CSC410 Tutorial

Safety, Liveness, and Fairness

Yi Li

Nov 20, 2017
LTL in System Verification

• Three types of properties
  • Safety:
    • “something bad will not happen”
    • e.g., “the program will never produce a wrong result” (partial correctness)
  • Liveness:
    • “something good will happen” (but we don’t know when)
    • e.g., “the program will produce a result” (termination)
  • Fairness:
    • often used to model schedulers
    • e.g., “a scheduler never ignores some process forever”
Safety Properties

Safety:
“something bad will not happen”

Typical examples:

\[ \square \neg (reactor\_temp > 1000) \]
\[ \square \neg ((x = 0) \land \bigcirc \bigcirc \bigcirc (y = z/x)) \]
and so on.....

Usually: \[ \square \neg .... \]
Liveness Properties

Liveness:
“something good will happen”

Typical examples:

◊ rich
◊ \((x > 5)\)
□ \((start \Rightarrow \diamond terminate)\)
□ \((Trying \Rightarrow \diamond Critical)\)

and so on.....

Usually: ◊ ....
Fairness Properties

Often only really useful when scheduling processes, responding to messages, etc.

Strong Fairness:

“if something is attempted/requested infinitely often, then it will be successful/allocated infinitely often”

Typical example:

\[ \square \Diamond \text{ready} \Rightarrow \square \Diamond \text{run} \]
Safety vs. liveness for state-transition graphs

Safety: those properties whose violation always has a finite witness
("if something bad happens on an infinite run, then it happens already on some finite prefix")

Liveness: those properties whose violation never has a finite witness
("no matter what happens along a finite run, something good could still happen later")
Safety: the properties that can be checked on finite executions

Liveness: the properties that cannot be checked on finite executions

(they need to be checked on infinite executions)

This is much easier.
Categorization of LTL properties

- **Safety**: nothing bad should happen.
  
  \[ \Phi = \neg \text{crit}_1 \lor \neg \text{crit}_2 \]

- **Invariants**: a condition $\Phi$ holds for all reachable states.

- **Safety Properties**: any infinite path does not have a bad finite prefix.
  
  \[ \Box (\text{red} \rightarrow \neg \Diamond \text{green}) \]

- **Liveness**: something good will happen in future.
  
  Any finite prefix can be extended to a trace that satisfies the property.
  
  \[ \Box (\text{request} \rightarrow \Diamond \text{response}) \]
For any LTL property $P$ over $AP$ there exists a safety property $P_{\text{safe}}$ and a liveness property $P_{\text{live}}$ (both over $AP$) such that,

$$P = P_{\text{safe}} \cap P_{\text{live}}$$
Fairness

unconditional LTL fairness:

\[ \square \diamond \Psi \]

every process gets its turn infinitely often.

strong LTL fairness:

\[ \square \diamond \Phi \quad \rightarrow \quad \square \diamond \Psi \]

every process that is enabled infinitely often gets its turn infinitely often.

weak LTL fairness:

\[ \diamond \square \Phi \quad \rightarrow \quad \square \diamond \Psi \]

every process that is continuously enabled from a certain time instant on gets its turn infinitely often.

Fair satisfaction relation: \[ TS \models_{\mathcal{F}} \phi \]
Extended Example: Fairness

```plaintext
global nat s, t
local nat m
while(true):
    m=t++  // Acquire a ticket
    while(m>s):
        busy wait
        skip
    // Critical section
    s++  // Exit critical
```
Example 1: mutual exclusion (safety)

\[ \mathcal{KM} \models \square \neg (C_1 \land C_2) \]
Example 1: mutual exclusion (safety)

\( N = \text{noncritical}, \ T = \text{trying}, \ C = \text{critical} \)

\( N_1, N_2 \) turn=0

User 1

User 2

N1, T2 turn=2

T1, C2 turn=2

KM = \[\square \neg (C_1 \land C_2) \]

YES: There is no reachable state in which \((C_1 \land C_2)\) holds!
Example 2: mutual exclusion (liveness)

\[\mathcal{KM} \models \diamond C_1 \]
Example 2: mutual exclusion (liveness)

N = noncritical,  T = trying,  C = critical

User 1  User 2

KM $\models \Diamond C_1$ ?

NO: the blue cyclic path is a counterexample!
Example 3: mutual exclusion (liveness)

\( K\mathcal{M} \models \Box (T_1 \Rightarrow \Diamond C_1) \)

\(N = \text{noncritical}, \ T = \text{trying}, \ C = \text{critical}\)

User 1

User 2
Example 3: mutual exclusion (liveness)

N = noncritical,  T = trying,  C = critical

KM ⊨ □(T₁ ⇒ ◇C₁)?

YES: in every path if T₁ holds afterwards C₁ holds!
Example 4: mutual exclusion (fairness)

\[ \mathcal{K} \models \square \diamond C_1 \]
Example 4: mutual exclusion (fairness)

\[ \mathcal{KM} \models \square \diamondsuit C_1 \]

NO: the blue cyclic path is a counterexample!
Example 4: mutual exclusion (strong fairness)

N = noncritical, T = trying, C = critical

User 1  User 2

\[ K_M \models \square \Diamond T_1 \Rightarrow \square \Diamond C_1 \]
Example 4: mutual exclusion (strong fairness)

\[ \mathcal{KM} \models \square \Diamond T_1 \Rightarrow \square \Diamond C_1 \]?

**YES:** every path which visits \( T_1 \) infinitely often also visits \( C_1 \) infinitely often!
References

- https://www.cs.uoregon.edu/research/summerschool/summer06/lectures/Qadeer063.pdf