CS310

Chomsky Normal Form

Section: 2.1 page 106

Pushdown Automata

Sections: 2.2

page 109

October 10, 2014

Quick Review

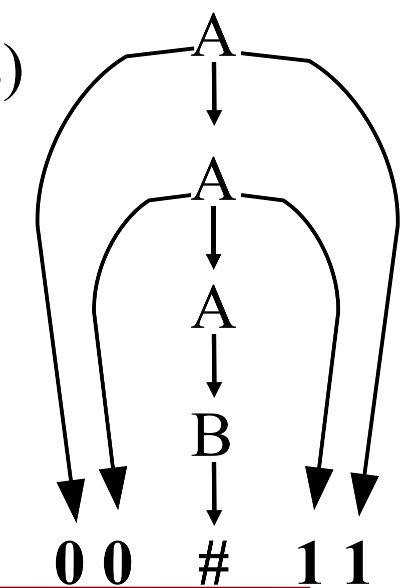
- (CFG) 4-tuple (V, ∑, R, S)
  - V finite set of variables
  - ∑ finite set of terminals
  - R set of rules of form:
    - variable **->** (string of variables and terminals)
  - S ∈ V, start variable
  - L(G) = { w ∈ ∑* | S **-*>** w}
    - w is in ∑* and can be derived from S

Example

A **->** 0A1

A **->** B

B **->** #



Chomsky Normal Form

- CNF presents a grammar in a standard, simplified form:

  A-> BC

  A -> *a*

  S -> ε

  - Where A,B,C are variables and B and C are not the start variable
  - *a* is a terminal
  - The rule S -> ε is allowed so the language can generate the empty string (optional)

CNF Benefits

- Easier to prove statements about CFG's when in CNF

- Any CFG can be converted to CNF
- Remove productions:

  A -> ε to empty

  A -> B Unit rule

  A -> s, s contains a terminal and $|s| > 1$

  A -> s, $|s| > 2$

  $s \in \{ V \cup \Sigma \}^*$

Removing A -> ε

S -> UAV

A -> ε

- A variable A is ***nullable*** if A-*> ε

  Find all nullable variables

  Remove all ε transitions

If T -> X_1AX_2 and A is nullable

then add T -> X_1X_2

Example

S -> TU

T -> AB

A -> aA | ε

B -> bB | ε

U -> ccA | B

Nullable variables?

Productions removed?

Productions added?

Removing A -> B (Unit Productions)

A-> B

B-> s

$s \in \{ V \cup \Sigma \}^*$

- A variable B is A-derivable if A-*>B

Find all A-derivable variables for each A

Remove all unit transitions

If B -> s and B is A-derivable

then add A -> s

Example

S -> TU | T | U

B -> bB | b

T -> AB | A | B

U -> ccA | B | cc

A -> aA | a

S-derivable:

T-derivable:

U-derivable:

Productions removed:

Productions added:

Remove $A \rightarrow S_1 a S_2$

$A \rightarrow S_1 a S_2$

$a \in \sum$, S_1 and S_2 are strings, at least one is not empty

Create

$X_a \rightarrow a$

$A \rightarrow S_1 X_a S_2$

Then fix up $A \rightarrow S_1 X_a S_2$

– why? what rule is violated?

– how?

Remove $A \rightarrow S_1X_aS_2$

$A \rightarrow S_1X_aS_2$

$A \rightarrow$

Put in to CNF

$S \rightarrow ASA \mid aB$

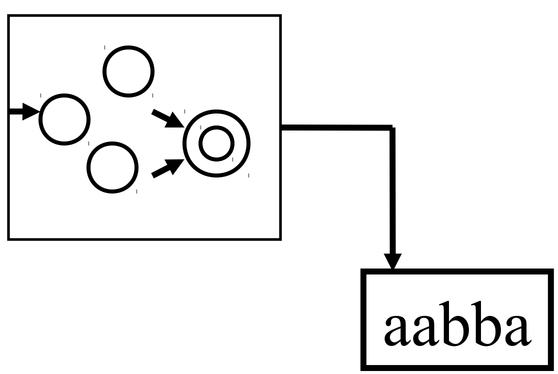
$A \rightarrow B \mid S$

$B \rightarrow b \mid \varepsilon$

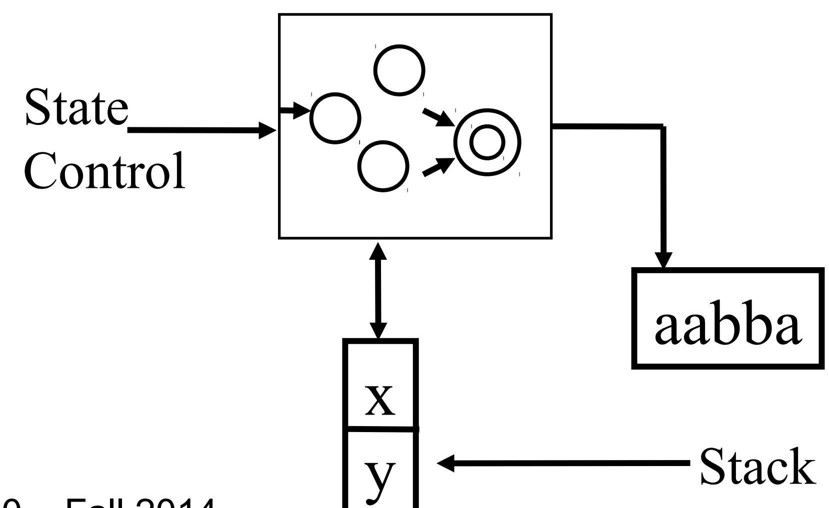
Pushdown Automata

- Machine to recognize Context Free Language
- Similar to an NFA, but contains a *stack*
  - An FA with memory added (LIFO!)

FA



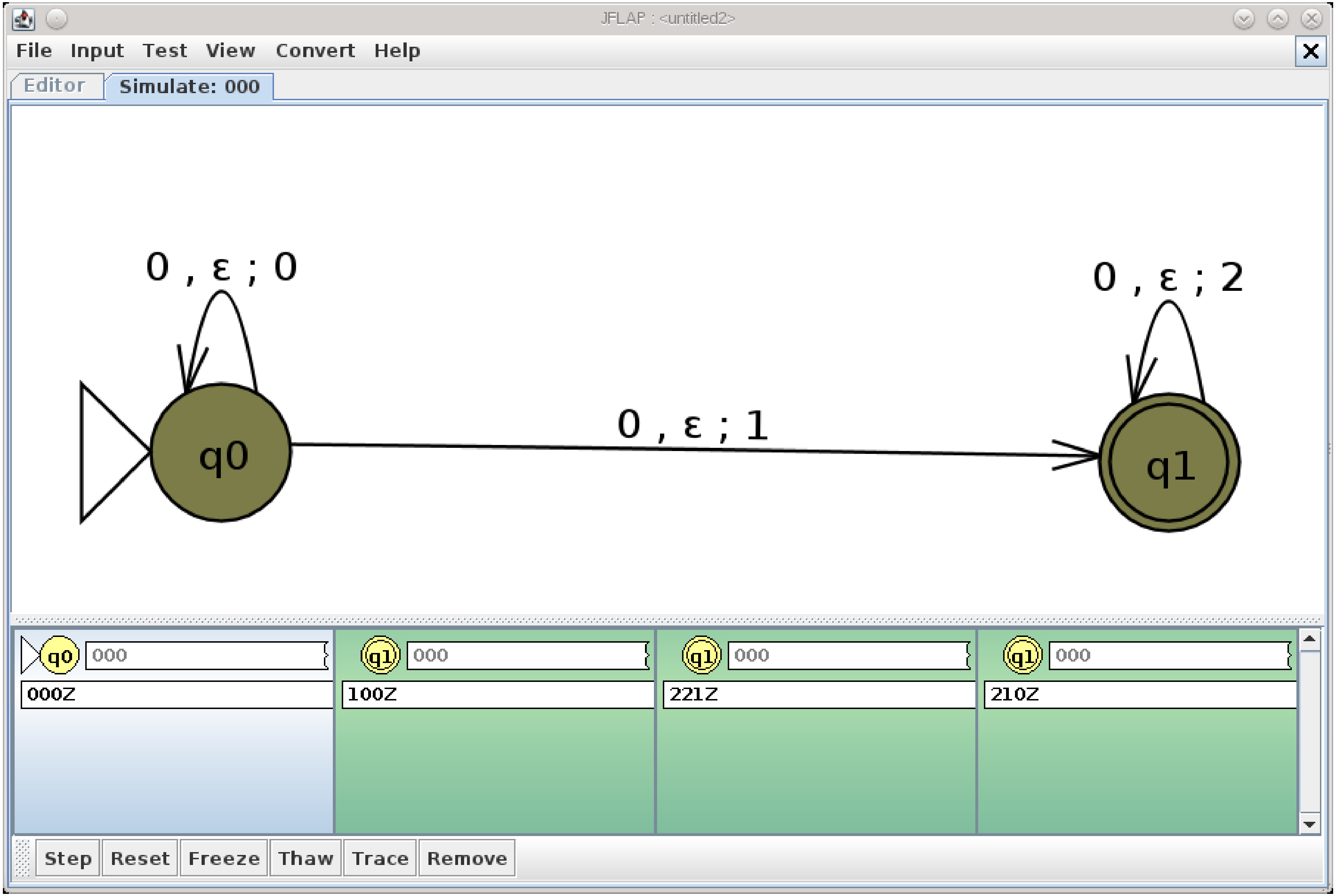
Pushdown Automata



Pushdown Automata

- PDA may be deterministic or nondeterministic
  - Not equivalent! (unlike DFA & NFA)
  - NPDA equivalent to CFG.
    - each process has its own stack
- Define certain (state, input) to push data onto the stack
- Combine input string with stack data for δ

PDA


JFLAP : <untitled2>
File Input Test View Convert Help
Editor
Simulate: 000
0 , ε ; 0
0 , ε ; 1
0 , ε ; 2
q0
q1
000
000Z
100Z
221Z
210Z
Step
Reset
Freeze
Thaw
Trace
Remove

Pushdown Automata (Informally)

S -> X

X -> (X) | XX | ε

What language? Regular?

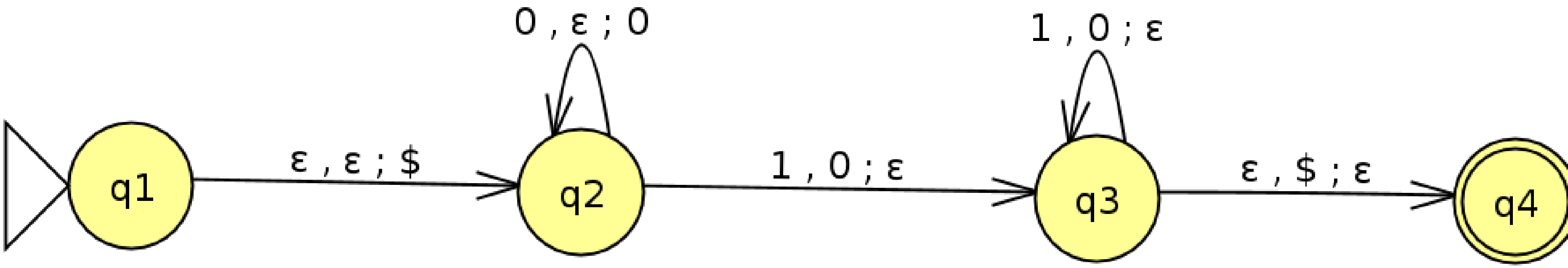
How would you solve this problem using a stack (forget the Pushdown Automata)?

Formal Definition

- 6-tuple!
  - Q: set of states
  - Σ: input alphabet
  - Γ: stack alphabet
  - $\delta: Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \rightarrow P(Q \times \Gamma_\varepsilon)$
    - input and top of stack to transition
    - Do not read or write from stack: $\Gamma_\varepsilon = \varepsilon$
  - $q_0 \in Q$: start state
  - $F \subseteq Q$: set of accept states

Example (Non-deterministic)

- $\{ 0^n 1^n \mid n > 0 \}$



Input	0			1			ε		
Stack	0	$	ε	0	$	ε	0	$	ε
q1	ø	ø	ø	ø	ø	ø	ø	ø	{(q2,$)}
q2	ø	ø	{(q2,0)}	{(q3, ε)}	ø	ø	ø	ø	ø
q3	ø	ø	ø	{(q3, ε)}	ø	ø	ø	{(q4, ε)}	ø
q4	ø	ø	ø	ø	ø	ø	ø	ø	ø

Practice

- $\{ ww^R \mid w \in \{0, 1\}^* \}$

hint: push symbols onto the stack, at each point guess that the middle of the string

has been reached and begin popping from stack

Examples

- Build a PDA for:

  $\{w \mid w \in \{0,1\}^*\}$

  $\{w\#w^R \mid w \in \{0,1\}^*\}$

  $\{0^n1^n ; n >= 0\}$

  $\{w \mid w \in \{0,1\}^*$; w contains an equal number of 0s and 1s$\}$

  $\{w \mid w \in \{0,1\}^*$; w contains more 1s than 0s$\}$

  $\{w \mid w \in \{0,1\}^*$; w contains an unequal number of 1s and 0s$\}$

  $\{wy \mid w \in \{0,1\}^*, y \in \{0,1\}^*$; y is the string w with every character flipped (0->1, 1->0)$\}$

- Read p 119 – 122 for next time!