Pushdown Automata

Mark Greenstreet, CpSc 421, Term 1, 2006/07

- Formal Definition of Pushdown Automata
- Equivalence of PDAs and CFGs

Formalizing Pushdown Automata

- A Pushdown automaton (PDA) is a 6-tuple
  - Q: a finite set of states;
  - Σ: the input alphabet, a finite set;
  - Γ: the stack alphabet, a finite set;
  - $\delta : Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \ \rightarrow \ 2^{Q \times (\Gamma \cup \{\epsilon\})}$, the transition relation;
  - $q_0 \in Q$: the start state; and
  - $F \subseteq Q$: the set of accepting states.

The Transition Relation

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \;\rightarrow\; 2^{Q \times (\Gamma \cup \{\epsilon\})}$$

- If $(q', g') \in \delta(q, c, g)$, then the PDA can read input symbol c while in state q with g on the top-of-the-stack; it then can move to state q' and replace g with g'.
- δ is a relation.
- c, g, or g' can be ϵ.

The Transition Relation

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \;\rightarrow\; 2^{Q \times (\Gamma \cup \{\epsilon\})}$$

- If $(q', g') \in \delta(q, c, g)$, then the PDA can read input symbol c while in state q with g on the top-of-the-stack; it then can move to state q' and replace g with g'.
- δ is a relation.
  - If there is more than one choice for (q', g') such that $(q', g') \in \delta(q, c, g)$, then the PDA makes a non-deterministic choice. As with NFAs, the PDA accepts a string if there is any way to make the non-deterministic choices to lead to an accepting state.
  - If $\delta(q, c, g) = \emptyset$, then the PDA rejects. As with an NFA, there may still be other non-determistic choices that lead to an accepting state.
- c, g, or g' can be ϵ.

The Transition Relation

$$\delta : Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \;\rightarrow\; 2^{Q \times (\Gamma \cup \{\epsilon\})}$$

- If $(q', g') \in \delta(q, c, g)$, then the PDA can read input symbol c while in state q with g on the top-of-the-stack; it then can move to state q' and replace g with g'.
- δ is a relation.
- c, g, or g' can be ϵ.
  - If c is ϵ, then the move doesn't consume any input.
  - If g is ϵ, then the move doesn't consume the the top-of-stack value. If $g = \epsilon$ and $g' \neq \epsilon$, we say that the machine pushes g' onto the stack. This is how the stack grows.
  - If g' is ϵ, then the move removes g from the stack without replacing it. We say that the machine pops g off of the stack. This is how the stack shrinks.
  - If neither g nor g' are ϵ, the move replaces g with g' as the top-of-stack symbol.

Configurations

A configuration is a 3-tuple, (q, s, γ) where

- $q \in Q$ denotes the current state of the PDA.
- $s \in \Sigma^*$ denotes the unread input.
- $\gamma \in \Gamma^*$ denotes the string of symbols on the stack. For example, $\gamma = XYYZ$ indicates that there are four symbols on the stack with X at the top-of-the-stack.

Moves

- Let $P = (Q, \Sigma, \Gamma, \delta, q_0, F)$ be a PDA.
- We write $(q, s, \gamma) \xrightarrow[P]{1} (q', s', \gamma')$ to indicate that P can move from configuration (q, s, γ) configuration (q', s', γ') in one step.
- We write $(q, s, \gamma) \xrightarrow[P]{*} (q', s', \gamma')$ iff P can move from configuration (q, s, γ) to configuration (q', s', γ') in zero or more steps.

Moves

- Let $P = (Q, \Sigma, \Gamma, \delta, q_0, F)$ be a PDA.
- We write $(q, s, \gamma) \xrightarrow[P]{1} (q', s', \gamma')$ to indicate that P can move from configuration (q, s, γ) configuration (q', s', γ') in one step. In particular. $(q, s, \gamma) \xrightarrow[P]{1} (q', s', \gamma')$ iff

$$\begin{array}{rl}
\exists c \in (\Sigma \cup \{\epsilon\}). & (c \text{ is the input symbol that } P \text{ reads, if any}) \\
\exists g \in (\Gamma \cup \{\epsilon\}). & (g \text{ is the top-of-stack symbol that } P \text{ reads, if any}) \\
\exists g' \in (\Gamma \cup \{\epsilon\}). & (g' \text{ is the new top-of-stack symbol that } P \text{ writes, if any}) \\
\exists \beta \in \Gamma^*. & (\beta \text{ is the rest of the string of symbols on the stack} \\
\multicolumn{2}{r}{(s = c \cdot s') \wedge (\gamma = g \cdot \beta) \wedge (\gamma' = g' \cdot \beta) \wedge ((q', g') \in \delta(q, c, g))}
\end{array}$$

- We write $(q, s, \gamma) \xrightarrow[P]{*} (q', s', \gamma')$ iff P can move from configuration (q, s, γ) to configuration (q', s', γ') in zero or more steps.

Moves

- Let $P = (Q, \Sigma, \Gamma, \delta, q_0, F)$ be a PDA.
- We write $(q, s, \gamma) \xrightarrow[P]{1} (q', s', \gamma')$ to indicate that P can move from configuration (q, s, γ) configuration (q', s', γ') in one step.
- We write $(q, s, \gamma) \xrightarrow[P]{*} (q', s', \gamma')$ iff P can move from configuration (q, s, γ) to configuration (q', s', γ') in zero or more steps.

$$
\begin{array}{ll}
 & (q, s, \gamma) \xrightarrow[P]{*} (q', s', \gamma') \\
\Leftrightarrow & \quad (q, s, \gamma) = (q', s', \gamma'), \\
\vee & \exists (q", s", \gamma"). \, ((q, s, \gamma) \xrightarrow[P]{*} (q", s", \gamma")) \wedge ((q", s", \gamma") \xrightarrow[P]{1} (q', s
\end{array}
$$

Acceptance

A PDA accepts a string iff it can reach an accepting state after reading the entire string:

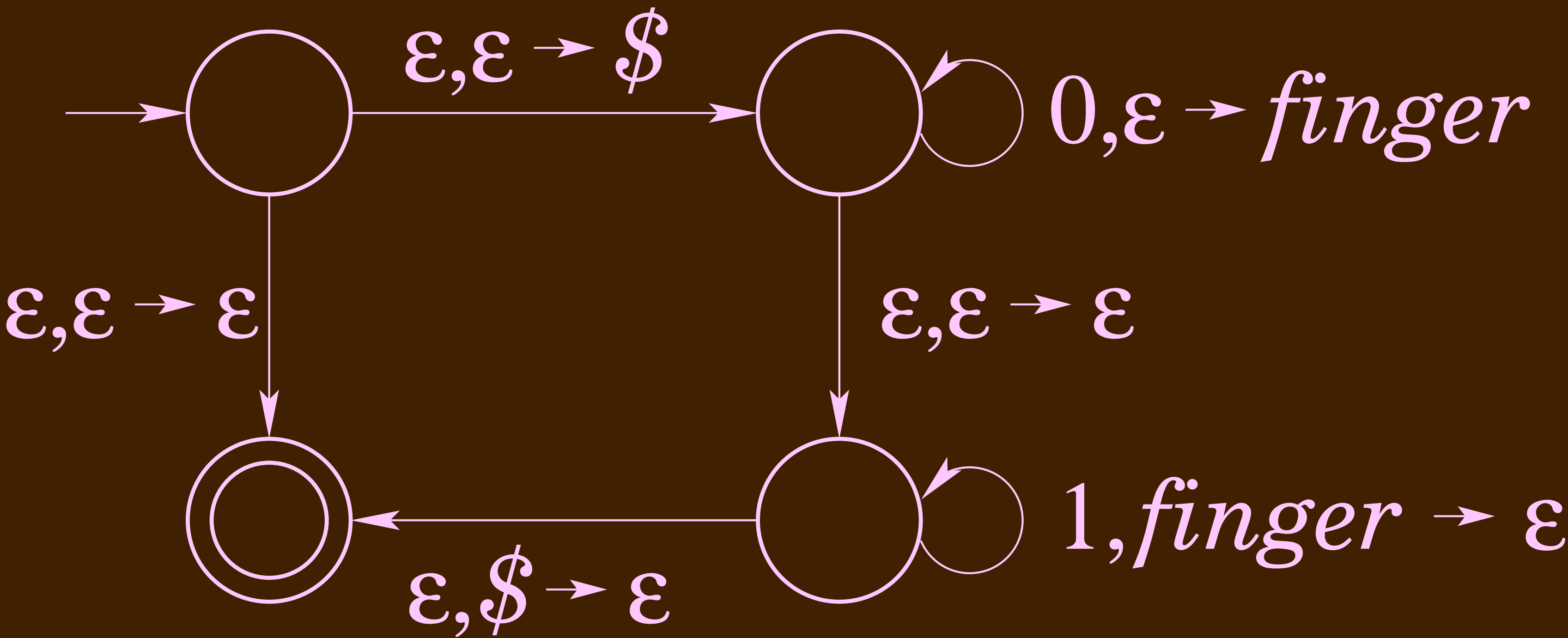
- $P = (Q, \Sigma, \Gamma, \delta, F)$ accepts w iff

$$(q_0, w, \epsilon) \xrightarrow[P]{*} (q', \epsilon, \gamma)$$

  For some $q' \in F$ and some $\gamma \in \Gamma^*$.

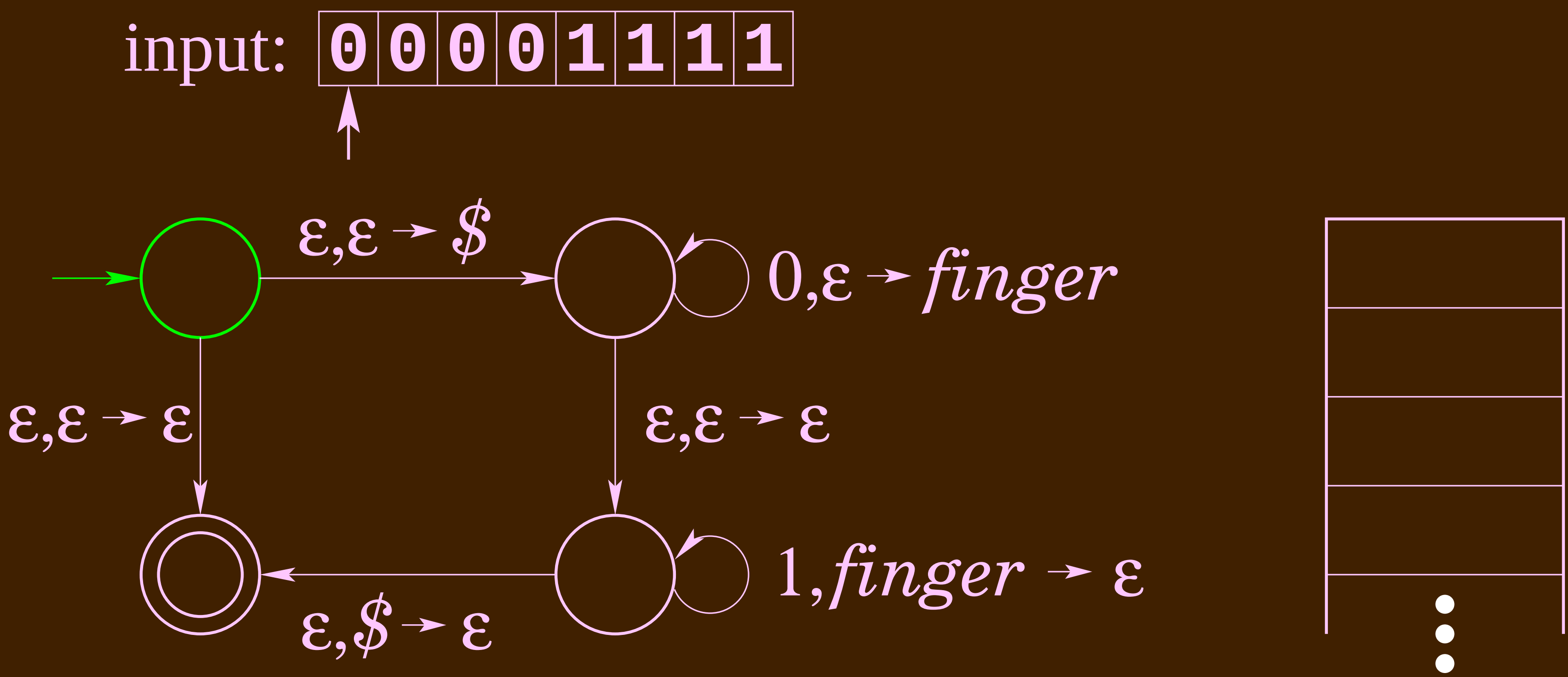
- The language recognized by P is the set of all strings that P accepts.

An Example

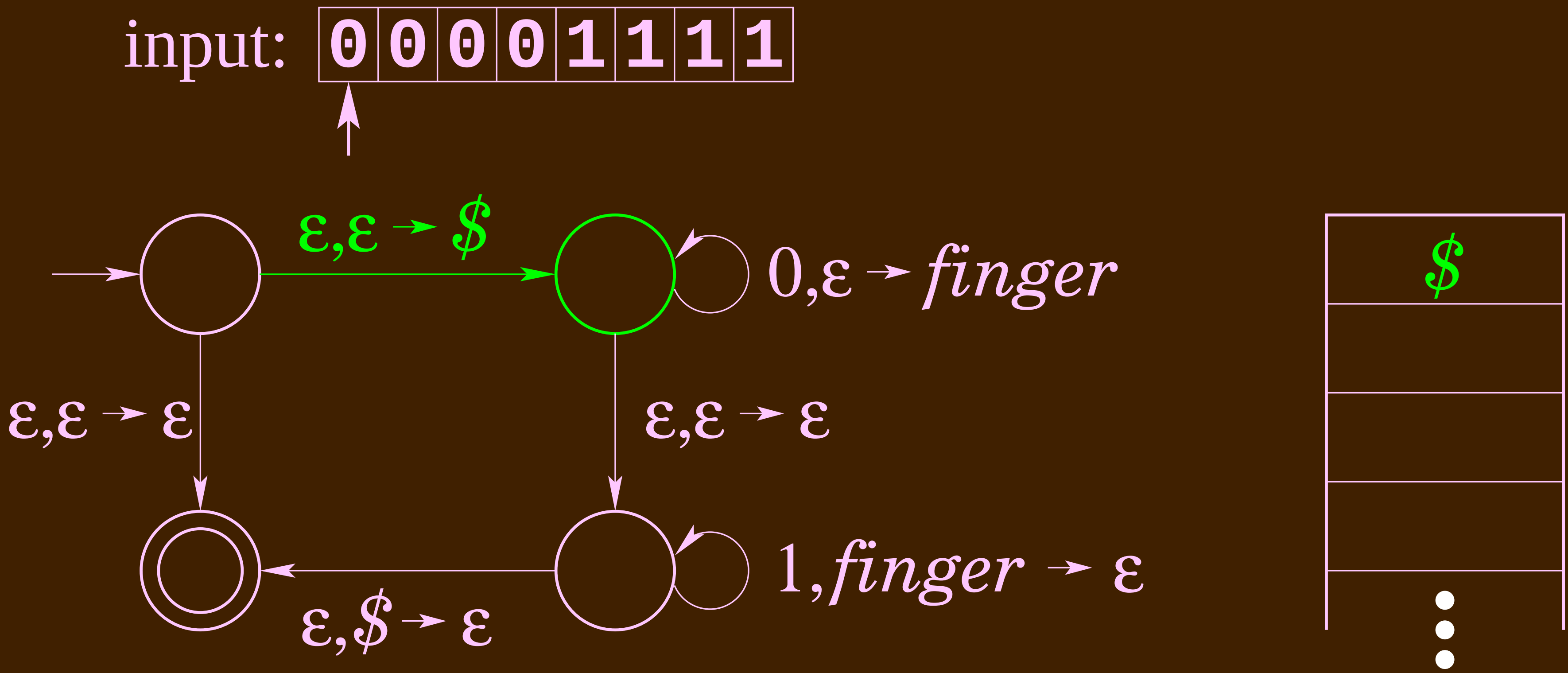


$\Sigma =$ **{0, 1}** $\qquad \Gamma = \{\$, \mathit{finger}\}$

An Example



An Example



An Example