Program correctness
Using temporal logics

Marcello Bonsangue
LTL equivalences

- **De Morgan-based**
  - $\neg F\phi \equiv G\neg \phi$
  - $\neg X\phi \equiv X\neg \phi$

  X-self duality: on a path each state has a unique successor

- **Until reduction**
  - $F\phi \equiv T U \phi$
  - $F\phi \equiv T U \phi$
LTL: Adequate sets of connectives

**Theorem**: The set of operators $T, \neg, \land, U, X$ is adequate for LTL.

- $\square \phi U \psi \equiv \neg (E[\neg \psi U (\neg \phi \land \neg \psi)] \lor AG \neg \psi)$
- $\square \phi R \psi \equiv \neg (\neg \phi U \neg \psi)$
- $\square \phi W \psi \equiv \psi R(\phi \lor \psi)$
Other LTL equivalences

- \( \mathbf{G} \phi \equiv \phi \land X \mathbf{G} \phi \)
- \( \mathbf{F} \phi \equiv \phi \lor X \mathbf{F} \phi \)
- \( \phi \mathbf{U} \psi \equiv \psi \lor (\phi \land X \phi \mathbf{U} \psi) \)

**Theorem:** \( \phi \mathbf{U} \psi \equiv \neg (\neg \neg \neg \neg \psi \mathbf{U} (\neg \phi \land \neg \psi)) \land \mathbf{F} \psi \)
Verification goals

- Formulating properties requires some expertise
- Today we present categories of fundamental properties commonly used for system verification
  - reachability properties
  - safety properties
  - liveness properties
  - fairness properties
Reachability

- A **reachability** property states that some particular situation can be reached
  - **Simple**
    - “We can obtain $n < 0$”
    - “We can enter a critical section”
  - **Conditional**
    - “We can enter a critical section without traversing $n = 0$”
  - **Any**
    - “we can always return to the initial state”
Reachability in LTL

- LTL misses the existential quantifier on paths, thus it can only express reachability negatively:
  
  something is not reachable

- Simple reachability
  
  $\square \neg G(n \geq 0)$
  $\square \neg G(\text{no\_critic\_sec})$
Safety

- A safety property states that, under certain conditions, an event never occurs
  - “Two processes will never be both in their critical section”
  - “A memory overflow will never occur”

- In general, safety statements express that an undesirable event will not occur.
- The negation of a reachability property is a safety property (and the other way around)
Safety in LTL

- Typically expressed by the combinator G in LTL
- Examples
  - $G(\neg\text{critic\_sec}_1 \land \neg\text{critic\_sec}_2)$
  - $G(\neg\text{overflow})$

- Conditional safety
  - “As long the key is not in, the car won’t start”
    - $\neg\text{start} \mathcal{W} \text{key}$
    - $\neg\text{start} \mathcal{U} \text{key}$ as we are not required to have the key in some day
Liveness

- A **liveness** property states that, under certain conditions, an event will ultimately occur
  - “Any request will be satisfied”
  - “The light will turn green”
  - “after the rain, the sunshine”

- **Liveness is not reachability**
  - “The light will turn green (some day, regardless of the system behavior)”
  - vs.
  - “It is possible for the light (some day) to turn green”
Liveness

- In general, liveness statements express that happy event will occur in the end

- Termination is a liveness property:
  - “The program will terminate”
Liveness in LTL

- Typically expressed by the combinator F

- Examples
  - $G(req \Rightarrow Fsat)$ in LTL

- In LTL $\phi_1 U \phi_2$ is a liveness property, whereas $\phi_1 W \phi_2$ is a safety property
Deadlock

- A **deadlock** property states that, the system can never be in a situation in which no progress is possible.

- Safety? Liveness?
  - Deadlock freeness in LTL
    - $\text{GX T}$
    - whatever state may be reached (G) there exists an immediate successor state (X T)
Fairness

- A *fairness* property states that, under certain conditions, an event will occur (or will fail to occur) infinitely often

- “If access to a critical section is infinitely often requested, then access will be granted infinitely often"
Fairness in LTL

- Typically expressed by the combinators
  - GF (infinitely often)
  - FG (eventually always)

- Examples
  - GF critic_in ∨ FG¬ critic_req