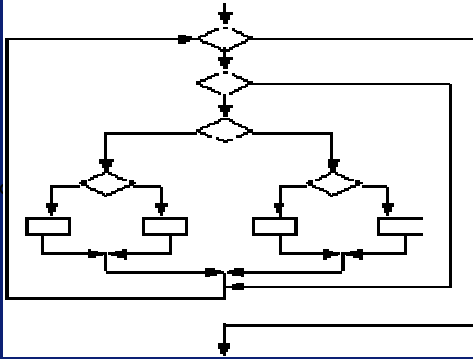


Spring 2009



Program correctness

Using temporal logics

Marcello Bonsangue



Leiden Institute of Advanced Computer Science
Research & Education

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

$$\top, \neg, \wedge, U, X$$

is adequate for LTL.

- $\phi U \psi \equiv \neg(E[\neg\psi U(\neg\phi \wedge \neg\psi)] \vee AG \neg\psi)$

- $\phi R \psi \equiv \neg(\neg\phi U \neg\psi)$

- $\phi W \psi \equiv \psi R(\phi \vee \psi)$



Other LTL equivalences

- $G\phi \equiv \phi \wedge XG\phi$
- $F\phi \equiv \phi \vee XF\phi$
- $\phi U \psi \equiv \psi \vee (\phi \wedge X\phi U \psi)$
- Theorem: $\phi U \psi \equiv \neg(\neg\psi U (\neg\phi \wedge \neg\psi)) \wedge F\psi$



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

- $\neg G(n \geq 0)$

- $\neg G(\text{no_critic_sec})$



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
 - $G(\neg \text{critic_sec}_1 \wedge \neg \text{critic_sec}_2)$
 - $G(\neg \text{overflow})$
- Conditional safety

“As long the key is not in, the car won’t start”

 - $\neg \text{start} \text{ W key}$
 - $\neg \text{start} \text{ U key}$ 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
 - $G(\text{req} \Rightarrow F\text{sat})$ in LTL
- In LTL $\phi_1 \mathbf{U} \phi_2$ is a liveness property, whereas $\phi_1 \mathbf{W} \phi_2$ 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~~

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
 - $GF \text{ critic_in} \vee FG \neg \text{critic_req}$

