Reasoning about Time and Reliability

Probabilistic CTL model checking

Daniel Bruns

Institut für theoretische Informatik
Universität Karlsruhe

13. Juli 2007

Seminar „Theorie und Anwendung von Model Checking“

Outline

Motivation

- Communication networks need to be error free and reliable. In addition, they often operate under real time conditions, meaning they must meet certain deadlines in order to work correctly.
- Particularly, this becomes vital when wireless networks are used – where significant packet loss is inevitable.
- In this talk, a temporal logic will be presented in which propositions such as "There is a probability of at least 99% that a message is received at most 6*ms* after it is sent." can be expressed.

Motivation

- Communication networks need to be error free and reliable. In addition, they often operate under real time conditions, meaning they must meet certain deadlines in order to work correctly.
- Particularly, this becomes vital when wireless networks are used – where significant packet loss is inevitable.
- In this talk, a temporal logic will be presented in which propositions such as "There is a probability of at least 99% that a message is received at most 6*ms* after it is sent." can be expressed.

An extension to CTL with time and probability

- Probabilistic real time CTL (PCTL) is an extension to the well-known temporal logic CTL (Computation Tree Logic).
- PCTL extends this concept by both a discrete time structure allowing real time statements as well as probabilities for these events by which hard and soft deadlines can be modeled.
- There exist polynomial time model checking algorithms suitable for small structures.

Outline

Syntax

Definition

- Atomic propositions are state formulae.
- If φ and ψ are state formulae, then so are $\neg\varphi$, $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$ and $(\varphi \rightarrow \psi)$.
- If φ and ψ are state formulae and $\tau \in \mathbb{N} \cup \infty$, then $(\varphi \ \mathrm{U}^{\leq\tau} \ \psi)$ and $(\varphi \ \mathrm{W}^{\leq\tau} \ \psi)$ are path formulae.
- If F is a path formula and $\rho \in [0, 1]$, then $[F]_{\geq\rho}$ and $[F]_{>\rho}$ are state formulae.

We will use $\varphi \ \mathrm{U}^{\leq\tau}_{\geq\rho} \ \psi$ as shorthand for $[\varphi \ \mathrm{U}^{\leq\tau} \ \varphi]_{\geq\rho}$.

PCTL: some intuition

- $\varphi \; \mathrm{U}^{\leq 4}_{\geq 0.99} \; \psi$ means there is a probability of at least 99% that both ψ will come true within 4 time units and φ holds till ψ comes true.
- $\varphi \; \mathrm{W}^{\leq 4}_{\geq 0.99} \; \psi$ means there is a probability of at least 99% that either the above condition holds or φ holds for at least 4 time units.
- If the underlying structure is modelled wisely, "time units" can be substituted with real time units such as "seconds" or "milliseconds".

PCTL: some intuition

- $\varphi\ \mathrm{U}^{\leq 4}_{\geq 0.99}\ \psi$ means there is a probability of at least 99% that both ψ will come true within 4 time units and φ holds till ψ comes true.
- $\varphi\ \mathrm{W}^{\leq 4}_{\geq 0.99}\ \psi$ means there is a probability of at least 99% that either the above condition holds or φ holds for at least 4 time units.
- If the underlying structure is modelled wisely, "time units" can be substituted with real time units such as "seconds" or "milliseconds".

PCTL: some intuition

- $\varphi \, \mathrm{U}^{\leq 4}_{\geq 0.99} \, \psi$ means there is a probability of at least 99% that both ψ will come true within 4 time units and φ holds till ψ comes true.
- $\varphi \, \mathrm{W}^{\leq 4}_{\geq 0.99} \, \psi$ means there is a probability of at least 99% that either the above condition holds or φ holds for at least 4 time units.
- If the underlying structure is modelled wisely, "time units" can be substituted with real time units such as "seconds" or "milliseconds".

Expressing CTL through PCTL

Example

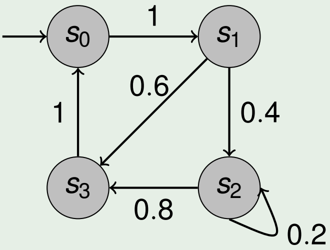
Since PCTL extends CTL, its properties can be expressed by extreme values of time and probability.

- generally: $\mathrm{G}\varphi \equiv \varphi \,\mathrm{W}^{\leq\infty}$ *false*
- finally: $\mathrm{F}\varphi \equiv$ *true* $\mathrm{U}^{\leq\infty}\, \varphi$
- until operator: $\varphi \mathrm{U} \psi \equiv \varphi \,\mathrm{U}^{\leq\infty}\, \psi$
- universal path quantifier: $\forall F \equiv [F]_{\geq 1}$
- existential path quantifier: $\exists F \equiv [F]_{>0}$

Logic structure

The structure over which PCTL formulae are evaluated is a finite automaton with labels on states and probabilistic transitions. Each transition corresponds to one time step.

Example



Formal definition of the structure

Definition

A structure $\mathcal{K}$ is a tuple $(\mathcal{S}, s_\perp, \mathcal{T}, L)$ where

- $\mathcal{S}$ is a finite set of states,
- $s_\perp \in \mathcal{S}$ is the initial state,
- $\mathcal{T}$ is a probabilistic transition function $\mathcal{T} : \mathcal{S}^2 \to [0, 1]$, such that $\sum_{t \in \mathcal{S}} \mathcal{T}(s, t) = 1$ for all states $s \in \mathcal{S}$ and
- L is a labeling function assigning sets of atomic formulae to states ($L : \mathcal{S} \to 2^{\{atom\}}$).

Formal definition of the structure

Definition

A structure $\mathcal{K}$ is a tuple $(\mathcal{S}, s_\perp, \mathcal{T}, L)$ where

- $\mathcal{S}$ is a finite set of states,
- $s_\perp \in \mathcal{S}$ is the initial state,
- $\mathcal{T}$ is a probabilistic transition function $\mathcal{T} : \mathcal{S}^2 \to [0, 1]$, such that $\sum_{t \in \mathcal{S}} \mathcal{T}(s, t) = 1$ for all states $s \in \mathcal{S}$ and
- L is a labeling function assigning sets of atomic formulae to states ($L : \mathcal{S} \to 2^{\{atom\}}$).

Formal definition of the structure

Definition

A structure $\mathcal{K}$ is a tuple $(\mathcal{S}, s_\perp, \mathcal{T}, L)$ where

- $\mathcal{S}$ is a finite set of states,
- $s_\perp \in \mathcal{S}$ is the initial state,
- $\mathcal{T}$ is a probabilistic transition function $\mathcal{T} : \mathcal{S}^2 \to [0, 1]$, such that $\sum_{t \in \mathcal{S}} \mathcal{T}(s, t) = 1$ for all states $s \in \mathcal{S}$ and
- L is a labeling function assigning sets of atomic formulae to states ($L : \mathcal{S} \to 2^{\{atom\}}$).

Prerequisites for semantics

Definition

1. A path beginning in state s_0 is an infinite sequence $(s_0, s_1, \ldots)$. Let the set of paths (beginning in s_0) be $\mathcal{P}(s_0)$.
2. The n-th state of a path π is denoted $\pi[n]$, a finite prefix of length n is denoted $\pi|_n := (s_0, \ldots, \pi[n])$.
3. For each state s a probability measure $\mu_s : \mathcal{P} \to [0,1]$ is defined for each finite sequence $(s_0, \ldots, s_n)$ by

$$\mu_s(\{\pi \in \mathcal{P} : \pi|_n = (s_0, \ldots, s_n)\}) = \prod_{i=1}^{n} \mathcal{T}(s_{i-1}, s_i)$$

Prerequisites for semantics

Definition

1. A path beginning in state s_0 is an infinite sequence $(s_0, s_1, \ldots)$. Let the set of paths (beginning in s_0) be $\mathcal{P}(s_0)$.
2. The n-th state of a path π is denoted $\pi[n]$, a finite prefix of length n is denoted $\pi|_n := (s_0, \ldots, \pi[n])$.
3. For each state s a probability measure $\mu_s : \mathcal{P} \to [0, 1]$ is defined for each finite sequence $(s_0, \ldots, s_n)$ by

$$\mu_s(\{\pi \in \mathcal{P} : \pi|_n = (s_0, \ldots, s_n)\}) = \prod_{i=1}^{n} \mathcal{T}(s_{i-1}, s_i)$$

Prerequisites for semantics

Definition

1. A path beginning in state s_0 is an infinite sequence $(s_0, s_1, \ldots)$. Let the set of paths (beginning in s_0) be $\mathcal{P}(s_0)$.
2. The n-th state of a path π is denoted $\pi[n]$, a finite prefix of length n is denoted $\pi|_n := (s_0, \ldots, \pi[n])$.
3. For each state s a probability measure $\mu_s : \mathcal{P} \to [0, 1]$ is defined for each finite sequence $(s_0, \ldots, s_n)$ by

$$\mu_s(\{\pi \in \mathcal{P} : \pi|_n = (s_0, \ldots, s_n)\}) = \prod_{i=1}^{n} \mathcal{T}(s_{i-1}, s_i)$$

Semantics

Definition

$s \models a$ iff $a \in L(s)$

$s \models \neg\varphi$ iff $s \not\models \varphi$

$s \models \varphi \wedge \psi$ iff $s \models \varphi$ and $s \models \psi$

$\pi \models \varphi\ \mathrm{U}^{\leq\tau}\ \psi$ iff there is a $\iota \leq \tau$ with $\pi[\iota] \models \psi$ and $\pi[\kappa] \models \varphi$ for all $\kappa < \iota$

$\pi \models \varphi\ \mathrm{W}^{\leq\tau}\ \psi$ iff $\pi \models \varphi\ \mathrm{U}^{\leq\tau}\ \psi$ or $\pi[\kappa] \models \varphi$ for all $\kappa \leq \tau$

$s \models [F]_{\geq\rho}$ iff $\mu_s(\{\pi \in \mathcal{P}(s) \mid \pi \models F\}) \geq \rho$

Finally, we define $\models_{\mathcal{K}} \varphi$ iff $s_\perp \models \varphi$.

Semantics

Definition

$s \models a$ iff $a \in L(s)$
$s \models \neg\varphi$ iff $s \not\models \varphi$
$s \models \varphi \wedge \psi$ iff $s \models \varphi$ and $s \models \psi$
$\pi \models \varphi\ \mathrm{U}^{\leq\tau}\ \psi$ iff there is a $\iota \leq \tau$ with $\pi[\iota] \models \psi$ and $\pi[\kappa] \models \varphi$ for all $\kappa < \iota$
$\pi \models \varphi\ \mathrm{W}^{\leq\tau}\ \psi$ iff $\pi \models \varphi\ \mathrm{U}^{\leq\tau}\ \psi$ or $\pi[\kappa] \models \varphi$ for all $\kappa \leq \tau$
$s \models [F]_{\geq\rho}$ iff $\mu_s(\{\pi \in \mathcal{P}(s) \mid \pi \models F\}) \geq \rho$

Finally, we define $\models_{\mathcal{K}} \varphi$ iff $s_{\perp} \models \varphi$.

Semantics

Definition

$s \models a$	iff $a \in L(s)$
$s \models \neg\varphi$	iff $s \not\models \varphi$
$s \models \varphi \wedge \psi$	iff $s \models \varphi$ and $s \models \psi$
$\pi \models \varphi\ \mathrm{U}^{\leq\tau}\ \psi$	iff there is a $\iota \leq \tau$ with $\pi[\iota] \models \psi$ and $\pi[\kappa] \models \varphi$ for all $\kappa < \iota$
$\pi \models \varphi\ \mathrm{W}^{\leq\tau}\ \psi$	iff $\pi \models \varphi\ \mathrm{U}^{\leq\tau}\ \psi$ or $\pi[\kappa] \models \varphi$ for all $\kappa \leq \tau$
$s \models [F]_{\geq\rho}$	iff $\mu_s(\{\pi \in \mathcal{P}(s) \mid \pi \models F\}) \geq \rho$

Finally, we define $\models_{\mathcal{K}} \varphi$ iff $s_{\perp} \models \varphi$.

Semantics

Definition

$$
\begin{array}{ll}
s \models a & \text{iff } a \in L(s) \\
s \models \neg\varphi & \text{iff } s \not\models \varphi \\
s \models \varphi \wedge \psi & \text{iff } s \models \varphi \text{ and } s \models \psi \\
\pi \models \varphi \,\mathrm{U}^{\leq\tau}\, \psi & \text{iff there is a } \iota \leq \tau \text{ with } \pi[\iota] \models \psi \text{ and} \\
 & \qquad\qquad \pi[\kappa] \models \varphi \text{ for all } \kappa < \iota \\
\pi \models \varphi \,\mathrm{W}^{\leq\tau}\, \psi & \text{iff } \pi \models \varphi \,\mathrm{U}^{\leq\tau}\, \psi \text{ or } \pi[\kappa] \models \varphi \text{ for all } \kappa \leq \tau \\
s \models [F]_{\geq\rho} & \text{iff } \mu_s(\{\pi \in \mathcal{P}(s) \mid \pi \models F\}) \geq \rho
\end{array}
$$

Finally, we define $\models_{\mathcal{K}} \varphi$ iff $s_\perp \models \varphi$.

Outline

1. Probabilistic real time CTL
2. PCTL model checking
3. Calculation example

A model checking algorithm for PCTL

- In this section a model checking algorithm will be presented, which determines whether a given structure $\mathcal{K}$ models some formula φ, i.e. $\models_{\mathcal{K}} \varphi$.
- When it finishes, each state will be labeled with the complete set of subformulae of φ which hold in that state. If the initial state $s_{\perp}$ is labeled with φ, then the structure is a model for φ.
- The algorithm is based on the original model checking algorithm for CTL.

A model checking algorithm for PCTL

- In this section a model checking algorithm will be presented, which determines whether a given structure $\mathcal{K}$ models some formula φ, i.e. $\models_{\mathcal{K}} \varphi$.
- When it finishes, each state will be labeled with the complete set of subformulae of φ which hold in that state. If the initial state $s_{\perp}$ is labeled with φ, then the structure is a model for φ.
- The algorithm is based on the original model checking algorithm for CTL.

A model checking algorithm for PCTL

- In this section a model checking algorithm will be presented, which determines whether a given structure $\mathcal{K}$ models some formula φ, i.e. $\models_{\mathcal{K}} \varphi$.
- When it finishes, each state will be labeled with the complete set of subformulae of φ which hold in that state. If the initial state $s_{\perp}$ is labeled with φ, then the structure is a model for φ.
- The algorithm is based on the original model checking algorithm for CTL.

Labeling states with formulae

1. We define an extended labeling function $\mathcal{L}$ for every subformula of φ. Initially, all states are labeled with atomic propositions: $\forall s \in \mathcal{S} : \mathcal{L}(s) := L(s)$
2. Suppose the subformulae of φ have been ordered in size (of logic connectives). Then from the smallest to φ itself change the labels of all states:
   - For a subformula $\neg\psi$ set $\mathcal{L}(s) := \mathcal{L}(s) \cup \{\neg\psi\}$ if $\psi \notin \mathcal{L}(s)$
   - For a subformula $(\varphi_1 \wedge \varphi_2)$ set $\mathcal{L}(s) := \mathcal{L}(s) \cup \{\varphi_1 \wedge \varphi_2\}$ if $\varphi_1, \varphi_2 \in \mathcal{L}(s)$.
   - For a subformula $(\varphi_1 \ \mathrm{U}^{\leq \tau}_{\geq p} \ \varphi_2)$ use the following algorithm based on matrix multiplication.

Labeling states with formulae

1. We define an extended labeling function $\mathcal{L}$ for every subformula of φ. Initially, all states are labeled with atomic propositions: $\forall s \in \mathcal{S} : \mathcal{L}(s) := L(s)$
2. Suppose the subformulae of φ have been ordered in size (of logic connectives). Then from the smallest to φ itself change the labels of all states:
   - For a subformula $\neg\psi$ set $\mathcal{L}(s) := \mathcal{L}(s) \cup \{\neg\psi\}$ if $\psi \notin \mathcal{L}(s)$
   - For a subformula $(\varphi_1 \wedge \varphi_2)$ set $\mathcal{L}(s) := \mathcal{L}(s) \cup \{\varphi_1 \wedge \varphi_2\}$ if $\varphi_1, \varphi_2 \in \mathcal{L}(s)$.
   - For a subformula $(\varphi_1 \ \mathrm{U}^{\leq \tau}_{\geq \rho} \ \varphi_2)$ use the following algorithm based on matrix multiplication.

Labeling states with formulae

1. We define an extended labeling function $\mathcal{L}$ for every subformula of φ. Initially, all states are labeled with atomic propositions: $\forall s \in \mathcal{S} : \mathcal{L}(s) := L(s)$
2. Suppose the subformulae of φ have been ordered in size (of logic connectives). Then from the smallest to φ itself change the labels of all states:
   - For a subformula $\neg\psi$ set $\mathcal{L}(s) := \mathcal{L}(s) \cup \{\neg\psi\}$ if $\psi \notin \mathcal{L}(s)$
   - For a subformula $(\varphi_1 \wedge \varphi_2)$ set $\mathcal{L}(s) := \mathcal{L}(s) \cup \{\varphi_1 \wedge \varphi_2\}$ if $\varphi_1, \varphi_2 \in \mathcal{L}(s)$.
   - For a subformula $(\varphi_1 \ \mathrm{U}^{\leq \tau}_{\geq \rho} \ \varphi_2)$ use the following algorithm based on matrix multiplication.

Labeling states with formulae

1. We define an extended labeling function $\mathcal{L}$ for every subformula of φ. Initially, all states are labeled with atomic propositions: $\forall s \in \mathcal{S} : \mathcal{L}(s) := L(s)$
2. Suppose the subformulae of φ have been ordered in size (of logic connectives). Then from the smallest to φ itself change the labels of all states:
   - For a subformula $\neg\psi$ set $\mathcal{L}(s) := \mathcal{L}(s) \cup \{\neg\psi\}$ if $\psi \notin \mathcal{L}(s)$
   - For a subformula $(\varphi_1 \wedge \varphi_2)$ set $\mathcal{L}(s) := \mathcal{L}(s) \cup \{\varphi_1 \wedge \varphi_2\}$ if $\varphi_1, \varphi_2 \in \mathcal{L}(s)$.
   - For a subformula $(\varphi_1 \; \mathrm{U}^{\leq \tau}_{\geq \rho} \; \varphi_2)$ use the following algorithm based on matrix multiplication.

Labeling for $(\varphi_1 \, \mathrm{U}^{\leq \tau}_{\geq \rho} \, \varphi_2)$

Assume $\tau < \infty$. Let $s_1, \ldots, s_n$ be the states of $\mathcal{S}$.

- Define a $n \times n$ transition matrix $A = (\alpha_{ij})$ by

$$\alpha_{ij} = \begin{cases} \mathcal{T}(s_i, s_j) & \text{if } \varphi_1 \in \mathcal{L}(s_i) \text{ and } \varphi_2 \notin \mathcal{L}(s_i) \\ 1 & \text{else if } i = j \\ 0 & \text{otherwise} \end{cases}$$

  By this, some transition probabilities are given: Rows represent the “from-states” and columns represent the “to-states”.
- Thus, the matrix A^τ intuitively gives transition probabilities for τ transition steps.

Labeling for ($\varphi_1 \; \mathrm{U}^{\leq \tau}_{\geq \rho} \; \varphi_2$) (contd.)

- Define a vector $P = (p_k)$ of size n with

$$p_k = \begin{cases} 1 & \text{if } \varphi_2 \in \mathcal{L}(s_k) \\ 0 & \text{otherwise} \end{cases}$$

  Intuitively, this represents the set of states in which φ_2 (the postcondition) is true.
- Calculate the vector $P' := A^{\tau} \cdot P$. This represents the probabilities of each state modeling the until-formula.
- Thus, for each state s_k redefine $\mathcal{L}(s_k) := \mathcal{L}(s_k) \cup \{\varphi_1 \; \mathrm{U}^{\leq \tau}_{\geq \rho} \; \varphi_2\}$ if the k-th entry in P' is at least ρ.

Labeling for $(\varphi_1 \ \mathrm{U}^{\leq \tau}_{\geq \rho} \ \varphi_2)$ (contd.)

- Define a vector $P = (p_k)$ of size n with

$$p_k = \begin{cases} 1 & \text{if } \varphi_2 \in \mathcal{L}(s_k) \\ 0 & \text{otherwise} \end{cases}$$

  Intuitively, this represents the set of states in which φ_2 (the postcondition) is true.
- Calculate the vector $P' := A^\tau \cdot P$. This represents the probabilities of each state modeling the until-formula.
- Thus, for each state s_k redefine $\mathcal{L}(s_k) := \mathcal{L}(s_k) \cup \{\varphi_1 \ \mathrm{U}^{\leq \tau}_{\geq \rho} \ \varphi_2\}$ if the k-th entry in P' is at least ρ.

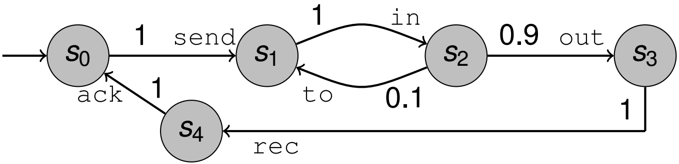
Labeling in special cases

- The case with $\tau = \infty$ must be treated separately.
- Other extreme cases can be calculated with separate algorithms for optimization reasons.
- The weak until operator W can be transformed into a formula with just U.
- There is another algorithm for calculation of P' with slightly different complexity.

Outline

Verification of a communication protocol

As an example, we will now verify a simple (fictional) communication protocol. It provides error free communication from a sender to a receiver over a lossy medium. We assume that acknowledgements are never lost though.



We also assume the message loss probability to be 10%.

Properties of interest

- An interesting question to pose is, if the system is able to meet certain deadlines, for example if there is a probability of at least 99% that a message is received (s_4) at most 6 time units after it is send (s_0).
- For a single event, this can be expressed by

$$s_0 \longrightarrow \mathrm{F}^{\leq 6}_{\geq 0.99} s_4$$

where F is a generalized finally-operator with time and probability.

Properties of interest

- An interesting question to pose is, if the system is able to meet certain deadlines, for example if there is a probability of at least 99% that a message is received (s_4) at most 6 time units after it is send (s_0).
- For a single event, this can be expressed by

$$s_0 \longrightarrow \mathrm{F}^{\leq 6}_{\geq 0.99} s_4$$

where F is a generalized finally-operator with time and probability.

Proving the assertion

- First of all, we translate the formula into a "pure until-formula" and name the subformulae:

$$\varphi = \underbrace{s_0}_{\varphi_1} \rightarrow \underbrace{\left(\underbrace{true}_{\varphi_2}\ \mathrm{U}^{\leq 6}_{\geq 0.99}\ \underbrace{s_4}_{\varphi_3}\right)}_{\varphi_4}$$

- Now states will labeled with atomic propositions:
  - Label s_0 with φ_1.
  - Label every state with φ_2.
  - Label s_4 with φ_3.

Proving the assertion

- First of all, we translate the formula into a “pure until-formula” and name the subformulae:

$$\varphi = \underbrace{s_0}_{\varphi_1} \rightarrow \underbrace{\left(\underbrace{true}_{\varphi_2} \; \mathrm{U}^{\leq 6}_{\geq 0.99} \; \underbrace{s_4}_{\varphi_3}\right)}_{\varphi_4}$$

- Now states will labeled with atomic propositions:
  - Label s_0 with φ_1.
  - Label every state with φ_2.
  - Label s_4 with φ_3.

Construction of matrix A and vector P

We construct the matrix A and the vector P according to the definition given earlier:

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0.1 & 0 & 0.9 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \qquad P = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Labeling for $\varphi_4 = \varphi_2 \, \mathrm{U}^{\leq 6}_{\geq 0.99} \, \varphi_3$

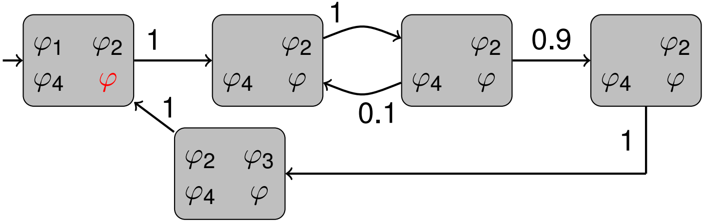
Next, we calculate $P' = A^6 \cdot P$:

$$A^6 = \begin{pmatrix} 0 & 0 & 0.01 & 0 & 0.99 \\ 0 & 0.001 & 0 & 0.009 & 0.99 \\ 0 & 0 & 0.01 & 0 & 0.99 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \qquad P' = \begin{pmatrix} 0.99 \\ 0.99 \\ 0.99 \\ 1 \\ 1 \end{pmatrix}$$

Since every entry in P' is at least 0.99, we conclude that every state is to be labeled with φ_4.

The initial state is labeled with φ

Finally, every state is labeled with $\varphi = (\varphi_1 \rightarrow \varphi_4)$ since every state is labeled with φ_4.



Since the initial state s_0 is labeled with φ, we have shown that the given protocol is a model for our assumption.

Summary

- The motivation for a logic like PCTL stems largely from an application view.
- We have seen practical usefulness for PCTL-expressible properties.
- The model checking algorithm shown is implemented in the PRISM model checking tool.

H. Hansson and B. Jonsson.
A logic for reasoning about time and reliability.
Technical Report R90013, Swedish Institute of Computer Science and Department of Computer Systems, Uppsala University, Dec. 1994.