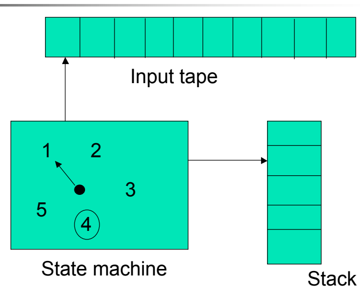
Pushdown Automata

Pushdown Automata

- A pushdown automata (PDA) is essentially:
  - An NFA with a stack
  - A "move" of a PDA will depend upon
    - Current state of the machine
    - Current symbol being read in
    - Current symbol popped off the top of the stack
  - With each "move", the machine can
    - Move into a new state
    - Push symbols on to the stack

Pushdown Automata



Pushdown Automata

- The stack
  - The stack has its own alphabet
  - Included in this alphabet is a special symbol used to indicate an empty stack. (z)
- Note that the basic PDA is non-deterministic!

Pushdown Automata

- Let's formalize this:
  - A pushdown automata (PDA) is a 7-tuple:
    - $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ where
      - Q = finite set of states
      - Σ = tape alphabet
      - Γ = stack alphabet (may have symbols in common w/ Σ)
      - $q_0 \in Q$ = start state
      - $z \in \Gamma$ = initial stack symbol
      - $F \subseteq Q$ = set of accepting states
      - δ = transition function

Pushdown Automata

- About this transition function δ:
  - During a move of a PDA:
    - At most one character is read from the input tape
      - λ transitions are okay
    - The topmost character is popped from the stack
    - The machine will move to a new state based on:
      - The character read from the tape
      - The character popped off the stack
      - The current state of the machine
    - 0 or more symbols from the stack alphabet are pushed onto the stack.

Pushdown Automata

- Formally:
  - δ: Q x ($\Sigma \cup \{\lambda\}$) x $\Gamma \rightarrow$ (finite subsets of Q x Γ^*)
  - Domain:
    - Q = state
    - ($\Sigma \cup \{\lambda\}$) = symbol read off tape
    - Γ = symbol popped off stack
  - Range
    - Q = new state
    - Γ^* = symbols pushed onto the stack

Pushdown Automata

- Example:
  - δ (q, a, a) = (p, aa)
    - Meaning:
      - When in state q,
      - Reading in an a from the tape
      - With an a popped off the stack
    - The machine will
      - Go into state p
      - Push the string "aa" onto the stack

Pushdown Automata

- Configuration of a PDA
  - Gives the current "configuration" of the machine
  - (p, x, α) where
    - p is the current state
    - x is a string indicating what remains to be read on the tape
    - α is the current contents of the stack.

Pushdown Automata

- Move of a PDA:
  - We can describe a single move of a PDA:
    - (q, x, α) $\mapsto$(p, y, β)
    - If:
      - x = ay, $\alpha = \gamma X$, β = YX
        - And
      - δ (q, x, γ) includes (p, Y) or
      - δ (q, ε, γ) includes (p, Y) and x = y.

Pushdown Automata

- Moves of a PDA
  - We can write:
    - (q, x, α) $\overset{*}{\mapsto}$ (p, y, β)
    - If
      - You can get from one configuration to the other by applying 0 or more moves.

Pushdown Automata

- Strings accepted by a PDA by Final State
  - Start at (q_0, x, z)
    - Start state q_0
    - X on the input tape
    - Empty stack
  - End with (q, λ, β)
    - End in an accepting state ($q \in F$)
    - All characters of x have been read
    - Some string on the stack (doesn't matter what).

Pushdown Automata

- Strings accepted by a PDA (Final State)
  - Let $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ be a PDA
  - x is accepted by M if
    - $(q_0, x, z) \vdash^* (q, \lambda, \beta)$
    - Where
      - $q \in F$
      - $\beta \in \Gamma^*$

Pushdown Automata

- The language accepted by a PDA
  - Let $M = (Q, \Sigma, \Gamma, q_0, z, F, \delta)$ be a PDA
  - The language accepted by M by final state,
    - Denoted L(M) is
    - The set of all strings x that are accepted by M by final state

Pushdown Automata

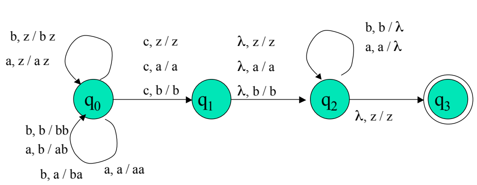
- Let's look at an example:
  - $L = \{ xcx^r \mid x \in \{ a,b \}^* \}$
  - Basic idea for building a PDA
    - Read chars off the tape until you reach the 'c'.
    - As you read chars push them on the stack
    - After reading the c, match the chars read with the chars popped off the stack until all chars are read
    - If at any point the char read does not match the char popped, the machine "crashes"

Pushdown Automata

- Let's look at an example:
  - $L = \{ xcx^r \mid x \in \{ a,b \}^* \}$
  - The PDA will have 4 states
    - State 0 (initial) : reading before the 'c'
    - State 1: read the 'c'
    - State 2 :read after 'c', comparing chars
    - State 3: (accepting): move only after all chars read and stack empty

Pushdown Automata

- Let's look at an example:
  - $L = \{ xcx^r \mid x \in \{ a,b \}^* \}$



PDA Example

- Transition for abcba
  - $(q_0, abcba, Z) \vdash (q_0, bcba, a)$ // push a
  - $\vdash (q_0, cba, ba)$ // push b
  - $\vdash (q_1, ba, ba)$ // goto 1
  - $\vdash (q_2, ba, ba)$ // λ trans
  - $\vdash (q_2, a, a)$ // pop b
  - $\vdash (q_2, \lambda, z)$ // pop a
  - $\vdash (q_3, \lambda, z)$ // Accept!

PDA Example

- Transition for abcb
  - $(q_0, abcb, z) \mapsto (q_0, bcb, a)$ // push a
  - $\mapsto (q_0, cb, ba)$ // push b
  - $\mapsto (q_1, b, ba)$ // goto 1
  - $\mapsto (q_2, b, ba)$ // ε trans
  - $\mapsto (q_2, \lambda, a)$ // pop b
  - Nowhere to go // Reject!

Pushdown Automata

- I bet you're wondering if JFLAP can handle PDAs!
  - Yes, it can…
  - Let's take a look.

Pushdown Automata

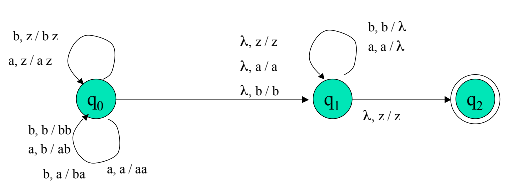
- Let's look at another example:
  - $L = \{ xx^r \mid x \in \{ a,b \}^* \}$
  - Basic idea for building a PDA
    - Much like last example, except
      - This time we don't know when to start popping and comparing
      - Since PDAs are non-deterministic, this is not a problem

Pushdown Automata

- Let's look at another example:
  - $L = \{ xx^r \mid x \in \{ a,b \}^* \}$
  - The PDA will have 3 states
    - State 0 (initial) : reading before the center of string
    - State 1: read after center of string, comparing chars
    - State 2 (accepting): after all chars read, stack should be empty
  - The machine can choose to go from state 0 to state 1 at any time:
    - Will result in many "wrong" set of moves
    - All you need is one "right" set of moves for a string to be accepted.

Pushdown Automata

- Let's look at an example:
  - $L = \{ xx^r \mid x \in \{ a,b \}^* \}$



PDA Example

- Let's see a bad transition set for abba
  - $(q_0, abba, z) \mapsto (q_0, bba, a)$ // push a
  - $\mapsto (q_0, ba, ba)$ // push b
  - $\mapsto (q_0, a, bba)$ // push b
  - $\mapsto (q_1, a, bba)$ // λ trans
  - Nowhere to go // Reject!

PDA Example

- Let's see a good transition set for abba
  - $(q_0, abba, z) \vdash (q_0, bba, a)$ // push a
  - $\vdash (q_0, ba, ba)$ // push b
  - $\vdash (q_1, ba, ba)$ // ε trans
  - $\vdash (q_1, a, a)$ // pop b
  - $\vdash (q_1, \lambda, Z)$ // pop a
  - $\vdash (q_2, \lambda, Z)$ // Accept!

Pushdown Automata

- "Let's go to the video tape"
  - Actually JFLAP…

Pushdown Automata

- Strings accepted by a PDA by <u>Final State</u>
  - Start at (q_0, x, z)
    - Start state q_0
    - X on the input tape
    - Empty stack
  - End with (q, λ, β)
    - End in an accepting state ($q \in F$)
    - All characters of x have been read
    - Some string on the stack (doesn't matter what).

Pushdown Automata

- Strings accepted by a PDA (Final State)
  - Let $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ be a PDA
  - x is accepted by M if
    - $(q_0, x, z) \vdash^* (q, \lambda, \beta)$
    - Where
      - $q \in A$
      - $\beta \in \Gamma^*$

Pushdown Automata

- Strings accepted by a PDA by <u>Empty Stack</u>
  - Start at (q_0, x, z)
    - Start state q_0
    - X on the input tape
    - Empty stack
  - End with (q, λ, λ)
    - End in <u>any</u> state
    - All characters of x have been read
    - Stack is empty

Pushdown Automata

- Strings accepted by a PDA (Empty Stack)
  - Let $M = (Q, \Sigma, \Gamma, \delta, q_0, z, F)$ be a PDA
  - x is accepted by M if
    - $(q_0, x, z) \vdash^* (q, \lambda, \lambda)$
    - Where
      - $q \in Q$

Pushdown Automata

- The language accepted by a PDA
  - Let M = $(Q, \Sigma, \Gamma, q_0, z, F, \delta)$ be a PDA
  - The language accepted by M by final state,
    - Denoted L(M) is
    - The set of all strings x that are accepted by M by final state
  - The language accepted by M by empty stack,
    - Denoted N(M) is
    - The set of all strings x that are accepted by M by empty stack
- We will show that all languages accepted by a PDA by final state will be accepted by an equivalent PDA by empty stack and visa versa

Final State vs. Empty Stack

- The two means by which a PDA can accept are equivalent wrt the class of languages accepted
  - Given a PDA M such that L = L(M), there exists a PDA M' such that L = N(M')
  - Given a PDA M such that L = N(M), there exists a PDA M' such that L = L(M')

Final State → Empty Stack

- Final State → Empty Stack
  - Given a PDA $P_F = (Q, \Sigma, \Gamma, \delta_F, q_0, z, F)$ and L = L (P_F) then there exists a PDA P_N such that L = N (P_N)
  - We will build such a PDA

Accept by Empty Stack

- Final State → Empty Stack
  - Basic idea
    - Transitions of P_N will mimic those of P_F
    - Create a new state in P_N that will empty the stack.
    - The machine can move into this new state whenever the machine is in an accepting state of P_F

Accept by Empty Stack

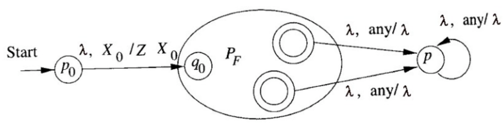
- Final State → Empty Stack
  - We must be careful though
    - P_F may crash when the stack is empty.
    - In those cases we need to assure that P_N does not accept
    - To solve this:
      - Create a new empty stack symbol X_0 which is placed on the stack before P_F s empty stack marker (z)
      - z will only be popped by the new "stack emptying state
    - The first move of P_N will be to place zX_0 on P_N stack.

Final State → Empty Stack

- Final State → Empty Stack
  - P_N = (
    - $Q \cup \{p_o, p\}$,
    - Σ,
    - $\Gamma \cup \{X_0\}$
    - δ_N
    - p_0,
    - X_0)

Accept by Empty Stack

- Final State → Empty Stack



Empty Stack → Final State

- Empty Stack → Final State
  - Given a PDA $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0)$ and $L = N(P_N)$ then there exists a PDA P_F such that $L = L(P_F)$
  - We will build such a PDA
  - Actually, you will…Exercise 17

Reality Check

- Pushdown Automata
  - NFAs with a stack
  - Move depends on tape symbol, state, and top of stack.
  - Move involves popping stack, moving to new state and pushing onto stack
  - Basic PDA is nod-deterministic.
  - Accept by final state
  - Accept by empty stack

Next time

- PDAs…the perfect machine for CFLs…
- Questions?