Pushdown Automata

A PDA adds memory to finite automata, in the form of a stack. Upon reading an input, we can operate as usual on the finite automata, taking transitions, but we can also push and pop a symbol from the stack as well. As we push a symbol to the stack (which may be infinite), it "pushes down" the other symbols on the stack, hence the term "pushdown" automata.

PDAs add power to finite automata, in fact, they yield the power of the context-free languages (see @75).

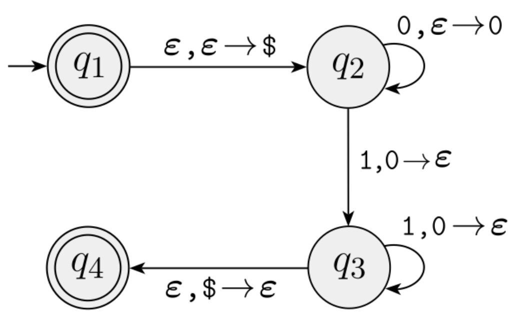
Key Definition (PDA): A push-down automata is a 6-tuple $(Q, \Sigma, \Gamma, \delta, q_0, F)$ such that:

1. Q is a **finite** set of states,
2. Σ is a **finite** set of symbols distinct from Q, i.e., $Q \cap \Sigma = \varnothing$ for the input alphabet,
3. Γ is a **finite** set of symbols for the stack alphabet,
4. $\delta : Q \times \Sigma_\epsilon \times \Gamma_\epsilon \to P(Q \times \Gamma_\epsilon)$ is the transition function,
5. $q_0 \in Q$ is the start state,
6. $F \subseteq Q$ is the set of final states (possibly empty).

Example: As noted in @75, $\{0^n 1^n\}$ is context-free because we have a grammar that produces it. I can also construct a PDA, which we can describe informally like so:

Construct an NFA immediately, without reading an input (i.e., ϵ), places $\$$ on the stack, to initialize the empty stack. The NFA proceeds reading the input, and upon each 0, pushes the 0 onto the stack. When the NFA reads the first 1, it begins to pop a zero, for each one it reads. When it is done reading in put, the NFA verifies that the only symbol remaining is $\$$, which means exactly as many 1's follow the 0's. Any other than the expected input forces a hang-state (so the machine will reject those).

Formally, the PDA can be written as a state machine like so (source: Sipser):



The way to read each transition is, for example, $0, \epsilon \to 0$, we say: "read input 0 and pop epsilon and push 0" (and go to state q2), alternatively some like to read the stack operation portion (e.g., $a \to b$) as a replacement (replace a with b).