Miles's office hours
2-4 on Tuesday.

CSE 105

THEORY OF COMPUTATION

"Winter" 2018

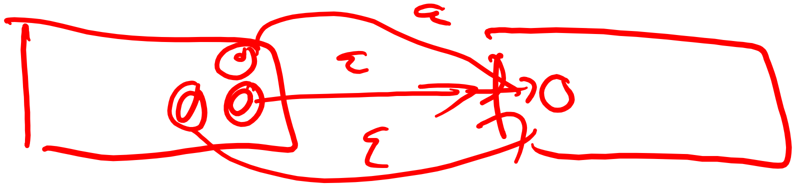
http://cseweb.ucsd.edu/classes/wi18/cse105-ab/

Today's learning goals

Sipser Section 2.1

- Identify the components of a formal definition of a CFG
- Derive strings in the language of a given CFG
- Determine the language of a given CFG
- Design a CFG generating a given language

Closure



The class of languages over Σ that are recognizable by PDA is closed under …

A. ~~Complementation, by flipping accept/ reject states.~~
B. Union, by adding a spontaneous move from a new start state to the start states.
C. Concatenation, by adding spontaneous moves from accept states of one machine to the start state of the second.
D. Kleene star, by adding a fresh start state and spontaneous moves from accept states to the old start state.
E. I don't know.

Closure

- Formal definition:

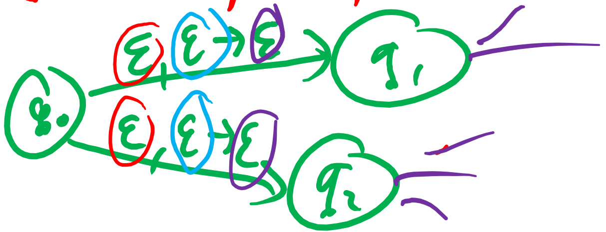
$M_1 = (Q_1, \Sigma, \Gamma, \delta_1, q_1, F_1)$

$M_2 = (Q_2, \Sigma, \Gamma, \delta_2, q_2, F_2)$

$N = M_1 \cup M_2.$

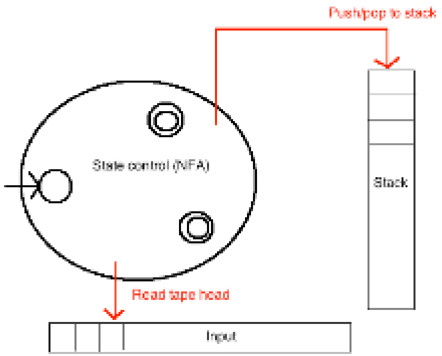
$N = (Q_1 \cup Q_2 \cup \{q_0\}, \Sigma, \Gamma, \delta, q_0, F_1 \cup F_2)$

$$\delta(q,x,s) = \begin{cases} \delta_1(q,x,s) & \text{if } q \in Q_1 \\ \delta_2(q,x,s) & \text{if } q \in Q_2 \\ \{(q_1,\varepsilon),(q_2,\varepsilon)\} & \text{if } q = q_0,\ x=\varepsilon,\ s=\varepsilon \\ \emptyset & \text{otherwise.} \end{cases}$$



PDA: NFA as ??: RegExp

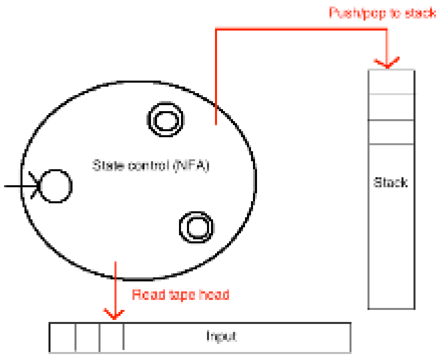
- Automata
  - String is read by the machine one character at a time, from left to right.
  - Determine if computation is successful by checking if entire string was read, and if land at an accept state.



Note: PDA based on **N**FA; can't always be determinized.

PDA: NFA as ??: RegExp

- Automata
  - String is read by the machine one character at a time, from left to right.
  - Determine if computation is successful by checking if entire string was read, and if land at an accept state.
- Regular expressions and ??
  - Derive all strings in the language by following rules for required patterns.



Context-free grammar

Informally, a collection of rules used to *create* string.

CFGs *generate* languages.

Some sample rules:

$$S \to aTb$$
$$T \to aT$$
$$T \to bTS$$
$$S \to \varepsilon$$

More formally...

Context-free grammar

Sipser Def 2.2, page 102

(V, Σ, R, S) 4-tuple.

Variables: finite set of (usually upper case) variables **V**

Terminals: finite set of alphabet symbols **Σ**

$V \cap \Sigma = \emptyset$

Rules/Productions: finite set of allowed transformations **R**

$A \to u \qquad A \in V, u \in (V \cup \Sigma)^*$

Start variable: origination of each derivation **S**

Derivation

Set of ~~rules~~ variables

Set of terminals

G = ({S}, {0}, R, S)

set of rules

with the rules

① ②

R = {S → 0S, S → 0}

Sample derivation:

S => 0S => 00S => 000

Start variable

One-step application of rule

String of terminals

Context-free language

Sipser p. 104

The **language generated by a CFG** (V, Σ, R, S) is

{ w in Σ^* | starting with the Start variable and applying sequence of rules, can derive w on RHS}

If G = (V, Σ, R, S) the language generated by G is denoted L(G).

Notation:

$$S \Longrightarrow^* w$$

Terminology: sequence of rule applications is **derivation**

Context-free language

Sipser p. 104

The language generated by CFG (V, Σ, R, S) is

{ w in Σ^* | starting with the Start variable and applying sequence of rules, can derive w on RHS}.

What is the language of the CFG ({S}, {0}, R, S) with the rules

R = {S → 0S, S → 0}?

A. {0}
B. {0, 0S}
C. ~~{0, 00, 000, …}~~
D. {ε, 0, 00, 000, …}
E. I don't know.

Context-free language

Sipser p. 104

What is the language of the CFG ({S}, {0,1}, R, S) with rules

$$R = \left\{ \begin{array}{l} S \rightarrow 0S \\ S \rightarrow 1S \\ S \rightarrow \varepsilon \end{array} \right\}$$

$S \rightarrow 0S \mid 1S \mid \varepsilon$

A. $L(0^*1^*)$
B. $L(0^* \cup 1^*)$
C. $L((0 \cup 1)^*)$ = language of all possible binary strings.
D. $L((0^*1^*))^*$
E. I don't know.

Designing a CFG

S
→ aT
→ abV
→ abbW
→ abba

L = { abba }

Which CFG generates L?

A. ({S, T, V, W} , {a, b}, {S → aT , T → bV , V → bW , W → a }, S)
B. ({Q}, {a,b}, { Q → abba } , Q)
C. ({X, Y} , {a,b}, { X → aYa , Y → bb } , X)
D. All of the above
E. None of the above

X
→ aYa
→ abba

Context-free languages

- L (00*)
- L((0U1)*)
- { abba }

regular.

Is any nonregular set context-free?

What about the languages that are recognized by PDAs?

Designing a CFG

We know this
set is not
regular!

$L = \{ a^n b^n \mid n \geq 0 \}$

$G = (V, \Sigma, R, \text{Start Variable})$.

Designing a CFG

$G = (\{S\}, \{a, b\}, R, S)$

$$L = \{ a^n b^n \mid n \geq 0 \}$$

One approach:

- what is shortest string in the language?
- how do we go from shorter strings to longer ones?

$$R = \begin{cases} ① \; S \to aSb \\ ② \; S \to ab \\ ③ \; S \to \varepsilon \end{cases}$$

$S \xrightarrow{②} ab$

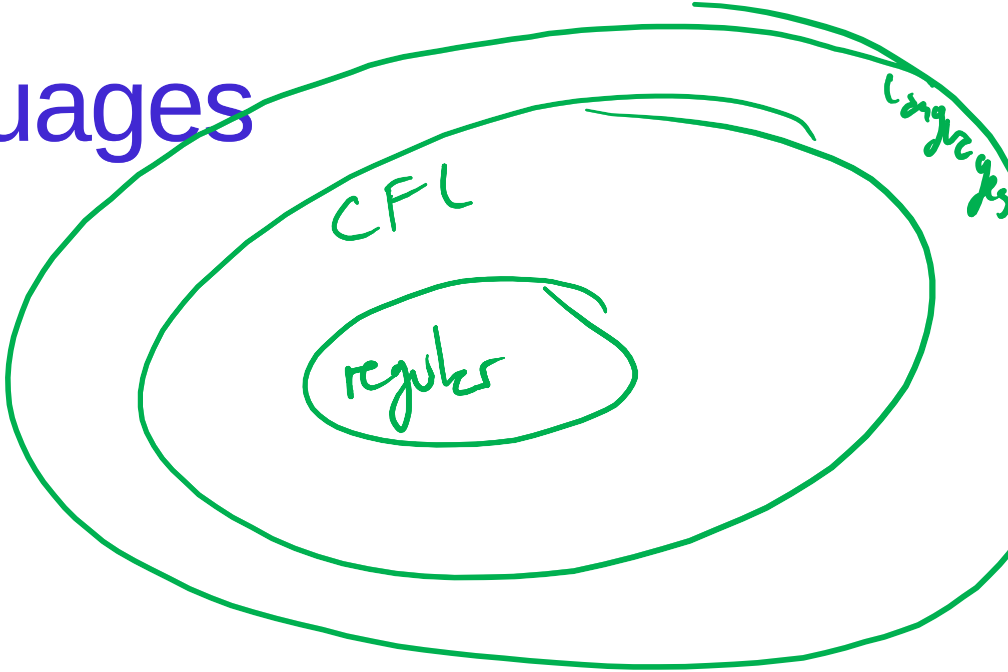
$S \xrightarrow{①} aSb$

$aSb \xrightarrow{③} a\varepsilon b$

ab

Context-free languages

- $L(00^*)$
- $L((0 \cup 1)^*)$
- { abba }
- $\{ a^n b^n \mid n \geq 0 \}$



Ex: $\{ 0^i 1^j \mid j \geq i \geq 0 \}$ recognizable by a PDA

exercise.

PDAs and CFGs are equally expressive

Theorem 2.20: A language is context-free if and only if some nondeterministic PDA recognizes it.

Consequences

- Quick proof that every regular language is context free
- To prove closure of class of CFLs under a given operation, can choose two modes of proof (via CFGs or PDAs) depending on which is easier

For next time

Exam 1 next class Wednesday, February 7

- **Bring ID, pen**
- **Bring note card (half page, double sided, handwritten)**
- **Check assigned seat on Piazza**
- **Piazza will be inactive from 8AM to 3:30PM on Wednesday**