# 303: Systems Verification

Lecture 3: LTL and CTL

Alessio Lomuscio

Thanks to Michael Huth and Mark Ryan
for some of the material in these notes.

Temporal logic provides a powerful language for reasoning about systems' behaviour. Temporal logics are a special family of modal logics.
Several temporal logics exist modelling different aspects of temporal evolution. Here we will focus on two: LTL and CTL.
They both assume time as a *sequence of discrete events*.
Models of time as *continuous sequence of events* or *intervals* exist - ask for pointers if interested.

# Linear Temporal Logic (LTL)

LTL assumes time is a linear sequence of determined discrete events. The modal box $\Box$ is written as $G$ representing "forever in the future". Its dual $\Diamond$ is represented by $F$ representing "at some point in the future".

- $G\phi$ represents situations in which $\phi$ is "**G**oing to be true forever in the future" (i.e., "$\phi$ is forever true from now on").
- $F\phi$ encodes situations in which "$\phi$ will be true at some **F**uture point", ie "at some future point $\phi$ becomes true".

# LTL - additional operators

In comparison to first order logic $G$ behaves like a $\forall$, while $F$ behaves like a $\exists$.
Two additional operators are often used in applications: "until" ($U$) and "next" ($X$).

- $X$ is a unary operator: $X\phi$ intuitively represents a situation where "$\phi$ holds at the ne**X**t time instant".
- $U$ is a binary operator, i.e., it is applied to two formulas. The formula $\phi U \psi$ intuitively represents a situation where "$\phi$ holds continuously in the future **U**ntil $\psi$ becomes true (at least once)".

Observe that $\psi$ has to become true at some point for the $U$ to hold (even if $\phi$ holds forever).

# LTL Syntax

## Definition

The syntax of LTL is given by the following BNF.

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid X\phi \mid G\phi \mid \phi U \phi$$

We assume that $F\phi ::= \neg G \neg \phi$.
We also assume that $\phi R \psi ::= \neg(\neg\phi U \neg\psi)$.
R stands for **R**elease and is the dual of U.

# LTL semantics

**Definition (LTL Models and Paths)**

A model for LTL is a Kripke model $M = (W, R, \pi)$ such that $R$ is a serial relation (i.e., a relation such that any state is related to at least another). A *path* in a model is an infinite sequence of states $w_0, w_1, ...$ such that $(w_i, w_{i+1}) \in R$, for any $i \geq 0$.

Note that any state may belong to more than one path, ie for any state there may be more than one successor (depending on which path we are considering). However, when considering a path, each state has a unique successor.

A path $\rho$ represents a possible evolution of the system. $\rho^i$ indicates the suffix of $\rho = s_0, s_1, \ldots$ starting at $s_i$. Note that since the suffix is infinite, $\rho^i$ is a path for any $i$.

# LTL Satisfaction (1/2)

We define satisfaction on states by first defining satisfaction on paths.

### Definition (LTL Satisfaction on paths)

Given a formula $\phi$, a model $M$ and a path $\rho = s0, s1, s2, \ldots$ on $M$, satisfaction for the LTL connectives is defined as follows.

$$\begin{array}{lll}
(M,\rho) \models p & \text{iff} & s_0 \in \pi(p) \\
(M,\rho) \models \neg\phi & \text{iff} & \text{not } (M,\rho) \models \phi \\
(M,\rho) \models \phi \wedge \psi & \text{iff} & (M,\rho) \models \phi \text{ and } (M,\rho) \models \psi \\
(M,\rho) \models X\phi & \text{iff} & (M,\rho^1) \models \phi. \\
(M,\rho) \models G\phi & \text{iff} & \text{for each } i \geq 0 \text{ we have } (M,\rho^i) \models \phi. \\
(M,\rho) \models \phi U \psi & \text{iff} & \text{there exists an } j \geq 0 \text{ such that} \\
 & & (M,\rho^j) \models \psi \text{ and} \\
 & & (M,\rho^k) \models \phi \text{ for all } 0 \leq k < j.
\end{array}$$

# LTL - Some observations

Note that the operator $U$ insists on $\psi$ becoming true at some point on the path. Also, if $\psi$ is not satisfied at the start of the path, for $\phi U \psi$ to be satisfied, $\phi$ needs to be satisfied at the start of the path.

Occasionally we may need to use a weaker version, or *weak until* which relaxes this constraint:

$$\phi W \psi ::= (\phi U \psi) \vee G\phi$$

The "release" connective $R$ is the dual of until $U$ ($\phi R \psi ::= \neg(\neg\phi U \neg\psi)$) as $F$ is the dual of $G$ ($F\phi ::= \neg G \neg \phi$).
$\phi R \psi$ represents a situation where $\phi$ **R**eleases $\psi$; in other words, $\psi$ needs to hold until $\phi$ becomes true.

As is customary in modal logic, the evaluation of a formula may depend on several states of a model; in LTL it may depend on infinitely many states.

# LTL Satisfaction (2/2)

Following the previous definition we can now give the key definition for satisfaction of LTL formulas.

## Definition (LTL Satisfaction on states)

Given a formula $\phi$, a model $M$ and a state $s$ in $M$, we say that $\phi$ is true at $s$ in $M$, written $(M, s) \models \phi$, if for all paths $\rho$ originating from $s$ we have that $(M, \rho) \models \phi$.

So in the case of LTL, the standard modal satisfaction definition on states involves quantification over *all possible futures*.

# Exercise 1

- Give the semantics of “at some point in the future” as above.
- Construct an LTL model that satisfies $(M, w) \models Gp \wedge (qUp)$.
- Show that there is no model that satisfies $(M, w) \models Gp \wedge F\neg p$.
- Give a model $M$ a path $\rho = s_0, s_1, ...$ in which $(M, \rho) \models Gp$ and $(M, s_1) \not\models Gp$.

# Exercise 2, pp 246 textbook

Consider the model
$\mathcal{M} =$
$(\{q_1, q_2, q_3, q_4\}, \{(q_1, q_2), (q_2, q_2), (q_3, q_1), (q_3, q_2), (q_3, q_4), (q_4, q_3)\},$
$\pi(a) = \{q_3, q_4)\}, \pi(b) = \{q_2, q_4)\})$.
For each of the formulas $\phi$:

1. $Ga$
2. $aUb$
3. $aUX(a \wedge \neg b)$
4. $X\neg b \wedge G(\neg a \vee \neg b)$
5. $X(a \wedge b) \wedge F(\neg a \wedge \neg b)$

(i) Find a path from the initial state $q_3$ which satisfies $\phi$.

(ii) Determine whether $\mathcal{M}, q_3 \models \phi$.

# Computation Tree Logic (CTL)

LTL allows us to talk about the temporal evolution of a system; satisfaction at a state involves quantification over *all* possible paths originating from that state.

It may happen though that we would like to check whether or not something happens in one path but not in all. This is important in applications, as we will see in later lectures.

CTL accommodates this need. The crucial difference from LTL is that CTL's syntax allows one to *quantify explicitly* over paths.

# Computation Tree Logic (CTL) Syntax

### Definition

The syntax of CTL is defined by the following BNF.

$$\phi ::= p \mid \neg\phi \mid \phi \wedge \phi \mid EX\phi \mid EG\phi | E(\phi U \phi)$$

- $EX\phi$ represents the fact that "there exists a (possible computation) path originating from the current state such that at the next state $\phi$ holds".
- $EG\phi$ represents the fact that "there exists a path from the current state such that $\phi$ holds forever in the future".
- $E(\phi U \psi)$ represents the fact that "there exists a path from the current state such that $\phi$ holds until $\psi$ becomes true".

# CTL - observations

Also in this case dual operators can be defined.
For example, $AX\phi ::= \neg EX \neg \phi$
$AX\phi$ represents the fact that "in all paths from the current state $\phi$ is true at the next state".

So in CTL we have two different path quantifiers: $E$, and $A$. $E$ encodes an existential quantification on paths ("there exists a path"), whereas $A$ encodes a universal one ("for all paths").
Similarly to $AX$, expressions containing the operators $AG, AF, AU$ can be expressed by rewriting expressions on $EX, EU, EG$.

# Exercise: Reading of other CTL operators

Write the intuitive meaning for the following CTL formulas in English, assuming the $A$ path quantifier is read as "For all possible paths".

- $AG\phi$,
- $AF\phi$,
- $A(\phi U \psi)$,
- $AX\phi \vee EG\neg\phi$,
- $AFEG\phi$.

# Semantics of CTL

### Definition (CTL satisfaction)

Given a formula $\phi$, a model $M = (W, R, \pi)$ and a state $s$ in $M$, the satisfaction of $\phi$ at $s$ in $M$, $(M, s) \models \phi$, is defined inductively as follows ($\neg, \wedge$ defined as usual):

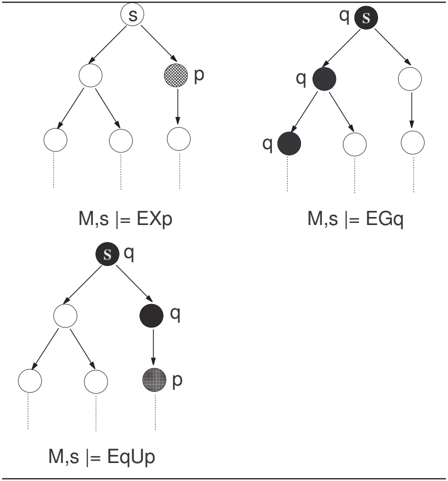
$(M, s) \models p$ iff $s \in \pi(p)$.

$(M, s) \models EX\phi$ iff there exists a path $s_0, s_1, s_2, \ldots$ from $s = s_0$ such that $(M, s_1) \models \phi$.

$(M, s) \models EG\phi$ iff there exists a path $s_0, s_1, s_2, \ldots$ from $s = s_0$ such that $(M, s_i) \models \phi$ for any $i \geq 0$.

$(M, s) \models E(\phi U \psi)$ iff there exists a path $s_0, s_1, s_2, \ldots$ from $s = s_0$ for which there exists an $i \geq 0$ such that $(M, s_i) \models \psi$ and $(M, s_j) \models \phi$ for all $0 \leq j < i$.

# CTL operators



M,s |= EXp

M,s |= EGq

M,s |= EqUp

# Satisfaction of CTL $A\phi$ formulas

Formally define the satisfaction clause for the CTL formulas below, assuming that the $A$ path quantifier denotes "For all possible paths".

- $AG\phi$
- $AF\phi$
- $A(\phi U \psi)$

# Exercise

- Construct a CTL model that satisfies $(M, w) \models A(Gp \wedge (qUp))$.
- Consider whether there is a CTL model that satisfies the formula $AG(p \wedge EF\neg p)$.
- Present the formal satisfaction definition for $AU, AF, AG$ similarly to the definition above, assuming that $A$ is the path quantifier corresponding to "for all paths".

# Summary on what studied so far

- Modal languages as formal unambiguous specification languages.
- Soundly based on heritage of mathematical techniques.
- Temporal (LTL/CTL) logic as powerful specification language.