CTL exercises

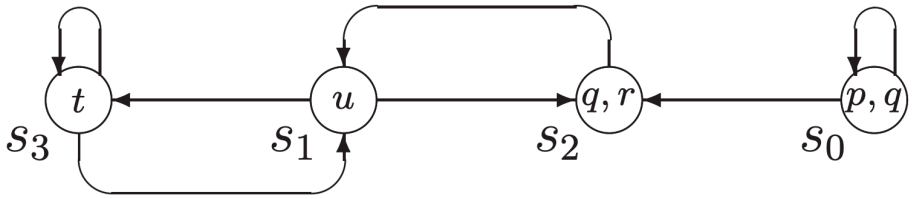
Definitions

1. We want to work on computation trees, which model paths through states in the system. For example:
2. We typically say that we have a model $\mathcal{M}$ that represents our program (e.g. a control flow graph or state machine), and want to prove that some property holds in a state s.
3. We have the following operators (in addition to propositional operators $\vee, \wedge, \neg, \implies$):
   a. Path quantifiers:
      i. **A** "for every path" (compare to $\forall$ in propositional logic/set theory)
      ii. **E** "there exists a path" (compare to $\exists$ in propositional logic/set theory)
   b. Linear-time operators:
      i. Xp — p holds **next** time.
      ii. Fp — p holds sometime in the **future**
      iii. Gp — p holds **globally** in the future
      iv. pUq — p holds **until** q holds

Exercises

Translate these into English (natural language) requirements statements.

1. EF$(Started \wedge \neg Ready)$
2. AG$(Request \implies$ AF$Ack)$
3. AG(AF$DeviceEnabled)$
4. AG(EF$Restart)$
5. Bonus: another thing we want to be able to do is look at a Computation Tree and see if the given expression holds. The image below represents a tree $\mathcal{M}$.



For the CTL expression $\mathcal{M}, s \models \mathsf{EF}(\neg r \wedge \mathsf{EX} r)$

- Translate this expression into the English semantics. Recall that $\mathcal{M}$ is our Model of the computation tree, s represents a state in the model provided, and r is some future state we want to model (e.g., a null pointer).
- create a set S which holds all the states where the expression is true.