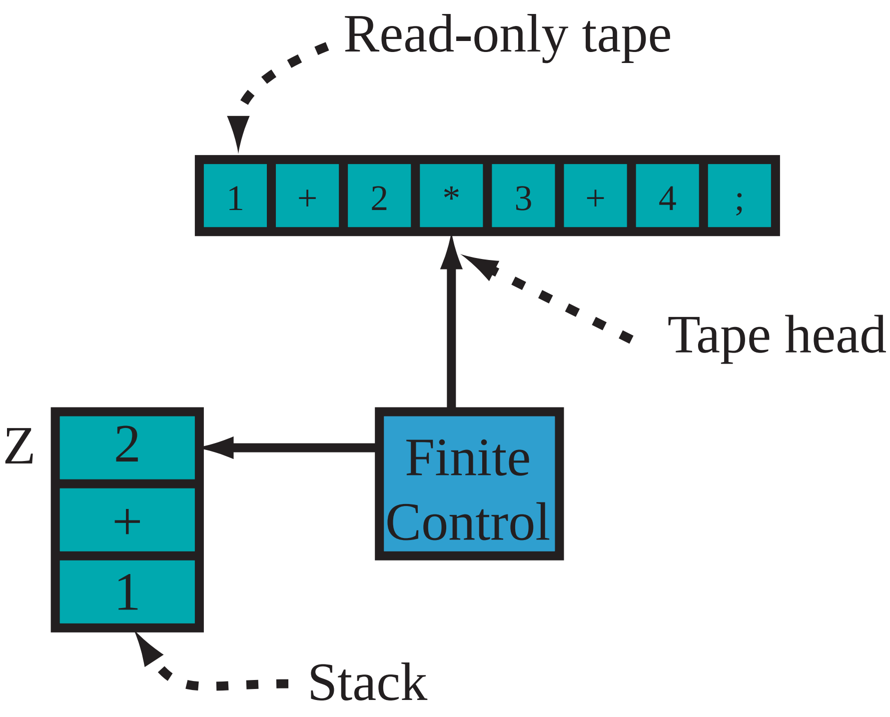
CS20a: summary (Oct 17, 2002)

- Context-free languages
  - *Grammars G = (V, T, P, S)*
  - *Grammar simplification*
    - *Eliminate epsilon productions (A -> epsilon)*
    - *Eliminate unit productions (A -> B)*
    - *Eliminate useless symbols*
  - *Normal forms*
    - *Chomsky*
    - *Griebach (not covered in lecture)*
- Next: pushdown automata
  - *N-PDA = CFG*
  - *D-PDA < CFG*

Pushdown automata

- A PDA has
  - *An* input tape
  - *A* finite control
  - *A* stack
- First, we'll consider nondeterministic PDAs
- Actions:
  - *Read: Examine input, move tape head right, and replace the top of the stack with a new symbol (nondeterministically--there may be many choices)*
  - *Think: manipulate stack without reading*

PDA machine



1. In state q
    a. read a symbol c, stack symbol Z
    b. move the tape head right
    b. goto state delta(q, c, Z).1
    d. replace Z with delta(q, c, Z).2

--or:
    a. goto state delta(q, ε, Z).1
    b. replace Z with delta(q, c, Z).2

2. Accept iff the FA is in a final state after reading the last symbol

Acceptance conditions

- Two traditional conditions
  - *Process the input, and accept if the stack is empty*
  - *Process the input, and accept if the PDA is in a final state*

PDA: formal definition

A (nondetermistic) PDA is a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$

- Q is a set of states
- Σ is an input alphabet
- Γ is a stack alphabet
- q_0 is the start state
- Z_0 is a stack symbol called the *start symbol*
- $F \subseteq Q$ is a set of final states
- $\delta : Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma \rightarrow 2^{Q \times \Gamma^*})$

PDA execution: reading a symbol

- Consider $\delta(q, a, Z) = \{(p_1, \gamma_1), \ldots, (p_m, \gamma_n)\}$
- Then the PDA can:
  - enter some state p_i
  - pop the stack (the symbol Z)
  - push all elements γ_i right-to-left (so the first symbol of γ_i is at the top)
  - advance the tape head

PDA execution: epsilon transition

- Consider $\delta(q, \epsilon, Z) = \{(p_1, \gamma_1), \ldots, (p_m, \gamma_n)\}$
- Then the PDA can:
  - enter some state p_i
  - pop the stack (the symbol Z)
  - push all elements γ_i right-to-left
  - **does not** advance the tape head

Balanced parentheses

$$M = (\{q_1\}, \{\underline{)}, \underline{(}\}, \{S, \underline{(}\}, \delta, S, \{\})$$

$$\begin{aligned}
\delta(q_1, \underline{(}, S) &= \{(q_1, \underline{(})\} \\
\delta(q_1, \underline{(}, \underline{(}) &= \{(q_1, \underline{(})\} \\
\delta(q_1, \underline{)}, \underline{(}) &= \{(q_1, \epsilon)\} \\
\delta(q_1, \epsilon, S) &= \{(q_1, \epsilon)\}
\end{aligned}$$

Accepting $w\$w^R$

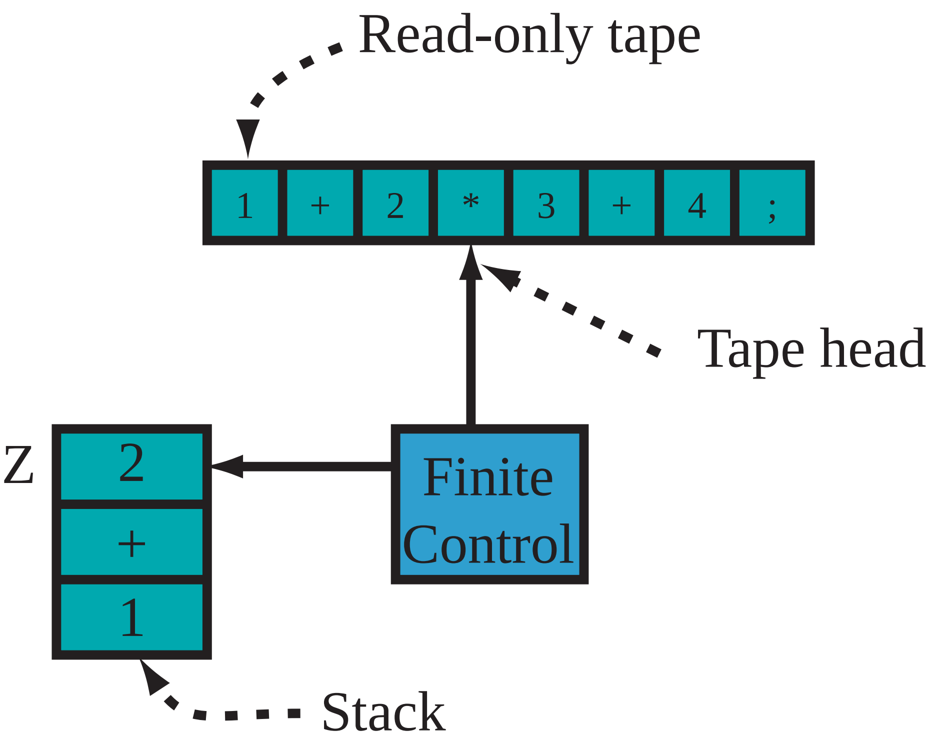
- (Machine given in class)

Instantaneous descriptions

An *instantaneous description* (ID) is a 3-tuple (q, w, γ)

- $q \in Q$ is a state
- $w \in \Sigma^*$ is the rest of the input
- $\gamma \in \Gamma^*$ is the contents of the stack, read top-to-bottom

ID



ID = (q, "*3+4;", "2+1")
where q is the current machine state

ID transitions

- Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA
- Let $a \in \Sigma \cup \{\epsilon\}$ be an input symbol
- Then, $(q, aw, Z\alpha) \rightarrow_M (p, w, \beta\alpha)$ if $\delta(q, a, Z)$ contains (p, β)
- Define $\rightarrow_M^*$ as the transitive closure of $\rightarrow_M$ (so it is reflexive, symmetric, and transitive)

Executions

- An execution $\sigma = \{\sigma_1, \ldots, \sigma_n\}$ is a string $\sigma \in ID^*$, where
  - $\sigma_i \rightarrow_M \sigma_{i+1}$

Acceptance

- A machine $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ *accepts* string w if
  - (empty stack) there exists an execution
    $$(q_0, w, Z_0) \rightarrow_M (q, \epsilon, \{\})$$
  - (final state) there exists an execution
    $$(q_0, w, Z_0) \rightarrow_M (q, \epsilon, \gamma)$$
    and $q \in F$

Deterministic PDAs

- Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ be a PDA
- Then M is *deterministic* if:
  - Whenever $\delta(q, \epsilon, Z) \neq \{\}$, then $\delta(q, a, Z) = \{\}$ for any $a \in \Sigma$
  - $|\delta(q, a, Z)| \leq 1$ for any $a \in \Sigma$
- Note: ww^R is accepted by a nondetermistic PDA, but not by any deterministic PDA

Empty stack vs final state

- Final state PDA -> empty stack PDA
  - *Simulate: whenever the PDA reaches a final state, empty the stack*
- Empty stack PDA -> final state PDA
  - *Add a special marker $ at the bottom of the stack*
  - *Add transitions delta(q, epsilon, S) → qf for some new state qf in F*

Compiling a CFG to a PDA

Theorem If $G = (V, T, P, S)$, then there is a PDA M where $L(G) = L(M)$ (empty stack)

Greibach Normal Form (GNF)

Greibach normal form every context-free language L without ϵ can be generated by a CFG wherr each production has the form $A \to a\alpha$ where a is a terminal, and α is a sequence of symbols.

GNF Construction: part 1

Proof

- Let $G = (V, T, P, S)$ be a CFG in Chomsky normal form where $L = L(G)$
- This means all productions have the form $A \to a$ or $A \to BC$
- First, number the productions
- Next, require that if $A_i \to A_j \alpha$ is a production, then $j > i$

Production ordering

By (complete) induction on the productions

- Consider production i
- For $j <$, assume all productions have the form $A_j a$ or $A_j \rightarrow A_k \alpha$ where $k > j$
- If $A_i \rightarrow A_l \alpha$, and $l < i$ then substitute A_l for its right-hand-side
- Repeat, until $A_i \rightarrow A_m \alpha$ for $m \geq l$

GNF productions

- For each $A_i \to A_j\alpha$, expand A_j so that the production starts with a terminal
- Each rhs for B_i starts with a terminal or a A_i
- If A_i, expand to get a terminal

Building the PDA from the GNF

To build the PDA for a grammar $G = (V, T, P, S)$ in GNF

- Each production has the form $A \to a_1 \ldots a_n X_1 \ldots X_m$
- Define $M = (\{q\}, T, V, \delta, q, S, \{\})$
- Let $\delta(q, a, A)$ contains (q, γ) iff $A \to a\gamma \in P$

Constructed forms

Now we have productions of the following forms:

- $A_i \rightarrow A_j \alpha$ for $j > i$
- $A_i \rightarrow a\alpha$ for terminal a
- $B_i \rightarrow \gamma$ for $\gamma \in (V \cup \{B_1, \ldots, B_m\})^*$

Self-productions

- Next, suppose $A_i \rightarrow A_i\alpha$,
- Let $A_i A_i \alpha_1 \mid \cdots \mid A_i \alpha_n$ be the productionw ith rhs starting with A_i
- Let $A_i \rightarrow \beta_1 \mid \cdots \mid \beta_m$ be the remaining productions
- Add a new nonterminal B, and productions

$$\begin{array}{rcl} A_i & \rightarrow & \beta_j \\ A_i & \rightarrow & \beta_j B \\ B & \rightarrow & \alpha_i \\ B & \rightarrow & \alpha_i B \end{array}$$

Building a CFG from a PDA

Theorem If M is a PDA, then there is a CFG $G = (V, T, P, S)$ s.t. $L(M) = L(G)$

- Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$
- Let $V = \{[q, A, p] \mid q, p \in Q \wedge A \in \Gamma\} \cup S$
- Productions
  - $S \rightarrow [q_0, Z_0, q]$ for each $q \in Q$
  - $[q, A, q_{m+1}] \rightarrow a[q_1, B_1, q_2] \cdots [q_m, B_m, q_{m+1}]$ for each

$$(q_1, B_1 \ldots B_m) \in \delta(q, a, A)$$