SS 2017
Software Verification
Probabilistic modelling – PCTL Model checking

Prof. Dr. Holger Schlingloff \(^{1,2}\)
Dr. Esteban Pavese \(^1\)

(1) Institut für Informatik der Humboldt Universität
(2) Fraunhofer Institut für offene Kommunikationssysteme FOKUS
Recap

- What is a Discrete Time Markov Chain?
- For what sort of problems are they useful?
- What is a probability measure?
- How did we define (so far) outcomes and events on a DTMC?
- How do we calculate the probability of these events?
The logic PCTL

- PCTL stands for *Probabilistic Computational Tree Logic*
  - introduced by Hansson/Jonsson in 1994
  - essentially the same as pCTL introduced by de Alfaro / Bianco in 1995
- Very similar to CTL, but quantitative operators A and E are replaced by a *probabilistic operator P*
- Introduces a bounded version of Until
PCTL - Examples

\[ \text{send} \implies P_{\geq 0.999} \Diamond \text{receive} \]

If at the current state we \textit{send}, then with probability at least 0.999, the message is eventually received.

\[ P = \frac{1}{6} \Diamond (\text{die with 6 pips}) \]

Following the Knuth/Yao protocol with coins, a die with 6 points is generated with probability 1/6.
PCTL – Syntax and semantics

\[ \phi := true | a | \phi \land \phi | \neg \phi | P_{\sim p} \psi \]

\[ \psi := X \phi | \phi U \leq^k \phi | \phi U \phi \]

As was the case with CTL, \( \phi \) denote state formulae, \( \psi \) denote path formulae, which are quantified by the \( P \) operator. CTL syntactic sugar applies (F, G)

\[ \sim \in \{ <, \leq, =, \geq, > \}; \ k \in \mathbb{N} \]
PCTL – Syntax and semantics

- PCTL formulae are always *state* formulae

- We usually say a DTMC satisfies a formula iff its initial state satisfies the formula. We will be explicit if we refer to a state other than the initial

- The semantics for the non-probabilistic operators are exactly as in CTL
PCTL – Syntax and semantics

- Semantics for the operator $P$
  - intuitively, $s \models P_\sim p \psi$ if the probability mass for all paths stemming from $s$, that satisfy $\psi$, is $\sim p$
  - for example, $send \implies P_{\geq 0.1}X receive$ means that, if we send at this state, the paths where the immediate next step is to receive have a probability of at least 0.1.
PCTL – Syntax and Semantics

- As already noted, CTL syntactic sugar applies
- There are also some probabilistic equivalences

\[
P_{<p} F \phi = P_{1-p} G \neg \phi
\]

\[
P_{\geq p} F \phi = P_{1-p} G \neg \phi
\]

- and all alike
PCTL vs. CTL

- Some PCTL formulae bear some relationship with CTL when we consider extreme (0 or 1) probabilities
- That is, *qualitative* properties rather than *quantitative*

\[ P_{>0} F \phi \text{ is the same as } EF \phi \]

\[ P_{\geq 1} F \phi \text{ is the same as } AF \phi \]
Almost!

- The probabilistic version is a bit weaker.

In CTL $s_0 \models AF\{heads\}$ is false.

In PCTL $s_0 \models P_{\geq 1}F\{heads\}$ is true.

The difference is the only failing trace is infinite, and has probability zero.
PCTL -- Measurability

- PCTL properties predicate about sets of paths over the DTMC
- All of these sets of paths are *measurable*:
  - Operator $X$ denotes cones induced by a prefix of length 1. Since the model is finite, so are these cones
  - Operator $U^{\leq k}$ considers all prefixes of size up to $k$. Again, a finite number
  - Operator $U$ considers all finite prefixes. In this case, we have an infinite (but countable) union
PCTL model checking

• Even though we define satisfaction of PCTL for the initial state, the model checking procedure is global

• That is, the basic model checking procedure computes
  - given a PCTL formula $\phi$ and a DTMC
  - the procedure outputs the set $\text{Sat}(\phi)$ of states that satisfy $\phi$
PCTL model checking

- The basic algorithm resembles that of CTL model checking
  - First, we obtain the set of subformulae of $\phi$
  - Iterating through these subformulae (order by size), we label each state with the subformula iff the state satisfies said subformula
  - At the end of the procedure, we look at the states that are labeled with $\phi$
- All the difficulty lies in the labelling.
PCTL model checking

- Non-probabilistic formulae are easy

\[
\begin{align*}
\text{Sat}(true) & = S \\
\text{Sat}(a) & = \{ s \in S | a \in L(s) \} \\
\text{Sat}(\phi \land \psi) & = \text{Sat}(\phi) \cap \text{Sat}(\psi) \\
\text{Sat}(\neg \phi) & = S \setminus \text{Sat}(\phi)
\end{align*}
\]

- In order to solve the probabilistic operator, we need to calculate the probability of satisfying the inner formula for each state
PCTL model checking

- Let's define

\[ Pr(s, \phi) = \mu(\{\pi | \pi_0 = s \land \pi \models \phi\}) \]

- that is, the cumulative probability of all paths starting at \( s \) that satisfy the given formula

- we define satisfiability for probabilistic formula like so
PCTL model checking

- Probabilistic next (X) operator

\[ s \models P_{\sim p} X \phi \iff Pr(s, X \phi) \sim p \]

- If we already computed the set Sat(\(\phi\)), then

\[
Pr(s, X \phi) = \sum_{t \in Sat(\phi)} P(s, t)
\]
Example

\[ P_{\geq 0.9}(X(\neg \text{try} \lor \text{ok})) \]
\[ (\neg \text{try} \lor \text{ok}) = \{s_0, s_2, s_3\} \]
\[ Pr(s_0, X(\neg \text{try} \lor \text{ok})) = 0 \]
\[ Pr(s_1, X(\neg \text{try} \lor \text{ok})) = 0.98 + 0.01 \]
\[ Pr(s_2, X(\neg \text{try} \lor \text{ok})) = 1 \]
\[ Pr(s_3, X(\neg \text{try} \lor \text{ok})) = 1 \]
PCTL model checking

- Probabilistic next (X) operator
- It is simpler to compute all at once
- Define

\[ \nu_i^\phi = 1 \text{ if } s_i \models \phi, \ 0 \text{ otherwise} \]

- and consider the stochastic adjacency matrix \( P \), then the probability of satisfaction of X operator is defined for all states at once

\[ Pr(X\phi) = P \times \nu^\phi \]
Example

\[ P = \begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0.01 & 0.01 & 0.98 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix} \quad v^\phi = \begin{pmatrix}
1 \\
0 \\
1 \\
1
\end{pmatrix} \]

\[ P \times v^\phi = \begin{pmatrix}
0 \\
0.99 \\
1 \\
1
\end{pmatrix} \]
PCTL model checking

- Probabilistic bounded until \((U^{\leq k})\) operator

\[ s \models P_{\sim p} \phi_1 U^{\leq k} \phi_2 \iff Pr(s, \phi_1 U^{\leq k} \phi_2) \sim p \]

- We will also leverage on the sets \(\text{Sat}(\phi)\)

- First we can easily compute some sets

\[ S^0 = S \setminus (\text{Sat}(\phi_1) \cup \text{Sat}(\phi_2)) \]

\[ S^1 = \text{Sat}(\phi_2) \]
Bounded Until

• Let $S^? = S \setminus (S^0 \cup S^1)$, we solve for $S^?$ by the equation system

\[
Pr(s, \phi_1 U \leq^k \phi_2) = \begin{cases} 
1 & \text{if } s \in S^1 \\
0 & \text{if } s \in S^0 \\
0 & \text{if } s \in S^? \land k = 0 \\
\sum_{t \in S} P(s, t) \times Pr(t, \phi_1 U \leq^{k-1} \phi_2) & \text{otherwise}
\end{cases}
\]

• it is a huge (and boring) equation system...
Bounded Until

- But! If multiplying by $P$ gave us one step, let’s multiply $k$ times by $P$!
- Actually, let’s leverage $S^0$ and $S^1$
  \[
P'(s, t) = \begin{cases} 
P(s, t) & \text{if } s \in S^? \\
1 & \text{if } s \in S^1 \land s = t \\
0 & \text{otherwise}
\end{cases}
\]
- and finally
  \[
\nu^\phi_1 U^{\leq 0} \phi_2 = \nu^\phi_2
\]
  \[
\nu^\phi_1 U^{\leq k} \phi_2 = P' \times \nu^\phi_1 U^{\leq k-1} \phi_2
\]
Bounded Until

- This method requires $k$ matrix/vector multiplications (each quadratic in the number of states)

- ... so it only makes sense for smallish $k$

- *usually* matrix multiplication is improved by a D&C approach
  - .. not in this case. $P$ is usually sparse
Example

\[ P_{>0.98}(\text{trueU}^{\leq 2} \text{ok}) \]

\[ S^1 = \{s_3\} \quad S^0 = \emptyset \quad S^? = \{s_0, s_1, s_2\} \]

\[ P' = P \quad \nu^{\text{trueU}^{\leq 0} \text{ok}} = \nu^{\text{ok}} = (0 \ 0 \ 0 \ 1) \]

\[ \nu^{\text{trueU}^{\leq 2} \text{ok}} = P' \times \nu^{\text{trueU}^{\leq 1} \text{ok}} \]

\[ = P' \times P' \times \nu^{\text{trueU}^{\leq 0} \text{ok}} = P' \times P' \times \nu^{\text{ok}} \]
Example

\[
\begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0.01 & 0.01 & 0.98 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\times
\begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0.01 & 0.01 & 0.98 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\times
\begin{pmatrix}
0 \\
0 \\
0 \\
1
\end{pmatrix}
= 
\begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0.01 & 0.01 & 0.98 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1
\end{pmatrix}
\times
\begin{pmatrix}
0.98 \\
0 \\
0 \\
1
\end{pmatrix}
= 
\begin{pmatrix}
0.9898 \\
0 \\
1
\end{pmatrix}
\]
PCTL model checking

- Unbounded until operator

- We actually saw this one last class!

- We will also precompute some sets

\[ S^0 = \text{Sat}(P_{\leq 0} \phi_1 U \phi_2) \]

\[ S^1 = \text{Sat}(P_{\geq 1} \phi_1 U \phi_2) \]
Computing $S^0$

- that is, all states that cannot satisfy the qualitative until
- First, compute $S^{\text{maybe}}$

$$S^{\text{maybe}} = \text{Sat}(P_{>0} \phi_1 U \phi_2)$$

- i.e. find all states that reach a $\phi_2$ state by a finite path not leaving $\phi_1$ states
  - this is a simple graph-based operation
- Now, $S^0 = S \setminus S^{\text{maybe}}$
Computing $S^1$

- Simply reuse $S^0$

- Compute $S^{\text{maybe not}}$, those states that can reach $S^0$ by a finite path (i.e., the states from which we can go into non-satisfaction states)

- Now $S^1 = S \setminus S^{\text{maybe not}}$
Unbounded Until

- Now solve the (smaller) equation system

\[
Pr(s, \phi_1 U \phi_2) = \begin{cases} 
1 & \text{if } s \in S^1 \\
0 & \text{if } s \in S^0 \\
\sum_{t \in S} P(s, t) \times Pr(t, \phi_1 U \phi_2) & \text{otherwise}
\end{cases}
\]

- This equation system has less unknowns than states
  - in particular, we remove those from \( S^1 \) and \( S^0 \)
- Solve by Gaussian elimination or iterative convergent approaches
Example

\[ P_{\geq 0.99}(try \cup ok) \]

\[
S^0 = \{ s_0, s_2 \} \quad S^1 = \{ s_3 \}
\]

\[
x_0 = 0 \\
x_1 = 0.01 \cdot x_1 + 0.01 \cdot x_2 + 0.98 \cdot x_3 \\
x_2 = 0 \\
x_3 = 1
\]