Tecniche di Specifica e di Verifica

CTL*, CTL and LTL

CTL* language I

Syntax Let **AP** a finite set of ***atomic propositions***. We define by mutual induction the following set of formulae:

(*state formulae*)

0 If $\mathbf{p} \in \mathbf{AP}$, then **p** is a ***state*** formula.

1 If ψ and ψ' are ***state*** formulae, then so are $\neg\psi$ and $\psi \vee \psi'$, $\psi \wedge \psi'$.

2 If ψ and ψ' are ***path*** formulae, then $\mathbf{E}\psi$ and $\mathbf{A}\psi$ are ***state*** formulae .

CTL* language I

Syntax ...

(***path formulae***)

3 if y is a ***state*** formula, then y is a ***path*** formula.

4 if y and y' are ***path*** formulae, then so are Øy and y Ú y', y Ù y'.

5 if y and y' are ***path*** formulae, then so are **X**y and y **U** y'.

CTL* semantics I

Semantics Given the standard definitions

K = (S, S_0, R, AP, L), **s Î S**, **L: S ® 2^{AP}** and

path of **K**: p = s_0 s_1 s_2…. where (s_i s_{i+1}) Î **R**:

0 **K, s ⊨ p** iff **p Î L(s).**

1 for ***propositional formulae***

- **K, s ⊨** Øy iff ***not*** **K, s ⊨** y
- **K, s ⊨** y $_1$ Ú y $_2$ iff **K, s ⊨** y $_1$ or **K, s ⊨** y $_2$.
- **K, s ⊨** y $_1$ Ù y $_2$ iff **K, s ⊨** y $_1$ and **K, s ⊨** y $_2$.

2 **K,s ⊨ E**y (**K,s ⊨ A**y) iff for some (for all) path p=**s** s_1 s_2…., **K,**p **⊨** y

CTL* semantics II

Semantics ...

3 **K,** p ⊨ **p** iff **K,** $\mathbf{s_0}$ ⊨ **p** .

4 for propositional formulare

– **K,** p ⊨ Øy iff ***not*** **K,** p ⊨ y

– **K,** p ⊨ y $_{\mathbf{1}}$ Ú y $_{\mathbf{2}}$ iff **K,** p ⊨ y $_{\mathbf{1}}$ or **K,** p ⊨ y $_{\mathbf{2}}$.

– **K,** p ⊨ y $_{\mathbf{1}}$ Ù y $_{\mathbf{2}}$ iff **K,** p ⊨ y $_{\mathbf{1}}$ and **K,** p ⊨ y $_{\mathbf{2}}$.

5 ***temporal operators***

– **K,**p ⊨ **X**y iff **K,**$p^{\mathbf{1}}$ ⊨ y

– **K,**p ⊨ y **U** y **’** iff for some **j**, **K,**$p^{\mathbf{j}}$ ⊨ y **’**, and for all **k<j**, **K,**$p^{\mathbf{k}}$ ⊨ y

CTL language definition

CTL can be defined as the ***sub-labguage*** of **CTL***[*] by replacing items 3-5 of the above definition, by the following:

3' if y and y' are ***state*** formulae, then **X**y and y **U** y' are ***path*** formulae.

LTL, CTL and CTL*

LTL (state): j ::= **A** y

(path): y ::= ***p*** ½ Ø y ½ y_1 Ú y_2 ½**X** y ½ y_1 **U** y_2

CTL (state): j ::= ***p*** ½ Ø j ½ j_1 Ú j_2 ½**E** y

(path): y ::= **X** j ½ j_1 **U** j_2

CTL* (state): j ::= ***p*** ½ Ø j ½ j_1 Ú j_2 ½**E** y

(path): y ::= j ½ Ø y ½ y_1 Ú y_2 ½**X** y ½ y_1 **U** y_2

LTL and CTL*

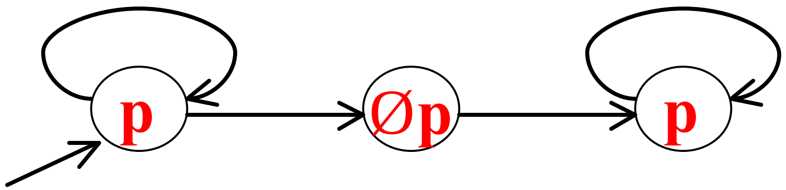
Theorem:[Clarke] For every **CTL*** formula y, an equivalent **LTL** (it it exists) must be of the form **A*f*(y)** where ***f*(y)** is equal to y with all the path quantifiers eliminated.

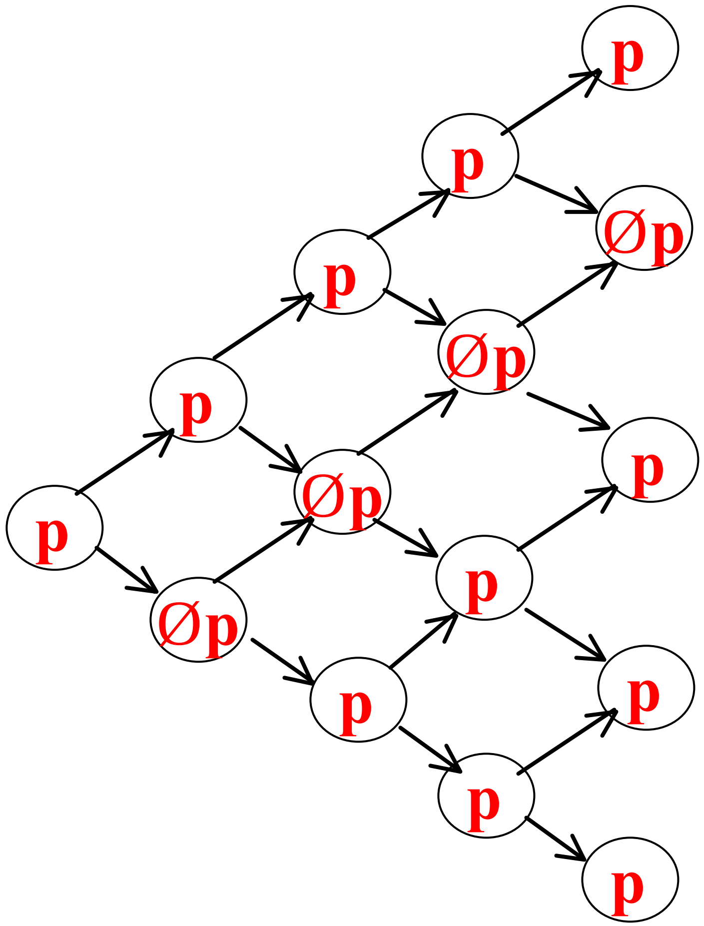
LTL vs CTL

In LTL, we could write:
A FG *p* , which means "on all paths, there is some state from which ***p*** will forever hold" (i.e. **Ø *p*** holds finitely often). `

There is no equivalent of this LTL formula in CTL.

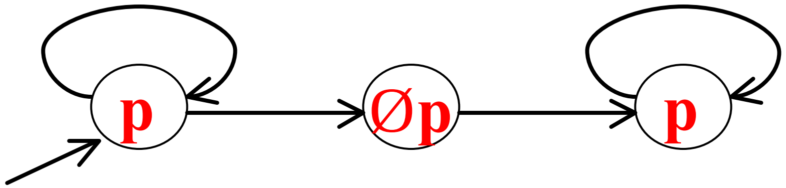
For example, in the following model, **A FG *p*** holds, but the formula **AF AG *p*** does not.

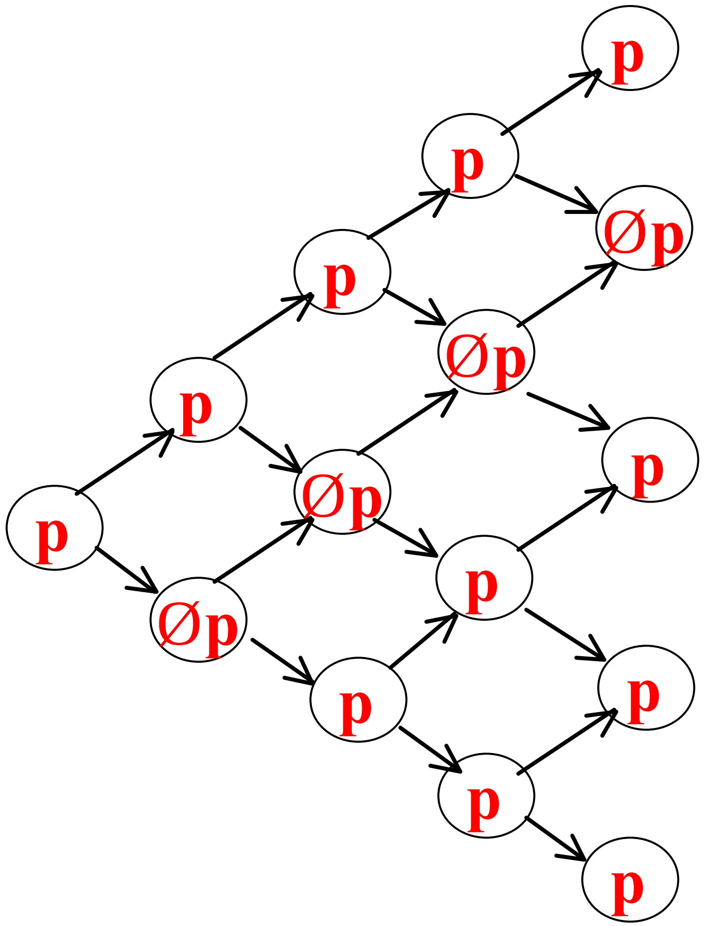




LTL vs CTL

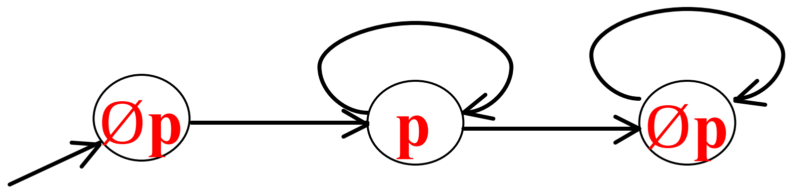
Similarly the LTL formula **AF(*p* Ù X *p*)** has no equivalent in CTL.
Two attempts are:
AF(*p* Ù AX *p*)
But in the model on the right, the LTL formula is true while the CTL formula is false

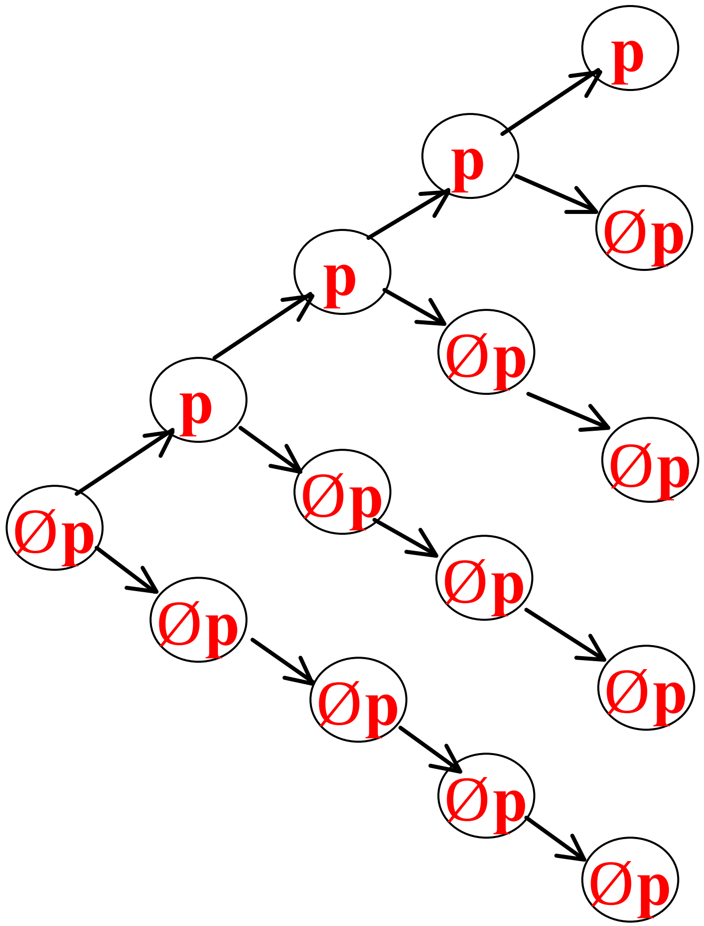




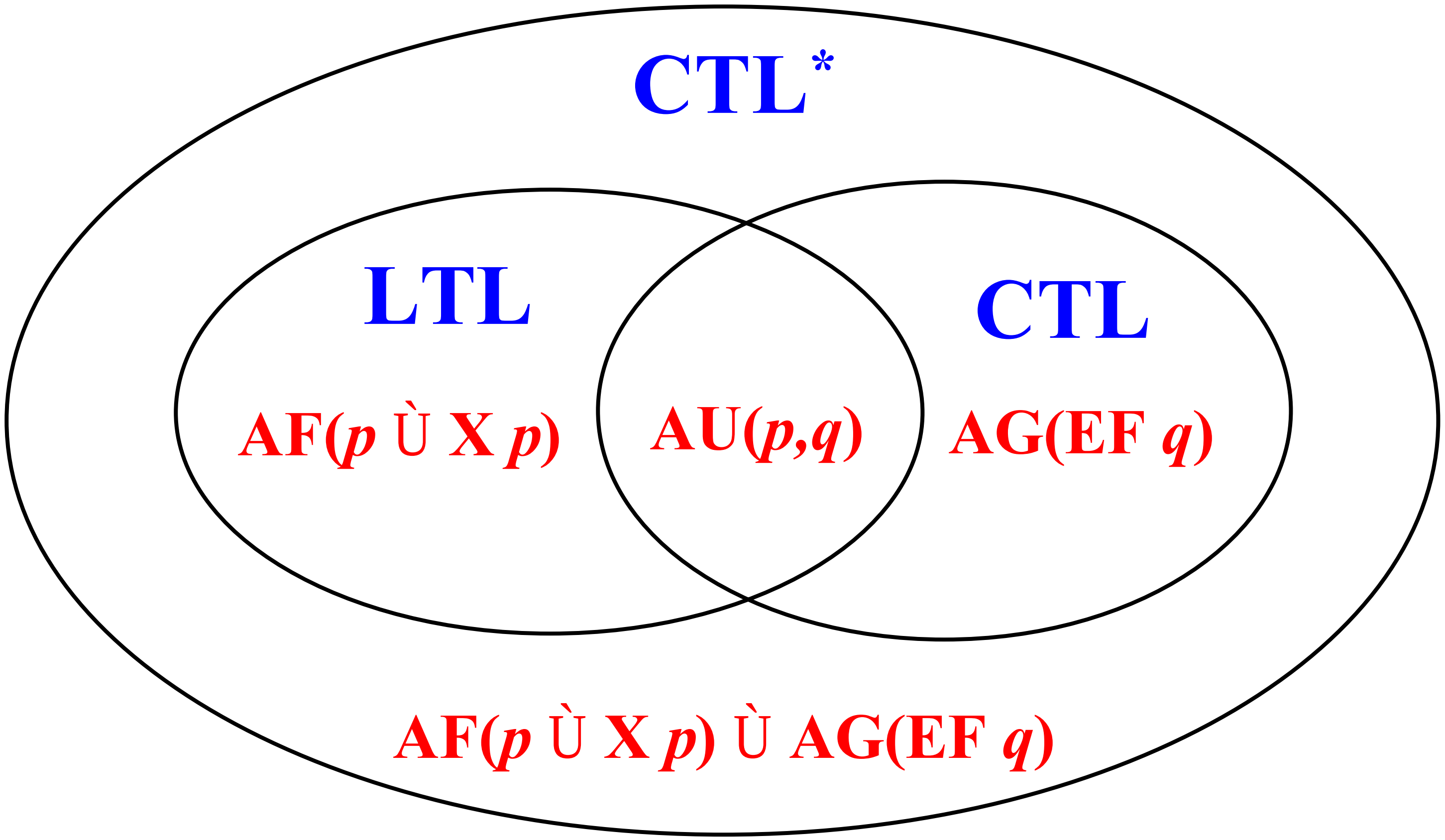
LTL vs CTL

Similarly the LTL formula **AF(*p* Ù X *p*)** has no equivalent in CTL.
Two attempts are:
AF(*p* Ù AX *p*)
and
AF(*p* Ù EX *p*)
But in the model on the right, the LTL formula is false while the second CTL formula is true.





LTL vs CTL vs CTL*



LTL vs CTL vs CTL*

- $A\,GF\,\varphi$ is a **LTL** formula which *can be expressed* in **CTL** by the *equivalent* formula $AG\,AF\,\varphi$.
- For any φ and ψ the **LTL** formula $A(GF\,\varphi \rightarrow \psi)$ is *not expressible* in **CTL**, in particular it is *not equivalent to* $((AG\,AF\,\varphi) \rightarrow \psi)$.
- In other words, *fairness constraints cannot be expressed* directly in **CTL**.