CS4271
Critical Systems & their Verification

Review Questions and Revision Exercises
Abhik Roychoudhury
Topic covered: Linear Time Temporal Logic

Feb 2 , 2012




## Warm-up Exercise on LTL

- i) Express the G operator in terms of the F operator.
- ii) Express the F operator in terms of the U operator.
  - Feel free to use any of the propositional logic operators.

- $G\,\varphi = \neg F \neg \varphi$
- $F\varphi = (true\ U\ \varphi)$




## Exercise – (1)

- The light is *always* green.
  - **G green**
- *Whenever* the light is red, it *eventually* becomes green.
  - **G( red ⇒ F green )**
- *Whenever* the light is green, it remains green *until* it becomes yellow.
  - **G(green ⇒ (green U yellow))**
- *Whenever* the light is yellow, it becomes red *immediately after*.
  - **G( yellow ⇒ X red)**
- Encode these properties in LTL.




## Exercise – (2)

- Check whether the four LTL properties in the previous slide are satisfied by our simple controller.


Traces $s0^{\omega}$, $s0^{*}s1s2^{\omega}$, $s0^{*}s1s2^{*}s0^{\omega}$ ....

**G green** Not satisfied, why?
**G(red ⇒ F green )** Not satisfied, why?
**G(green ⇒ (green U yellow))** Not satisfied, why?
**G( yellow ⇒ X red)** Satisfied, why?




## LTL Exercise – (3)

Consider a resource allocation protocol where n processes $P_1,\ldots,P_n$ are contending for exclusive access of a shared resource. Access to the shared resource is controlled by an arbiter process. The atomic proposition $req_i$ is true only when $P_i$ explicitly sends an access request to the arbiter. The atomic proposition $gnt_i$ is true only when the arbiter grants access to $P_i$. Now suppose that the following LTL formula holds for our resource allocation protocol.

- $G\,(req_i \Rightarrow F\,gnt_i)$




## LTL Exercise – (3)

- i) Explain in English what the property means.
- ii) Is this a desirable property of the protocol ?
- iii) Suppose that the resource allocation protocol has a distributed implementation so that each process is implemented in a different site. Does the LTL property affect the communication overheads among the processes in any way ?

## Answer

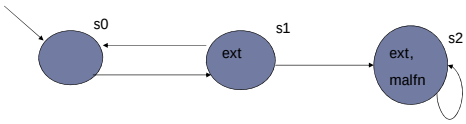
- i) Whenever a request is made by a process, it is eventually granted.
- ii) Yes, this is a desirable property since it avoids starvation of issued requests.
- iii) Communication overheads depend on whether the request is communicated only once, or it is held high until the request is granted. Consider the alternative property
  - $G(req_i \; U \; gnt_i)$




## LTL Exercise (4)

i) What are the traces of this spring model ?

ii) Does this model satisfy the formula "The spring always remain extended" ? How to write it in LTL and check its correctness in this model ?



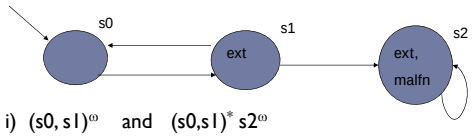

iii) What do the properties FG ext and GF ext denote ?

iv) Are they true in this spring model ? If not how many traces violating each property can you find ?

v) What about the property ¬ ext U malfn ?




## Answer:




i) $(s0, s1)^{\omega}$ and $(s0,s1)^{*}\, s2^{\omega}$

ii) *Gext*, none of the traces actually satisfy it.

iii) *FGext* --- the spring eventually remains extended forever
*GFext* --- the spring is extended infinitely often.

iv) *FGext* is false, since $(s0, s1)^{\omega}$ does not satisfy the property
*GFext* is true in the spring model.

v) $\neg$ *ext U malfn* is false, since $(s0, s1)^{\omega}$ does not satisfy the property




## LTL Exercise – (5)

- Specify the following as LTL properties. *p, q, r are atomic propositions.*
  - *(a) p is false until r occurs, but r may not occur at all.*
  - (b) *Always, after q occurs p must remain false.*

- **Answer:**
  - $G\neg r \vee (\neg pUr)$
  - $G(q \Rightarrow G(\neg p))$




## LTL Exercise – (6)

- Show that the following pairs of temporal logic formula are not equivalent. You should construct an example system model which satisfies one of them but not the other. You may assume that p,q are atomic propositions.
- i) FG p and GF p
- ii) (true U p) and (p U true)
- iii) (p U q) and (q R p)

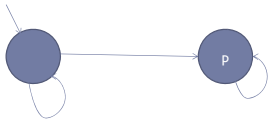



## Answers (6-i)

- Satisfies GFp but does not satisfy FGp

## Answers – (6-ii)

- trueUp is the LTL formula Fp and pUtrue is the formula true. So, any system model not satisfying Fp is an example







## Answers – (6-iii)

- qRp is true even if q never holds. But pUq cannot be true if q never holds. So, the following system model satisfies qRp but does not satisfy pUq, since it is possible to loop at the initial state and thereby q never holds.