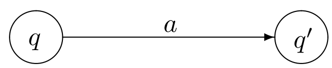
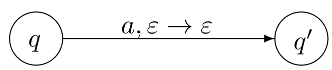
Finite Automata (FA) $\to$ Context-Free Grammars (CFG)

FA $\to$ PDA. It is easy to transform a finite automaton to a pushdown automaton: all we need to do is to push anything and not to pop anything. In other word, every transition of the finite automaton

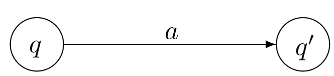


is transformed into:



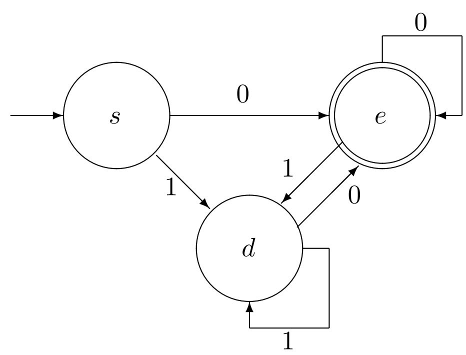
FA $\to$ CFG: algorithm. Let us show how to transform a finite automaton (FA) into a context-free grammar (CFG).

- To each state q of the FA, introduce a new variable Q.
- The variable corresponding to the starting state will be the starting variable of the new CFG.
- For each transition of the finite automaton

  we add a rule $Q \to aQ'$.
- For each final state f of the FA, we add a rule $F \to \varepsilon$.

FA $\to$ CFG: example. Let us consider the FA for recognizing even unsigned integers:



By applying the general algorithm to this FA, we get a CFG with the starting variable S and the following rules:

$$S \to 0E$$

$$S \to 1D$$

$$E \to 0E$$

$$E \to 1D$$

$$D \to 0E$$

$$D \to 1D$$

$$E \to \varepsilon$$

How to derive words in the resulting grammar. Derivations in this grammar follow, step-by-step, the way the original finite automaton accepts a word. Let us illustrate it on the example of the word 10 accepted by the original finite automaton:

- we start in the start state s; this corresponds to the starting variable S;
- then, we use the fact that once we are in the state s and we see the symbol 1, then we go the state d; this transition corresponds to the rule $S \to 1D$, so the generation so far is:
$$\underline{S} \to 1D;$$
- then, we use the fact that once we are in the state d and we see the symbol 0, then we go to the state e; this transition corresponds to the rule $D \to 0E$, so generation so far is
$$\underline{S} \to 1\underline{D} \to 10E;$$

- we have read all the symbols of the word, and we are in the final state; for the FA, this means that the word 10 is accepted; for CFG, we need to use the rule $E \to \varepsilon$ corresponding to the final state e; thus, we get the following derivation of the word 10:

$$\underline{S} \to 1\underline{D} \to 10\underline{E} \to 10.$$

So, we have indeed derived the word 10 in the grammar.

Practice. Try the same construction for different finite automata.