Modeling and Verification of Probabilistic Systems
Lecture 17: Continuous Stochastic Logic

Joost-Pieter Katoen

Lehrstuhl für Informatik 2
Software Modeling and Verification Group

http://www-i2.informatik.rwth-aachen.de/i2/mvps11/

July 4, 2011

Overview

1. CSL Syntax

2. CSL Semantics

3. CSL Model Checking

4. Complexity

5. Summary

Overview

1. CSL Syntax

2. CSL Semantics

3. CSL Model Checking

4. Complexity

5. Summary

Continuous Stochastic Logic

- CSL is a language for formally specifying properties over CTMCs.
- It is a branching-time temporal logic based on CTL.
- Formula interpretation is Boolean, i.e., a state satisfies a formula or not.
- Like in PCTL, the main operator is $\mathbb{P}_J(\varphi)$
  - where φ constrains the set of paths and J is a threshold on the probability.
  - it is the probabilistic counterpart of $\exists$ and $\forall$ path-quantifiers in CTL.
- The new features are a timed version of the next and until-operator.
  - $\bigcirc^I \Phi$ asserts that a transition to a Φ-state can be made at time $t \in I$.
  - $\Phi \, U^I \, \Psi$ asserts that a Ψ-state can be reached via Φ-states at time $t \in I$.

CTMCs — A transition system perspective

Continuous-time Markov chain

A CTMC $\mathcal{C}$ is a tuple $(S, \mathbf{P}, r, \iota_{\text{init}}, AP, L)$ with:

- S is a countable nonempty set of states
- $\mathbf{P} : S \times S \to [0,1]$, transition probability function s.t. $\sum_{s'} \mathbf{P}(s, s') = 1$
- $r : S \to \mathbb{R}_{>0}$, rate assigning function
- $\iota_{\text{init}} : S \to [0,1]$, the initial distribution with $\sum_{s \in S} \iota_{\text{init}}(s) = 1$
- AP is a set of atomic propositions.
- $L : S \to 2^{AP}$, the labeling function, assigning to state s, the set $L(s)$ of atomic propositions that are valid in s.

Residence time

The average residence time in state s is $\frac{1}{r(s)}$.

CSL syntax   [Baier, Katoen & Hermanns, 1999]

Continuous Stochastic Logic: Syntax

CSL consists of state- and path-formulas.

- CSL *state formulas* over the set AP obey the grammar:

$$\Phi ::= \text{true} \;\bigg|\; a \;\bigg|\; \Phi_1 \wedge \Phi_2 \;\bigg|\; \neg\Phi \;\bigg|\; \mathbb{P}_J(\varphi)$$

where $a \in AP$, φ is a path formula and $J \subseteq [0,1]$, $J \neq \varnothing$ is a non-empty interval.

- CSL *path formulae* are formed according to the following grammar:

$$\varphi ::= \bigcirc^I \Phi \;\bigg|\; \Phi_1 \, \mathsf{U}^I \, \Phi_2$$

where Φ, Φ_1, and Φ_2 are state formulae and $I \subseteq \mathbb{R}_{\geqslant 0}$ an interval.

Abbreviate $\mathbb{P}_{[0,0.5]}(\varphi)$ by $\mathbb{P}_{\leqslant 0.5}(\varphi)$ and $\mathbb{P}_{]0,1]}(\varphi)$ by $\mathbb{P}_{>0}(\varphi)$.

Continuous Stochastic Logic

- CSL *state formulas* over the set AP obey the grammar:

$$\Phi ::= \text{true} \;\bigg|\; a \;\bigg|\; \Phi_1 \wedge \Phi_2 \;\bigg|\; \neg\Phi \;\bigg|\; \mathbb{P}_J(\varphi)$$

where $a \in AP$, φ is a path formula and $J \subseteq [0,1]$, $J \neq \varnothing$.

- CSL *path formulae* are formed according to the following grammar:

$$\varphi ::= \bigcirc^I \Phi \;\bigg|\; \Phi_1 \, \mathsf{U}^I \, \Phi_2$$

where Φ, Φ_1, and Φ_2 are state formulae and $I \subseteq \mathbb{R}_{\geqslant 0}$ an interval.

Intuitive semantics

- $s_0 \, t_0 \, s_1 \, t_1 \ldots \models \Phi \, \mathsf{U}^I \, \Psi$ if Ψ is reached at $t \in I$ and prior to t, Φ holds.
- $s \models \mathbb{P}_J(\varphi)$ if probability that paths starting in s fulfill φ lies in J.

Overview

Derived operators

$$\Diamond\Phi \;=\; \text{true}\,\mathsf{U}\,\Phi$$

$$\Diamond^I\Phi \;=\; \text{true}\,\mathsf{U}^I\Phi$$

$$\mathbb{P}_{\leqslant p}(\Box\Phi) \;=\; \mathbb{P}_{>1-p}(\Diamond\neg\Phi)$$

$$\mathbb{P}_{(p,q)}(\Box^I\Phi) \;=\; \mathbb{P}_{[1-q,1-p]}(\Diamond^I\neg\Phi)$$

Example properties

- ▶ Transient probabilities to be in *goal* state at time point 4:

$$\mathbb{P}_{\geqslant 0.92}\left(\Diamond^{=4}\,goal\right)$$

- ▶ With probability $\geqslant 0.92$, a goal state is reached legally:

$$\mathbb{P}_{\geqslant 0.92}\left(\neg\,illegal\;\mathsf{U}\;goal\right)$$

- ▶ ... in maximally 137 time units: $\quad\mathbb{P}_{\geqslant 0.92}\left(\neg\,illegal\;\mathsf{U}^{\leqslant 137}\,goal\right)$
- ▶ ... once there, remain there almost surely for the next 31 time units:

$$\mathbb{P}_{\geqslant 0.92}\left(\neg\,illegal\;\mathsf{U}^{\leqslant 137}\;\mathbb{P}_{=1}(\Box^{[0,31]}\,goal)\right)$$

Paths in a CTMC

Timed paths

Paths in CTMC $\mathcal{C}$ are maximal (i.e., infinite) paths of alternating states and time instants:

$$\pi \;=\; s_0 \xrightarrow{t_0} s_1 \xrightarrow{t_1} s_2\cdots$$

such that $s_i \in S$ and $t_i \in \mathbb{R}_{>0}$. Let $Paths(\mathcal{C})$ be the set of paths in $\mathcal{C}$ and $Paths^*(\mathcal{C})$ the set of finite prefixes thereof.

Notations

- ▶ Let $\pi[i] := s_i$ denote the $(i+1)$-st state along the timed path π.
- ▶ Let $\pi\langle i\rangle := t_i$ the time spent in state s_i.
- ▶ Let $\pi@t$ be the state occupied in π at time $t \in \mathbb{R}_{\geqslant 0}$, i.e. $\pi@t := \pi[i]$ where i is the smallest index such that $\sum_{j=0}^{i}\pi\langle j\rangle > t$.

CSL semantics (1)

Notation

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

Satisfaction relation for state formulas

The satisfaction relation $\models$ is defined for CSL state formulas by:

$$
\begin{aligned}
s &\models a & &\text{iff}\quad a \in L(s)\\
s &\models \neg\Phi & &\text{iff}\quad \text{not } (s \models \Phi)\\
s &\models \Phi \wedge \Psi & &\text{iff}\quad (s \models \Phi) \text{ and } (s \models \Psi)\\
s &\models \mathbb{P}_J(\varphi) & &\text{iff}\quad Pr(s \models \varphi) \in J
\end{aligned}
$$

where $Pr(s \models \varphi) = Pr_s\{\pi \in Paths(s) \mid \pi \models \varphi\}$.

This is as for PCTL, except that Pr is the probability measures on cylinder sets of timed paths in CTMC $\mathcal{C}$.

CSL semantics (2)

Satisfaction relation for path formulas

Let $\pi = s_0\, t_0\, s_1\, t_1\, s_2 \ldots$ be an infinite path in CTMC $\mathcal{C}$.

The satisfaction relation $\models$ is defined for state formulas by:

$$\pi \models \bigcirc^I \Phi \quad \text{iff} \quad s_1 \models \Phi \;\wedge\; t_0 \in I$$

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

Standard next- and until-operators

- $X\Phi \;\equiv\; \bigcirc^I \Phi$ with $I = \mathbb{R}_{\geqslant 0}$.
- $\Phi\, U\, \Psi \;\equiv\; \Phi\, U^I\, \Psi$ with $I = \mathbb{R}_{\geqslant 0}$.

Measurability

CSL measurability

For any CSL path formula φ and state s of CTMC $\mathcal{C}$,
the set $\{\, \pi \in \mathit{Paths}(s) \mid \pi \models \varphi \,\}$ is measurable.

Proof:

Rather straightforward; left as an exercise.

Overview

CSL model checking

CSL model checking problem

Input: a finite CTMC $\mathcal{C} = (S, \mathbf{P}, r, \iota_{\mathrm{init}}, AP, L)$, state $s \in S$, and CSL state formula Φ

Output: yes, if $s \models \Phi$; no, otherwise.

Basic algorithm

In order to check whether $s \models \Phi$ do:

1. Compute the satisfaction set $Sat(\Phi) = \{\, s \in S \mid s \models \Phi \,\}$.
2. This is done recursively by a bottom-up traversal of Φ's parse tree.
   - The nodes of the parse tree represent the subformulae of Φ.
   - For each node, i.e., for each subformula Ψ of Φ, determine $Sat(\Psi)$.
   - Determine $Sat(\Psi)$ as function of the satisfaction sets of its children:
     e.g., $Sat(\Psi_1 \wedge \Psi_2) = Sat(\Psi_1) \cap Sat(\Psi_2)$ and $Sat(\neg \Psi) = S \setminus Sat(\Psi)$.
3. Check whether state s belongs to $Sat(\Phi)$.

Core model checking algorithm

Propositional formulas

$Sat(\cdot)$ is defined by structural induction as follows:

$$
\begin{aligned}
Sat(\text{true}) &= S \\
Sat(a) &= \{\, s \in S \mid a \in L(s) \,\}, \text{ for any } a \in AP \\
Sat(\Phi \wedge \Psi) &= Sat(\Phi) \cap Sat(\Psi) \\
Sat(\neg \Phi) &= S \setminus Sat(\Phi).
\end{aligned}
$$

Probabilistic operator $\mathbb{P}$

In order to determine whether $s \in Sat(\mathbb{P}_J(\varphi))$, the probability $Pr(s \models \varphi)$ for the event specified by φ needs to be established. Then

$$Sat(\mathbb{P}_J(\varphi)) = \{\, s \in S \mid Pr(s \models \varphi) \in J \,\}.$$

Let us consider the computation of $Pr(s \models \varphi)$ for all possible φ.

The next-step operator

Recall that: $s \models \mathbb{P}_J(\bigcirc^I \Phi)$ if and only if $Pr(s \models \bigcirc^I \Phi) \in J$.

Lemma

$$Pr(s \models \bigcirc^I \Phi) = \underbrace{\left(e^{-r(s)\cdot \inf I} - e^{-r(s)\cdot \sup I} \right)}_{\text{probability to leave } s \text{ in interval } I} \cdot \sum_{s' \in Sat(\Phi)} \mathbf{P}(s, s').$$

Algorithm

Considering the above equation for all states simultaneously yields:

$$\left(Pr(s \models \bigcirc \Phi) \right)_{s \in S} = \mathbf{b}_I^T \cdot \mathbf{P}$$

with $\mathbf{b}_I$ is defined by $b_I(s) = e^{-r(s)\cdot \inf I} - e^{-r(s)\cdot \sup I}$ if $s \in Sat(\Phi)$ and 0 otherwise, and $\mathbf{b}_I^T$ is the transposed variant of $\mathbf{b}_I$.

Time-bounded until (1)

Recall that: $s \models \mathbb{P}_J(\Phi \, \mathsf{U}^{\leqslant t} \, \Psi)$ if and only if $Pr(s \models \Phi \, \mathsf{U}^{\leqslant t} \, \Psi) \in J$.

Lemma

Let $S_{=1} = Sat(\Psi)$, $S_{=0} = S \setminus (Sat(\Phi) \cup Sat(\Psi))$, and $S_? = S \setminus (S_{=0} \cup S_{=1})$. Then:

$$
Pr(s \models \Phi \, \mathsf{U}^{\leqslant t} \, \Psi) =
\begin{cases}
1 & \text{if } s \in S_{=1} \\
0 & \text{if } s \in S_{=0} \\
\displaystyle\int_0^t \sum_{s' \in S} \mathbf{R}(s, s') \cdot e^{-r(s)\cdot x} \cdot Pr(s' \models \Phi \, \mathsf{U}^{\leqslant t-x} \, \Psi)\, dx & \text{otherwise}
\end{cases}
$$

This is a slight generalisation of the Volterra integral equation system for timed reachability.

Time-bounded until (2)

Let $S_{=1} = Sat(\Psi)$, $S_{=0} = S \setminus (Sat(\Phi) \cup Sat(\Psi))$, and $S_? = S \setminus (S_{=0} \cup S_{=1})$. Then:

$$
Pr(s \models \Phi \, \mathsf{U}^{\leqslant t} \, \Psi) =
\begin{cases}
1 & \text{if } s \in S_{=1} \\
0 & \text{if } s \in S_{=0} \\
\displaystyle\int_0^t \sum_{s' \in S} \mathbf{R}(s, s') \cdot e^{-r(s)\cdot x} \cdot Pr(s' \models \Phi \, \mathsf{U}^{\leqslant t-x} \, \Psi)\, dx & \text{otherwise}
\end{cases}
$$

Recall lemma from the previous lecture

$$
\underbrace{Pr(s \models \overline{F} \, \mathsf{U}^{\leqslant t} \, G)}_{\text{timed reachability in } \mathcal{C}}
= \underbrace{Pr(s \models \Diamond^{=t} G)}_{\text{in } \mathcal{C}[F \cup G]}
= \underbrace{\underline{p}(t) \text{ with } \underline{p}(0) = \mathbf{1}_s}_{\text{transient prob. in } \mathcal{C}[F \cup G]}.
$$

Phrased using CSL state formulas

$$
\underbrace{Pr(s \models \Phi \, \mathsf{U}^{\leqslant t} \, \Psi)}_{\text{timed reachability in } \mathcal{C}}
= \underbrace{Pr(s \models \Diamond^{=t} \Psi)}_{\text{in } \mathcal{C}[Sat(\neg\Phi) \cup Sat(\Psi)]}
= \underbrace{\underline{p}(t) \text{ with } \underline{p}(0) = \mathbf{1}_s}_{\mathcal{C}[Sat(\neg\Phi) \cup Sat(\Psi)]}.
$$

Time-bounded until (3)

Algorithm for checking $Pr(s \models \Phi \, U^{\leqslant t} \, \Psi) \in J$

1. If $t = \infty$, then use approach for until (as in PCTL): solve a system of linear equations.
2. Determine recursively $Sat(\Phi)$ and $Sat(\Psi)$.
3. Make all states in $S \setminus Sat(\Phi)$ and $Sat(\Psi)$ absorbing.
4. Uniformize the resulting CTMC with respect to its maximal rate.
5. Determine the transient probability at time t using s as initial distribution.
6. Return yes if transient probability of all Ψ-states lies in J, and no otherwise.

Time-bounded until (4)

Possible optimizations

1. Make all states in $S \setminus Sat(\exists(\Phi \, U \, \Psi))$ absorbing.
2. Make all states in $Sat(\forall(\Phi \, U \, \Psi))$ absorbing.
3. Replace the labels of all states in $S \setminus Sat(\exists(\Phi \Psi))$ by unique label zero.
4. Replace the labels of all states in $Sat(\forall(\Phi \, U \, \Psi))$ by unique label one.
5. Perform bisimulation minimization on all states.

The last step collapses all states in $S \setminus Sat(\exists(\Phi \, U \, \Psi))$ into a single state, and does the same with all states in $Sat(\forall(\Phi \, U \, \Psi))$.

Preservation of CSL-formulas

Bisimulation and CSL-equivalence coincide

Let $\mathcal{C}$ be a finitely branching CTMC and s, t states in $\mathcal{C}$. Then:

$$s \sim_m t \quad \text{if and only if} \quad s \text{ and } t \text{ are CSL-equivalent.}$$

Remarks

If for CSL-formula Φ we have $s \models \Phi$ but $t \not\models \Phi$, then it follows $s \not\sim_m t$. A single CSL-formula suffices!

Preservation of CSL-formulas

Weak bisimulation and CSL-without-next-equivalence coincide

Let $\mathcal{C}$ be a finitely branching CTMC and s, t states in $\mathcal{C}$. Then:

$$s \approx_m t \quad \text{if and only if} \quad s \text{ and } t \text{ are CSL-without-next-equivalent.}$$

Here. CSL-without-next is the fragment of CSL where the next-operator $\bigcirc$ does not occur.

Remarks

If for CSL-without-next-formula Φ we have $s \models \Phi$ but $t \not\models \Phi$, then it follows $s \not\approx_m t$.

Uniformization and CSL

Uniformization and CSL

For any finite CTMC $\mathcal{C}$ with state space S, $r \geqslant \max\{\, r(s) \mid s \in S \,\}$ and Φ a CSL-without-next-formula:

$$Sat^{\mathcal{C}}(\Phi) \;=\; Sat^{\mathcal{C}'}(\Phi) \quad \text{where } \mathcal{C}' = \textit{unif}(r, \mathcal{C}).$$

Uniformization and CSL

For any uniformized CTMC: CSL-equivalence coincides with CSL-without-next-equivalence.

Overview

1. CSL Syntax

2. CSL Semantics

3. CSL Model Checking

4. **Complexity**

5. Summary

Time complexity

Let $|\Phi|$ be the size of Φ, i.e., the number of logical and temporal operators in Φ.

Time complexity of CSL model checking

For finite CTMC $\mathcal{C}$ and CSL state-formula Φ, the CSL model-checking problem can be solved in time
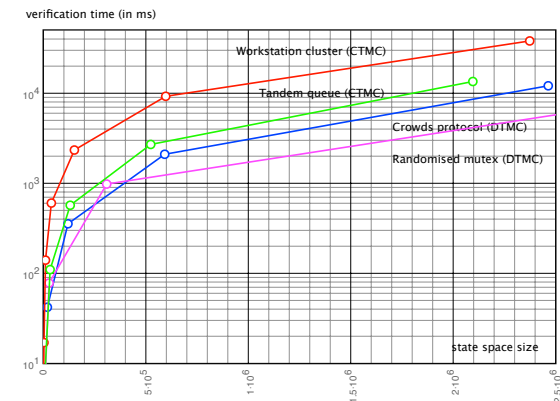
$$\mathcal{O}(\, poly(size(\mathcal{C})) \,\cdot\, t_{\max} \cdot |\Phi|\,)$$

where $t_{\max} \;=\; \max\{\, t \mid \Psi_1 \,\mathsf{U}^{\leqslant t}\Psi_2 \text{ occurs in } \Phi \,\}$ with and $t_{\max} = 1$ if Φ does not contain a time-bounded until-operator.

Some practical verification times

▶ command-line tool MRMC ran on a Pentium 4, 2.66 GHz, 1 GB RAM laptop.

▶ CSL formulas are time-bounded until-formulas.

Overview

Summary

- CSL is a variant of PCTL with timed next and timed until.
- Sets of paths fulfilling CSL path-formula φ are measurable.
- CSL model checking is performed by a recursive descent over Φ.
- The timed next operator amounts to a single vector-matrix multiplication.
- The time-bounded until-operator $U^{\leqslant t}$ is solved by uniformization.
- The worst-case time complexity is polynomial in the size of the CTMC and linear in the size of the formula.