CSC410

AZADEH FARZAN

FALL 2020
Computational Tree Logic (CTL)
Limitations of LTL

Corollary 3.18. Trace Equivalence and LT Properties

Let $TS$ and $TS'$ be transition systems without terminal states and with the same set of atomic propositions. Then:

$$\text{Traces}(TS) = \text{Traces}(TS') \iff TS \text{ and } TS' \text{ satisfy the same LT properties.}$$

There thus does not exist an LT property that can distinguish between trace-equivalent transition systems. Stated differently, in order to establish that the transition systems $TS$ and $TS'$ are not trace-equivalent it suffices to find one LT property that holds for one but not for the other.

Example 3.19. Two Beverage Vending Machines

Consider the two transition systems in Figure 3.8 that both model a beverage vending machine. For simplicity, the observable action labels of transitions have been omitted. Both machines are able to offer soda and beer. The left transition system models a beverage machine that after insertion of a coin nondeterministically chooses to either provide soda or beer. The right one, however, has two selection buttons (one for each beverage), and after insertion of a coin, nondeterministically blocks one of the buttons. In either case, the user has no control over the beverage obtained—the choice of beverage is under full control of the vending machine.

Let $\mathcal{AP} = \{\text{pay}, \text{soda}, \text{beer}\}$. Although the two vending machines behaved differently, it is not difficult to see that they exhibit the same traces when considering $\mathcal{AP}$, as for both machines traces are alternating sequences of pay and either soda or beer. The vending machines are thus trace-equivalent. By Corollary 3.18 both vending machines satisfy exactly the same LT properties. Stated differently, it means that there does not exist an LT property that distinguishes between the two vending machines.

These two transition systems satisfy the same set of LTL formulas. But they function in different ways.

They are trace equivalent.
model checking based on linear or branching time logics: the linear-vs-branching-time debate and provide arguments that justify the treatment of algorithms do exist.

As lightly, branching temporal logic that is suitable for discrete notion of time, and only future modalities. CTL is an important introduction.

This chapter considers Computation Tree Logic (CTL) and existential path quantifiers. For instance, the aforementioned property “for every computation it is always possible to return to the initial state” can be faithfully expressed by

\[ \exists s : i n a n y t a s t e(\Phi) \]

The property \[ \Phi \] is always refuted. The property \[ \forall s : i n a n y t a s t e(\Phi) \] is eventually reached. This does not, however, exclude the fact that there can also be computations for which this property does not hold, for instance, computations for which \[ s = s_0 \] is eventually reached. This does not, however, exclude the fact that there can also be computations for which this property does not hold, for instance, computations for which \[ s = s_0 \] is eventually reached. This does not, however, exclude the fact that there can also be computations for which this property does not hold, for instance, computations for which \[ s = s_0 \] is eventually reached. This does not, however, exclude the fact that there can also be.

The expressiveness of many linear and branching temporal logics is incomparable. This means that some properties that are expressible in a linear temporal logic cannot be expressed in certain branching temporal logics, and vice versa.

This means that some properties that are expressible in a linear temporal logic cannot be expressed in certain branching temporal logics, and vice versa. This means that some properties that are expressible in a linear temporal logic cannot be expressed in certain branching temporal logics, and vice versa.

The expressiveness of many linear and branching temporal logics is incomparable. The expressiveness of many linear and branching temporal logics is incomparable. The expressiveness of many linear and branching temporal logics is incomparable.

The expressiveness of many linear and branching temporal logics is incomparable. The expressiveness of many linear and branching temporal logics is incomparable. The expressiveness of many linear and branching temporal logics is incomparable.

\[ \{ x = 1, x \neq 0 \} \]

\[ \{ x \neq 0 \} \]

\[ \{ x = 0 \} \]
### CTL Syntax

**State Formula**

\[ \Phi ::= \text{true} \mid a \mid \Phi_1 \land \Phi_2 \mid \neg \Phi \mid \exists \varphi \mid \forall \varphi \]

**Path Formula**

\[ \varphi ::= \mathbf{O} \Phi \mid \Phi_1 \cup \Phi_2 \]

**Examples:**

- \( \exists \mathbf{O} (x = 1) \)
- \( \forall \mathbf{O} (x = 1) \)
- \( \exists (x = 1 \land \forall \mathbf{O} (x \geq 3)) \)
- \( \forall (\text{true} \cup (x = 1)) \)

**But not:**

- \( \exists (x = 1 \land \forall \mathbf{O} (x \geq 3)) \)
- \( \forall (\text{true} \cup (x = 1)) \)

needs a path quantifier!

needs a temporal operator!
Eventually and Always

Eventually: \[ \exists \Diamond \Phi = \exists (\text{true} \cup \Phi) \]
\[ \forall \Diamond \Phi = \forall (\text{true} \cup \Phi) \]

Always: \[ \exists \Box \Phi = \neg \forall \Diamond \neg \Phi \]
\[ \forall \Box \Phi = \neg \exists \Diamond \neg \Phi \]
Meaning of CTL: Examples

Figure 6.2: Visualization of semantics of some basic CTL formulae.
### CTL Semantics

- **Atomic propositions**:
  \[ s \models a \iff a \in \mathcal{L}(s) \]

- **Negation**:
  \[ s \models \neg \Phi \iff \text{not } s \models \Phi \]

- **Conjunction**:
  \[ s \models \Phi \land \Psi \iff (s \models \Phi) \text{ and } (s \models \Psi) \]

- **Existential path formula**:
  \[ s \models \exists \pi \varphi \iff \exists \pi \in \text{Paths}(s) \ s.t. \pi \models \varphi \]

- **Universal path formula**:
  \[ s \models \forall \pi \varphi \iff \forall \pi \in \text{Paths}(s) \ s.t. \pi \models \varphi \]

**Same as LTL**

- **Future**:
  \[ \pi \models \circ \Phi \iff \pi[1] \models \Phi \]

- **Until**:
  \[ \pi \models \Phi \lor \Psi \iff \exists j \geq 0. \ (\pi[j] \models \Psi \land (\forall 0 \leq k < j. \pi[k] \models \Phi)) \]
CTL for LTSs

Figure 6.4: Interpretation of several CTL formulae.
End of CTL