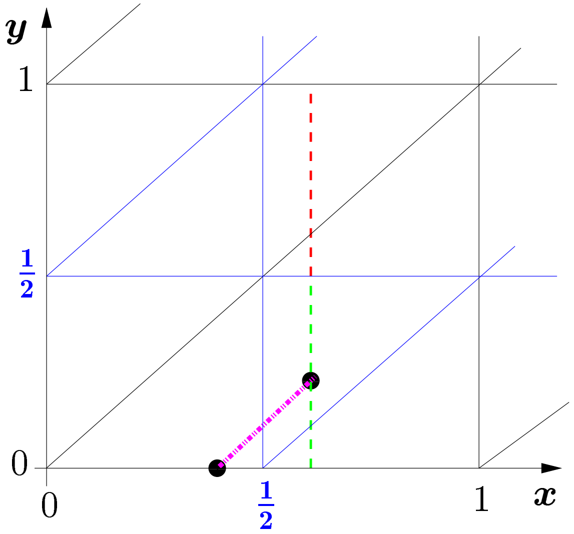
Continuous probabilistic TA

- Discrete probability often inadequate:
  - exact probability dependent on exact time point
  - need to model arrivals/departures, cf multimedia protocols
- Use continuous density functions
  - with bounded support in $\mathcal{R}^{\geq 0}$
  - allow to reset clocks according to c.d.f.s
  - Dirac distributions for deterministic resetting
- Region graph analysis inadequate:
  - one solution subdivide regions
- Approximate model checking for PTCTL:
  - more subdivision, smaller error
  - obtain bounds on error
- Problem: high complexity!

Why subdivide the region graph?

- Example by R. Alur: clocks x, y: initially:
  - y set to 0
  - x set in [0,1] according to a c.d.f.
- Say $x < 1$:
  - Time passes; before $x = 1$, nondeterministically, make a discrete transition
  - Reset y in [0,1] according to another c.d.f.
  - Two transitions: t_x and t_y with enabling conditions $x = 1$ and $y = 1$ respectively
    - Probability of t_y being enabled first?
    - Closer x is to 1 when y is reset, less likely that t_y enabled first ($y > x$)





  - If $y \in (0, 0.5)$, know x will reach 1 first
  - If $y \in (0.5, 1)$, don't know $\rightarrow$ error

Challenges & Future Plans

- Efficiency improvements
  - develop parallel engine
  - adapt fast numerical solutions to MTBDDs
  - approximate the probability

- Exploiting structure
  - abstraction, proof assistant – how?
  - (de)compositionality

- Property specifications
  - expected time
  - long-run average

- Probabilistic real-time verification
  - combine with real-time model checker?
  - efficient methods for continuous PTAs

- More case studies
  - gossip-style routing protocols
  - Crowds anonymity protocol [Shmatikov 2002]
  - dynamic power management [Shukla et al. 2002]