4/20 - LTL

http://cs.brown.edu/courses/cs195y/2020/pages/pdf/LTL.pdf

Recall that LTL has the following:

- A sequence of states where certain things are true or false
- Deals with an *infinite* repetition of states
    - Not finite/arbitrary number of times
    - To convert finite traces to infinite ones, we can just repeat the last state forever

Then, we can talk about certain traces and see if they satisfy LTL formulas:

- Formulas that are evaluated over traces
    - Talk about “always” (G, globally)
    - Talk about “eventually” (F, finally)
- We can also have formulas that are evaluated over a state
- To get the next state, we essentially just drop the first state
    - Evaluate internal formula on the new first state
- We can deal with just one operator at a time!

However, things like *availability* are not expressible in LTL:

- Things of the form “there exists a trace where…”
- E.g. I want the brakes to always be available, not necessarily that you’re braking all the time
    - Exists option of a feature

How can we enrich LTL to express these properties:

- We can add path quantification (A, E)
- This is called CTL - Computation Tree Logic
    - Also a temporal logic
- The more expressive the logic, the harder it is to model-check with that language
    - That’s why many model checkers stick with a limited logic