

CS6910: Testing/Verification of Concurrent Programs

Temporal Logic
1996 Turing Award

Properties

How are properties specified?

formula: $AG (Step \rightarrow AF ! Frame)$

model \models formula

model: Verilog PCI design description

PCI Local Bus Controller

Safety Property

- Safety property
 - "Something bad must not happen"
 - E.g.: System should not crash
 - E.g.: Mutual exclusive use of a shared resource
 - Error trace is finite

Liveness Property

- Liveness property
 - "Something good must happen"
 - E.g.: Every packet sent must be received at its destination
 - E.g.: A bus resource allocator will eventually grant the use of the bus
 - Error trace is infinite

Propositional Logic

- Used to reason about static situations
- Formulas are built using atomic propositions and propositional operators
 - Atomic proposition $p \in AP$
 - Negation $\neg p$
 - Conjunction $p \wedge q$
 - Disjunction $p \vee q$
 - Implication $p \rightarrow q$

Problem with Propositional Logic


- Propositional logic is good for describing "static" situations
 - P holds only in $s0$ and $s1$
- How to describe dynamic behaviors such as:
 - Will q eventually happen?
 - Will p always happen?
- Dynamic behavior is important
 - Security protocols
 - Hardware/software
 - Operating systems, ...

Temporal Logic


- Originates from philosophy
- Used to reason about properties with a qualitative notion of time
- Formulas are built using
 - Standard propositional operators such as \neg, \wedge, \vee
 - Temporal operators such as
 - always
 - eventually
 - next-time
 - until

Atomic State Properties

- Boolean formula over state variables




req




req \wedge \neg ack

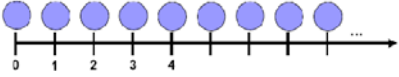
Temporal Operator "Always"

- $G p$: G stands for globally or always p
- $G p$ is true for a path if p holds at all states (points of time) along the path p


$p =$ 

$\neg p =$ 

$G p$





$\neg G p$



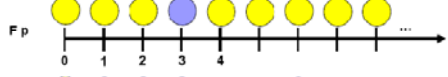
Temporal Operator "Eventually"

- $F p$ stands for eventually p
- $F p$ is true for a path if p holds at some state (point in time) along the path p


$p =$ 

$\neg p =$ 

$F p$





$\neg F p$
 $(G \neg p)$



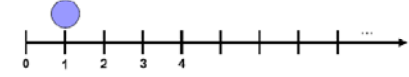
Temporal Operator "Next"

- $X p$ stands for next p
- $X p$ is true for a path if p holds at the next state (point in time) along the path p

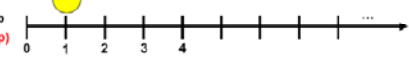
$p =$ 

$\neg p =$ 

$X p$





$\neg X p$
 $(X \neg p)$





Temporal Operator "Until"

- $p U q$ stands for p until q
- $p U q$ is true for a path if q holds at some state along the path, and p is true in all states before that state


$p =$ 

$\neg p =$ 

$q =$ 

$\neg q =$ 

$p U q$



Temporal Operators and Relationships

- The temporal operators G , F , X and U express properties along single computation paths.
- Can you express $G p$ purely in terms of F , p and propositional operators?
- Can you express $F p$ in terms of U and propositional operators?

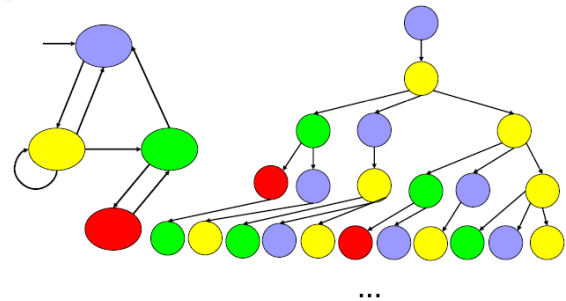
Examples in Temporal Logic

- "No more than one processor (in a 2-processor system) should have a cache line in write mode"
 - $wr1 / wr2$ are respectively true if processor 1 / 2 has the line in write mode
- "The grant signal must be asserted at some time after the request signal is asserted"
 - Signals: grant, req
- "A request signal must receive an acknowledge and the request should stay asserted until the acknowledge signal is received"
 - Signals: req, ack

Path Quantifiers in Temporal Logic

- Using the temporal operators so far we can only express properties over a single computation path
 - Linear Temporal Logic (LTL)
- "Path quantifiers" allow us to reason over a tree of possible executions
 - Computation Tree Logic (CTL)

Infinite Computation Tree



Path Quantifiers

- Two additional operators: A (all) and E (exists)
- Corresponding properties hold in states (not paths)
- $A p$: Property p holds along all computation paths starting from the state in which $A p$ holds
- $E p$: Property p holds along at least one path starting from the state in which $E p$ holds

Specifying Safety Properties

- $A G p$ stands for p holds globally on all paths
- $A G p$ is true for a state if property p holds globally along all computation paths starting from the state

Safety property:

Nothing bad will happen

Example: Mutual exclusion

Formula: $AG \neg(p1_lock \wedge p2_lock)$

$p1$ and $p2$ cannot be in the lock



state simultaneously.

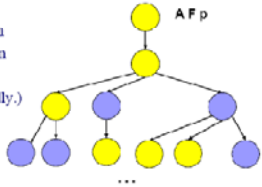


Specifying Liveness Properties

- AFp stands for p holds eventually on all paths
- AFp is true for a state if property p holds eventually along all computation paths starting from the state

Liveness property:
 Something good will happen
 Example: resource allocation
 Formula: $AF \text{ bus_grant}$
 (The bus is granted eventually.)

$p =$  $\neg p =$ 



Temporal Logic Examples

- "From any state it is possible to get to the reset state along some path"
 - Signal: reset
- "For any state, it must hold that the grant signal always be asserted some time after the request signal was asserted"
 - Signals: grant, req

Summary: Specifying Properties

Temporal Logic formulas (CTL) =

$p \in A, I$





$\neg f \mid (f \wedge g) \mid (f \vee g) \mid$

$EX f \mid EF f \mid EG f \mid E(f U g) \mid$

$AX f \mid AF f \mid AG f \mid A(f U g)$

Types of Correctness Properties:

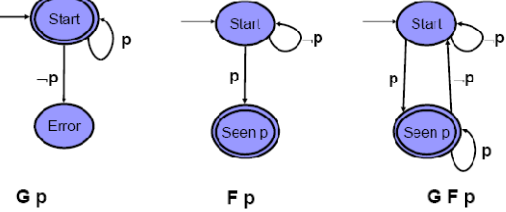
- **Safety properties:** nothing bad happens, invariants
 - e.g. no deadlock occurs
- **Liveness properties:** something good happens, progress
 - e.g. every request is eventually acknowledged
- **Precedence properties:** fixed ordering of events
 - e.g. service is done in order of requests received
- **Fairness properties:** assumptions on processes, scheduler
 - e.g. impartiality, justice, strong fairness

From Temporal Logic (LTL) to Monitors

- A monitor for a linear temporal logic formula
 - is a finite state machine
 - accepts exactly those behaviors that satisfy the temporal logic formula
 - "Accepts" means that the accepting state is visited infinitely often
 - Reason: Behaviors or traces of systems can have infinite length
 - Such automata are called Büchi automata
 - is often used to specify properties instead

Property Monitors



G p
F p
G F p

Specifying Correctness Properties (Summary)

- Temporal Logic: Useful for reasoning about behavior of an LTS using a qualitative notion of time
- Formulas are formed by using:
 - Standard Boolean operators (and &, or +, not !)
 - Temporal operators (always, eventually, next-time, until)
- Formulas are interpreted on sequences/trees over time
 - Linear Temporal Logic (LTL): Infinite sequence is considered, starting from the initial state
 - Computation Tree Logic (CTL): Infinite tree of computations is considered, starting from the initial state