Software Testing and Validation

A.A. 2025/2026

Corso di Laurea in Informatica

Logics in Model Checking

Igor Melatti

Università degli Studi dell'Aquila

Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

Beyond Invariants

- Invariants represent a huge share of properties to be verified on a system
- For many systems, one may be happy with invariants only
  - “nothing bad happens”, that’s all folks
- However, it is not always sufficient: a non-running system of course satisfies invariants
  - no starting states, thus no reachable states...

Safety vs. Liveness

- Safety properties: something bad must never happen
  - example: in the Peterson's protocol, it must not happen that both processes are accessing the resource (L3 in the Murphi model)
- Invariants are a special case of safety properties
  - there are some safety properties which are not invariants
  - however, they can be expressed with invariants by adding variables to the Kripke Structure
  - in the following, we will consider “invariants” and ”safety properties” as synonyms
- Liveness properties: something good will eventually happen
  - example: in the Peterson's protocol, both processes will eventually access the resource
  - not at the same time!
  - cannot be expressed with invariants

Safety vs. Liveness

- Notation: let $\mathcal{S}$ be a KS and φ be a formula in any logic
- $\mathcal{S} \models \varphi$ is true iff φ is true in $\mathcal{S}$
    - what this means depends on the logic, as we will see
- For most properties φ, if $\mathcal{S} \not\models \varphi$ then there exists a path $\pi \in \mathrm{Path}(\mathcal{S})$ which is a *counterexample*
    - by overloading the symbol $\models$, $\pi \not\models \varphi$
- For safety properties, $|\pi| < \infty$
    - $\mathcal{S}$ arrives to an *unsafe* state and that's it
- For liveness properties, $|\pi| = \infty$
    - since $\mathcal{S}$ is finite, this implies that π contains a loop (*lasso*) in its final part

Safety vs. Liveness

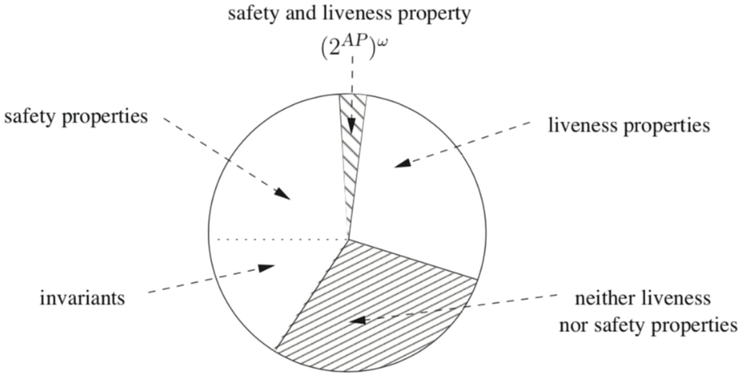
- Equivalent definition for a safety formula: given a finite counterexample, every extension still contains the error
- There is one formula which is both safety and liveness: the `true` invariant
    - it cannot have a counterexample...
- There are formulas which are neither safety nor liveness
    - their counterexample is not a path
- For typically used formulas, they are either safety or liveness properties

Safety vs. Liveness: Mathematical Definition

- Let a *model* σ be an infinite sequence of truth assignments to all $p \in AP$
  - $\sigma \in (2^{AP})^{\omega}$
  - could also be seen as a sequence of sets $P \subseteq AP$
  - given a path π of a KS $\mathcal{S}$, we can always obtain a model from π by replacing each $\pi(i)$ with $L(\pi(i))$
- It is possible to define if $\sigma \models \varphi$, for a given formula φ
- φ is a safety property if, for all σ s.t. $\sigma \not\models \varphi$, there exists j s.t. $\forall \sigma'.\sigma|_j = \sigma'|_j \rightarrow \sigma' \not\models \varphi$
  - i.e., given an (infinite) counterexample σ, there must exist a prefix p of σ s.t. all other models σ' having p as a prefix are again counterexamples
- φ is a liveness property if, for each prefix $w_0 \ldots w_i$, there exists σ s.t. $\sigma|_i = w_0 \ldots w_i$ and $\sigma \models \varphi$
  - i.e., a (finite) prefix of a model σ cannot be a counterexample, as you may always complete it in a “good” way

Safety vs. Liveness: Mathematical Definition

If we identify a property by the set of its models ($\varphi = \{\sigma \mid \sigma \models \varphi\}$)



Model Checking Logics: Preliminaries

- Model Checking logics are based on the concept of *execution* of a Kripke structure $\mathcal{S}$
  - thus, on $\pi \in \mathrm{Path}$
- Often, paths are directly viewed as a sequence of atomic propositions, rather than states
  - from $\pi = s_1, s_2, \ldots$ to $AP(\pi) = L(s_1), L(s_2), \ldots$
- Focusing on executions allows to model *time*
  - time in the sense that we have something coming before of something else (in a path...)
- Trade-off between
  - logics expressiveness: interesting properties can be written
  - logics efficiency: there is an efficient model checking algorithm to compute if $\mathcal{S} \models \varphi$

Model Checking Logics: Preliminaries

- We will focus on the two leading Model Checking logics: LTL and CTL
  - with some hints on CTL*
  - LTL (Linear-time Temporal Logic) established by Pnueli in 1977
  - CTL (Computation Tree Logic) established by Clarke and Emerson in 1981
  - used for IEEE standards:
    - PSL (Property Specification Language, IEEE Standard 1850)
    - SVA (SystemVerilog Assertions, IEEE Standard 1800).
- We will see syntax and semantics of both logics
  - syntax: how a valid formula is written
  - semantics: what a valid formula "means"
  - that is, when $\mathcal{S} \models \varphi$ holds

LTL Syntax

$$\Phi ::= p \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid (\Phi) \mid \mathbf{X}\Phi \mid \Phi_1 \ \mathbf{U} \ \Phi_2$$

- Other derived operators:
  - of course true, false, OR and other propositional logic connectors
  - future (or eventually): $\mathbf{F}\Phi = \mathrm{true}\ \mathbf{U}\ \Phi$
  - globally: $\mathbf{G}\Phi = \neg(\mathrm{true}\ \mathbf{U}\ \neg\Phi) = \neg\mathbf{F}\neg\Phi$
  - release: $\Phi_1\ \mathbf{R}\ \Phi_2 = \neg(\neg\Phi_1\ \mathbf{U}\ \neg\Phi_2)$
  - weak until: $\Phi_1\ \mathbf{W}\ \Phi_2 = (\Phi_1\ \mathbf{U}\ \Phi_2) \vee \mathbf{G}\Phi_1$
- Other notations:
  - next: $\mathbf{X}\Phi = \bigcirc\Phi$
  - $\mathbf{G}\Phi = \Box\Phi$
  - $\mathbf{F}\Phi = \Diamond\Phi$
- We are dropping *past operators*, thus this is *pure future LTL*

LTL Semantics

- Goal: formally defining when $\mathcal{S} \models \varphi$, being $\mathcal{S}$ a KS and φ an LTL formula
  - we say that $\mathcal{S}$ *satisfies* φ, or φ *holds in* $\mathcal{S}$
- This is true when, for all paths π of $\mathcal{S}$, π satisfies φ
  - i.e., $\forall \pi \in \mathrm{Path}(\mathcal{S}).\ \pi \models \varphi$
  - symbol $\models$ is overloaded...
- For a given π, $\pi \models \varphi$ iff $\pi, 0 \models \varphi$
- Finally, to define when $\pi, i \models \varphi$, a recursive definition over the recursive syntax of LTL is provided
  - $\pi \in \mathrm{Path}(\mathcal{S}), i \in \mathbb{N}$

LTL Semantics for $\pi, i \models \varphi$

- $\pi, i \models p$ iff $p \in L(\pi(i))$
- $\pi, i \models \Phi_1 \wedge \Phi_2$ iff $\pi, i \models \Phi_1 \wedge \pi, i \models \Phi_2$
- $\pi, i \models \neg\Phi$ iff $\pi, i \not\models \Phi$
- $\pi, i \models \mathbf{X}\Phi$ iff $\pi, i+1 \models \Phi$
- $\pi, i \models \Phi_1 \; \mathbf{U} \; \Phi_2$ iff $\exists k \geq i : \; \pi, k \models \Phi_2 \wedge \forall i \leq j < k. \; \pi, j \models \Phi_1$

LTL Semantics for Added Operators

- It is easy to prove that:
  - $\forall \pi \in \mathrm{Path}(\mathcal{S}), i \in \mathbb{N}.\ \pi, i \models \mathrm{true}$
  - $\pi, i \models \mathbf{G}\Phi$ iff $\forall j \geq i.\ \pi, j \models \Phi$
  - $\pi, i \models \mathbf{F}\Phi$ iff $\exists j \geq i.\ \pi, j \models \Phi$
  - $\pi, i \models \Phi_1\ \mathbf{R}\ \Phi_2$ iff $\forall k \geq i.\ \pi, k \models \Phi_2 \vee \exists i \leq j < k :\ \pi, j \models \Phi_1$
    - i.e., $\forall k \geq i.\ \pi, k \not\models \Phi_2 \to \exists i \leq j < k :\ \pi, j \models \Phi_1$
    - i.e., $\forall k \geq i.\ \forall i \leq j < k.\ \pi, j \not\models \Phi_1 \to \pi, k \models \Phi_2$
  - $\pi, i \models \Phi_1\ \mathbf{W}\ \Phi_2$ iff $(\forall j \geq i.\ \pi, j \models \Phi_1) \vee (\exists k \geq i :\ \pi, k \models \Phi_2 \wedge \forall i \leq j < k.\ \pi, j \models \Phi_1)$
- For many formulas, it is silently required that paths are infinite
- That's why transition relations in KSs must be total

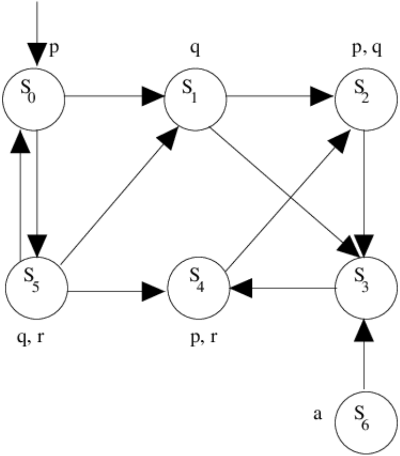
LTL Semantics: Typical Paths for Common Formulas

- For $p \in AP$, we will also consider p to be any set in $\{P \in 2^{AP} \mid p \in P\}$
  - that is, p is any subset of atomic propositions containing p
  - e.g., p may be any of $\{p\}, \{p, q\}, \{p, r, s\}$...
  - furthermore, $\bar{p} = \neg p \in \{P \in 2^{AP} \mid p \notin P\}$
    - e.g., $\bar{p}$ may be any of $\{q\}, \{q, r\}, \{r, s\}$...
  - finally, $\bot$ denotes any subset of atomic propositions
- If $\pi \models \mathbf{G}p$, then $\pi = p^\omega$
  - of course, this includes, e.g., $\pi = \{p, q\}\{p, r\}\{p\}\{p, q\}\{p\} \ldots$
  - $\pi, 3 \models \mathbf{G}p$: $\pi = \bot\bot\bot\ p^\omega$
- If $\pi \models \mathbf{F}p$, then $\pi = \bot^*\ p\ \bot^\omega$
- If $\pi \models p\ \mathbf{U}\ q$, then $\pi = \{p, \bar{q}\}^* q\ \bot^\omega$
- If $\pi \models p\ \mathbf{W}\ q$, then either $\pi = \{p, \bar{q}\}^* q\ \bot^\omega$ or $\pi = p^\omega$
- If $\pi \models p\ \mathbf{R}\ q$, then either $\pi = \{\bar{p}, q\}^\omega$ or $\pi = \{\bar{p}, q\}^* \{p, q\}\ \bot^\omega$
  - q must be kept holding till when a p appears and "releases" q...

Safety and Liveness Properties in LTL

- Given an LTL formula φ, φ is a safety formula iff $\forall \mathcal{S}.\ (\exists \pi \in \mathrm{Path}(\mathcal{S}) :\ \pi \not\models \varphi) \to \exists k :\ \pi|_k \not\models \varphi$
- Given an LTL formula φ, φ is a liveness formula iff $\forall \mathcal{S}.\ (\exists \pi \in \mathrm{Path}(\mathcal{S}) :\ \pi \not\models \varphi) \to |\pi| = \infty$
- All LTL formulas are either safety, liveness, or the AND of a safety and a liveness
  - being defined on paths, the counterexample is always a path
- Safety properties are those involving only **G**, **X**, true and atomic propositions
- Liveness are all those involving an **F** or a **U**
  - but beware of negations...
- Some formulas are both safety and liveness, like true, **G** true and so on

LTL Examples



$\mathcal{S} \models \mathbf{F}p$ since p holds in the first state

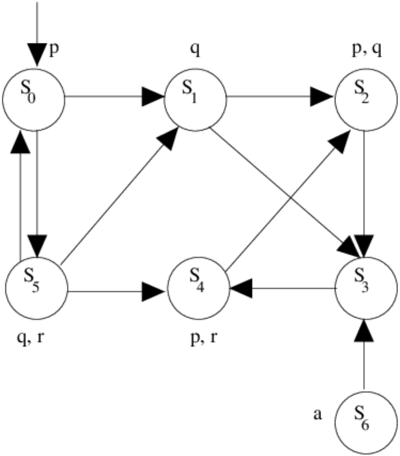
For full: let $\pi \in \mathrm{Path}(\mathcal{S})$

$\pi, 0 \models \mathbf{F}p$ with $j = 0$

recall: $\pi, i \models \mathbf{F}\Phi$ iff $\exists j \geq i.\ \pi, j \models \Phi$

$\pi, i \models p$ iff $p \in L(\pi(i))$

LTL Examples



$\mathcal{S} \not\models \mathbf{F}a$ since s_6 is not reachable from s_0

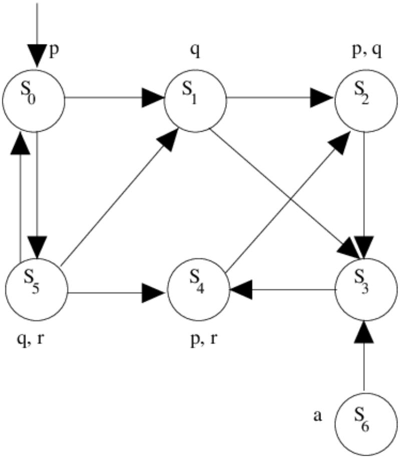
counterexample: $\pi = s_0 s_5 s_0 s_5 \ldots$

For full: $\pi, 0 \not\models \mathbf{F}a$ as, for all $j \geq 0$, $a \notin L(\pi(j))$

Counterexample is infinite, thus this is a liveness property

Any finite prefix of π is not a counterexample

LTL Examples



$\mathcal{S} \not\models \mathbf{G}p$ since there are many counterexamples, here is one: $\pi = s_0 s_5 s_0 s_5 \ldots$

For full: $\pi, 0 \not\models \mathbf{G}p$ with $j = 1$

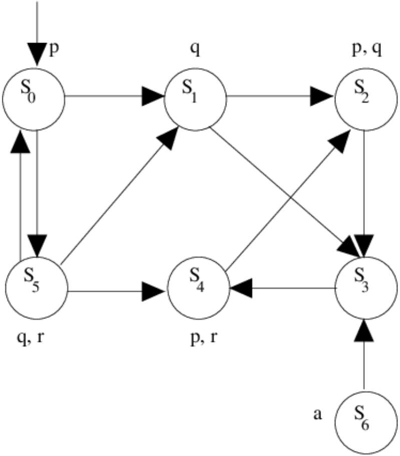
recall: $\pi, i \models \mathbf{G}\Phi$ iff $\forall j \geq i.\ \pi, j \models \Phi$

$\pi, i \models p$ iff $p \in L(\pi(i))$

Safety property, actually $\pi|_2$ is enough

Every path having $\pi|_2$ as a prefix is a counterexample

LTL Examples

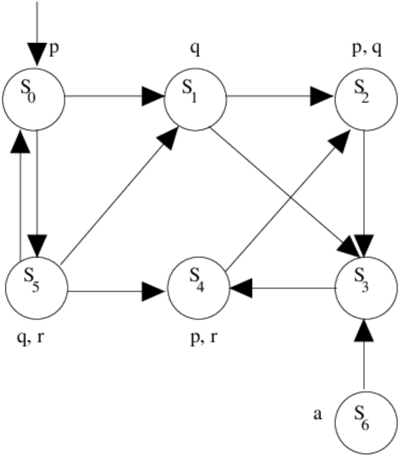


$\mathcal{S} \models \mathbf{G}\neg a$ since s_6 is not reachable from s_0

For full: let $\pi \in \mathrm{Path}(\mathcal{S})$ $\pi, 0 \models \mathbf{G}\neg a$ as the only state s with $a \in L(s)$ is s_6, which is not reachable from s_0

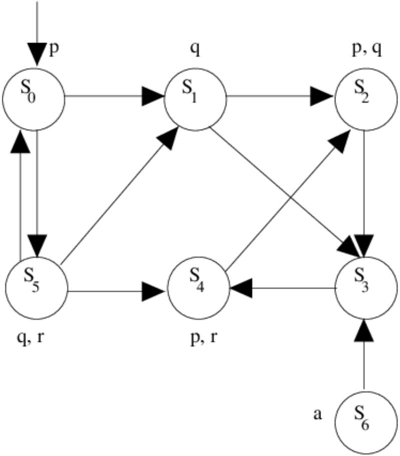
recall: $\pi \in \mathrm{Path}(\mathcal{S})$ implies $\pi(0) \in I$, thus $\pi(0) = s_0$ here

LTL Examples



$\mathcal{S} \models p \ \mathbf{U} \ q$ since $p \in L(s_0)$, $\mathrm{next}(s_0) = \{s_1, s_5\}$ and $q \in L(s_1) \wedge q \in L(s_5)$

LTL Examples

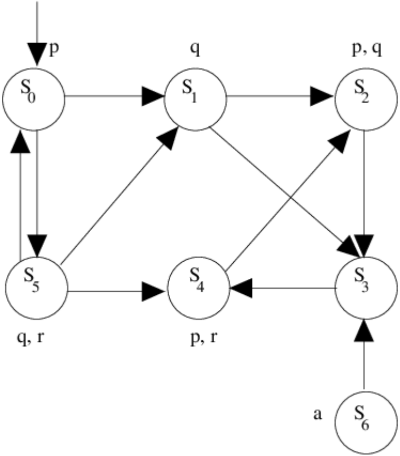


$\mathcal{S} \not\models p \ \mathbf{U} \ r$, a counterexample is $\pi = s_0 s_1 (s_2 s_3 s_4)$

Again this is a liveness formula, even if $\pi|_1$ would have been enough

In fact, you have to rule out $\{p, \bar{r}\}^\omega$...

LTL Examples



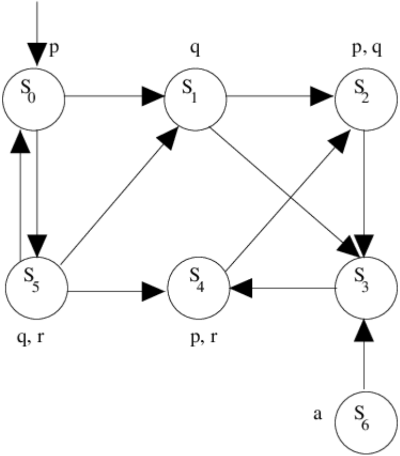
$\mathcal{S} \not\models \neg(p \mathbf{\ U\ } r)$, a counterexample is $\pi = (s_0 s_5)$

In fact, $(s_0 s_5), 0 \models p \mathbf{\ U\ } r$

Thus it may happen that $\mathcal{S} \not\models \Phi$ and $\mathcal{S} \not\models \neg(\Phi)$

Instead, it is impossible that $\mathcal{S} \models \Phi$ and $\mathcal{S} \models \neg(\Phi)$

LTL Examples



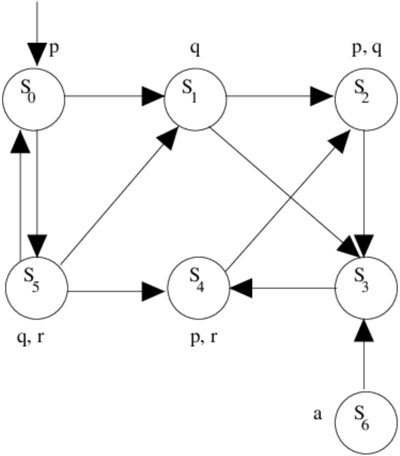
$\mathcal{S} \not\models q$, since s_0 is the only initial state and $q \notin L(s_0)$ (all paths in $\mathrm{Path}(\mathcal{S})$ must start from s_0)

$\mathcal{S} \models p$, since $p \in L(s_0)$

$\mathcal{S} \models \mathbf{X}q$, since $q \in L(s_1) \wedge q \in L(s_5)$

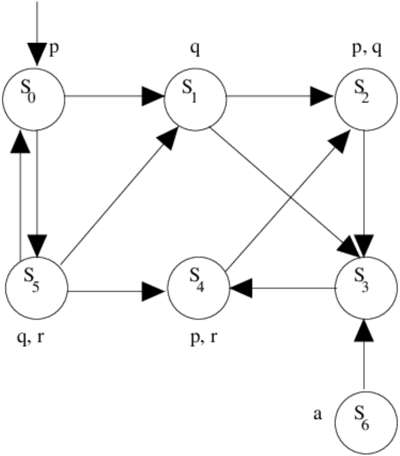
$\mathcal{S} \not\models \mathbf{XX}q$, since all states but s_5, s_6 are reachable in exactly 2 steps

LTL Examples



$\mathcal{S} \not\models \mathbf{FG}p$, a counterexample is $\pi = s_0 s_1 (s_2 s_3 s_4)$
Again this is a liveness formula

LTL Examples

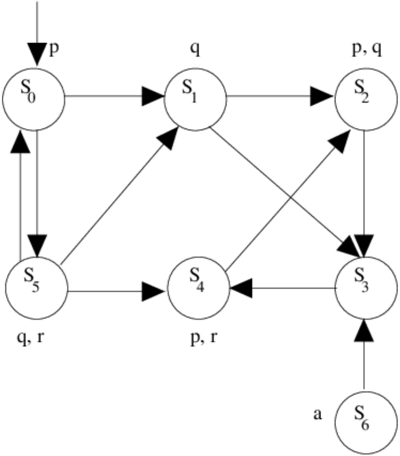


$\mathcal{S} \models \mathbf{GF}p$

All lassos are $s_0 s_5$ or $s_2 s_3 s_4$

In both such lassos, there are states in which p holds

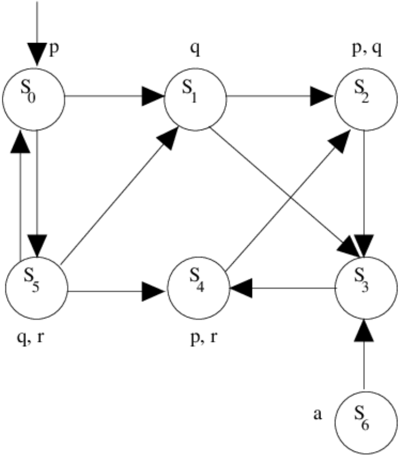
LTL Examples



$\mathcal{S} \models \mathbf{GF}p \vee \mathbf{FG}p$

Consequence of the two previous slides

LTL Examples



$\mathcal{S} \not\models \mathbf{G}(p \ \mathbf{U} \ q)$, a counterexample is $\pi = s_0 s_1 (s_2 s_3 s_4)$

$(p \ \mathbf{U} \ q)$ must hold at any reachable state

Ok in s_0, s_1, s_2, but not in s_3

LTL Non-Toy Examples

- Recall the Peterson's protocol: checking mutual exclusion is $\mathbf{G}(\neg(p \wedge q))$, being $p = \mathtt{P[1]} = \mathtt{L3}, q = \mathtt{P[2]} = \mathtt{L3}$
  - all invariants are of the form $\mathbf{G}P$, where P does not contain modal operators **X**, **U** or **F**
- Checking that both processes access to the critical section *infinitely often* is $\mathbf{GF}\ P[1] = L3 \wedge \mathbf{GF}\ P[2] = L3$
  - liveness property: no process is infinitely banned to access the critical section
- Even better: $\mathbf{G}\ (P[1] = L2 \to \mathbf{F}\ P[1] = L3)$
  - the same for the other process
  - since it is simmetric, this is actually enough

Equivalence Between LTL Properties

- Definition of equivalence between LTL properties:
  $\varphi_1 \equiv \varphi_2 \quad \text{iff} \quad \forall \mathcal{S}.\, \mathcal{S} \models \varphi_1 \Leftrightarrow \mathcal{S} \models \varphi_2$
  - equivalent: $\forall \sigma \ldots$
- Idempotency:
  - $\mathbf{FF}p \equiv \mathbf{F}p$
  - $\mathbf{GG}p \equiv \mathbf{G}p$
  - $p\ \mathbf{U}\ (p\ \mathbf{U}\ q) \equiv (p\ \mathbf{U}\ q)\ \mathbf{U}\ q \equiv p\ \mathbf{U}\ q$
- Absorption:
  - $\mathbf{GFG}p \equiv \mathbf{FG}p$
  - $\mathbf{FGF}p \equiv \mathbf{GF}p$
- Expansion (used by LTL Model Checking algorithms!):
  - $p\ \mathbf{U}\ q \equiv q \vee (p \wedge \mathbf{X}(p\ \mathbf{U}\ q))$
  - $\mathbf{F}p \equiv p \vee \mathbf{XF}p$
  - $\mathbf{G}p \equiv p \wedge \mathbf{XG}p$

CTL Syntax

$$\Phi ::= p \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid (\Phi) \mid \mathbf{EX}\Phi \mid \mathbf{EG}\Phi \mid \mathbf{E}\Phi_1 \ \mathbf{U} \ \Phi_2$$

- Other derived operators (besides true, false, OR, etc):
  - $\mathbf{EF}\Phi = \mathbf{E}\mathrm{true} \ \mathbf{U} \ \Phi$
    - cannot be defined using $\mathbf{E}\neg\mathbf{G}\neg\Phi$, as this is not a CTL formula
    - actually, it is a CTL* formula (see later)
    - in fact, you cannot place a negation between **E** and the subformula
  - $\mathbf{AF}\Phi = \neg\mathbf{EG}\neg\Phi$, $\mathbf{AG}\Phi = \neg\mathbf{EF}\neg\Phi$, $\mathbf{AX}\Phi = \neg\mathbf{EX}\neg\Phi$
  - $\mathbf{A}\Phi_1 \ \mathbf{U} \ \Phi_2 = (\neg\mathbf{E}\neg\Phi_2 \ \mathbf{U} \ (\neg\Phi_1 \wedge \neg\Phi_1)) \wedge \neg\mathbf{EG}\neg\Phi_2$
  - $\Phi_1\mathbf{AU}\Phi_2 = \mathbf{A}\Phi_1\mathbf{U}\Phi_2$, $\Phi_1\mathbf{EU}\Phi_2 = \mathbf{E}\Phi_1\mathbf{U}\Phi_2$

Comparison with LTL Syntax

$$\Phi ::= \mathrm{true} \mid p \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid (\Phi) \mid \mathbf{X}\Phi \mid \Phi_1 \,\mathbf{U}\, \Phi_2$$

- Essentially, all temporal operators are preceded by either **E** or **A**
  - with some care for **U**

CTL Semantics

- Goal: formally defining when $\mathcal{S} \models \varphi$, being $\mathcal{S}$ a KS and φ a CTL formula
- This is true when, for all initial states $s \in I$ of $\mathcal{S}$, $s \models \varphi$
  - thus, CTL is made of *state* formulas
  - LTL has *path* formulas
- To define when $s \models \varphi$, a recursive definition over the recursive syntax of CTL is provided
  - no need of an additional integer as for LTL syntax

CTL Semantics for $s \models \varphi$

- $\forall s \in S.\ s \models \mathrm{true}$
- $s \models p$ iff $p \in L(s)$
- $s \models \Phi_1 \wedge \Phi_2$ iff $s \models \Phi_1 \wedge s \models \Phi_2$
- $s \models \neg\Phi$ iff $s \not\models \Phi$
- $s \models \mathbf{EX}\Phi$ iff $\exists \pi \in \mathrm{Path}(\mathcal{S}, s).\ \pi(1) \models \Phi$
- $s \models \mathbf{EG}\Phi$ iff $\exists \pi \in \mathrm{Path}(\mathcal{S}, s).\ \forall j.\ \pi(j) \models \Phi$
- $s \models \mathbf{E}\Phi_1\ \mathbf{U}\ \Phi_2$ iff
  $\exists \pi \in \mathrm{Path}(\mathcal{S}, s) \exists k:\ \pi(k) \models \Phi_2 \wedge \forall j < k.\ \pi(j) \models \Phi_1$

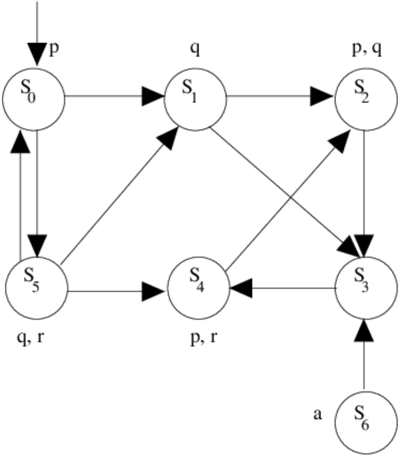
CTL Semantics for Added Operators

- It is easy to prove that:
  - $s \models \mathbf{AG}\Phi$ iff $\forall \pi \in \mathrm{Path}(\mathcal{S}, s).\ \forall j.\ \pi(j) \models \Phi$
  - $s \models \mathbf{AF}\Phi$ iff $\forall \pi \in \mathrm{Path}(\mathcal{S}, s).\ \exists j.\ \pi(j) \models \Phi$
  - analogously for **AU**, **AR**, **AW**
  - just replace $\forall$ with $\exists$ for **EF**, **ER**, **EW**
- Analogously to LTL, for many CTL formulas it is silently required that paths are infinite
- So again transition relations in KSs must be total

Safety and Liveness Properties in CTL

- Some CTL formulas may be neither safety nor liveness
  - being defined on states, the counterexample may be an entire computation tree
- Safety properties are those involving only **AG**, **AX**, true and atomic propositions
- Some formulas are both safety and liveness, like true, **AG** true and so on
- Liveness are formulas like **AF**, **AFAG**, **AU**
- **EF** or **EG** are neither liveness nor safety

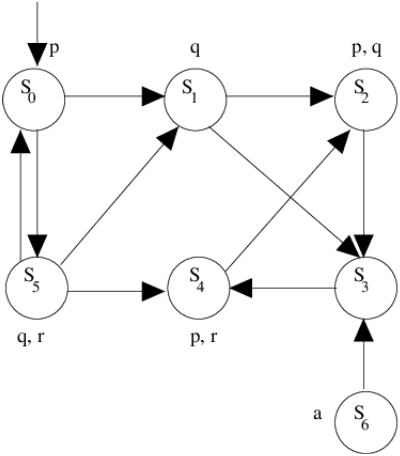
CTL Examples



$\mathcal{S} \models \mathbf{AF}p$ since p holds in the first state

For full: $s_0 \models \mathbf{F}p$ since $p \in L(s_0)$, thus, for all paths starting in s_0, p holds in the first state, so it holds eventually

CTL Examples



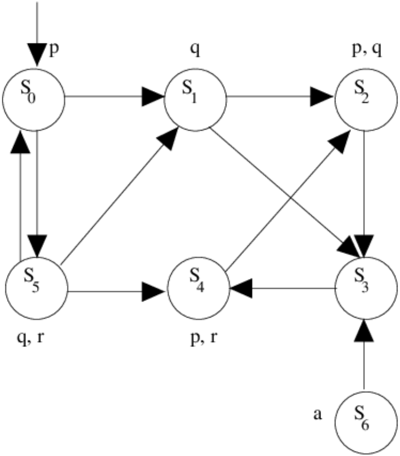
$\mathcal{S} \models \mathbf{EF}p$ for the same reason as above

If it holds for all paths, then it holds for one path

$\mathbf{AF}\Phi \rightarrow \mathbf{EF}\Phi$

The same holds for the other temporal operators $\mathbf{G}, \mathbf{U}$ etc

CTL Examples



$\mathcal{S} \not\models \mathbf{EF}a$ since s_6 is not reachable

Note that the counterexample cannot be a single path

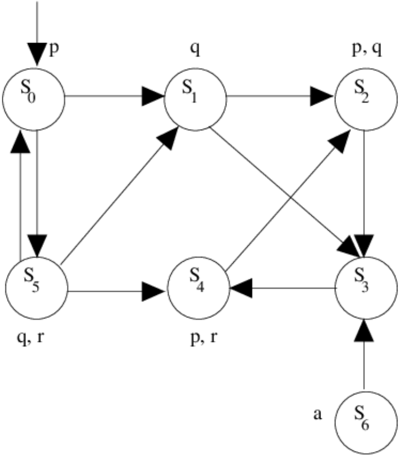
Since it would not enough to disprove existence

The full reachable graph must be provided

One could also show the tree of all paths

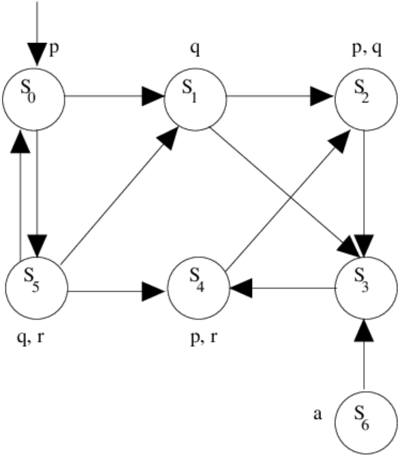
Neither safety nor liveness

CTL Examples



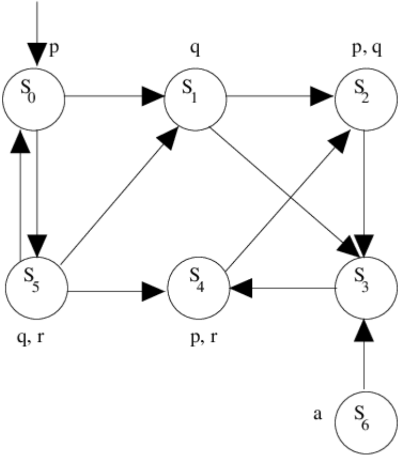
$\mathcal{S} \models \mathbf{A}(p\ \mathbf{U}\ q)$ since $p \in L(s_0)$, $\mathrm{next}(s_0) = \{s_1, s_5\}$ and $q \in L(s_1) \wedge q \in L(s_5)$

CTL Examples



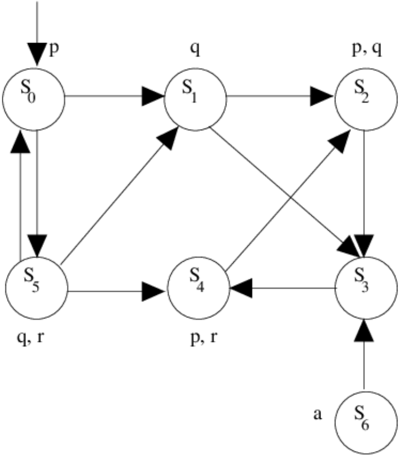
$\mathcal{S} \not\models \mathbf{A}(p\ \mathbf{U}\ r)$, a counterexample is $\pi = s_0 s_1 (s_2 s_3 s_4)$

CTL Examples



$\mathcal{S} \models \mathbf{E}(p\ \mathbf{U}\ r)$, an example is $\pi = (s_0 s_5)$

CTL Examples

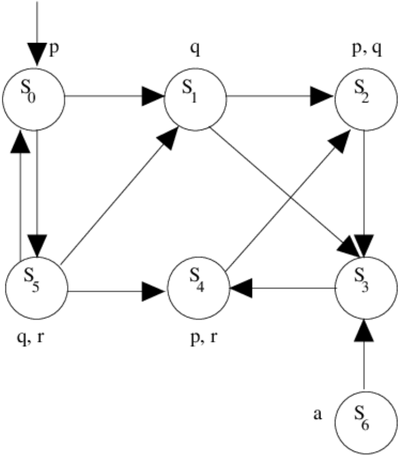


$\mathcal{S} \not\models \neg\mathbf{E}(p \ \mathbf{U} \ r)$, a counterexample is $\pi = (s_0 s_5)$

In fact, $\mathcal{S} \not\models \Phi$ iff $\mathcal{S} \models \neg(\Phi)$ whenever $|I| = 1$

In fact, the implicit for all is on initial states only, whilst it is on all paths for LTL...

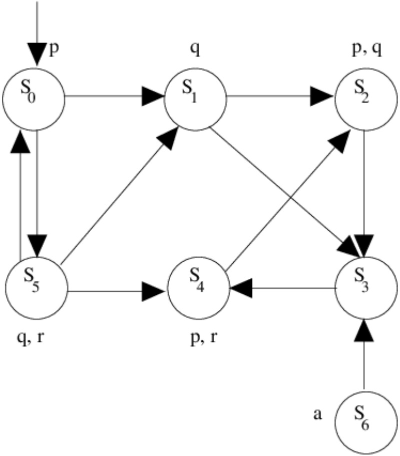
CTL Examples



$\mathcal{S} \not\models$ **AFAG**p, a counterexample is $\pi = s_0 s_1 (s_2 s_3 s_4)$

This is a liveness formula

CTL Examples

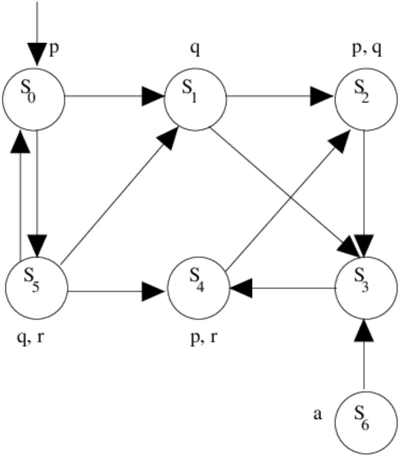


$\mathcal{S} \not\models$ **EFEG**p, a counterexample is again a computation tree

All lassos are $s_0 s_5$ or $s_2 s_3 s_4$

In both such lassos, there are states in which p does not hold

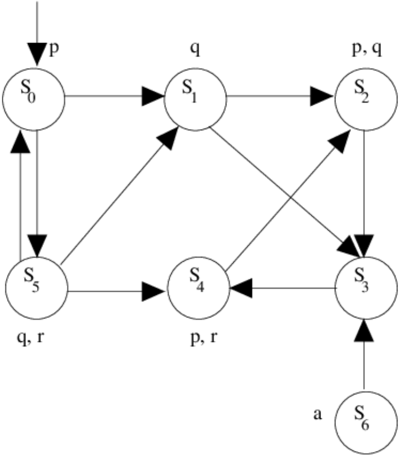
CTL Examples



$\mathcal{S} \not\models$ **AFEG**p, a counterexample is again a computation tree

Since $\mathcal{S} \not\models$ **EFEG**p...

CTL Examples



$\mathcal{S} \not\models$ **EFAG**p, a counterexample is again a computation tree
Since $\mathcal{S} \not\models$ **EFEG**p...

CTL Non-Toy Examples

- Recall the Peterson's protocol: checking mutual exclusion is $\mathbf{AG}(\neg(p \wedge q))$, being $p = \mathtt{P[1]} = \mathtt{L3}, q = \mathtt{P[2]} = \mathtt{L3}$
  - equivalent to LTL $\mathbf{G}p$
- It is always possible to restart: $\mathbf{AGEF}\ P[1] = L0 \wedge \mathbf{AGEF}\ P[2] = L0$

CTL vs. LTL: a Comparison

- Recall that $\varphi_1 \equiv \varphi_2$ iff $\forall \mathcal{S}.\, \mathcal{S} \models \varphi_1 \Leftrightarrow \mathcal{S} \models \varphi_2$
  - also holds (w.l.g.) when φ_1 is LTL and φ_2 is CTL
- Of course, some CTL formulas cannot be expressed in LTL
  - it is enough to put an **E**, since LTL always universally quantifies paths
  - so, there is not an LTL φ s.t. $\varphi \equiv \mathbf{EG}p$
    - no, $\mathbf{F}\neg p$ is not the same, why?
- So, one might think: LTL is contained in CTL
  - in the sense, for each LTL formula, there is a CTL equivalent formula
  - simply replace each temporal operator **O** with **AO**, that's it
  - let $\mathcal{T}$ be a translator doing this
  - for any LTL formula φ, $\varphi \equiv \mathcal{T}(\varphi)$
  - actually, $\mathbf{G}p \equiv \mathcal{T}(\mathbf{G}p) = \mathbf{AG}p$

UNIVERSITÀ DEGLI STUDI DELL'AQUILA

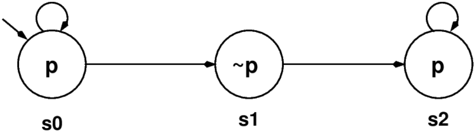
DISIM Dipartimento di Ingegneria e Scienze dell'Informazione e Matematica

CTL vs. LTL: a Comparison

- Theorem. Let φ be an LTL formula. Then, either i) $\varphi \equiv \mathcal{T}(\varphi)$ or ii) there does not exist a CTL formula ψ s.t. $\varphi \equiv \psi$
  - idea of proof: replacing with **E** is of course not correct, and temporal operators on paths are the same
- Corollary. There exists an LTL formula φ s.t., for all CTL formulas ψ, $\varphi \not\equiv \psi$
- Proof of corollary:
  - by the theorem above and the definitions, we need to find
    1. an LTL formula φ
    2. a KS $\mathcal{S}$
  - where $\mathcal{S} \models \varphi$ and $\mathcal{S} \not\models \mathcal{T}(\varphi)$
    - viceversa is not possible

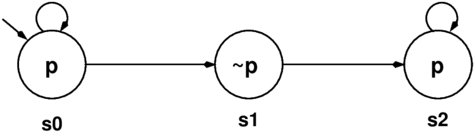
CTL vs. LTL: a Comparison

- For example, as for the LTL formula, we may take $\varphi = \mathbf{FG}p$
  - note instead that $\mathbf{GF}p \equiv \mathbf{AGAF}p$
- For example, as for the KS $\mathcal{S}$, we may take



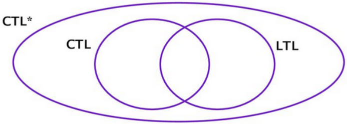
- We have that $\mathcal{S} \models \mathbf{FG}p$, but $\mathcal{S} \not\models \mathbf{AFAG}p$
- Thus, CTL requires "more" than the corresponding LTL

CTL vs. LTL: a Comparison



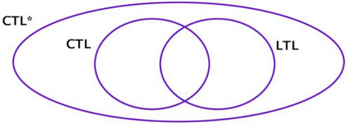
- $\mathcal{S} \not\models$ **AFAG**p means that
  $\neg(\forall \pi \in \mathrm{Path}(\mathcal{S}).\ \exists j:\ \forall \rho \in \mathrm{Path}(\mathcal{S}, \pi(j)).\ \forall k.\ p \in \rho(k))$
  $= \exists \pi \in \mathrm{Path}(\mathcal{S}).\ \forall j:\ \exists \rho \in \mathrm{Path}(\mathcal{S}, \pi(j)).\ \exists k.\ p \notin \rho(k)$
- In our $\mathcal{S}$, $\pi = s_0^\omega$: in fact, at any point of π, you may branch and go through $\neg p$ instead...
- $\mathcal{S} \models$ **FG**p means that $\forall \pi \in \mathrm{Path}(\mathcal{S}).\ \exists j:\ \forall k \geq j.\ p \in \pi(k)$
- Thus, there is not a CTL formula equivalent to **FG**p
- Furthermore, there is not an LTL formula equivalent to **AFAG**p

CTL, LTL and CTL*



- CTL* introduced in 1986 (Emerson, Halpern) to include both CTL and LTL
- No restrictions on path quantifiers to be 1-1 with temporal operators, as in CTL
- State formulas: $\Phi ::= \mathrm{true} \mid p \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \mathbf{A}\Psi \mid \mathbf{E}\Psi$
- Path formulas: $\Psi ::= \Phi \mid \Psi_1 \wedge \Psi_2 \mid \neg\Psi \mid \Psi_1 \mathbf{U} \Psi_2 \mid \mathbf{F}\Psi \mid \mathbf{G}\Psi$

CTL, LTL and CTL*



- The intersection between CTL and LTL is both syntactic and “semantic”
- Some formulas are both CTL and LTL in syntax: all those involving only boolean combinations of atomic propositions
- “Semantic” intersection: some LTL formulas may be expressed in CTL and vice versa, using different syntax
  - **AGAF**p and **GF**p
  - **AG**p and **G**p
  - etc