Quantitative $\mu$-calculus and CTL based on constraint semirings

2nd Workshop on Quantitative Aspects of Programming Languages

Alberto Lluch Lafuente, Ugo Montanari
Dipartimento di Informatica, Università di Pisa
Introduction

- Classical Model Checking: reason about qualitative (boolean) aspects of systems.
- Quantitative Model Checking: reason about quantitative aspects of systems.
  ⇒ Examples: probabilistic, discounted, durational.
- Classical CSP: Problems with boolean constraints.
- Soft CSP: Problems with soft constraints.
  ⇒ Examples: probabilistic, fuzzy, weighted, multi-valued, etc.
- C-semirings as framework for Soft CSP
  ⇒ C-semirings as framework for Quantitative Model Checking?
Mutual Exclusion: Boolean Reasoning

- Can process \( p \) reach the c.s.?
  \[
  [\text{EF}_{p\text{-in\_cs}}(s_0) = true
  \]

- Can process \( p \) reach the c.s. and stay forever?
  \[
  [\text{EFEG}_{p\text{-in\_cs}}(s_0) = true
  \]
Mutual Exclusion: Reasoning about Processes

- Which processes can reach the c.s.?
- Which processes can reach the c.s. and stay forever?
Mutual Exclusion: Reasoning about Costs

- Min. cost for a process to use the c.s.?
- Min. cost for a process to use the c.s. forever?
**Mutual Exclusion: Reasoning about Probabilities**

- Max. probability for a process to use the c.s.?
- Max. probability for a process to use the c.s. forever?
C-Semirings (Definition)

A c-semiring is a tuple \( \langle A, +, \times, 0, 1 \rangle \) such that:

- \( A \) is a (possibly infinite) set;
- \( 0 \) and \( 1 \) are elements of \( A \);
- \( + : A \times A \rightarrow A \)
  - associative, commutative and idempotent;
  - \( a + 0 = a \) and \( a + 1 = 1 \).
- \( \times : A \times A \rightarrow A \)
  - associative and commutative;
  - distributes over \( + \);
  - \( a \times 0 = 0 \) and \( a \times 1 = a \).
C-Semirings (Instances and Application to QoS)

- Boolean semiring: \( \langle \{true, false\}, \lor, \land, false, true \rangle \)
  \( \Rightarrow \) Service/link availability, etc.

- Optimization semiring: \( \langle \mathbb{R}^+, \min, +, +\infty, 0 \rangle \)
  \( \Rightarrow \) Bandwidth, price, etc.

- Probabilistic c-semiring: \( \langle [0, 1], \max, \cdot, 0, 1 \rangle \)
  \( \Rightarrow \) Availability rate, performance, etc.

- Fuzzy c-semiring: \( \langle [0, 1], \max, \min, 0, 1 \rangle \)

- Set-based c-semiring: \( \langle 2^N, \cup, \cap, \emptyset, N \rangle \)
  \( \Rightarrow \) Capabilities, access rights, etc.
C-Semirings (Properties I)

- Cartesian products, exponentials and power constructions of c-semirings are c-semirings.
  \[ \Rightarrow \text{Combine multiple criteria, e.g. multiple QoS attributes.} \]

- The additive operation induces a partial order:
  \[ a \leq_S b \text{ iff } a + b = b \text{ (equiv. } \exists c \mid a + c = b). \]
  \[ \Rightarrow \text{E.g., partial order for } \langle \mathbb{R}^+, \min, +, +\infty, 0 \rangle \text{ is } \geq: \]
  \[ a \geq b \text{ iff } \min(a, b) = b. \]
C-Semirings (Properties II)

- \( \langle A, \leq_S \rangle \) is a complete lattice:
  - Every subset of \( A \) has a lub or join (\( \sqcup \)) which is \( + \);
  - Every subset of \( A \) has a glb or meet (\( \sqcap \)).
    \( \Rightarrow \) E.g., meet is \( \text{max} \) in \( \langle \mathbb{R}^+, \text{min}, +, +\infty, 0 \rangle \).
  - In practical cases the lattice is distributive.

- Sometimes the multiplicative operation is idempotent:
  - \( \times \) coincides with meet (\( \sqcap \));
  - \( + \) distributes over \( \times \);
  - \( \langle A, \leq_S \rangle \) is a distributive lattice.
    \( \Rightarrow \) E.g., boolean, fuzzy, set-based c-semirings.
Transition Systems

- Transition systems $M = \langle S, T \rangle$, where:
  - $S$ is a set of states;
  - $T \subseteq S \times S$ is a set of transitions.

- We assume $M$ to be image-finite, $T$ to be total.

- Runs of a system are maximal paths.

- A path is a sequence $s_0, s_1, s_2 \ldots$, such that $(s_i, s_{i+1}) \in T$. 
Boolean Model checking

- Which states of $M$ satisfy $\phi$?

$$[\phi] \subseteq 2^S$$

or

$$[\phi] : S \rightarrow \{true, false\}$$

C-Semiring Model Checking

- Which value of $A$ associates $\phi$ to each state of $M$?

$$[\phi] : S \rightarrow A.$$ 

, where $C = \langle A, +, \times, 0, 1 \rangle$ is a c-semiring.
Boolean CTL (Syntax)

\[ \phi ::= \text{true} \mid \text{false} \mid p \mid \neg \phi \mid \phi \lor \phi \mid \phi \land \phi \mid \kappa \psi \mid \kappa X \phi \]

\[ \kappa ::= E \mid A \]

\[ \psi ::= F \phi \mid G \phi \mid [\phi U \phi] \mid [\phi R \phi] \]

C-Semiring CTL (Syntax)

\[ \phi ::= a \mid v \mid f(\phi, \ldots, \phi) \mid \phi + \phi \mid \phi \times \phi \mid \kappa \psi \mid \kappa X \phi \]

\[ \kappa ::= \prod \mid \Sigma \mid \Pi \]

\[ \psi ::= F \phi \mid G \phi \mid [\phi U \phi] \mid [\phi R \phi] \]
C-CTL (Path Semantics over TS I)

\[
\begin{align*}
[a](s) &= a \\
[v](s) &= v(s) \\
[\phi_1 + \phi_2](s) &= [\phi_1](s) + [\phi_2](s) \\
[\phi_1 \times \phi_2](s) &= [\phi_1](s) \times [\phi_2](s) \\
[f(\phi_1, \ldots, \phi_n)](s) &= f([\phi_1](s), \ldots, [\phi_n](s))
\end{align*}
\]
C-CTL (Path Semantics over TS II)

\[ [\kappa X \phi](s) = \kappa_{(s, s') \in T} [\phi](s') \]
\[ [\kappa \psi](s) = \kappa_{p \in \gamma(s)} [\psi](p) \]
\[ [F \phi](p) = \sum_{i \geq 0} [\phi](s^p_i) \]
\[ [G \phi](p) = \prod_{i \geq 0} [\phi](s^p_i) \]
\[ [\phi_1 U \phi_2](p) = \sum_{i \geq 0} ([\phi_2](s^p_i) \times \prod_{0 \leq j < i} [\phi_1](s^p_j)) \]
\[ [\phi_1 R \phi_2](p) = \prod_{i \geq 0} ([\phi_2](s^p_i) + \sum_{0 \leq j < i} [\phi_1](s^p_j)) \]

, where \( \gamma(s) \) is the set of maximal paths starting at \( s \).
C-CTL Path Semantics (Temporal Operators)

Let $p = s_0, s_1, s_2, \ldots$

$$\begin{align*}
\mathcal{F}\phi(p) &= \mathcal{F}\phi(s_0) + \mathcal{F}\phi(s_1) + \mathcal{F}\phi(s_2) + \ldots \\
\mathcal{G}\phi(p) &= \mathcal{G}\phi(s_0) \times \mathcal{G}\phi(s_1) \times \mathcal{G}\phi(s_2) \times \ldots \\
\mathcal{U}\phi_1 \mathcal{U}\phi_2(p) &= \sum \left\{ \begin{array}{c}
\mathcal{U}\phi_1 \mathcal{U}\phi_2(s_1) \\
\mathcal{U}\phi_1 \mathcal{U}\phi_2(s_1) \times \mathcal{U}\phi_2(s_2) \\
\mathcal{U}\phi_1 \mathcal{U}\phi_2(s_1) \times \mathcal{U}\phi_1(s_2) \times \mathcal{U}\phi_2(s_3) \\
\vdots \\
\mathcal{U}\phi_2(s_1) \\
\mathcal{U}\phi_1(s_1) + \mathcal{U}\phi_2(s_2) \\
\mathcal{U}\phi_1(s_1) + \mathcal{U}\phi_1(s_2) + \mathcal{U}\phi_2(s_3) \\
\vdots 
\end{array} \right\} \\
\mathcal{R}\phi_1 \mathcal{R}\phi_2(p) &= \prod \left\{ \begin{array}{c}
\mathcal{R}\phi_1 \mathcal{R}\phi_2(s_1) \\
\mathcal{R}\phi_1 \mathcal{R}\phi_2(s_1) + \mathcal{R}\phi_2(s_2) \\
\mathcal{R}\phi_1(s_1) + \mathcal{R}\phi_2(s_2) + \mathcal{R}\phi_2(s_3) \\
\vdots 
\end{array} \right\}
\end{align*}$$
Mutual Exclusion: Reasoning about Costs $(\mathbb{R}^+, \text{min}, +, +\infty, 0)$

- Min. cost for a process to use the c.s.?
  \[ [\sum \mathbf{F} \text{cost}] (s_0) = 0 \]
- Min. cost for a process to use the c.s. forever?
  \[ [\sum \mathbf{F} \sum \mathbf{G} \text{cost}] (s_0) = 0 \]
Mutual exclusion (all together)

- Optimal QoS to reach the c.s.?
  \[
  \left[ \sum F_{qos} \right](s_0) = \{ (\{p\}, 0, 0.5), (\{q\}, 1, 1) \}
  \]

- Optimal QoS to reach the c.s. and stay forever?
  \[
  \left[ \sum F \sum G_{qos} \right](s_0) = \{ (\{p\}, 0, 0), (\{q\}, \infty, 1) \}
  \]
C-CTL (Fixpoint Semantics over TS)

\[
\begin{align*}
[[\kappa F \phi]]^f & = [[\mu z. \phi + \kappa X z]] \\
[[\kappa G \phi]]^f & = [[\nu z. \phi \times \kappa X z]] \\
[[\kappa [\phi_1 U \phi_2]]]^f & = [[\mu z. \phi_2 + (\phi_1 \times \kappa X z)]] \\
[[\kappa [\phi_1 R \phi_2]]]^f & = [[\nu z. \phi_2 \times (\phi_1 + \kappa X z)]]
\end{align*}
\]
Path vs. Fixpoint Semantics

If \( \times \) is idempotent . . .

- If \( \times \) is idempotent \( \forall \phi \in \text{c-CTL}: [\phi] = [\phi]^f \).

If \( \times \) is not idempotent . . .

- \( \forall \phi \in \text{c-} \{\sum, \prod}\text{CTL}: [\phi] \leq_V [\phi]^f \).
- In practice: \( \forall \phi \in \text{c-CTL}: [\phi] \leq_V [\phi]^f \)
Verification Algorithms for $c$-CTL

If $\times$ is idempotent . . .

- $a \times a \times \ldots = a$ and $a+a+\ldots = a$
  - $\Rightarrow$ we can consider acyclic paths only.
- $O(|S|)$ fixpoint iterations are sufficient.

If $\times$ is not idempotent . . .

- Path and fixpoint semantics require different algorithms.
- Fixpoint iteration not feasible: might require $\omega$ iterations.
- Things are easier if $a \times a \times \ldots = 0$ (unless $a=1$)
Bisimulation

• \( sR s' \) whenever:
  
  \begin{itemize}
    \item \( v(s) = v(s') \) for all \( v \in AV \);
    \item \( s \to s_1 \) then \( s' \to s'_1 \mid s_1 R s'_1 \);
    \item \( s' \to s'_1 \) then \( s \to s_1 \mid s_1 R s'_1 \).
  \end{itemize}

• If \( \times \) is idempotent...

\[ \forall \phi \in c\text{-}CTL: sR s' \text{ then } \llbracket \phi \rrbracket(s) = \llbracket \phi \rrbracket(s') \]

• If \( \times \) is not idempotent...

\[ \forall \phi \in c\{-\{\sum, \Box\}\text{-}CTL: sR s' \text{ then } \llbracket \phi \rrbracket(s) = \llbracket \phi \rrbracket(s') \]
Bisimulation (Counterexample)

\[ s_0 R s'_0 \text{ but... } [\prod Xa](s_0) = a \times a \neq a = [\prod Xa](s'_0) \]
**Simulation**

- $sHs'$ whenever:
  - $u(s) \leq_S u(s')$ for all $u \in AV$;
  - $s \rightarrow s_1$ then $s' \rightarrow s'_1 \mid s_1 H s'_1$.

- If $\times$ is idempotent:
  \[
  \forall \phi \in \text{c-CTL}: sHs' \text{ then } \llbracket \phi \rrbracket(s) \leq_S \llbracket \phi \rrbracket(s')
  \]

- If $\times$ is not idempotent...
  \[
  \forall \phi \in \text{c-} \{\sum, \prod\} \text{CTL}: sRs' \text{ then } \llbracket \phi \rrbracket(s) \leq_S \llbracket \phi \rrbracket(s')
  \]
Captured Problems

- Graph problems:
  ⇒ Reachability, (multi-criteria) path optimization, etc.

- Boolean and quasi-boolean model checking:
  ⇒ Multi-valued CTL [Chechik et al.,03].

- Some probabilistic model checking approaches:
  ⇒ Fuzzy CTL over transition systems [de Alfaro et al.,04].

- Some discounted model checking problems (via $c-\mu$):
  ⇒ Discounted CTL [de Alfaro et al., 04].
Conclusions

• We have extended the \( \mu \)-calculus and CTL to c-semirings.
• The usual connection between both logics is not general.
• Model checking CTL when \( \times \) is idempotent can be done via fixpoint iteration.
• The framework captures some quantitative model checking problems.
• Future work: algorithms, WAN applications, extension to spatial logics, etc.