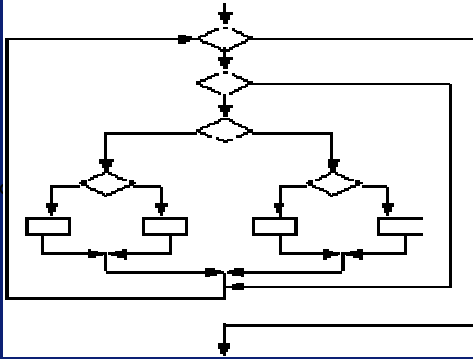


Spring 2009



Program correctness

Linear Time Temporal Logic

Marcello Bonsangue



Leiden Institute of Advanced Computer Science
Research & Education

Formal Verification

Verification techniques comprise

- **a modelling framework** \mathcal{M}
to describe a system
- **a specification language** ϕ
to describe the properties to be verified
- **a verification method** $\mathcal{M} \models \phi, \Gamma \vdash \phi$
to establish whether a model satisfies a property



Motivations

- For an elevator system, consider the requirements:
 - any request must ultimately be satisfied
 - the elevator never traverses a floor for which a request is pending without satisfying it
- Both concern the **dynamic behavior** of the system. They can be formalized using a time-dependent notation, like

$$z(t) = 1/2gt^2$$

for the free-falling elevator



Example

- In first order logic, with
 - $E(t)$ = elevator position at time t
 - $P(n,t)$ = pending request at floor n at time t
 - $S(n,t)$ = servicing of floor n at time t

Any request must ultimately be satisfied

$$\forall t \forall n (P(n,t) \Rightarrow \exists t' > t : S(n,t'))$$

The elevator never traverse a floor for which a request is pending without satisfying it

$$\forall t \forall t' > t \forall n (P(n,t) \wedge E(t') \neq n \wedge \exists t'' < t' : E(t'') = n) \Rightarrow \exists t'' < t' : S(n,t'')$$



Temporal Logic

- First order logic is too cumbersome for these specifications
- Temporal logic is a logic tailored for describing properties involving time
 - the time parameter t disappears
 - temporal operators mimic linguistic constructs
 - **always, until, eventually**
 - the truth of a proposition depend on the state on which the system is



LTL: the language

- **Atomic propositions** $p_1, p_2, \dots, q, \dots$
 - to make statements about states of the system
 - elementary descriptions which in a given state of the system have a well-defined truth value:
 - the printer is busy
 - nice weather
 - open
 - $x+2=y$

 - Their choice depend on the system considered



LTL: the language

■ Boolean combinators

<input type="checkbox"/> true	\top
<input type="checkbox"/> false	\perp
<input type="checkbox"/> negation	\neg
<input type="checkbox"/> conjunction	\wedge
<input type="checkbox"/> disjunction	\vee
<input type="checkbox"/> implication	\Rightarrow

Note: read $p \Rightarrow q$ as “*if p then q*” rather than “*p implies q*”.

Try $(1 = 2) \Rightarrow \text{Sint_Klas_exists}$



LTL: the language

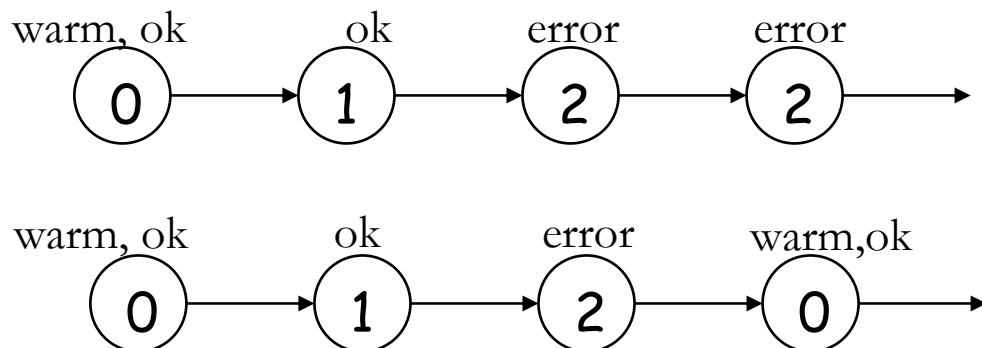
- **Temporal combinators** allows to speak about the sequencing of states along a computation (rather than about states individually)

- **neXt**

X

- $X\phi$ = *in the next state ϕ holds*

- Examples: $XX\text{error}$ and $XXX\text{ok}$

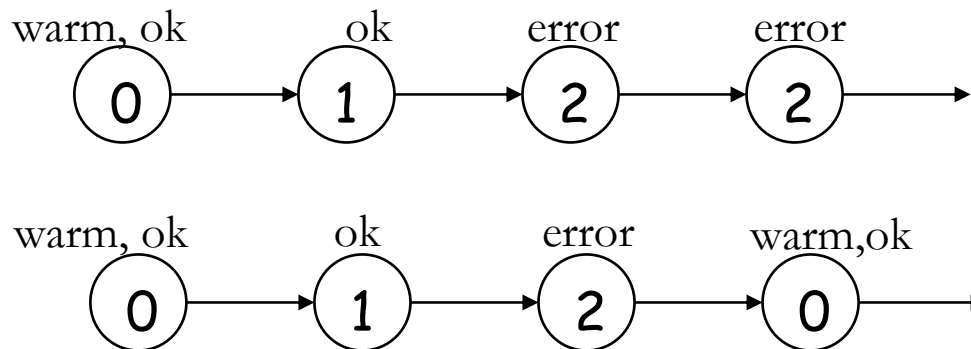


LTL: Temporal combinators

■ Future

F

- $F\phi$ = *in some future state ϕ holds* (at least once and without saying in which state)
- For example, $\text{warm} \Rightarrow F\text{ok}$ holds if we are in a “warm” state then we will be in an “ok” state.



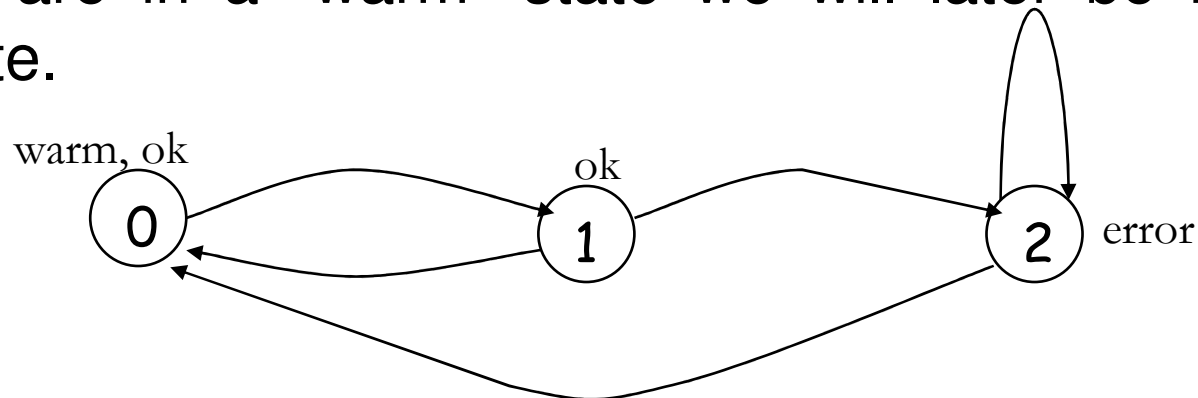
LTL: Temporal combinators

■ Globally

G

- $G\phi$ = in all future states ϕ always holds
- It is the dual of F: $G\phi = \neg F\neg\phi$

- For example $G(\text{warm} \Rightarrow F\text{ok})$ holds if at any time when we are in a “warm” state we will later be in an “ok” state.



- $G(\text{warm} \Rightarrow X\neg\text{warm})$? $G(\text{ok} \Rightarrow X\text{warm})$?



LTL: Temporal combinators

■ Until

U

- $\phi_1 U \phi_2 = \phi_2$ will hold in some future state, and in all intermediate states ϕ_1 will hold.

■ Weak until

W

- $\phi_1 W \phi_2 = \phi_1$ holds in all future states until ϕ_2 holds
- it may be the case ϕ_2 will never hold



LTL: Temporal combinators

■ Release

R

□ $\phi_1 R \phi_2 = \phi_2$ holds in all future state up to (and including) a state when ϕ_1 holds (if ever).

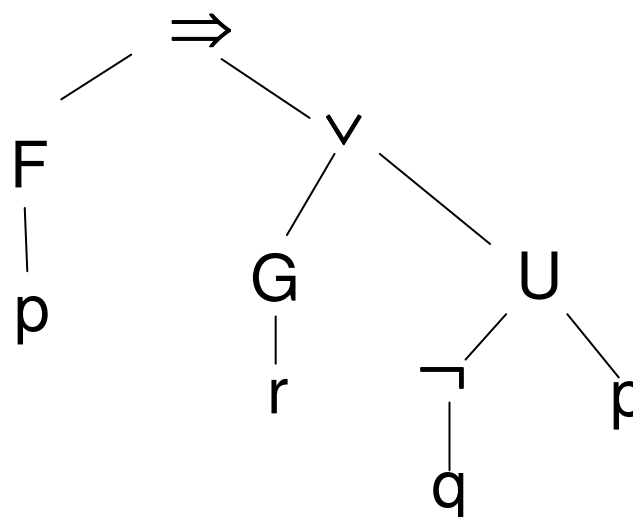
□ It is the dual of U: $\phi_1 R \phi_2 = \neg(\neg\phi_1 U \neg\phi_2)$



LTL - Priorities

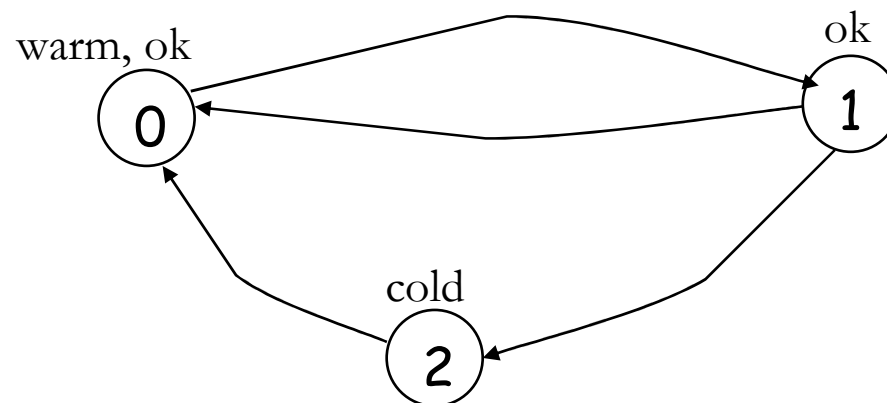
- Unary connectives bind most tightly
 - \neg , X, F, G
- Next come U, R and W
- Finally come \wedge , \vee and \Rightarrow

$Fp \Rightarrow Gr \vee \neg qUp$



LTL models: Transition Systems

- **Transition system:** $\langle S, \rightarrow, L \rangle$
 - S set of states
 - $L: S \rightarrow \mathcal{P}(\text{Atoms})$ labelling function
 - $\rightarrow \subseteq S \times S$ transition relation
 - Every state s has some successor state s' with $s \rightarrow s'$
- A system evolves from one **state** to another under the action of a **transition**
- We label a state with propositions that hold in that state



Computation paths

- **Path**: an infinite sequence π of states such that each consecutive pair is connected by a transition

$$0 \rightarrow 1 \rightarrow 2 \rightarrow 0 \rightarrow \dots$$

- For $i \geq 1$, we write π^i for the suffix of a path π starting at i .



Semantics (I)

- Let $M = \langle S, \rightarrow, L \rangle$ be a transition system, and $\pi = s_1 \rightarrow s_2 \rightarrow \dots$ a path of M .

- $\pi \models \top$ always
- $\pi \models p$ iff $p \in I(s_1)$
- $\pi \models \neg\phi$ iff $\pi \not\models \phi$
- $\pi \models \phi_1 \wedge \phi_2$ iff $\pi \models \phi_1$ and $\pi \models \phi_2$



Semantics (II)

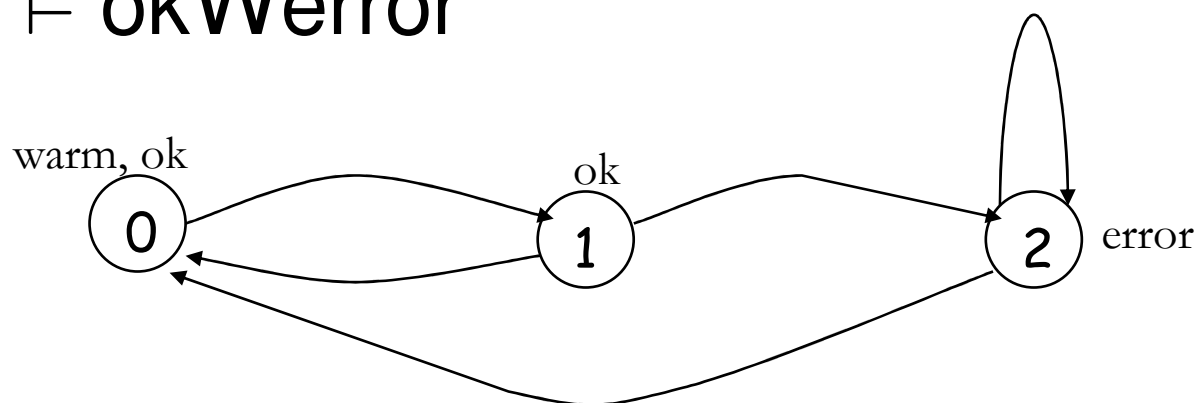
- $\pi \models X\phi$ iff $\pi^2 \models \phi$
- $\pi \models F\phi$ iff there is $1 \leq i$ such that $\pi^i \models \phi$
- $\pi \models G\phi$ iff for all $1 \leq i$, $\pi^i \models \phi$
- $\pi \models \phi_1 U \phi_2$ iff there is $1 \leq i$ such that $\pi^i \models \phi_2$
and for all $j < i$, $\pi^j \models \phi_1$
- $\pi \models \phi_1 W \phi_2$ iff either $\pi \models \phi_1 U \phi_2$ or for all $1 \leq i$, $\pi^i \models \phi_2$
- $\pi \models \phi_1 R \phi_2$ iff either there is $1 \leq i$ such that $\pi^i \models \phi_1$
and for all $j \leq i$, $\pi^j \models \phi_2$
or for all $1 \leq k$, $\pi^k \models \phi_2$



System properties

- $M, s \models \phi$ iff $\pi \models \phi$ for every path π of M starting from the state s

- $M, 0 \models \text{ok}W\text{error}$



- $M, 0 \not\models \text{ok}U\text{error}$ (Why?)

