Hoare logic and Model checking

Part II: Model checking

Lecture 8: Temporal logic

Jean Pichon-Pharabod
University of Cambridge

CST Part II – 2020/21

Recap

In the previous lecture, we saw how temporal models can be used to model various systems.

In this lecture, we will look at how temporal logic can be used to specify the behaviour of temporal models.

Why not use first-order logic?

Why not model time explicitly in first-order logic with equality and $<$, and have variables represent time points?

For example:

$$\forall t_1.\, p(t_1) \Rightarrow (\exists t_2.\, t_1 < t_2 \wedge q(t_2))$$

- ✓ It works.
- ✓ It has a well-understood theory.
- ✗ It is very error-prone.
- ✗ Is is very expensive to check.

Paths and states

Two intuitive things to consider: states, and paths.

- CTL* allows both,
- CTL (computation tree logic) focuses on states,
- LTL (linear temporal logic) focuses on paths.

When using model checking, one generally picks (a language based on) either LTL or CTL.
To describe model checking, CTL* makes things clearer.

We will first focus on the implication-free fragment.

Syntax of the implication-free fragment of CTL*

Given a fixed set of atomic propositions AP,

$\psi, \ldots \in$ StateProp ::=			$\phi, \ldots \in$ PathProp ::=		
$\bot$	$\mid$	false	$\phi_1 \wedge^{\mathsf{p}} \phi_2$	$\mid$	conjunction
$\top$	$\mid$	true	$\phi_1 \vee^{\mathsf{p}} \phi_2$	$\mid$	disjunction
$\psi_1 \wedge^{\mathsf{s}} \psi_2$	$\mid$	conjunction	$\mathsf{injs}\ \psi$	$\mid$	state property
$\psi_1 \vee^{\mathsf{s}} \psi_2$	$\mid$	disjunction	$\mathsf{X}\ \phi$	$\mid$	next
$\mathsf{injp}\ p$	$\mid$	atomic predicate	$\mathsf{F}\ \phi$	$\mid$	future
$\mathsf{A}\ \phi$	$\mid$	universal	$\mathsf{G}\ \phi$	$\mid$	generally
$\mathsf{E}\ \phi$		existential	$\phi_1\ \mathsf{U}\ \phi_2$		until

We almost always omit injp and injs.

Informal semantics of the implication-free fragment of CTL*

- injp p: the current state satisfies atomic proposition p
- A ϕ: all paths starting from the current state satisfy ϕ
- E ϕ: some path starting from the current state satisfies ϕ
- injs ψ: the first state of the current path satisfies ψ
- G ϕ: every suffix of the current path satisfies ϕ
- F ϕ: some suffix of the current path satisfies ϕ
- X ϕ: the tail of the current path satisfies ϕ
- ϕ_1 U ϕ_2: some suffix of the current path satisfies ϕ_2, and all the suffixes of the current path of which that path is a suffix satisfy ϕ_1

Example propositions in the implication-free fragment of CTL*

- E (F (injs (injp p))): there is a state reachable from the current state that satisfies atomic proposition p
- E (F (injs (injp p $\wedge^{s}$ injp q))): there is a state reachable from the current state that satisfies both atomic proposition p and atomic proposition q
- (E (F (injs (injp p)))) $\wedge^{s}$ (E (F (injs (injp q)))): there is a state reachable from the current state that satisfies atomic proposition p, and a reachable state that satisfies proposition q
- E ((F (injs (injp p))) $\wedge^{p}$ (F (injs (injp q)))): there is a path from the current state, along which there is a state satisfying atomic proposition p, and a state satisfying atomic proposition q
- E (X (injs (injp p))): there is a successor state satisfying atomic proposition p

Example propositions in the implication-free fragment of CTL*

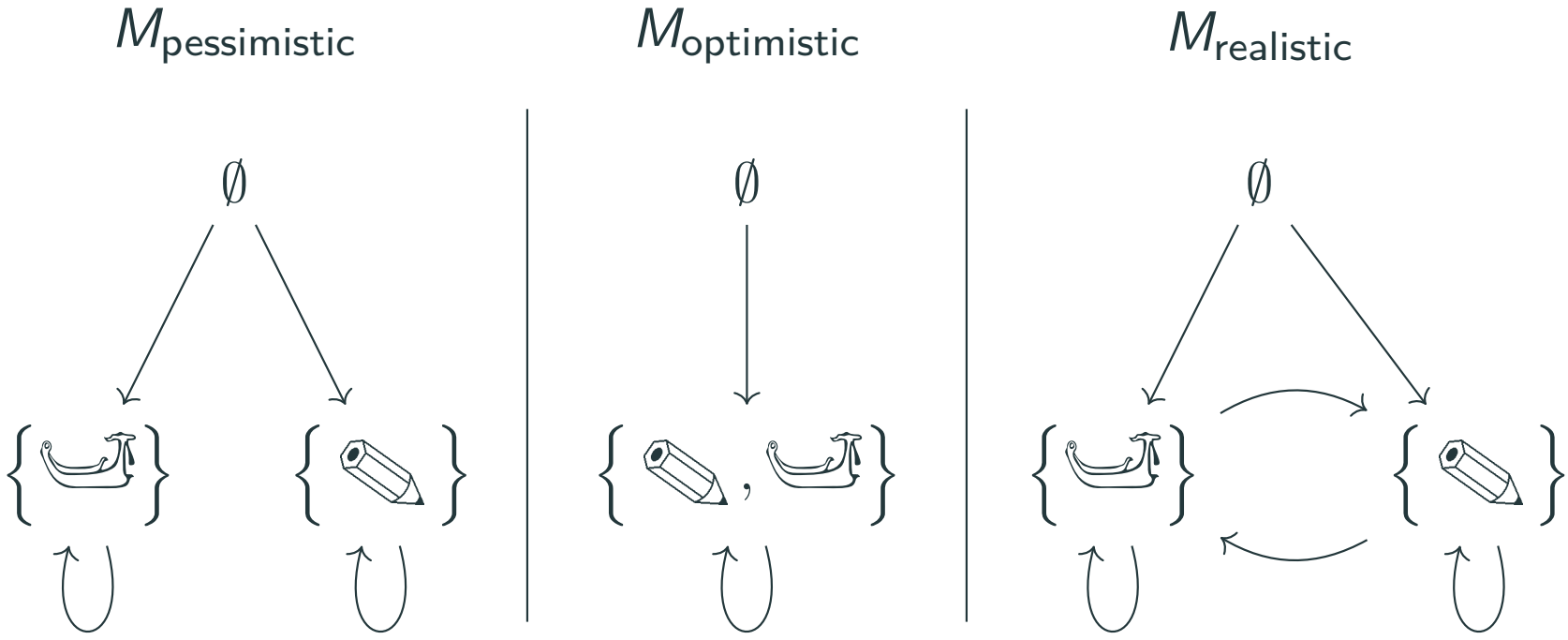
- A (G (injp p)): p always holds (in any path)
- E (G (injp p)): there is one path where p always holds
- A (G (A (F (injp idle)))): the tea & coffee machine always goes back to an idle state
- A (F (A (G (injp broken)))): the tea & coffee machine ends up permanently broken

Path conjunction vs. state conjunction

- E (F (injs ((injp p) $\wedge^{s}$ (injp q)))): there is a state that is reachable from the current state and that satisfies both p and q
- E ((F (injs (injp p))) $\wedge^{p}$ (F (injs (injp q)))): there is a state that is reachable from the current state and that satisfies p, and a state reachable that is reachable from the current state and that satisfies q

Example of path conjunction vs. state conjunction

"At Cambridge, you can row and study"



Semantics of the implication-free fragment of CTL*

We define whether M satisfies ψ,

$$① \vDash ② \quad \in \mathsf{TModel} \to \mathsf{StateProp} \to \mathsf{Prop}$$
$$M \vDash \psi \stackrel{def}{=} \forall s \in M.S.\ M.S_0\ s \to s \vDash_M \psi$$

using two auxiliary mutually inductive predicates

$$② \vDash^{\mathsf{s}}_{①} ③ \in (M \in \mathsf{TModel}) \to M.S \to \mathsf{StateProp} \to \mathsf{Prop}$$
$$② \vDash^{\mathsf{p}}_{①} ③ \in (M \in \mathsf{TModel}) \to \mathsf{stream}\ M.S \to \mathsf{PathProp} \to \mathsf{Prop}$$

We write the arguments that remain constant through recursive calls in this shade of grey blue.

Semantics of the implication-free fragment of CTL*: state properties

$$
\begin{aligned}
s \models^{\mathsf{s}}_M \top &\stackrel{def}{=} \top \\
s \models^{\mathsf{s}}_M \bot &\stackrel{def}{=} \bot \\
s \models^{\mathsf{s}}_M \psi_1 \wedge^{\mathsf{s}} \psi_2 &\stackrel{def}{=} \left(s \models^{\mathsf{s}}_M \psi_1\right) \wedge \left(s \models^{\mathsf{s}}_M \psi_2\right) \\
s \models^{\mathsf{s}}_M \psi_1 \vee^{\mathsf{s}} \psi_2 &\stackrel{def}{=} \left(s \models^{\mathsf{s}}_M \psi_1\right) \vee \left(s \models^{\mathsf{s}}_M \psi_2\right) \\
s \models^{\mathsf{s}}_M \mathsf{injp}\ p &\stackrel{def}{=} M.\ell\ s\ p \\
s \models^{\mathsf{s}}_M \mathsf{A}\ \phi &\stackrel{def}{=} \left(\begin{array}{l} \forall \pi \in \mathsf{stream}\ M.S. \\ \quad \mathsf{IsPath}\ M\ \pi \rightarrow \pi\ 0 = s \rightarrow \pi \models^{\mathsf{p}}_M \phi \end{array}\right) \\
s \models^{\mathsf{s}}_M \mathsf{E}\ \phi &\stackrel{def}{=} \left(\begin{array}{l} \exists \pi \in \mathsf{stream}\ M.S. \\ \quad \mathsf{IsPath}\ M\ \pi \wedge \pi\ 0 = s \wedge \\ \quad \pi \models^{\mathsf{p}}_M \phi \end{array}\right)
\end{aligned}
$$

Semantics of the implication-free fragment of CTL*: path properties

$$\pi \models^{\mathsf{p}}_M \mathsf{injs}\ \psi \quad \stackrel{def}{=} (\pi\ 0) \models^{\mathsf{s}}_M \psi$$

$$\pi \models^{\mathsf{p}}_M \phi_1 \wedge^{\mathsf{p}} \phi_2 \quad \stackrel{def}{=} \left(\pi \models^{\mathsf{p}}_M \phi_1\right) \wedge \left(\pi \models^{\mathsf{p}}_M \phi_2\right)$$

$$\pi \models^{\mathsf{p}}_M \phi_1 \vee^{\mathsf{p}} \phi_2 \quad \stackrel{def}{=} \left(\pi \models^{\mathsf{p}}_M \phi_1\right) \vee \left(\pi \models^{\mathsf{p}}_M \phi_2\right)$$

$$\pi \models^{\mathsf{p}}_M \mathsf{X}\ \phi \quad \stackrel{def}{=} (\mathsf{tailn}\ M.S\ 1\ \pi) \models^{\mathsf{p}}_M \phi$$

$$\pi \models^{\mathsf{p}}_M \mathsf{F}\ \phi \quad \stackrel{def}{=} \exists n \in \mathbb{N}.\, (\mathsf{tailn}\ M.S\ n\ \pi) \models^{\mathsf{p}}_M \phi$$

$$\pi \models^{\mathsf{p}}_M \mathsf{G}\ \phi \quad \stackrel{def}{=} \forall n \in \mathbb{N}.\, (\mathsf{tailn}\ M.S\ n\ \pi) \models^{\mathsf{p}}_M \phi$$

$$\pi \models^{\mathsf{p}}_M \phi_1\ \mathsf{U}\ \phi_2 \stackrel{def}{=}$$

$$\exists n \in \mathbb{N}. \begin{pmatrix} \left(\forall k \in \mathbb{N}.\, 0 \leq k < n \rightarrow (\mathsf{tailn}\ M.S\ k\ \pi) \models^{\mathsf{p}}_M \phi_1\right) \wedge \\ (\mathsf{tailn}\ M.S\ n\ \pi) \models^{\mathsf{p}}_M \phi_2 \end{pmatrix}$$

Moving goats

If we extend our atomic propositions to include fine-grained descriptions of the different items, we can write:

$$\mathit{Safe} \wedge^{\mathsf{s}} \mathit{Live}$$

$$\mathit{Safe} \stackrel{def}{=} \mathsf{A}\ (\mathsf{G}\ \mathit{StateSafe})$$

$$\mathit{Live} \stackrel{def}{=} \mathsf{A}\ (\mathsf{F}\ (\mathit{Done}))$$

$$\mathit{StateSafe} \stackrel{def}{=} \text{❀}\text{-}\mathit{Safe} \wedge^{\mathsf{s}} \text{🐐}\text{-}\mathit{Safe}$$

$$\text{❀}\text{-}\mathit{Safe} \stackrel{def}{=} \begin{array}{l} \left(\text{❀}\text{🐐}\text{⛵}\right) \vee^{\mathsf{s}} \left(\text{❀}\text{🐐}\text{⛵}\right) \vee^{\mathsf{s}} \\ \left(\text{❀}\text{🐐}\right) \vee^{\mathsf{s}} \left(\text{🐐}\text{❀}\right) \end{array}$$

$$\text{🐐}\text{-}\mathit{Safe} \stackrel{def}{=} \ldots$$

$$\mathit{Done} \stackrel{def}{=} \text{❀}\text{🐐}\text{⛵}$$

We can also express the fact that the puzzle has a solution with

$$\mathsf{E}\ ((\mathsf{G}\ \mathit{StateSafe}) \wedge^{\mathsf{p}} (\mathsf{F}\ \mathit{Done}))$$

Informal specification of indicating

Rule 103
Signals warn and inform other road users, including pedestrians ([...]), of your intended actions. You should always

- give clear signals in plenty of time, having checked it is not misleading to signal at that time
- use them to advise other road users before changing course or direction, stopping or moving off
- [...]

Formal specification of indicating

$$\mathsf{A}\ (\mathsf{G}\ (\blacksquare \vee (\blacksquare\ \mathsf{U}\ \text{  })))$$

$$\mathsf{A}\ \mathsf{G}\ (\mathit{SpecSN} \vee^{\mathsf{p}} \mathit{SpecSI} \vee^{\mathsf{p}} \mathit{SpecTI})$$

$$\mathit{SpecSN} = ((\blacksquare \wedge^{\mathsf{s}} \text{  }) \wedge^{\mathsf{p}} ((\blacksquare \wedge^{\mathsf{s}} \text{})\ \mathsf{U}\ (\blacksquare \wedge^{\mathsf{s}} \text{})))$$

$$\mathit{SpecSI} = ((\blacksquare \wedge^{\mathsf{s}} \text{}) \wedge^{\mathsf{p}} ((\blacksquare \wedge^{\mathsf{s}} \text{})\ \mathsf{U}\ (\text{  } \wedge^{\mathsf{s}} \text{})))$$

$$\mathit{SpecTI} = ((\text{} \wedge^{\mathsf{s}} \text{}) \wedge^{\mathsf{p}} ((\text{} \wedge^{\mathsf{s}} \text{})\ \mathsf{U}\ (\blacksquare \wedge^{\mathsf{s}} \text{})))$$

If we want to allow cancelling: make the RHS of until in *SpecSI* have $\ldots \vee^{\mathsf{s}} (\blacksquare \wedge^{\mathsf{s}} \text{})$.

If we want to allow driving straight forever: make the RHS of *SpecSN* have $\ldots \vee^{\mathsf{s}} \mathsf{G}\ (\blacksquare \wedge^{\mathsf{s}} \text{})$; similarly for turning forever.

Implication

Unstable assertions

It may be more natural to use the following:

$$\mathsf{A}\ (\mathsf{G}\ (\text{🛞} \rightarrow \text{🔧}))$$
$$\mathsf{A}\ (\mathsf{G}\ ((\mathsf{E}\ (\mathsf{X}\ \text{🛞})) \rightarrow \text{🔧}))$$

but implication is not stable under abstraction.

We can add implication (and thereby negation) to our temporal logic:

- ✓ more intuitive,
- ✗ more brittle: it conflates not being labelled with p with being labelled with $\neg p$, and thus does not respect abstraction.

Syntax of CTL* with implication

$$\mathsf{StateProp} \in \mathsf{Set}$$

$$\psi, \ldots \in \mathsf{StateProp} ::=$$

$$\bot \mid \top \mid \neg^{\mathsf{s}}\, \psi \mid \psi_1 \wedge^{\mathsf{s}} \psi_2 \mid \psi_1 \vee^{\mathsf{s}} \psi_2 \mid \psi_1 \to^{\mathsf{s}} \psi_2 \mid$$

$$\mathsf{injp}\ p \mid \mathsf{A}\ \phi \mid \mathsf{E}\ \phi$$

$$\mathsf{PathProp} \in \mathsf{Set}$$

$$\phi, \ldots \in \mathsf{PathProp} ::=$$

$$\neg^{\mathsf{p}}\, \phi \mid \phi_1 \wedge^{\mathsf{p}} \phi_2 \mid \phi_1 \vee^{\mathsf{p}} \phi_2 \mid \phi_1 \to^{\mathsf{p}} \phi_2 \mid$$

$$\mathsf{injs}\ \psi \mid \mathsf{X}\ \phi \mid \mathsf{F}\ \phi \mid \mathsf{G}\ \phi \mid \phi_1\ \mathsf{U}\ \phi_2$$

Splitting

Checking a full CTL* property can be reduced to checking an implication-free CTL* property.

To do so, we need to represent the fact that an atomic proposition can be negated.
We do this by having two versions of each atomic property p:
$\oplus p$ corresponds to p, and $\ominus p$ corresponds to $\neg p$:

$$
\begin{aligned}
&\text{Inductive split } (AP \in \text{Set}) \in \text{Set} := \\
&\quad \oplus\text{-} \in AP \to \text{split } AP \\
&\mid \ominus\text{-} \in AP \to \text{split } AP
\end{aligned}
$$

Fragments

Fragments

Fragments of CTL*: CTL, LTL, ACTL*, ECTL*, ...

Fragments: CTL

CTL

CTL restricts CTL* so that path quantifiers and temporal operators always come together:

$$
\begin{aligned}
&\psi, \ldots \in \mathsf{StateProp} \in \mathsf{Set} ::= \\
&\quad \bot \mid \top \mid \neg^{\mathsf{s}}\, \psi \mid \\
&\quad \psi_1 \wedge^{\mathsf{s}} \psi_2 \mid \psi_1 \vee^{\mathsf{s}} \psi_2 \mid \psi_1 \to^{\mathsf{s}} \psi_2 \mid \\
&\quad \mathsf{injp}\ p \mid \mathsf{A}\ \phi \mid \mathsf{E}\ \phi \\
&\phi, \ldots \in \mathsf{PathProp} \in \mathsf{Set} ::= \\
&\quad \mathsf{X}\ (\mathsf{injs}\ \psi) \mid \mathsf{F}\ (\mathsf{injs}\ \psi) \mid \mathsf{G}\ (\mathsf{injs}\ \psi) \mid (\mathsf{injs}\ \psi_1)\ \mathsf{U}\ (\mathsf{injs}\ \psi_2)
\end{aligned}
$$

Limits of CTL

$$\neg \left(\begin{array}{l} \forall p \in AP.\, \exists \psi^{\mathsf{CTL}} \in \mathsf{StateProp}^{\mathsf{CTL}}. \\ \quad \forall M \in \mathsf{TModel}.\, (M \models \mathsf{F}\ (\mathsf{G}\ p)) \leftrightarrow (M \models \psi^{\mathsf{CTL}}) \end{array} \right)$$

Fragments: LTL

LTL

LTL restricts CTL* so that properties are only (universally quantified) path properties:

$$\psi, \ldots \in \mathsf{StateProp} ::= \mathsf{A}\ \phi$$
$$\phi, \ldots \in \mathsf{PathProp} ::=$$
$$\neg^{\mathsf{p}}\ \phi \mid \phi_1 \wedge^{\mathsf{p}} \phi_2 \mid \phi_1 \vee^{\mathsf{p}} \phi_2 \mid \phi_1 \rightarrow^{\mathsf{p}} \phi_2 \mid$$
$$\text{injs-injp}^{\text{WI}}\ p \mid \mathsf{X}\ \phi \mid \mathsf{F}\ \phi \mid \mathsf{G}\ \phi \mid \phi_1\ \mathsf{U}\ \phi_2$$

The leading A is kept implicit.

Limits of LTL

LTL cannot express things like "whenever p holds, it is possible to reach a state where q holds":

$$\neg \left(\begin{array}{l} \forall AP \in \mathsf{Set}.\, \forall p, q \in AP.\, \exists \psi^{\mathsf{LTL}} \in \mathsf{StateProp}^{\mathsf{LTL}}\ AP. \\ \quad \forall M \in \mathsf{TModel}\ AP. \\ \qquad (M \vDash \mathsf{A}\ (\mathsf{G}\ (p \rightarrow \mathsf{E}\ (\mathsf{F}\ q)))) \leftrightarrow (M \vDash \psi^{\mathsf{LTL}}) \end{array} \right)$$

CTL vs. LTL: Milner's tea & coffee machines

These can be used to tell the difference between CTL (can distinguish) and LTL (cannot distinguish), because their difference is about their branching structure

Limits of CTL*

CTL* cannot express things like "p holds at even steps".

CTL* also has lots of moving parts. The linear μ-calculus (itself a special case of the modal μ-calculus) is more expressive than CTL*, and has far fewer moving parts, but is quite fiddly.

ACTL* and ECTL*

The universal fragment of CTL*, ACTL*, where all E are under odd numbers of negations, and all A are under even numbers of negations, is well-behaved with respect to abstraction (see lecture 11).

ACTL* contains LTL, and ACTL, the intersection of ACTL* and CTL.
ACTL* is dual (for the negation operation) to the existential fragment of CTL*, ECTL*.

Quantifiers

Unlike in Hoare logic, there are no quantifiers, as they would make it difficult to mechanically check properties.

To make up for this, we can use property schemas with big operators or bounded quantifiers, and indexed atomic propositions, which stand for the expanded property.

For example $\bigwedge_{i=0}^{n} p_i$, for $n = 3$, is expanded to $p_1 \wedge p_2 \wedge p_3$.

So is $\bigwedge_{i \in S} p_i$, for $S = \{1, 2, 3\}$.

This is is not as general as quantifiers, as the value of n or S has to be known. Because this is done as a preprocessing phase, and does not change the language of properties.

Summary

Temporal logics can be used to specify temporal models.

In the next lecture, we will look at how model checking is used in practice.