Modal and Temporal Logic

Vivian Dang

Modal Logic & Temporal Logic

- Modal Logic deals with different possible “worlds” or “modes” that a proposition *P* might be true. *P* might be false in one mode but true in the other mode.
  - HML
  - Uses LTS to reason with
- Temporal Logic deals with truth that can change over time.
  - E.g. *P* can be:
    - “sometimes” true that says P is true now if at some point in the future P is true
    - “always” true says that P is always true in the future
  - Computation Tree Logic - CTL
  - Linear-Time Temporal Logic - LTL
  - Uses Kripke Structure

Kripke Structure

- States are named e.g. *a, b, c...*
- States are also labelled with sets of atomic propositions *L(a), L(b), L(c)...*
- A **path** π is defined to be an infinite complete trace, or a finite complete trace ending in a deadlock state.

$$\pi : a \rightarrow b \rightarrow c \rightarrow \ldots$$

Basic Logic syntax

This is what we've already seen in HML:

$$\varphi ::= P \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg \varphi \mid \varphi \longrightarrow \varphi$$

CTL - Syntax

$$\varphi ::= P \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \varphi \longrightarrow \varphi$$
$$\mid EX\varphi \mid AX\varphi \mid EF\varphi \mid AF\varphi \mid EG\varphi \mid AG\varphi$$
$$\mid E(\varphi U \varphi) \mid A(\varphi U \varphi)$$

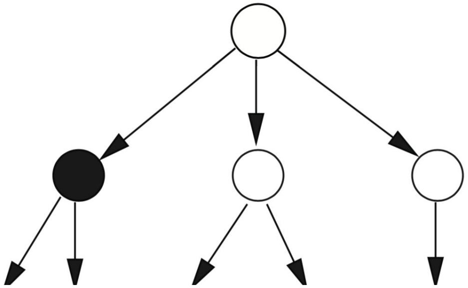
CTL - Semantics

Let’s define how a **state** (s) satisfies a CTL formula

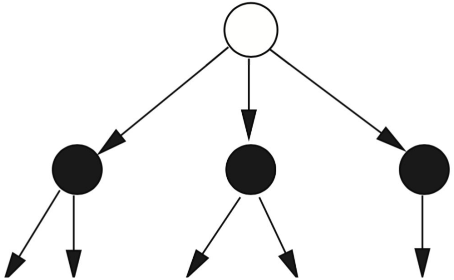
CTL - Next

$$s \vDash EX\varphi \iff \exists s'.\ (s \to s' \wedge s' \vDash \varphi)$$

$$s \vDash AX\varphi \iff \forall s'.\ (s \to s' \implies s' \vDash \varphi)$$



EX.black

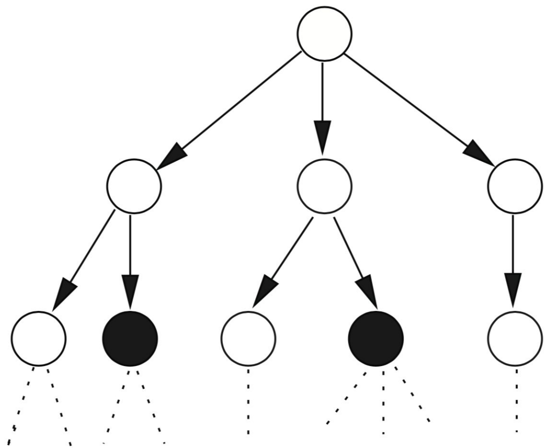


AX.black

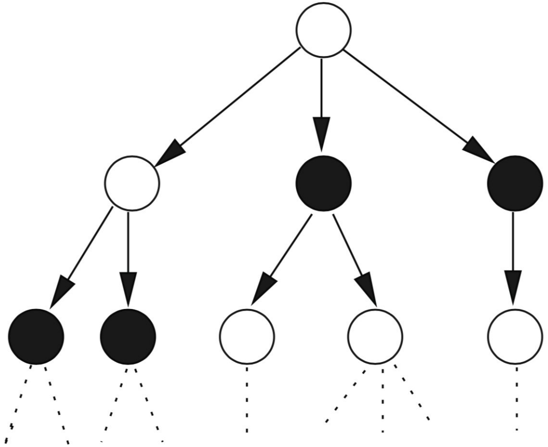
CTL - Finally (Eventually)

$$s \models EF\varphi \iff \exists s_1, s_2, \ldots, s_n.\ (s \to s_1 \to s_2 \to \ldots \to s_n \land s_n \models \varphi)$$

$$s \models AF\varphi \iff \forall s_1, s_2, \ldots, s_n.\ (s \to s_1 \to s_2 \to \ldots \to s_n \implies s_n \models \varphi)$$



EF.black

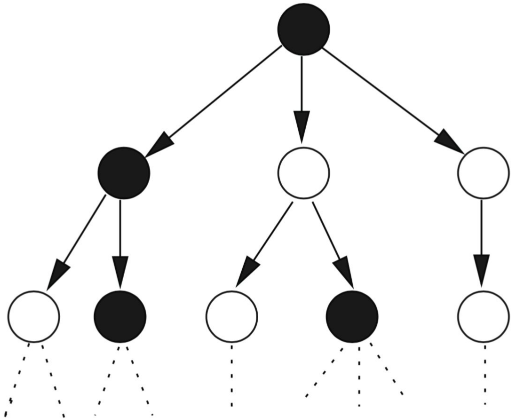


AF.black

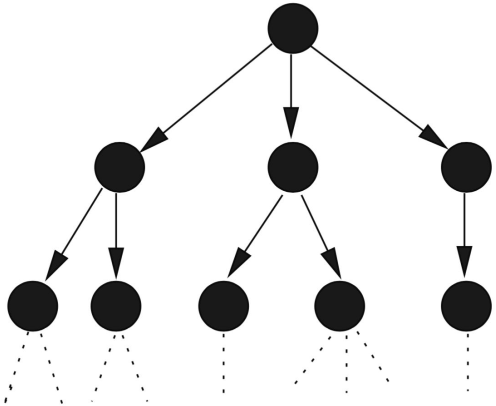
CTL - **G**lobally

$$s \models EG\varphi \iff \exists s_0, s_1, \ldots \ (s = s_0 \wedge s_0 \to s_1 \to \ldots \wedge \forall i.\ s_i \models \varphi)$$

$$s \models AG\varphi \iff \forall s_0, s_1, \ldots \ (s = s_0 \wedge s_0 \to s_1 \to \ldots \implies \forall i.\ s_i \models \varphi)$$



EG.black

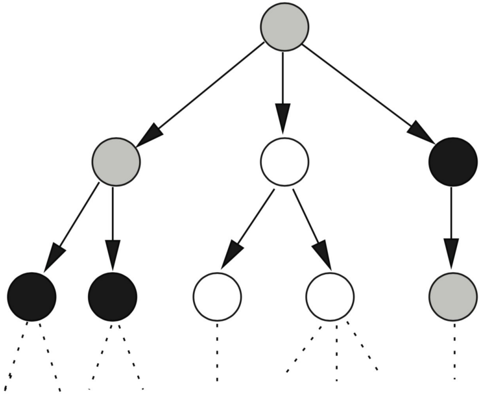


AG.black

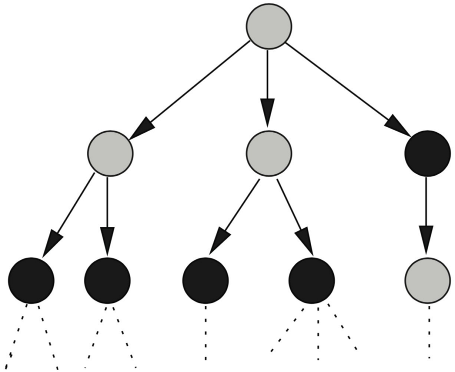
CTL - Until (Strong)

$$s \models E(\varphi U \psi) \iff \exists s_1, s_2, \ldots \left(s \to s_1 \to s_2 \to \ldots \wedge \left(\exists i.\ (\forall j < i.\ s_j \models \varphi) \wedge s_i \models \psi\right)\right)$$

$$s \models A(\varphi U \psi) \iff \forall s_1, s_2, \ldots \left(s \to s_1 \to s_2 \to \ldots \implies \left(\exists i.\ (\forall j < i.\ s_j \models \varphi) \wedge s_i \models \psi\right)\right)$$



E(grey U black)



A(grey U black)

CTL - Satisfiability

What it means to satisfy a CTL formula for a **transition system**

$$TS \models \varphi \iff \forall s_o \in I.\ s_o \models \varphi$$

CTL - examples

Let **P** be “I will read” and **Q** be “It’s raining outside”

- “I will read tomorrow, no matter what happens” → **AX.P**
- “I will read everyday from now on.” → **AG.P**
- “It’s *possible* I will *eventually* read someday, at least for one day” → **EF.P**
- “I will read everyday *until* it’s raining outside; once it’s raining, there’s no guarantee whether I will read or not” → **A.(PUQ)**

Recall: Kripke Structure

- States are named e.g. *a, b, c…*
- States are also labelled with sets of atomic propositions *L(a), L(b), L(c)…*
- A **path** is defined to be a series of states e.g.

$$\pi : a \to b \to c \to \dots$$

- A **trace** of a path *π* is defined to be the sequence of its atomic propositions:

$$\mathrm{trace}(\pi) = L(a)L(b)L(c)\dots$$

LTL - Syntax

$$\varphi ::= P \mid \varphi \wedge \varphi \mid \varphi \vee \varphi \mid \neg\varphi \mid \varphi \longrightarrow \varphi$$
$$\mid X\varphi \mid F\varphi \mid G\varphi \mid \varphi U \varphi$$

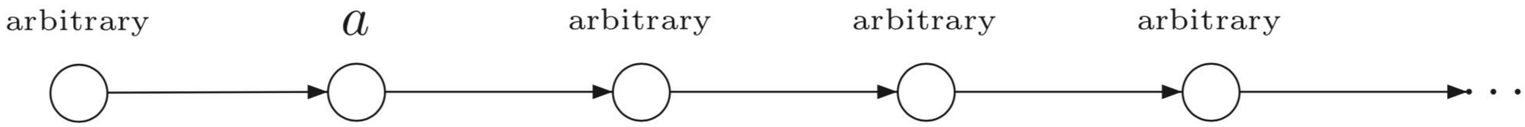
LTL - Semantics

Let's define how a **trace** (σ) satisfies an LTL formula

LTL - Next

$$\sigma = A_0 A_1 A_3 \ldots \models X\varphi \iff \sigma[1\ldots] = A_1 A_2 A_3 \ldots \models \varphi$$

Xa:



LTL - **F**inally (Eventually)

$$\sigma \models F\varphi \iff \exists j \geq 0.\ \sigma[j \ldots] \models \varphi$$

Fa:

LTL - Globally

$$\sigma \models G\varphi \iff \forall j \geq 0.\ \sigma[j\ldots] \models \varphi$$

Ga:

LTL - Until (Strong)

$$\sigma \models \varphi U \psi \iff \exists j \geq 0.\ \sigma[j \ldots] \models \psi \text{ and}$$
$$\sigma[i \ldots] \models \varphi, \text{ for all } 0 \leq i < j$$

a U b:

LTL - Satisfiability

What it means to satisfy an LTL formula for a **path**, **state**, and **transition system:**

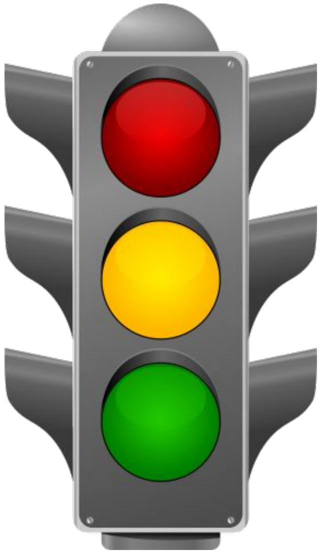
$$\pi \models \varphi \iff \mathrm{trace}(\pi) \models \varphi$$

$$s \models \varphi \iff \pi \models \varphi \text{ for all paths } \pi \text{ starting from } s$$

$$TS \models \varphi \iff \forall s_0 \in I.\ s_0 \models \varphi$$

LTL - example

Traffic lights with "**green**", "**amber**", "**red**" stage.



- **GF**.**green** $\wedge$ **GF**.**amber** $\wedge$ **GF**.**red**
  - green, amber, red infinitely often
- **G**.(**green** $\rightarrow \neg$ **X**.**red**)
  - once green, can't be red immediately
- **G**.(**green** $\rightarrow$ **X**.(**green** **U**.(**amber** **X**.(**amber** **U** **red**))))
  - once green, the light always becomes red eventually after being amber for some time

Thank you!

References

- Rob van Glabbeek - COMP6752 Lecture 10 video and notes
- Christel Baier, Joost-Pieter Katoen - Principles of model checking
- E.A. Emerson - Temporal and Modal Logic (Handbook of Theoretical Computer Science, Volume B: Formal Models and Semantics, pages 995–1072)
- https://en.wikipedia.org/wiki/Computation_tree_logic
- https://en.wikipedia.org/wiki/Linear_temporal_logic