Temporal Logic

Ralf Huuck

Outline

• Why not “standard” logic?
• What is temporal logic?
• LTL
• CTL*
• CTL
• Fairness

Model Checking Problem

? M ⪓ φ model, program satisfies, implements, refines property, specification

How to formalize the different components?

for this lecture we assume is M given as a Kripke structure, what about ⪓ and φ ?

Kripke Structure

Given a set of atomic propositions AP.

Kripke structure M=(S, s₀, →, μ) is defined by

• S set of states
• s₀ initial state
• → ⊆ S×S transition relation (total)
• μ:S → 2^AP labeling function

Any infinite run is accepting, i.e., like Buchi automaton where every state is a final state. Product as for NFA.
Example

\[ \begin{align*}
S_0 & \rightarrow \{a\} \\
S_1 & \rightarrow \{a,b\} \\
S_2 & \rightarrow \{b\}
\end{align*} \]

How to define properties formally?

- Kripke structure
- automata
- ω regular expression
- logics

Logic can provide succinct notation, “close” to natural language.

Propositional Logic

Syntax

\[ \phi ::= p \mid \neg \phi \mid \phi_1 \lor \phi_2 \]

Other connectivities (\&\& \iff \implies \ldots ) can be derived (see next slide)

Semantics

Given a state s in a Kripke structure M we define \( M,s \models \phi \) inductively by:

\[
\begin{align*}
M,s \models p & \iff p \in \mu(s) \\
M,s \models \neg \phi & \iff \text{not } M,s \models \phi \\
M,s \models \phi_1 \lor \phi_2 & \iff M,s \models \phi_1 \text{ or } M,s \models \phi_2
\end{align*}
\]

(\&\& \iff \implies \text{true,false} \ldots )

Propositional logic is good at describing “static” situations.
Propositional logic is good for describing "static" situations.

Example

\[ s_0 \vdash a \lor b \]
\[ M,s_1 \vdash a \]
\[ M,s_2 \vdash \neg a \]

Propositional logic is unsuitable for describing "dynamic" behavior.

How to describe:
- eventually \( b \) will happen?
- \( a \) will happen always again?

Dynamic Behavior

Important for reactive systems
- security protocols
- hardware
- operating systems
- embedded systems
- ...

Mutual Exclusion

process1
- ncs
- cs
- ncs

always only one process in cs
evantage process1 in cs
always eventually process2 in cs

process2
- ncs
- cs
- ncs

cs = critical section
ncs = none critical section
Temporal Logics

- originate from philosophy
- how to express statements including time?
- what is an appropriate model?
  - real-time vs discrete time
  - linear time vs branching time
    - (deterministic vs non-deterministic)
  - ...

(P)LTL

- Propositional Linear time Temporal Logic
- discrete time
- linear (deterministic) progression

LTL Syntax

- PLTL formula are inductively defined by:
  \[ \phi ::= p \mid \neg \phi \mid \phi_1 \lor \phi_2 \mid X \phi \mid F \phi \mid G \phi \mid \phi_1 U \phi_2 \]
  - p denotes atomic proposition
  - X denotes next-state operator
  - F denotes eventually/finally
  - G denotes always/globally
  - U denotes until

\[ \neg \text{on} \quad \text{on} \quad \neg \text{on} \quad \text{on} \quad \text{on} \quad \text{on} \quad \text{on} \quad \neg \text{on} \]
LTL semantics

- LTL formula $\varphi$ interpreted over infinite paths of states $\pi=s_0s_1s_2\ldots$
- we define LTL wrt Kripke structure $M$
- $M,\pi \models \varphi$ denotes $\varphi$ holds in a path $\pi$ of Kripke structure $M$.
- $M \models \varphi$ iff all paths of $M$ satisfy $\varphi$, i.e., for all $\pi$ in $M$ we have $M,\pi \models \varphi$

Paths in Kripke Structures

Paths in Kripke Structures

Semantics of LTL Operators

let $\pi^k$ denote suffix $s_k s_{k+1} s_{k+2} \ldots$ where $k \geq 0$

- $M,\pi \models p$ iff $s_0 \models p$, i.e., $p \in S(s_0)$
- $M,\pi \models \neg \varphi$ iff not $M,\pi \models \varphi$
- $M,\pi \models \varphi_1 \lor \varphi_2$ iff $M,\pi \models \varphi_1$ or $M,\pi \models \varphi_2$
- $M,\pi \models X\varphi$ iff $M,\pi^1 \models \varphi$, i.e., $s_1 s_2 s_3 \ldots$ satisfies $\varphi$
- $M,\pi \models F\varphi$ iff $\exists k \geq 0$ s.t. $M,\pi^k \models \varphi$
- $M,\pi \models G\varphi$ iff $\forall k \geq 0$ M, $\pi^k \models \varphi$
- $M,\pi \models \varphi_1 U \varphi_2$ iff $\exists k \geq 0$ s.t. $M,\pi^k \models \varphi_2$ and $\forall 0 \leq j < k$ we have $M,\pi^j \models \varphi_1$

Example

path $\pi$ satisfies:

- $M,\pi \models a$
- $M,\pi \models \neg b$
- $M,\pi \models Xb$
- $M,\pi \models Fa$
- $M,\pi \models FG(a \land b)$
- $M,\pi \models F(bUa)$
- $M,\pi \models FG(aUb)$
- what else?
Exercise

Which temporal operators can be expressed through one or more of the others?

Which cannot?

\[ M \models \phi \text{ iff } M, \pi \models \phi \text{ for all paths } \pi \]

Which properties satisfy this Kripke structure?

**CTL***

- Computational Tree Logic (star)
- discrete time
- branching (non-deterministic) progression
- describes properties of computation trees
- more powerful than LTL
Computation Trees

Unwinding Kripke Structure

Operators in CTL*

- temporal operators
  - same as LTL, describe properties of paths
- path quantifiers
  - A: for all paths (\(\forall\)) ...
  - E: there exists a path (\(\exists\)) ...
- more formally ...

CTL* Formulae

- propositional logic as underlying “static” logic
- two different types of formulae
  - state formula: properties of a state
  - path formulas: property of a path
- all LTL formulae are path formulae
- state formulae ≠ “static” propositional formulae
State Formulae

\[ \phi_s ::= p \mid \neg \phi_s \mid \phi_s \lor \phi_s \mid A \phi_s \mid E \phi_s \]

- \( \phi_s \) denotes state formula
- \( \phi_s \) path formula
- \( p \) atomic proposition
- \( A \phi_s \) and \( E \phi_s \) are state formulas
- set of all state formulae = set of all legal CTL* formulae

Path Quantifiers in State Formulae

- A and E are path quantifiers
- denote universal and existential quantification over paths starting in a certain state
- \( A \phi_s \) holds in a state \( s \)
  - iff for all paths starting in \( s \), \( \phi_s \) holds
- \( E \phi_s \) holds in a state \( s \)
  - iff there exists a path starting in \( s \), s.t. \( \phi_s \) holds

Path Formulae

\[ \phi_{\pi} ::= \phi_s \mid \neg \phi_{\pi} \mid \phi_{\pi_1} \lor \phi_{\pi_2} \mid X \phi_{\pi} \mid F \phi_{\pi} \mid G \phi_{\pi} \mid \phi_{\pi_1} U \phi_{\pi_2} \]

- every LTL formula is path formula
- all state formulae are also path formulae
- nesting: \( A(GF(A a \lor b)) \) (example tree?)

Semantics of CTL*

- define semantics w.r.t. Kripke structure \( M \)
  - \( M, s \models \phi \) denotes state formula \( \phi \) holds in a state \( s \) of \( M \)
  - \( M, \pi \models \phi \) denotes path formula \( \phi \) holds for path \( \pi \) in \( M \)
- \( \models \) defined inductively, as before
Semantics of CTL* State Formulae

- $M, s \models p$ iff $p \in \mu(s)$
- $M, s \models \neg \phi$ iff $M, s \not\models \phi$
- $M, s \models \phi_1 \land \phi_2$ iff $M, s \models \phi_1$ and $M, s \models \phi_2$
- $M, s \models A\phi$ iff for all paths $\pi$ starting in $s$, $M, \pi \models \phi$
- $M, s \models E\phi$ iff there exists a path $\pi$ starting in $s$, such that $M, \pi \models \phi$

Semantics of CTL* Path Formulae

- $M, \pi \models \phi_s$ iff $s_0 \models \phi_s$
- $M, \pi \models \neg \phi$ iff not $M, \pi \models \phi$
- $M, \pi \models \phi_1 \lor \phi_2$ iff $M, \pi \models \phi_1$ or $M, \pi \models \phi_2$
- $M, \pi \models X\phi$ iff $M, \pi^1 \models \phi$, i.e., $s_0 s_1 s_2 \ldots$ satisfies $\phi$
- $M, \pi \models F\phi$ iff $\exists k \leq 0$ s.t. $M, \pi^k \models \phi$
- $M, \pi \models G\phi$ iff $\forall k \leq 0$ s.t. $M, \pi^k \models \phi$
- $M, \pi \models \phi_1 U \phi_2$ iff $\exists k \leq 0$ s.t. $M, \pi^k \models \phi_2$
  
basically, same as before

LTL vs. CTL*

$\phi \in \text{LTL}$ implies $A\phi \in \text{CTL}^*$

$EF\phi \in \text{CTL}^*$ but not expressible in LTL (other examples?)

LTL strictly less expressive than CTL*

Examples

root state $s_0$ satisfies:

- $M, s_0 \models Aa$
- $M, s_0 \models A(\neg b)$
- $M, s_0 \models AXb$
- $M, s_0 \models EFa$
- $M, s_0 \models EFAa$
- what else?
M \models \phi (\phi \in \text{CTL}^*) \text{ iff } M,s_0 \models \phi \text{ for initial state } s_0

Which properties satisfies this Kripke structure?

**CTL**

a fragment of CTL*

**CTL Syntax**

\[ \phi ::= p \mid \neg \phi \mid \phi_1 \lor \phi_2 \mid AX\phi \mid EX\phi \mid AF\phi \mid EF\phi \mid AG\phi \mid EG\phi \mid A(\phi_1 U \phi_2) \mid E(\phi_1 U \phi_2) \]

- no arbitrary nesting
- path qualifiers and temporal operators alternate
- no Boolean combinations of path formulae

Not allowed: \( a \land Fq, E(A(F(a \lor b))) \), ..., what else?
Which CTL properties hold for this Kripke structure?

- LTL sublogic of CTL*
  - \( EF \varphi \in CTL^* \), not expressible in LTL
- CTL sublogic of CTL*
  - \( FG \varphi \in CTL^* \), not expressible in CTL
- LTL and CTL not comparable
  - \( FG \varphi \in LTL \), not expressible in CTL
  - \( EF \varphi \in CTL \), not expressible in LTL

Safety and Liveness

- safety: “never something bad will happen”
  \( AG \neg (in1 \land in2) \)
- liveness: “eventually something good will happen”
  \( EF \) safe

Rule of thumb: liveness properties iff counter example requires an infinite trace/infinite deep tree
Often liveness properties cannot be proven without certain assumptions, i.e., fairness.

**Weak/Strong Fairness**

- **Weak fairness**
  - If an event is continuously enabled, it will occur infinitely often
  - in LTL: $\text{GF (\neg \text{enabled} \lor \text{occurs})}$

- **Strong fairness**
  - If an event is infinitely often enabled it will occur infinitely often
  - in LTL: $\text{GF enabled} \Rightarrow \text{GF occurs}$

**Fairness and Model Checking**

- Weak/strong fairness can be expressed in LTL, however, not in CTL

- in **LTL model checking** fairness can be added directly as an assumption

- in **CTL model checking** fairness has to be build into the model checking algorithm
Summary

- temporal logic to specify behavior over time
- LTL: linear structure (for all paths)
- CTL(*): branching structure (selective paths)
- LTL, CTL sublogics of CTL*
- CTL, LTL not comparable
- different classes of properties (safety/liveness, fairness)

Next Lecture

- CTL model checking
- how does it work