Linear Temporal Logic

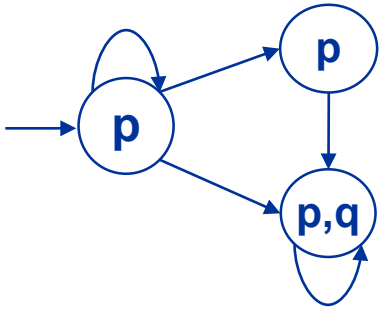
Moonzoo Kim
School of Computing
KAIST

Motivation for verification

- There is a great advantage in being able to verify the correctness of computer systems
  - This is most obvious in the case of safety-critical systems
    - ex. Cars, avionics, medical devices
  - Also applies to mass-produced embedded devices
    - ex. handphone, USB memory, MP3 players, etc
- Formal verification can be thought of as comprising three parts
  1. a system description language
  2. a requirement specification language
  3. a verification method to establish whether the description of a system satisfies the requirement specification.

Model checking

- Model checking
  - In a model-based approach, the system is represented by a model $\mathcal{M}$. The specification is again represented by a formula Φ.
    - The verification consists of computing whether $\mathcal{M}$ satisfies Φ $\mathcal{M} \models \Phi$
      - Caution: $\mathcal{M} \models \Phi$ represents satisfaction, not semantic entailment
- In model checking,
  - The model $\mathcal{M}$ is a transition systems and
  - the property Φ is a formula in temporal logic
    - ex. □ p, □ q, ◇ q, □ ◇ q



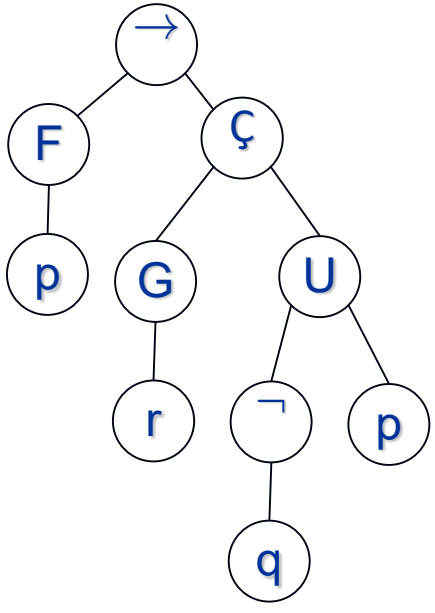
Linear time temporal logic (LTL)

- LTL models time as a sequence of states, extending infinitely into the future
  - sometimes a sequence of states is called a computation path or an execution path, or simply a path
- Def 3.1 LTL has the following syntax
  - $\Phi ::= T \mid \bot \mid p \mid \neg \Phi \mid \Phi \wedge \Phi \mid \Phi \vee \Phi \mid \Phi \rightarrow \Phi$

    $\mid X \Phi \mid F \Phi \mid G \Phi \mid \Phi U \Phi \mid \Phi W \Phi \mid \Phi R \Phi$

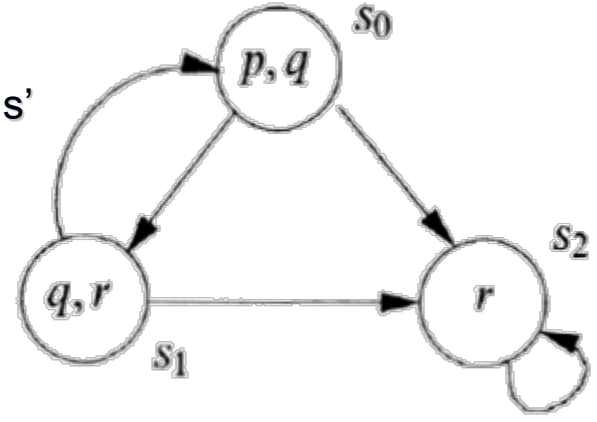
    where p is any propositional atom from some set Atoms
  - Operator precedence
    - the unary connectives bind most tightly. Next in the order come U, R, W, $\wedge$, $\vee$, and $\rightarrow$

F p → G r Ç ¬ q U p



Semantics of LTL (1/3)

- Def 3.4 A transition system (called model) $\mathcal{M}$ = (S, $\rightarrow$, L)
  - a set of states S
  - a transition relation $\rightarrow$ (a binary relation on S)
    - such that every s $\in$ S has some s' $\in$ S with s $\rightarrow$ s'
  - a labeling function L: S $\rightarrow \mathcal{P}$ (Atoms)
- Example
  - S={s_0,s_1,s_2}
  - $\rightarrow$={(s_0,s_1),(s_1,s_0),(s_1,s_2),(s_0,s_2),(s_2,s_2)}
  - L={(s_0,{p,q}),(s_1,{q,r}), (s_2,{r})}



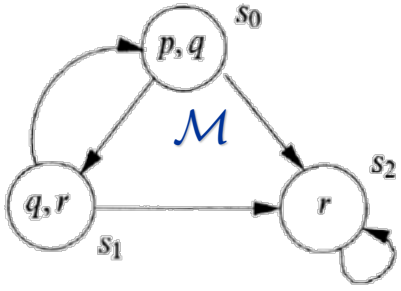
- Def. 3.5 A path in a model $\mathcal{M}$ = (S, $\rightarrow$, L) is an infinite sequence of states s_{i_1}, s_{i_2}, s_{i_3},… in S s.t. for each j$\geq$ 1, $s_{i_j} \rightarrow s_{i_{j+1}}$. We write the path as $s_{i_1} \rightarrow s_{i_2} \rightarrow$ …
  - From now on if there is no confusion, we drop the subscript index i for the sake of simple description
- We write π^i for the suffix of a path starting at s_i.
  - ex. π^3 is $s_3 \rightarrow s_4 \rightarrow$ …

Semantics of LTL (2/3)

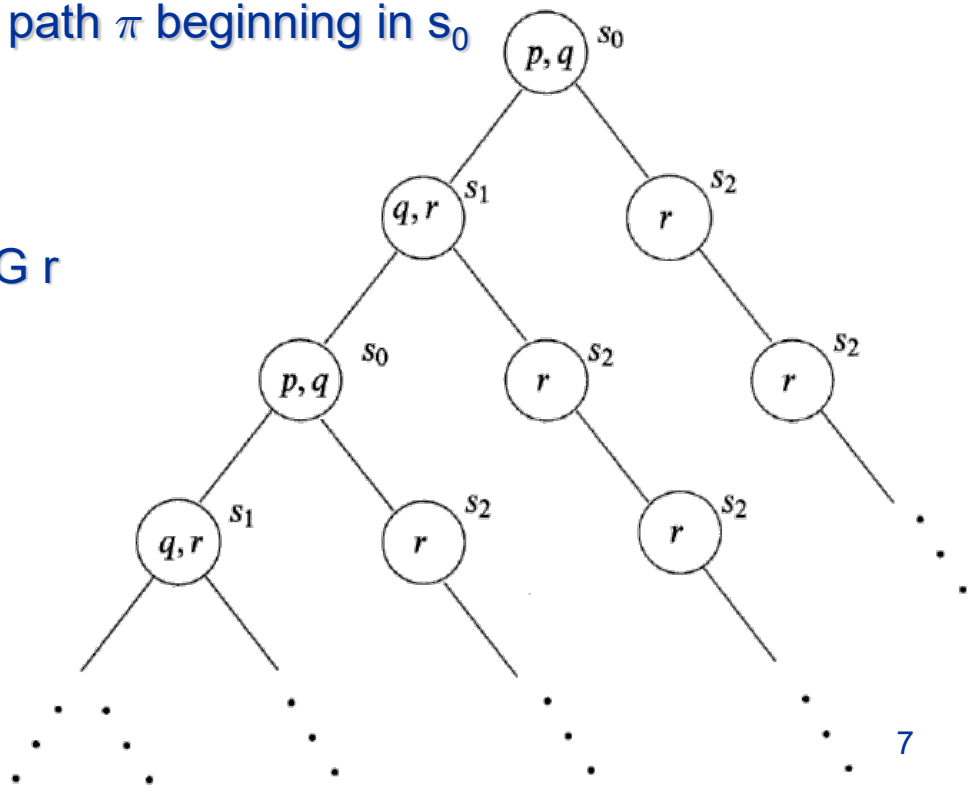
- Def 3.6 Let $\mathcal{M}$ = (S, $\rightarrow$, L) be a model and $\pi = s_1 \rightarrow \ldots$ be a path in $\mathcal{M}$. Whether π satisfies an LTL formula is defined by the satisfaction relation $\models$ as follows:
  - Basics: $\pi \models \top$, $\pi \not\models \bot$, $\pi \models p$ iff $p \in L(s_1)$, $\pi \models \neg\Phi$ iff $\pi \not\models \Phi$
  - Boolean operators: $\pi \models p \wedge q$ iff $\pi \models p$ and $\pi \models q$
    - similar for other boolean binary operators
  - $\pi \models X\,\Phi$ iff $\pi^2 \models \Phi$ (next $\circ$)
  - $\pi \models G\,\Phi$ iff for all $i \geq 1$, $\pi^i \models \Phi$ (always $\Box$)
  - $\pi \models F\,\Phi$ iff there is some $i \geq 1$, $\pi^i \models \Phi$ (eventually $\Diamond$)
  - $\pi \models \Phi U\,\psi$ iff there is some $i \geq 1$ s.t. $\pi^i \models \psi$ and for all j=1,…,i-1 we have $\pi^j \models \Phi$ (strong until)
  - $\pi \models \Phi W\,\psi$ iff either (weak until)
    - either there is some $i \geq 1$ s.t. $\pi^i \models \psi$ and for all j=1,…,i-1 we have $\pi^j \models \Phi$
    - or for all $k \geq 1$ we have $\pi^k \models \Phi$
  - $\pi \models \Phi R\,\psi$ iff either (release)
    - either there is some $i \geq 1$ s.t. $\pi^i \models \Phi$ and for all j=1,…,i we have $\pi^j \models \psi$
    - or for all $k \geq 1$ we have $\pi^k \models \psi$

Semantics of LTL (3/3)

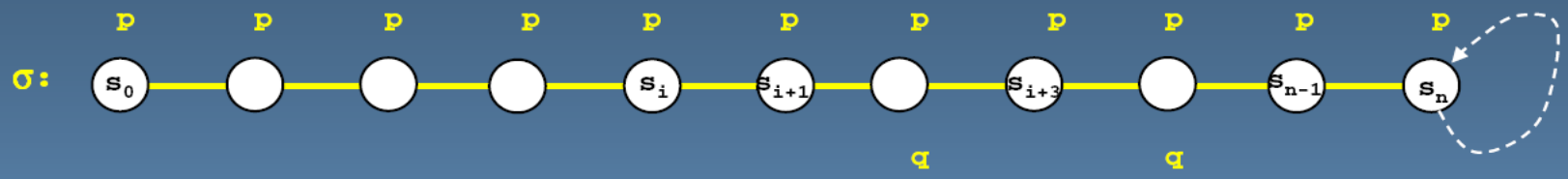
- Def 3.8 Suppose $\mathcal{M} = (S, \rightarrow, L)$ is a model, $s \in S$, and Φ an LTL formula. We write $\mathcal{M}, s \models \Phi$ if for every execution path π of $\mathcal{M}$ starting at s, we have $\pi \models \Phi$
  - If $\mathcal{M}$ is clear from the context, we write $s \models \Phi$



- Example
  - $s_0 \models p \wedge q$ since $\pi \models p \wedge q$ for every path π beginning in s_0
  - $s_0 \models \neg r$, $s_0 \models \top$
  - $s_0 \models X\, r$, $s_0 \not\models X\,(q \wedge r)$
  - $s_0 \models G\, \neg(p \wedge r)$, $s_2 \models G\, r$
  - For any s of $\mathcal{M}$, $s \models F(\neg q \wedge r) \rightarrow F\, G\, r$
    - Note that s_2 satisfies $\neg q \wedge r$
  - $s_0 \not\models G\, F\, p$
    - $s_0 \rightarrow s_1 \rightarrow s_0 \rightarrow s_1 \ldots \models G\, F\, p$
    - $s_0 \rightarrow s_2 \rightarrow s_2 \rightarrow s_2 \ldots \not\models G\, F\, p$
  - $s_0 \models G\, F\, p \rightarrow G\, F\, r$
  - $s_0 \not\models G\, F\, r \rightarrow G\, F\, p$



examples



[]p is satified at all locations in σ

<>p is satisfied at all locations in σ

[]<>p is satisfied at all locations in σ

<>q is satisfied at all locations except s_{n-1} and s_n

Xq is satisfied at s_{i+1} and at s_{i+3}

pUq (strong until) is satisfied at all locations except s_{n-1} and s_n

<>(pUq) (strong until) is satisfied at all locations except s_{n-1} and s_n

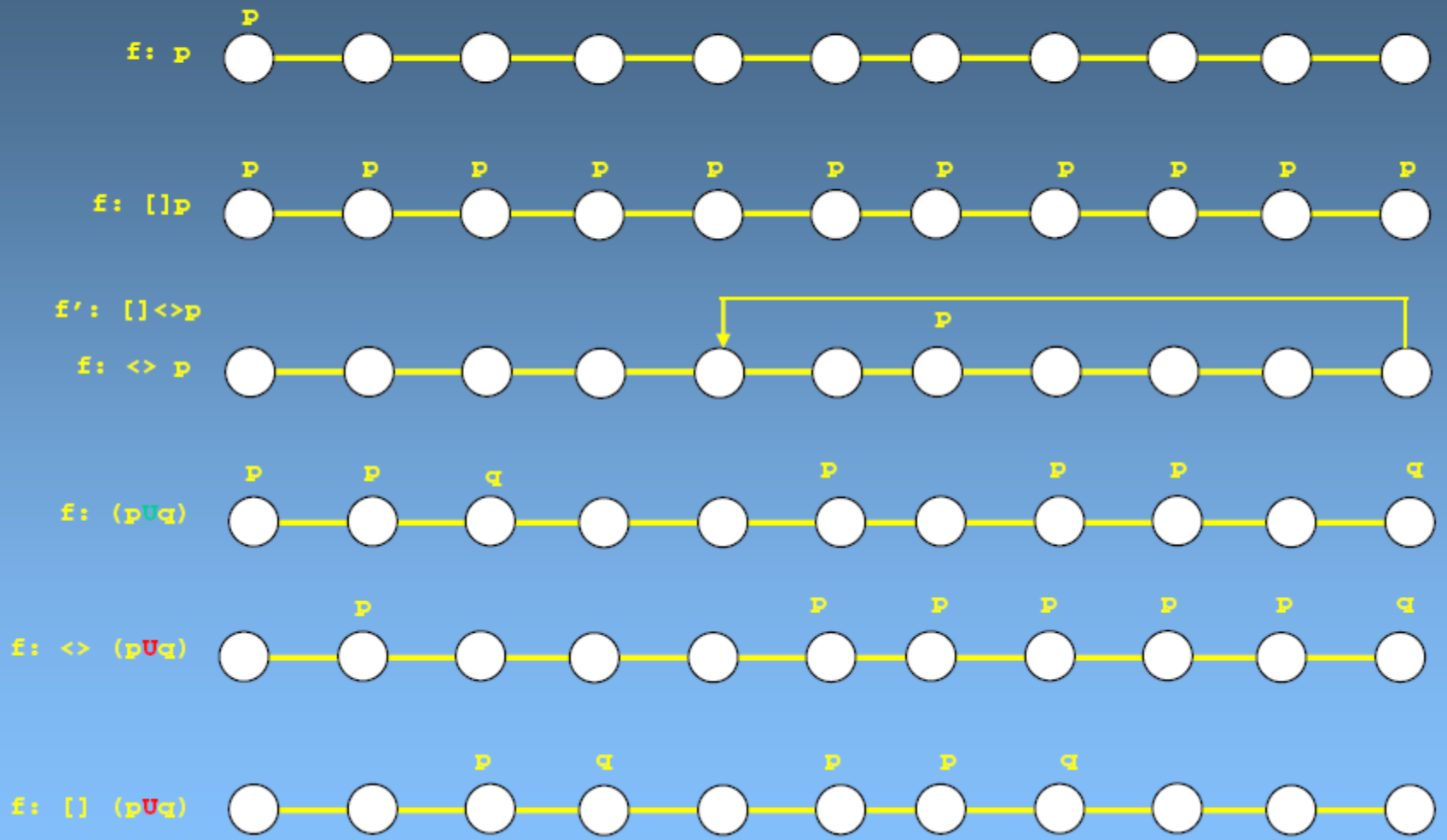
<>(pUq) (weak until) is satisfied at all locations

[]<>(pUq) (weak until) is satisfied at all locations

in model checking we are typically only interested in whether a temporal logic formula is satisfied for all runs of the system, starting in the initial system state (that is: at s_0)

slide quoted from Caltech 101b.2 "Logic Model Checking" by Dr.G.Holzmann

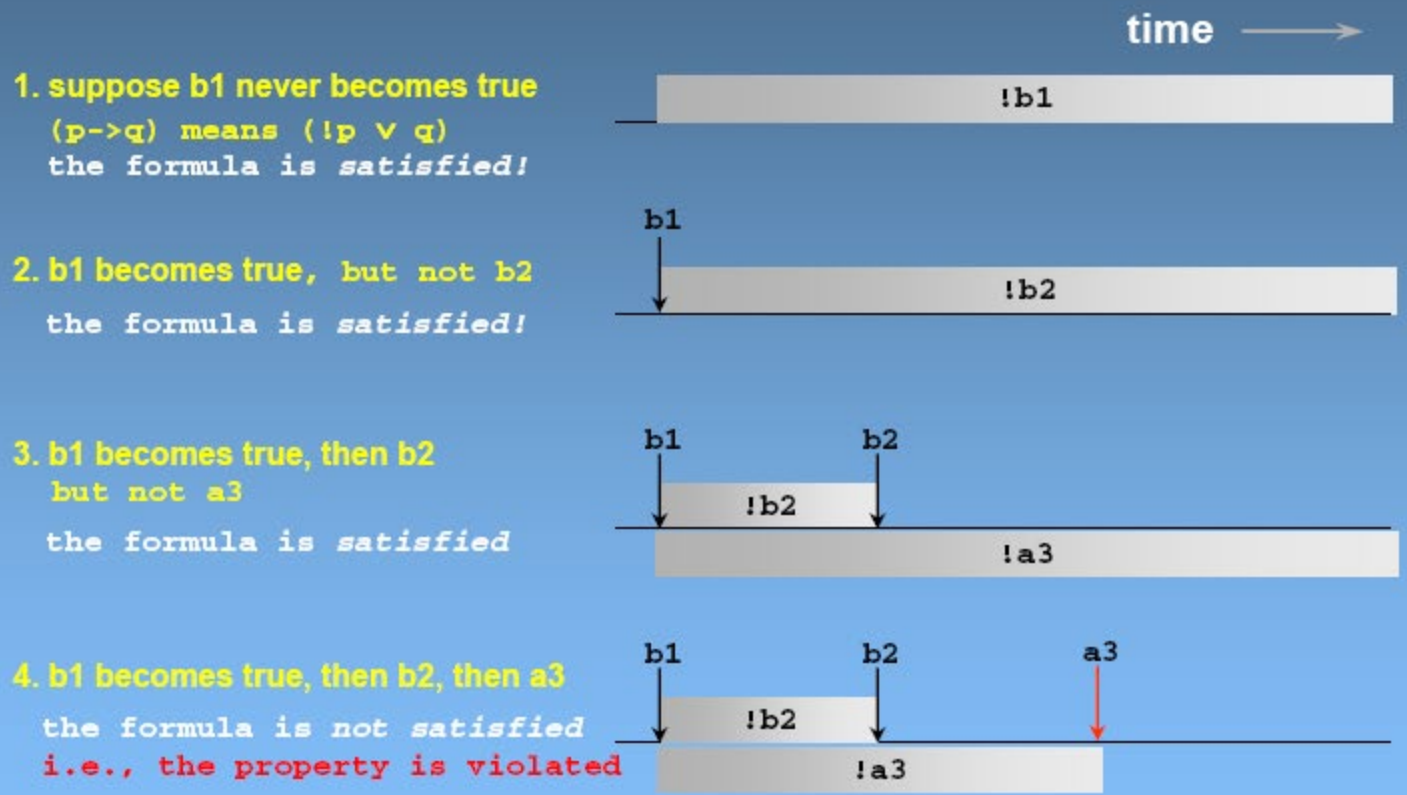
visualizing LTL formulae



slide quoted from Caltech 101b.2 "Logic Model Checking" by Dr.G.Holzmann

interpreting formulae...

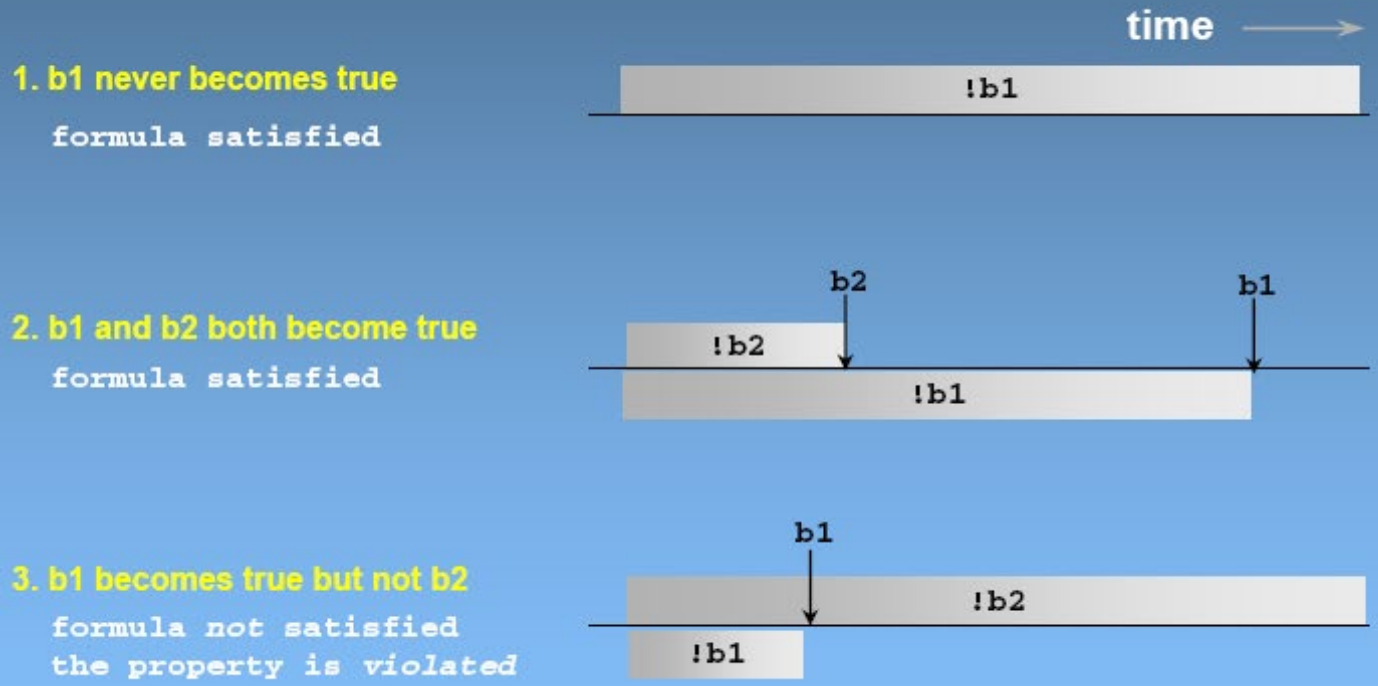
LTL: (<>(b1 && (!b2 U b2))) -> []!a3



slide quoted from Caltech 101b.2 "Logic Model Checking" by Dr.G.Holzmann

another example

LTL: (<>b1) -> (<>b2)



slide quoted from Caltech 101b.2 "Logic Model Checking" by Dr.G.Holzmann

Equivalences between LTL formulas

- Def 3.9 $\Phi \equiv \psi$ if for all models $\mathcal{M}$ and all paths π in $\mathcal{M}$: $\pi \models \Phi$ iff $\pi \models \psi$
- $\neg G\ \Phi \equiv F\ \neg\Phi$, $\neg F\ \Phi \equiv G\ \neg\Phi$, $\neg X\ \Phi \equiv X\ \neg\Phi$
- $\neg\ (\Phi\ U\ \psi) \equiv \neg\Phi\ R\ \neg\psi$, $\neg(\Phi\ R\ \psi) \equiv \neg\Phi\ U\ \neg\psi$
- $F\ (\Phi \vee \psi) \equiv F\ \Phi \vee F\ \psi$
- $G\ (\Phi \wedge \psi) \equiv G\ \Phi \wedge G\ \psi$
- $F\ \Phi \equiv T\ U\ \Phi$, $G\ \Phi \equiv \bot R \Phi$
- $\Phi\ U\ \psi \equiv \Phi\ W\ \psi \wedge F\ \psi$
- $\Phi\ W\ \psi \equiv \Phi\ U\ \psi \vee G\ \Phi$
- $\Phi\ W\ \psi \equiv \psi\ R\ (\Phi \vee \psi)$
- $\Phi\ R\ \psi \equiv \psi\ W\ (\Phi \wedge \psi)$

Practical patterns of specification

- For any state, if a request occurs, then it will eventually be acknowledge
  - G(requested $\rightarrow$ F acknowledged)
- A certain process is enabled infinitely often on every computation path
  - G F enabled
- Whatever happens, a certain process will eventually be permanently deadlocked
  - F G deadlock
- If the process is enabled infinitely often, then it runs infinitely often
  - G F enabled $\rightarrow$ G F running
- An upwards traveling lift at the second floor does not change its direction when it has passengers wishing to go to the fifth floor
  - G (fllor2 $\wedge$ directionup $\wedge$ ButtonPressed5 $\rightarrow$ (directionup U floor5)
- It is impossible to get to a state where a system has started but is not ready
  - Φ = G $\neg$(started $\wedge$ $\neg$ready)
  - What is the meaning of (intuitive) negation of Φ ?
    - For every path, it is possible to get to such a state (started $\wedge$ $\neg$ready).
    - There exists a such path that gets to such a state.
      - we cannot express this meaning directly
- LTL has limited expressive power
  - For example, LTL cannot express statements which assert the existence of a path
    - From any state s, there exists a path π starting from s to get to a restart state
    - The lift can remain idle on the third floor with its doors closed
  - Computation Tree Logic (CTL) has operators for quantifying over paths and can express these properties

Summary of practical patterns

G p	always p	invariance
F p	eventually p	guarantee
$p \rightarrow (F\ q)$	p implies eventually q	response
$p \rightarrow (q\ U\ r)$	p implies q until r	precedence
G F p	always, eventually p	recurrence (progress)
F G p	eventually, always p	stability (non-progress)
$F\ p \rightarrow F\ q$	eventually p implies eventually q	correlation