Problem 1

Describe simple algorithms to

1. Convert a regular expression into a context-free grammar.

We can construct the grammar recursively given the regular expression.

First, base cases

a	$S \to a$
ε	$S \to \varepsilon$
$\emptyset$	$S \to S$

Then given regular expression R and R';

$R \cup R'$	$S_0 \to S \mid S'$
RR'	$S_0 \to SS'$
R^*	$S_0 \to S_0 S \mid \varepsilon$

where S and S' are the start symbols of the CFG for R and R', respectively.

2. Convert an NFA to a context-free grammar

Given NFA $N = (Q, \Sigma, \delta, q_0, F)$

where $\delta : Q \times \Sigma_\varepsilon \rightarrow \mathcal{P}(Q)$

we construct $G = (V, \Sigma, R, S)$

$V = Q$

$S = q_0$

R defined below:

if $q_j \in \delta(q_i, a)$ add the rule

$q_i \rightarrow a q_j$

if $q_i \in F$ add the rule

$q_i \rightarrow \varepsilon$

The rules trace the transition of the NFA

Problem 2

(a) Prove that intersection of context-free language and a regular language is context-free.

If L is context-free and R is regular, then L has a pushdown automaton that accepts it and R has a DFA. We can construct a new PDA that simulate both the PDA of L and the DFA of R.

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

where $\delta_1 : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \longrightarrow \mathcal{P}(Q \times \Gamma_\varepsilon)$

and DFA $A = (Q_2, \Sigma, \delta_2, q_2, F_2)$

The new PDA $P' = (Q, \Sigma, \Gamma, \delta, q_0, F)$

where

- $Q = Q_1 \times Q_2$
- $q_0 = (q_1, q_2)$
- if $\delta_1(q_i, b, a) \ni (q_j, t)$ and $\delta_2(r_k, b) = r_\ell$

$$q_i \xrightarrow{b,\, a \to t} q_j \qquad q_i, q_j \in Q_1$$

$$r_k \xrightarrow{b} r_\ell \qquad r_k, r_\ell \in Q_2$$

Then

$\delta((q_i, r_k), b, a) \ni ((q_j, r_\ell), t)$: simulate both transitions

- $F = F_1 \times F_2$: accept if both accept.

(b) Show that

$$A = \{ w \mid w \in \{a,b,c\}^* \text{ and contains equal numbers of } a\text{'s}, b\text{'s and } c\text{'s} \}$$

is not context-free.

Consider the language

$$B = a^* b^* c^*, \text{ which is Regular}$$

$A \cap B = \{ a^n b^n c^n : n \geq 0 \}$ which is know to be not context-free.

If A is context-free, then we would have that the intersection of a context-free language and a regular language is not context-free, a contradiction.

Problem 3

In Chomsky normal form, every rule has the form

$$S \to \varepsilon$$
$$A \to BC \qquad B, C \neq S$$
$$A \to a$$

~~[illegible]~~ A string of length n has n literals. This means we need n variables to generate those literals, Since no variable, except S, can produce ε, and S never appears on the ~~left of~~ right of a rule. Therefore, starting from S, every application of a rule increases the string length by 1 because it adds one variable, ~~because each application of a rules~~ except the first rule which adds two variables. So we need $n-1$ such rules, in addition to the n rules of the form $A \to a$.

The total is $2n-1$.

Problem 4

Give an example of a language that is not CFL but does satisfy the 3 conditions of the pumping lemma for CFL.

Let $\Sigma = \{a, b, c\}$ and consider the language

$$L = \{0a^n b^n c^n \mid n \geq 0\} \cup \{0^k \Sigma^* \mid k \neq 1\}$$

The intersection of L with $0a^*b^*c^*$, which is a regular language, is $\{0a^n b^n c^n \mid n \geq 0\}$. Let's call this language L'. We can show that L' is not CFL. Therefore, L cannot be CFL because CFL $\cap$ Regular = CFL. Now we have to show two things:

1) L' is not CFL
2) Every string in L can be pumped. (for this see problem 6 in hw 2)

Proving L' is not CFL: Assume L' is CFL, then it has a context-free grammar G that generates it. In G, replace the literal 0 by ε. Then G generates $\{a^n b^n c^n \mid n \geq 0\}$ which we know is not CFL, a contradiction. Therefore, L' is not CFL.

Problem 5

Let G be:

$S \to aSb \mid bY \mid aY$

$Y \to bY \mid aY \mid \varepsilon$

G generates the complement of $\{a^n b^n : n \geq 0\}$. See problem 3b in hw 3.

A grammar for the complement of $L(G)$ is therefore:

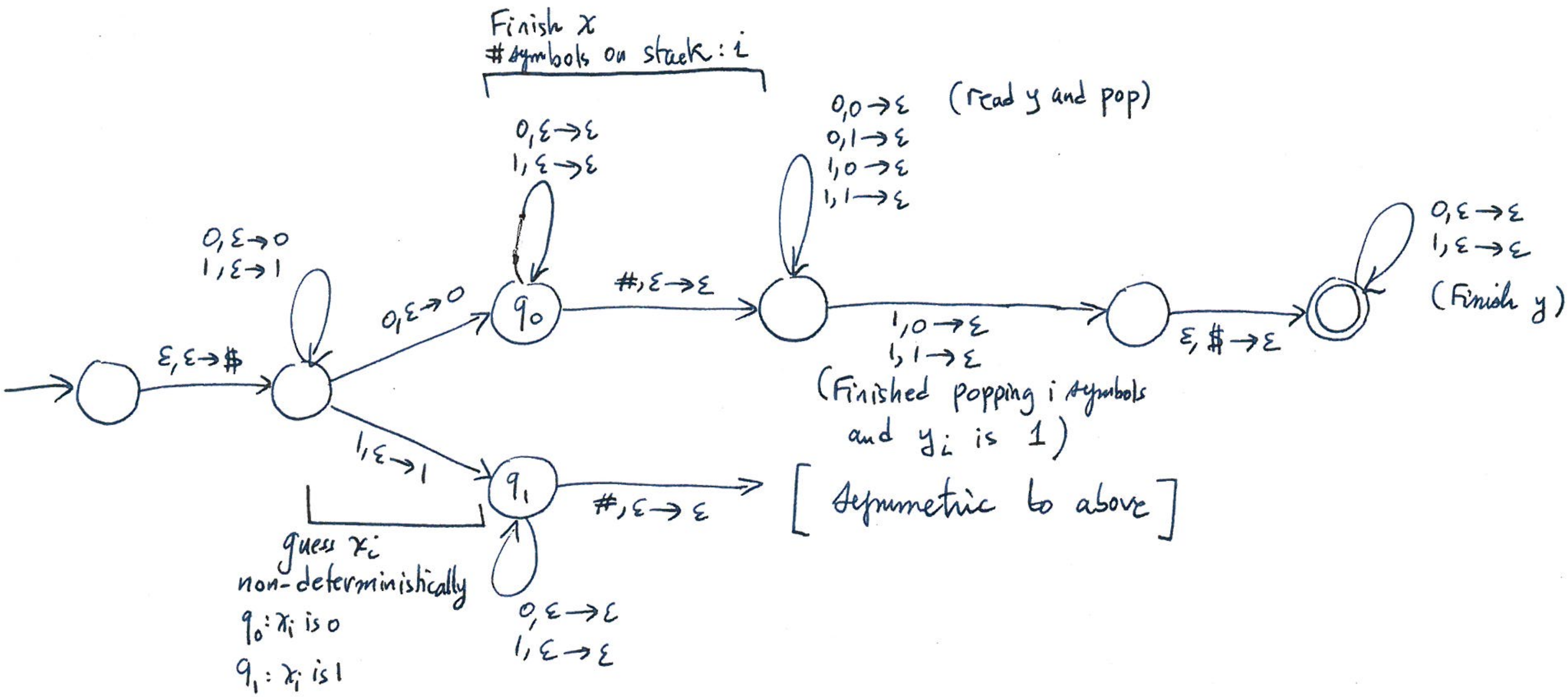
$S \to aSb \mid \varepsilon$

Problem 6:

Show that $\{ x\#y \mid x, y \in \{0,1\}^* \text{ and } x \neq y \}$ is CFL.

We check if $x_i \neq y_i$ for some i. We do it non-deterministically using a PDA

1) Read symbols from x and push them on stack
2) Non-deterministically guess that we have reach x_i and stop pushing. Finish reading x
3) Read $\#$
4) Start reading y until we get to y_i by popping symbols off the stack.
5) Check $y_i \neq x_i$, finish reading y.



$\varepsilon, \varepsilon \to \$$
$0, \varepsilon \to 0$
$1, \varepsilon \to 1$
$0, \varepsilon \to 0$
$1, \varepsilon \to 1$
guess x_i non-deterministically
q_0: x_i is 0
q_1: x_i is 1
q_0
q_1
Finish x
symbols on stack: i
$0, \varepsilon \to \varepsilon$
$1, \varepsilon \to \varepsilon$
$\#, \varepsilon \to \varepsilon$
$0, \varepsilon \to \varepsilon$
$1, \varepsilon \to \varepsilon$
$\#, \varepsilon \to \varepsilon$
$0,0 \to \varepsilon$
$0,1 \to \varepsilon$
$1,0 \to \varepsilon$
$1,1 \to \varepsilon$
(read y and pop)
$1,0 \to \varepsilon$
$1,1 \to \varepsilon$
(Finished popping i symbols and y_i is 1)
[Symmetric to above]
$\varepsilon, \$ \to \varepsilon$
$0, \varepsilon \to \varepsilon$
$1, \varepsilon \to \varepsilon$
(Finish y)