CTL* synthesis via LTL synthesis

Roderick Bloem¹, Sven Schewe², Ayrat Khalimov¹
in the next 30 minutes

• LTL/CTL* synthesis problem
• Why reduce CTL* synthesis to LTL synthesis?
  - unrealizable specifications
• Reduction
  - annotating trees with strategies
• Conclusion
LTL/CTL* synthesis problem by example

Specification:
• LTL formula: $G(r \rightarrow F g)$
• Inputs: $r$, outputs: $g$

Find a state machine with such inputs/outputs whose all executions satisfy the formula.

An example solution

Another solution
LTL/CTL* synthesis problem by example

Specification:
- CTL* formula: $\mathcal{A}G(r \rightarrow F g) \land \mathcal{A}GEF\neg g$
- Inputs: $r$, outputs: $g$

Find a state machine with such inputs/outputs whose all executions satisfy the formula.

An example solution

Another solution
why reduce CTL* synth. to LTL synthesis?

1. Handle unrealizable CTL* efficiently
2. Avoid building specialized CTL* synthesizers
   - re-use state-of-the-art LTL synthesizers
[Φ_{LTL}, I, O, type] is unrealizable ⇔
[¬Φ_{LTL}, O, I, ¬type] is realizable

Example:
• g ↔ Xr, I = \{r\}, O = \{g\} is unrealizable.

• \neg(g ↔ Xr), I = \{g\}, O = \{r\} is realizable:
output the negated first value of g.
unrealizable specifications: $\text{CTL}^*$

\[
[\Phi_{\text{CTL}^*}, I, O, \text{type}] \text{ is unrealizable } \iff \\
[\neg \Phi_{\text{CTL}^*}, O, I, \neg \text{type}] \text{ is realizable}
\]

Counterexample:

- $\text{AG}o, I = \{i\}, O = \{o\}$ is realizable:
  
  \text{always output } o.

- $\text{EF}\neg o, I = \{o\}, O = \{i\}$ is realizable:
steps in standard LTL/CTL* synthesis

LTL formula

CTL* formula

cannot negate CTL*

negation is cheap

nondet transitions --- formulas $E\varphi$

universal transitions --- formulas $A\varphi$

cannot negate

negation is EXPensive

alternating automaton

require system to resolve nondeterminism

universal automaton

check non-emptiness

(system or “unrealisable”)

(EXP)
our reduction

\[ \Phi_{CTL^*} \text{ is realizable } \iff \Phi_{LTL} \text{ is realizable} \]

the total blow-up is as before: \( \text{EXP} \)

system size can grow

\[ \text{CTL}^* \text{ formula} \]
\[ \approx \text{EXP} \]

\[ \text{LTL formula} \]
\[ \approx \text{EXP} \]

universal automaton

check non-emptiness

system or “unrealisable”

require system to resolve nondeterminism

\( \neg \) is cheap
automata for CTL*

- \( \text{EG EX}(g \land X(g \land F \neg g)) \)
- \( p_{EX} \equiv \text{EX}(g \land X(g \land F \neg g)) \)
- \( p_{EG} \equiv \text{EG} p_{EX} \)
\[ p_{EX} \equiv \text{EX}(g \land X(g \land F \neg g)) \]
\[ p_{EG} \equiv \text{EG}p_{EX} \]
Every state is additionally labeled with:

- **subformulas** → \{true, false\}
- **Q** → **Q** × **Direction**
blue and pink paths are equivalent: they merge into one

how many different paths can pass a node?

| Q |: the number of the nondet states!

annotated tree
core ideas of reduction

- “merging” paths are equivalent
  - max $|Q|$ non-equiv paths can pass through a node
- Assign a number $1 \ldots \max |Q|$ to each witness of $p_{EX}$
  - the whole witness is encoded by this number
  - require the witness to satisfy the LTL formula of $p_{EX}$
  - use the same number for equiv paths
newly annotated tree

Assign a number $1 \ldots \max |Q|$ to each witness of $p_{EX}$
- the whole witness is encoded by this number
- require the witness to satisfy the LTL formula of $p_{EX}$
- use the same number for equiv paths

$v_{EX} = 3$
$d_3 \mapsto r$
$d_2 \mapsto r$
$d_1 \mapsto r$
$d_4^* \mapsto r$

$v_{EX} = 2$
$d_2 \mapsto r$
$d_1 \mapsto r$
$d_4 \mapsto r$

$v_{EX} = 1$
$d_1 \mapsto r$
$d_4 \mapsto r$

$v_{EX} = 1, d_1 \mapsto r$
$v_{EQ} = 4, d_4 \mapsto r$
LTL formula

• For each subformula $E\varphi$:

$$\bigwedge_{i \in \{1 \ldots |Q|\}} G[ \nu_{E\varphi} = i \rightarrow (Gd_i \rightarrow \varphi')]$$ (1)

• For each subformula $A\varphi$:

$$G[ p_{A\varphi} \rightarrow \varphi']$$ (2)

• The LTL formula is

$$\bigwedge_{E\varphi} Eq.\,1 \land \bigwedge_{A\varphi} Eq.\,2$$
• For each subformula $E \varphi$:

$$\bigwedge_{i \in \{1 \ldots |Q|\}} G[\ l_{E \varphi} = i \rightarrow (G d_i \rightarrow \varphi')]$$

• For each subformula $A \varphi$:

$$G[\ p_{A \varphi} \rightarrow \varphi']$$

• The LTL formula is

$$\bigwedge_{E \varphi} Eq.\ 1 \land \bigwedge_{A \varphi} Eq.\ 2$$

• $\Phi_{LTL}$ is realizable $\iff$ $\Phi_{CTL^*}$ is realizable

• The complexity stays in 2EXP

• The system can get larger!
example: \( \text{EX}(g \land X(g \land X\neg g)) \)

\[
v \neq 0 \land \bigwedge_{i \in \{1, \ldots, 5\}} G[v = i \rightarrow (Gd_i \rightarrow X(g \land X(g \land X\neg g)))]
\]

A smallest system satisfying \( \Phi_{CTL^*} \):

A smallest system satisfying \( \Phi_{LTL} \):
We reduced CTL* synthesis to LTL synthesis without incurring a blow up.

Now we can use the reduction to handle unrealizable CTL* specifications and to re-use LTL synthesizers.