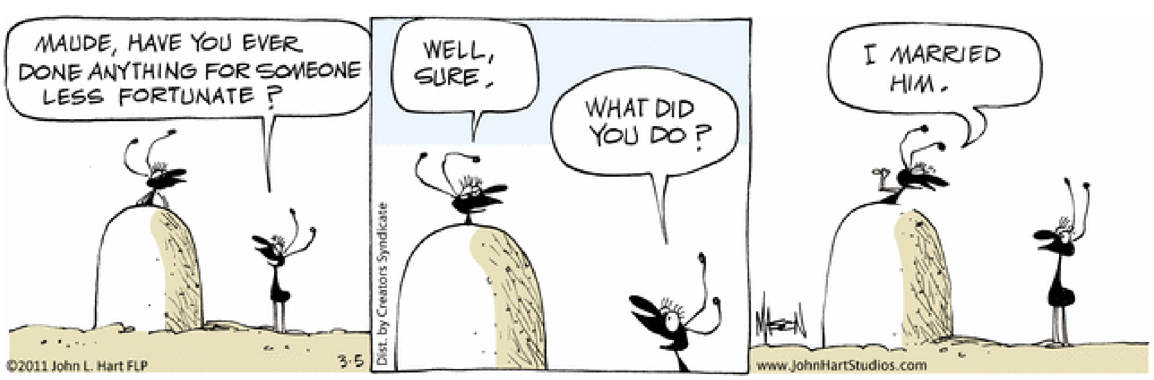
MA/CSSE 474
Theory of Computation

PDA examples

More About Nondeterminism

Your Questions?

- Previous class days' material
- Reading Assignments
- HW11 or 12 problems
- Anything else



Recap: Definition of a Pushdown Automaton

$M = (K, \Sigma, \Gamma, \Delta, s, A)$, where:

- K is a finite set of states
- Σ is the input alphabet
- Γ is the stack alphabet
- $s \in K$ is the initial state
- $A \subseteq K$ is the set of accepting states, and
- Δ is the transition relation. It is a finite subset of

Σ and Γ are not necessarily disjoint

$$\underbrace{(K \times (\Sigma \cup \{\varepsilon\}) \times \Gamma^*)}\ \times\ \underbrace{(K \times \Gamma^*)}$$

state | input symbol or ε | string of symbols to pop from top | state | string of symbols to push on stack

What does an individual element of Δ look like?

Recap: Definition of a Pushdown Automaton

A **configuration** of M is an element of $K \times \Sigma^* \times \Gamma^*$.

The **initial configuration** of M is (s, w, ε), where w is the input string.

Recap: Yields

Let c be any element of $\Sigma \cup \{\varepsilon\}$,
Let γ_1, γ_2 and γ be any elements of Γ^*, and
Let w be any element of Σ^*.

Then:
$(q_1, cw, \gamma_1\gamma) \vdash_M (q_2, w, \gamma_2\gamma)$ iff $((q_1, c, \gamma_1), (q_2, \gamma_2)) \in \Delta$.

Let $\vdash_M^*$ be the reflexive, transitive closure of $\vdash_M$.

C_1 ***yields*** configuration C_2 iff $C_1 \vdash_M^* C_2$

Recap: Nondeterminism

If M is in some configuration (q_1, s, γ) it is possible that:

- Δ contains exactly one transition that matches.
- Δ contains more than one transition that matches.
- Δ contains no transition that matches.

Recap: Computations

A ***computation*** by M is a finite sequence of configurations C_0, C_1, …, C_n for some $n \geq 0$ such that:

- C_0 is an initial configuration
- C_n is of the form (q, ε, γ), for some state $q \in K_M$ and some string γ in Γ^*
- $C_0 \vdash_M C_1 \vdash_M C_2 \vdash_M \ldots \vdash_M C_n$.

Recap: Accepting Computation

A computation C of M is an ***accepting computation*** iff:

- $C = (s, w, \varepsilon) \vdash_M^* (q, \varepsilon, \varepsilon)$, and
- $q \in A$.

M ***accepts*** a string w iff at least one of its computations accepts.

Other paths may:

- Read all the input and halt in a nonaccepting state
- Read all the input and halt in an accepting state with a non-empty stack
- Loop forever and never finish reading the input
- Reach a dead end where no more input can be read

The ***language accepted by*** M, denoted $L(M)$, is the set of all strings accepted by M.

Rejecting

A computation *C* of *M* is a ***rejecting computation*** iff:

- $C = (s, w, \varepsilon) \vdash_M^* (q, \varepsilon, \alpha)$,
- *C* is not an accepting computation, and
- *M* has no moves that it can make from (q, ε, α).

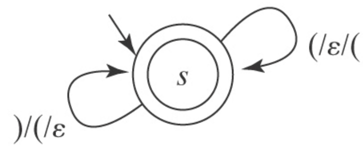
M ***rejects*** a string *w* iff all of its computations reject.

Note that it is possible that, on input *w*, *M* neither accepts nor rejects.

PDA examples

Construct PDAs to recognize specific languages

A PDA for Bal



$M = (K, \Sigma, \Gamma, \Delta, s, A)$, where:

$K = \{s\}$ the states

$\Sigma = \{(,)\}$ the input alphabet

$\Gamma = \{(\}$ the stack alphabet

$A = \{s\}$

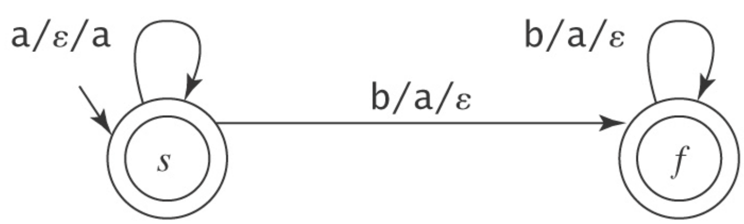
Δ contains:

((*s*, (, ε), (*s*, ( )) **

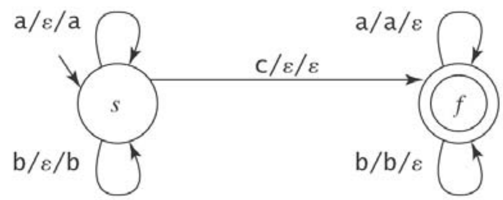
((*s*,), ( ), (*s*, ε))

**Important: This does not mean that the stack is empty

A PDA for $A^nB^n = \{a^nb^n: n \geq 0\}$



A PDA for $\{wcw^R: w \in \{a, b\}^*\}$



M = (*K*, Σ, Γ, Δ, *s*, *A*), where:

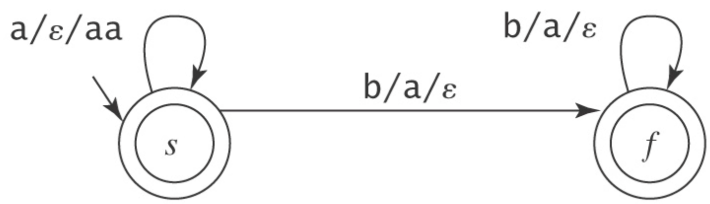
K = {*s*, *f*}	the states
Σ = {a, b, c}	the input alphabet
Γ = {a, b}	the stack alphabet
A = {*f*}	the accepting states

Δ contains:
((*s*, a, ε), (*s*, a))
((*s*, b, ε), (*s*, b))
((*s*, c, ε), (*f*, ε))
((*f*, a, a), (*f*, ε))
((*f*, b, b), (*f*, ε))

How can we modify this PDA to accept $\{ww^R: w \in \{a, b\}^*\}$?

A PDA for $\{a^n b^{2n}: n \geq 0\}$

A PDA for $\{a^n b^{2n} : n \geq 0\}$



A PDA for PalEven $=\{ww^R : w \in \{a, b\}^*\}$

$S \rightarrow \varepsilon$
$S \rightarrow aSa$
$S \rightarrow bSb$

This one is nondeterministic

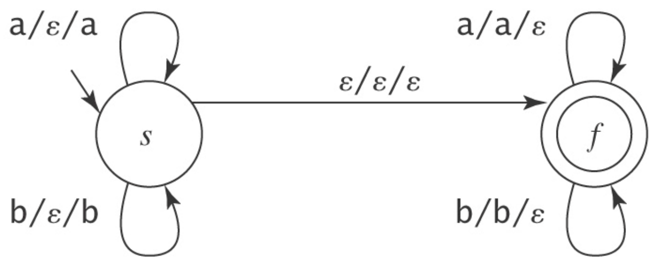
A PDA:

A PDA for PalEven $=\{ww^R: w \in \{a, b\}^*\}$

$S \to \varepsilon$
$S \to aSa$
$S \to bSb$

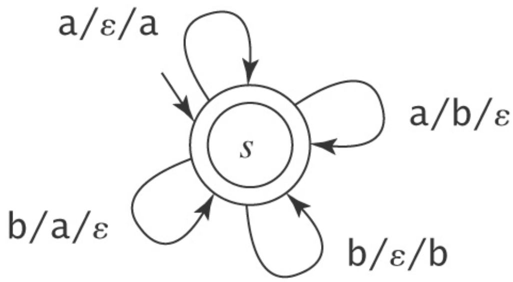
This one is nondeterministic

A PDA:



A PDA for $\{w \in \{a, b\}^* : \#_a(w) = \#_b(w)\}$

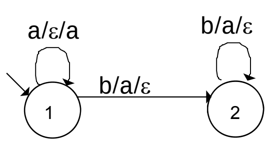
A PDA for $\{w \in \{a, b\}^* : \#_a(w) = \#_b(w)\}$



More on Nondeterminism
Accepting Mismatches

$L = \{a^m b^n : m \neq n; m, n > 0\}$

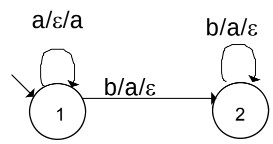
Start with the case where $n = m$:



More on Nondeterminism Accepting Mismatches

$L = \{a^m b^n : m \neq n; m, n > 0\}$

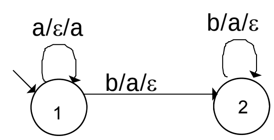
Start with the case where $n = m$:



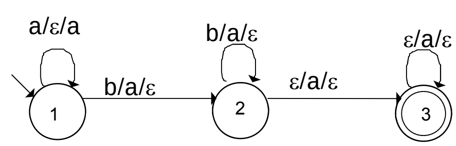
- If stack and input are empty, halt and reject.
- If input is empty but stack is not ($m > n$) (accept):
- If stack is empty but input is not ($m < n$) (accept):

More on Nondeterminism Accepting Mismatches

$L = \{a^m b^n : m \neq n; m, n > 0\}$

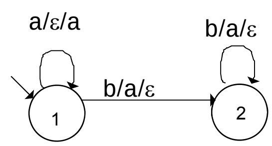


- If input is empty but stack is not ($m > n$) (accept):

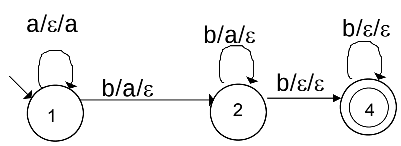


More on Nondeterminism
Accepting Mismatches

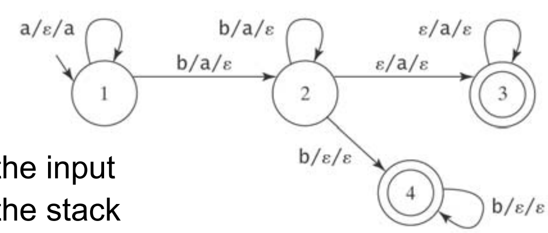
$L = \{a^m b^n : m \neq n;\ m, n > 0\}$



- If stack is empty but input is not ($m < n$) (accept):



$L = \{a^m b^n : m \neq n;\ m, n > 0\}$



- State 4: Clear the input
- State 3: Clear the stack
- A non-deterministic machine!

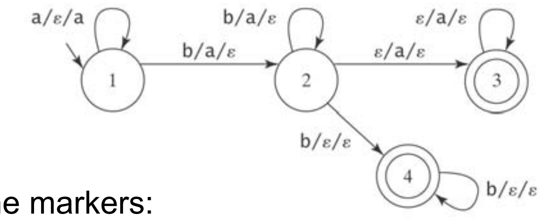
  What if we could

  detect end of input (as we can in real-world situations)?

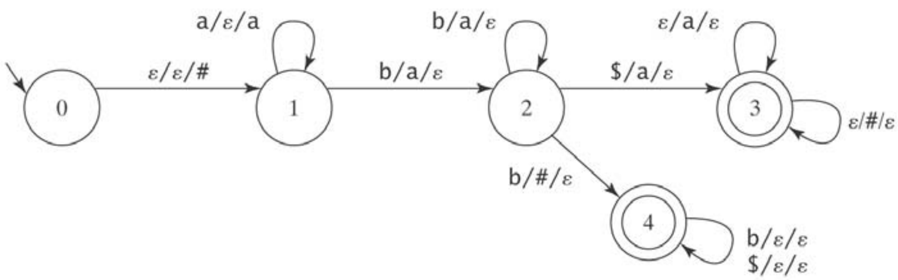
  detect empty stack?
- Add end-of-input marker \$ to Σ
- Add bottom-of-stack marker # to Γ

Reducing Nondeterminism

- Original non-deterministic model



- With the markers:



The Power of Nondeterminism

Consider $A^nB^nC^n = \{a^nb^nc^n : n \geq 0\}$.

PDA for it?

The Power of Nondeterminism

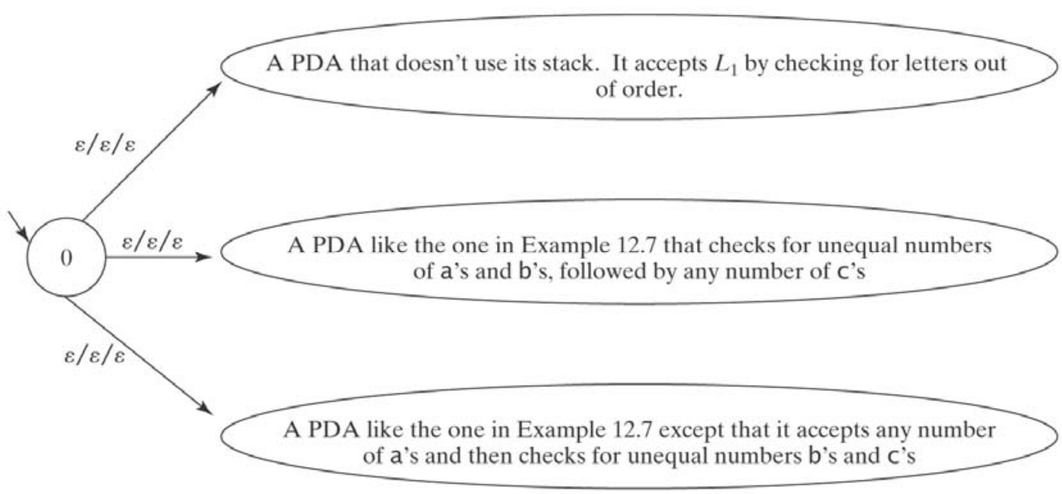
Consider $A^nB^nC^n = \{a^nb^nc^n: n \geq 0\}$. PDA for it?

Now consider $L = \neg A^nB^nC^n$. L is the union of two languages:

1. $\{w \in \{a, b, c\}^*$: the letters are out of order$\}$, and

2. $\{a^ib^jc^k: i, j, k \geq 0$ and $(i \neq j$ or $j \neq k)\}$ (in other words, unequal numbers of a's, b's, and c's).

A PDA for $L = \neg A^nB^nC^n$



$L = \{a^n b^m c^p : n, m, p \geq 0 \text{ and } n \neq m \text{ or } m \neq p\}$

$S \rightarrow NC$	/* $n \neq m$, then arbitrary c's
$S \rightarrow QP$	/* arbitrary a's, then $p \neq m$
$N \rightarrow A$	/* more a's than b's
$N \rightarrow B$	/* more b's than a's
$A \rightarrow \text{a}$	
$A \rightarrow \text{a}A$	
$A \rightarrow \text{a}A\text{b}$	
$B \rightarrow \text{b}$	
$B \rightarrow B\text{b}$	
$B \rightarrow \text{a}B\text{b}$	
$C \rightarrow \varepsilon \mid \text{c}C$	/* add any number of c's
$P \rightarrow B'$	/* more b's than c's
$P \rightarrow C'$	/* more c's than b's
$B' \rightarrow \text{b}$	
$B' \rightarrow \text{b}B'$	
$B' \rightarrow \text{b}B'\text{c}$	
$C' \rightarrow \text{c} \mid C'\text{c}$	
$C' \rightarrow C'\text{c}$	
$C' \rightarrow \text{b}C'\text{c}$	
$Q \rightarrow \varepsilon \mid \text{a}Q$	/* prefix with any number of a's

Closure question

- Is the set of context-free languages closed under complement?