1. Let $L$ be the language with definition

$$L = \{a^i b^j \mid 0 \leq i \leq j \leq 2i\}.$$ 

a) Give a recursive definition of $L$.

b) Use your recursive definition to construct a context free grammar whose language is $L$.

a) A recursive definition is:

BASIS: $\lambda \in L$

RECURSIVE STEP: If $u \in L$ then $au$ and $aub^2$ are in $L$.

CLOSURE: A string is in $L$ if it can be obtained from the basis by a finite number of applications of the recursive step.

b) 

$$G : S \rightarrow aSb \mid aSb^2 \mid \lambda$$

2. Let $L$ be the language with recursive definition

Basis: $\lambda, a, b, c \in L$.

Recursive Step: If $u \in L$ then $au$ and $ub$ are in $L$.

Closure: A string is in $L$ if it can be obtained from the basis by a finite number of applications of the recursive step.

a) Prove $L \subseteq \{a^ib^j \mid i \geq j \geq 0\}$

Let $w \in L$. If $w \in L_0$, then $w \in \{\lambda, a, b, c\}$, all of which are in $\{a^ib^j \mid i \geq j \geq 0\}$ with $i, j$ or both 0.

Assume by induction that $L_{i-1} \subseteq \{a^ib^j \mid i, j \geq 0\}$ and $\{a^ib^j \mid i, j \geq 0\}$. If $w \in L_i$, then $w = au$ or $w = ub$ for $u \in L_{i-1}$. So $u = a^ib^j$ or $u = a^icb^j$, hence $w \in \{a^{i+1}b^j, a^ib^{j+1}, a^ib^j+1\}$, all of which are in $\{a^ib^j \mid i, j \geq 0\}$.

So by induction $L \subseteq \{a^ib^j \mid i, j \geq 0\}$.

b) Prove $\{a^ib^j \mid i, j \geq 0\} \cup \{a^ib^j \mid i, j \geq 0\} \subseteq L$

Let $w = a^ib^j$ and $w' = a^icb^j$. We want to show $w \in L$.

We use induction on $n = i + j$.

Base case: if $i + j = 0$, then $i = j = 0$, and $w = \lambda$ and $w' = c$ are both in $L_0 \subseteq L$.

Induction hypothesis: The elements of $\{a^ib^j \mid i, j \geq 0\} \cup \{a^ib^j \mid i, j \geq 0\}$ in $L$ for $i + j = n - 1$.

Let $i + j = n > 0$.

So $a^{-1}b^j$, $a^ib^j$, $a^icb^j$ and $a^ib^j$ are all in $L$ by the inductive hypothesis if they are defined, that is, if $i - 1$ and $j - 1$ are not negative, and they cannot both be negative since $i + j > 0$. By the recursive step, $w$ and $w'$ are in $L$. 

1 of 9
c) What is the set $L_2$ generated by the recursive definition.

$L_0 = \{\lambda, a, b, c\}$,
$L_1 = \{\lambda, a, b, c, aa, ab, ac, bb, bc\} = \{\lambda, a, b, c, aa, ab, ac, bb, cb\}$,
$L_2 = \{\lambda, a, b, c, aa, ab, ac, ab, bb, bc, aaa, aab, aac, abb, acb, aab, abb, acb, bbb, cbb\}$.

3. Let $L$ be the subset of $\{a, b\}^*$ consisting of those strings which contain either $ab$ or $ba$.

a) Give a regular expression for $L$.

The string either starts with $a$ or starts with $b$, and is in the set as soon as the other letter appears. $(a^*b \cup b^*a)(a \cup b)^*$.

b) Give a regular grammar for $L$.

[Hint: It helps if your regular expression in part a is very simple. If it isn’t, it is better to start from scratch.]

$$
G : S \rightarrow aA \mid bB \\
A \rightarrow aA \mid bC \mid b \\
B \rightarrow bB \mid aC \mid a \\
C \rightarrow aC \mid bC \mid a \mid b
$$

Note that since the empty string is not in the language, we do not need $\lambda$-rules.

c) Find a Deterministic Finite Automaton which recognizes $L$. 
4. Find the minimum Deterministic Finite Automaton which is equivalent to

\[ q_0 \xrightarrow{a} q_1, q_0 \xrightarrow{b} q_6, q_1 \xrightarrow{b} q_2, q_1 \xrightarrow{a} q_4, q_2 \xrightarrow{b} q_3, q_2 \xrightarrow{a} q_5, q_3 \xrightarrow{a} q_4, q_4 \xrightarrow{a} q_6, q_6 \xrightarrow{a} q_0, q_6 \xrightarrow{b} q_4 \]

First, without applying the algorithm, states \( q_4 \) and \( q_5 \) are indistinguishable since the \( a \) and \( b \) transitions map to the exact same states. So the automaton is equivalent to:

\[ q_0 \xrightarrow{b} q_1, q_1 \xrightarrow{a} q_2, q_1 \xrightarrow{b} q_2, q_2 \xrightarrow{b} q_3, q_2 \xrightarrow{a} q_4, q_3 \xrightarrow{a} q_5, q_4 \xrightarrow{a} q_6, q_6 \xrightarrow{a} q_0, q_6 \xrightarrow{b} q_4 \]

Initially, all states are inequivalent to the final state \( q_3 \), so they those pairs are marked by a 1, illustrated by an edge in the diagram on the left below. Applying \( b \) to states \( q_1 \) and \( q_2 \), applying \( a \) to states \( q_4 \) and \( q_56 \), and applying to \( a \) to states \( q_2 \) and \( q_1 \), yields pairs of distinguishable states, and which are also marked with a 1, giving 3 new edges in the diagram.

Applying \( b \) to \( q_0 \) and \( q_1 \), a to \( q_0 \) and \( q_56 \), \( b \) to \( q_1 \) and \( q_4 \), and a to \( q_2 \) and \( q_56 \) give distinguishable pairs, adding 4 new edges.

Lastly, applying \( a \) to \( q_1 \) and \( q_56 \) shows the last pair are distinguishable, and minimal diagram has 6 states.

[Note: Keeping track of the distinguishable states in the algorithm marking a graph is only practical for automata with very few states. For 8 or more is seems better to use a grid, like we did in class, or 16 or more, program a computer.]
6. Let $G$ be the grammar given by

$$G : S \rightarrow A \mid B$$

$$A \rightarrow abA \mid \lambda$$

$$B \rightarrow aBb \mid \lambda.$$ 

a) Give a set theoretic description of $L(G)$.

$$\{(ab)^j \mid j \geq 0\} \cup \{a^j b^j \mid j \geq 0\}$$

b) Show the grammar is ambiguous.

$ab$ has to rightmost derivations:

$$S \Rightarrow A \Rightarrow abA \Rightarrow ab$$

and

$$S \Rightarrow B \Rightarrow aAb \Rightarrow ab$$

c) Show that there is no regular expression for $L(G)$.

All strings in $L$ have $n_a = n_b$.

Let $K$ be the number of states in an automaton $M$ with $L_M = L$. Consider the string $a^K b^K$. The pumping lemma says that $a^K b^K$ can be factored as $uvw$ with length$(uv) \leq K$ $v$ a pumpable string. So $uv$ consists only of $a$’s, and pumping $v$ gives a string with more $a$’s than $b$’s, but such a string is not in $L$.

So $L$ is not regular.

7. Use expression graphs to find a regular repression for the Language of the following incompletely deterministic Finite Automaton.

To algorithm requires us to start with an automaton with a single final state, which we have.

To delete a state, which must and labeled edges for every pair consisting of an edge into that state, and an edge out of it. The easiest one to delete is $q_2$.

$q_1$ has two incoming and two outgoing edges, so there are four edges to be added, two of which are loops.
(because of font problems, the unions are indicated by commas.)

$q_3$ also has to incoming, and two outgoing edges, but it also has a loop labeled $cc$, so $(cc)^*$ must be in the middle of every new edge label.

From this we can now read off the regular expression

$$[(b^2 \cup (cb \cup bc)(bb)^*cb)(cb \cup bc)(cc)^*a][((a(cc)^*a) \cup (a(cc)^*cb)(b^2 \cup (cb \cup bc)(bb)^*cb)(cb \cup bc)(cc)^*a]^*$$

8. Let $G$ be the grammar given by

$$\begin{align*}
G : S & \rightarrow AB | BCS \\
A & \rightarrow aA | C \\
B & \rightarrow bb | b \\
C & \rightarrow cC | \lambda
\end{align*}$$

a) Convert $G$ to an equivalent essentially non-contracting grammar.

There is only one $\lambda$-rule, $\text{Null}(G) = \{A, C\}$. So

$$\begin{align*}
G'' : S & \rightarrow AB | BCS | B | BS \\
A & \rightarrow aA | C | a \\
B & \rightarrow bb | b \\
C & \rightarrow cC | c
\end{align*}$$

And since $S$ is not nullable, we do not add $S \rightarrow \lambda$.

b) If there are any chain rules in your answer to part a), covert to an equivalent essentially non-contracting grammar without chain rules.

$\text{Chain}(S) = \{S, B\}$ $\text{Chain}(A) = \{A, C\}$ $\text{Chain}(B) = \{B\}$ $\text{Chain}(C) = \{C\}$ So

$$\begin{align*}
G'' : S & \rightarrow AB | BCS | Bb | b | BS \\
A & \rightarrow aA | cC | c | a \\
B & \rightarrow bb | b \\
C & \rightarrow cC | c
\end{align*}$$
c) Convert the Chomsky normal form.

In $G''$ there is only one right hand side of length greater than 2, so

$$
G'' : S \rightarrow AB \mid BD \mid BJ_b \mid b \mid BS \\
A \rightarrow J_a A \mid J_c C \mid c \mid a \\
B \rightarrow J_b B \mid b \\
C \rightarrow J_c C \mid c \\
D \rightarrow CS \\
J_a \rightarrow a \\
J_b \rightarrow b \\
J_c \rightarrow c
$$

8. Remove the left recursion from the following rule:

$$
A \rightarrow ABC \mid ACD \mid AAA \mid BC \mid DAACA
$$

The right hands sides $\{ABC, ACD, AAA\}$ are left recursive, and $\{BC, BAACA\}$ are “escapes”.

So

$$
A \rightarrow BC \mid DAACA \mid BCA' \mid DAACAA' \\
A' \rightarrow BC \mid CD \mid AA \mid BCA' \mid CDA' \mid AAA'
$$

has no left recursion.


First note that $q_1$ can be eliminated and replaced with an $a$ arrow from $q_0$ to $q_3$. 

6 of 9
Second, note that states $q_5$ and $q_3$ are indistinguishable, so the automaton is equivalent to the four state machine

We can now more quickly compute $t$.

<table>
<thead>
<tr>
<th>$\lambda$ - closure</th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_0$</td>
<td>${q_0, q_2, q_3}$</td>
<td>${q_2, q_3}$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>${q_2, q_3}$</td>
<td>${q_2, q_3, q_4}$</td>
</tr>
<tr>
<td>$q_3$</td>
<td>$q_3$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>$q_4$</td>
<td>$q_4$</td>
<td>$q_4$</td>
</tr>
</tbody>
</table>

And we check that the last two columns are unions of $\lambda$-closures, which they are. With $t$ we can compute a deterministic machine:

Although it was not asked, this deterministic machine is not the minimum state machine since $q_{234}$, $q_{34}$ and $q_4$ are all indistinguishable, (any string from them lands on an accepting state.) So the machine reduces to
and we see that the language is the set of strings which are either empty, start with a, or consist only of b’s.

10. Construct a Push Down Automaton which accepts

\[ \{a^i b^j c^k \mid (i = j) \lor (j = k)\} \]

It is easiest to specify by state diagram:

![State Diagram](image)

Notice it is non-deterministic, and has plenty of \(\lambda\)-transitions. (In the diagram \(\lambda\) is \(l\)).

11. Let the language \(L\) be defined by

\[ L = \{a^i b^j c^k \mid i + k = j\} \]

a) Show that there is no regular expression for \(L\).

For this we use the ordinary pumping lemma. Let \(L\) be recognized by an automaton with \(N\) states. The \(a^N b^i c^k\) with \(N + k = j\) factors into \(uvw\), with the length of \(uv\) less at most \(N\) and \(v\) a pumpable string. So the pumpable string consists of only a’s, and so we can increasing the number of a’s only, violating the contraint \(i + k = j\). So \(L\) is not regular.

b) Construct a Push Down Automaton which accepts \(L\).

![State Diagram](image)

11. Show that there is no Push Down Automaton which accepts

\[ \{a^i b^{2i} a^i\} \]

For this we need the pumping lemma for context free grammars.

\(L\) has words of arbitrary length, so we can find one \(a^i b^{2i} a^i = uvwxy\) where \(v\) and \(x\) are simultaneously pumpable and at least one has non-trivial length. Neither \(v\) nor \(x\) can have two different letters, so, since there are at most two pumpable factors, one of the three factors of \((a^i)(b^{2i})(a^i)\), hence all three are unchanged, and the word cannot be pumped.
Thus the language is not context free.

11. Show that there is no Push Down Automaton which accepts

$$\{a^n \mid n = k^2\}$$

For this we need the pumping lemma for context free grammars.

There is a number $K$ so that every string of length at least $K$ can be factored as $uvwxy$ where $v$ and $x$ are simultaneously pumpable and at least one has non-trivial length, and total two pumpable strings is at most $K$.

So consider $a^{K^2} = uvwxy$ with length$(vx) = j \leq K$. Then $uv^2wx^2y = a^{K^2+j}$. and $K^2 + j \leq K^2 + K < K^2 + 2K + 1 = (K+1)^2$. So $K^2 + j$, laying between $K^2$ and $(K+1)^2$ is not a perfect square, and so $a^{K^2}$ is not pumpable, so the language is not context free. 
