Probabilistic computational tree logic (pCTL)

- Temporal logic for describing properties of MCs
- Extension of the temporal logic CTL
  - Key addition is the probabilistic operator $P$
  - Which “replaces” CTL’s universal and existential path quantification

screencast by Anne Remke, University of Twente
Probabilistic computational tree logic (pCTL)

**State-formulas**

\[ \Phi ::= a \mid \neg \Phi \mid \Phi \lor \Phi \mid P_{\leq p}(\varphi) \]

with \( a \in \text{AP} \), probability \( p \) and comparison operator \( \leq \)

\[ P_{\leq p}(\varphi) \] probability that paths fulfill \( \varphi \) is \( \leq p \)
Probabilistic computational tree logic (pCTL)

Path-formulas

\[ \varphi ::= X \Phi \mid \Phi U \leq k \Psi \]

with integer \( k \)

- \( X \Phi \): next state fulfills \( \Phi \)
- \( \Phi U \leq k \Psi \): \( \Phi \) holds along the path until \( \Psi \) holds within \( k \) steps
Example pCTL requirements

- Probability of not going down in the next state is at least 95%:

- Probability of going down within 5 steps is at most 1%:

- Probability of going down within 5 steps after continuously operating with at least 2 processors is at most 1%:

⇒ Complicated measures can be specified by nesting of operators
Model-checking pCTL

- Checking whether a state \( s \) in an L-DTMC satisfies a pCTL formula \( \Phi \) is performed in the same way as for CTL:
  - Compute \textit{recursively} the set \( \text{Sat}(\Phi) \) of states that satisfy \( \Phi \)
  - Check whether state \( s \) belongs to \( \text{Sat}(\Phi) \)

- For the non-probabilistic part: as for CTL

- How to compute \( \text{Sat}(\Phi) \) for the probabilistic operators?
Basic algorithm proceeds by induction on parse tree of $\Phi$.
Assume that $\Phi = (\neg \text{fail} \land \text{try}) \rightarrow P_{>0.95}[\neg \text{fail} \cup \text{succ}]$:
Bottom-up Computation

- For the non-probabilistic operators:
  - $Sat(a) = \{ s \in S \mid a \in L(s) \}$;
  - $Sat(\neg \Phi) = S \setminus Sat(\Phi)$;
  - $Sat(\Phi_1 \land \Phi_2) = Sat(\Phi_1) \cap Sat(\Phi_2)$

- For the probabilistic part $P_{\downarrow p}[\varphi]$:
  - Compute $\text{Prob}(s, \varphi)$ for all states $s \in S$. 
Model-checking pCTL's Next

- \( s \in \text{Sat}(\mathcal{P}_{\leq p}(\varphi)) \) iff \( \text{Prob}(s, \varphi) \leq p \)

- For next: \( \varphi = X \Phi \)

- Recursive computation: assume \( \text{Sat}(\Phi) \) states are known

- We have
  \[
  \text{Prob}(s, X \Phi) = \sum_{s' \in \text{Sat}(\Phi)} P(s, s')
  \]
Example

\[ P \geq 0.8 [X b] \]

\[
P = \begin{pmatrix}
0 & 0.1 & 0.9 & 0 & 0 & 0 \\
0.4 & 0 & 0 & 0.6 & 0 & 0 \\
0 & 0 & 0.1 & 0.1 & 0.5 & 0.3 \\
0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 & 0.7 & 0.3
\end{pmatrix}
\]
Model-checking pCTL’s $k$-bounded Until

- $s \in \text{Sat}(\mathcal{P}_{\leq p}(\varphi))$ iff $\text{Prob}(s, \Phi \cup \leq^k \Psi) \leq p$

\[
\text{Prob}(s, \Phi \cup \leq^k \Psi) =
\]

- 1, if $s \in \text{Sat}(\Psi)$
- $\sum_{s' \in S} P(s, s') \cdot \text{Prob}(s', \Phi \cup \leq^{k-1} \Psi)$, for $k > 0$ and $s \in \text{Sat}(\Phi) \setminus \text{Sat}(\Psi)$
- 0, otherwise

- Solution via fixed-point iteration algorithm
- More efficient algorithms for $\mathcal{P}_{>0}(\Phi \cup \leq^k \Psi)$ and $\mathcal{P}_{\geq1}(\Phi \cup \leq^k \Psi)$
Teaser

Check out my next pencast to see how this can be done more efficiently!