Linear Temporal Logic
Safety requirements vs liveness requirements

**Safety**: nothing bad *ever* happens

**Liveness**: something good *eventually* happens

Means system is functioning as intended

System requirements are often liveness requirements
What are some liveness requirements for the AC?
How would you **monitor** that a liveness requirement is fulfilled?
Verifying some liveness properties

Saying something *eventually* happens is the same thing as saying that it is *not* the case that it always *doesn’t* happen.

Can we use invariant verification to check this?
Linear Temporal Logic (LTL)

Assume you have *some* execution trace

LTL operators are propositional logic operators PLUS:

- G (globally/always)
- F (eventually/finally)
- X (next state)
- U (until)
FG vs G, GF vs F, FG vs GF

e repeats an infinite number of times for an infinite trace
LTL examples on FSMs

Safety property is an *invariant* if property $p$ holds for all reachable states of $S$.

Liveness property *holds* for $S$ if it holds for all possible traces of $S$.

Lee/Seshia Chapter 13, exercise 2
LTL means we can specify liveness properties with $F$. Can we specify safety properties more easily with LTL, too?
Safety properties with LTL

Use “G” to say a property holds for every state.
Can use “X” to express statefulness/history without a monitor state machine.
Limits of LTL

\[ G(\text{even}(x) \rightarrow ((X-\text{even}(x)) \land (XX\text{even}(x))) \]

But if you don’t know if you started a sequence with an odd number or even number, you cannot write

\[ (\text{even}(x) \land X-\text{even}(x)) \]
Automated model checking and LTL

These are covered more deeply in Alur’s textbook

If interested: take CS1710!

**Buchi Automata:** automata that “accept” a certain LTL formula

Can be automatically constructed

Using nested DFS, show repeatability for negation of LTL formula holds
More verification techniques

Automated verification

Symbolic model checking: represents a set of states symbolically as a logic formula and does symbolic (algebraic) computation

What about timed/hybrid automata?

Symbolic reachability analysis for linear hybrid automata (special class of HA)
Symbolic model checking for a different kind of logic (signal temporal logic)
Assisted proof engines (differential dynamic logic)
An active area of research!
Summary: pros/cons of verification?