1. (15 pts) Let \( L \) be the language with definition

\[
L = \{a^i b^j c^k \mid 1 \leq j \leq 3, i = k\}.
\]

a) Give a recursive definition of \( L \).

A recursive definition is:

BASIS: \( b, b^2, b^3 \in L \)

RECURSIVE STEP: If \( u \in L \) then \( auc \in L \).

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

Following a recursive definition, our grammar could be

\[
G : S \rightarrow aSb \mid aSc \mid X
\]

\[
X \rightarrow b \mid bb \mid bbb
\]

c) Construct a pushdown automaton which accepts \( L \).

The state diagram is

```

q0 \( \xrightarrow{l/1} \) q1 \( \xrightarrow{a \cdot l/A} \) q2 \( \xrightarrow{b \cdot l/l} \) q3
\( \xrightarrow{l/l} \) q4
```

```
2. (10 pts) Let $L$ be the language with recursive definition

Basis: $\lambda \in L$.

Recursive Step: If $u \in L$ then $a^2u$ and $ub^3$ 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) Find $L_3$

$L_0 = \{\lambda\}$.
$L_{i+1}$ is result of applying the recursive step in all possible ways to $L_i$. The recursive definition says

$$L = \bigcup L_i$$

After the first application of the recursive step we have $L_1 = \{a^2, b^3\}$.
After the second application of the recursive step we have $L_2 = \{a^4, a^2b^3, b^6\}$.
After the third application of the recursive step we have

$$L_3 = \{a^6, a^4b^3, a^2b^6, b^9\}.$$

b) Prove $L \subseteq \{a^{2i}b^{3j} \mid i, j \geq 0\}$

We show this inductively, on the recursively defined set. 
[Part a) was a hint that it is much easier if you use the sets $L_i$ generated from the recursive definition.]

$L \subseteq \{a^{2i}b^{3j} \mid i, j \geq 0\}$ Since $L$ is recursively defined, we show $L_i \subseteq \{a^{2i}b^{3j} \mid i, j \geq 0\}$ for all $i$.

Base Case: $L_0 = \{\lambda\}$, and $L = a^{2i}b^{3j}$ taking $i = j = 0$, so $L_0 \subseteq \{a^{2i}b^{3j} \mid i, j \geq 0\}$.

Inductive Step: Assume $L_i \subseteq \{a^{2i}b^{3j} \mid i, j \geq 0\}$ for some $i$. We want to show $L_{i+1} \subseteq \{a^{2i}b^{3j} \mid i, j \geq 0\}$.

Let $w \in L_{i+1}$. We have $w$ is the recursive step for some $u \in L_i$, and $u = a^{2i}b^{3j}$ for some $i, j \geq 0$ by the inductive hypothesis.

So either $w = a^2u$ or $w = ub^3$.

$a^2u = a^2a^{2i}b^{3j} = a^{2(i+1)}b^{3j} \in \{a^{2i}b^{3j} \mid i, j \geq 0\}$ and $ub^3 = a^{2i}b^{3j}b^3 = a^{2i}b^{3(j+1)} \in \{a^{2i}b^{3j} \mid i, j \geq 0\}$.

So $w \in \{a^{2i}b^{3j} \mid i, j \geq 0\}$, and $L_{i+1} \subseteq \{a^{2i}b^{3j} \mid i, j \geq 0\}$, as required.

So $L_i \subseteq \{a^{2i}b^{3j} \mid i, j \geq 0\}$ for all $i$ by induction.

So $L = \{a^{2i}b^{3j} \mid i, j \geq 0\}$. 

2 of 8
3. (15 pts) Give regular expressions for each of the following subsets of \( \{a, b, c\}^* \).

a) Strings for which the number of a’s is even.

\[
((a(b \cup c)^*a) \cup b \cup c)^* 
\]

b) Strings for which each b is followed by at least one a or at least 2 c’s.

\[
(a, c, ba, bcc)^* 
\]

c) Strings which either start with a or have an even number of letters.

\[
(a(a \cup b \cup c)^* \cup ((a \cup b \cup c)(a \cup b \cup c)))^* 
\]
4. (15 pts) Let $L$ be the subset of $\{a, b\}^*$ consisting of those strings which contain $abb$ and $bba$. [Note: $bbabb$ is in this language.]

a) Give a regular expression for $L$.

Both subwords must occur, perhaps with intersection. One must occur first. The expression $(abb(a, b)^* bba, bbabb, abba)$ gives strings with prefix $abb$ and suffix $bba$. Similarly he expression $(bba(a, b)^* abb, bbabb)$ gives strings with prefix $bba$ and suffix $abb$. Altogether we can take

$$((a, b)^* (abb(a, b)^* bba, bbabb, abba) \cup (bba(a, b)^* abb, bbabb)) (a, b)^*$$

b) Give a regular grammar for $L$.

We will use the following variables

$S$ - prefix empty  
$A$ - prefix prefix is just $b$  
$B$ - prefix is $b^2 b^*$  
$C$ - prefix is $(b^2 b^*)(ab)^* a$  
$D$ - prefix is $(b^2 b^*)(ab)^* ab$  
$E$ - prefix has only isolated $b$’s and ends in $a$.  
$F$ - prefix has only isolated $b$’s and ends in $ab$.  
$G$ - prefix has only has $ab^2$, but no $b^2 a$, and ends in $b^2$.

$$G : S \rightarrow aE \mid bA$$
$$A \rightarrow aE \mid bB$$
$$B \rightarrow aC \mid bB$$
$$C \rightarrow aC \mid bD$$
$$D \rightarrow bZ \mid aC$$
$$E \rightarrow aE \mid bF$$
$$F \rightarrow aE \mid bG$$
$$G \rightarrow aZ \mid bG$$
$$Z \rightarrow aZ \mid bZ \mid a \mid b$$

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 $L$. 

![Deterministic Finite Automaton](image-url)
5. (15 pts) Consider the following deterministic finite automaton.

![Automaton Diagram]

a) Show that states $q_1$ and $q_5$ are distinguishable.

This is very simple. Since $\delta(q_1, a) = q_4$ is non-accepting and $\delta(q_4, a) = q_5$ which is accepting, $q_1$ and $q_5$ are distinguishable.

[Note: You can’t show indistinguishability by saying different words applied at the states result the same state is irrelevant. So the fact that $\delta(q_5, a) = \delta(q_1, b)$ does not show distinguishability.]

b) Find the minimal equivalent deterministic finite automaton.

Clearly $q_2$ and $q_3$ are indistinguishable, since any $w$ transitions from them to an accepting state.

Also $q_1$ and $q_4$ are indistinguishable, since both $a$ and $b$ transition them to the exact same states, so regardless of whether $w$ starts with $a$ or $b$, $\hat{\delta}(q_1, w) = \hat{\delta}(q_4, w)$.

So the given automaton is equivalent to

![Diagram of Equivalent Automaton]

We have already shown $q_1$ and $q_5$ are distinguishable, and accepting state is distinguishable from all the rest. So we only have to check $q_0$ versus $q_{14}$, which are distinguishable by $b$, and $q_0$ versus $q_5$, which are distinguishable by $a$.

c) Find a regular expression for the language of the automaton.

$$(ab^*a \cup ba^*b)(a \cup b)^*$$
6. (15 pts) Let \( G \) be the grammar given by

\[
G : S \to aAb | aBb^2 | aCb^3 | \lambda \\
A \to aAb | \lambda \\
B \to aBb^2 | \lambda \\
C \to aCb^3 | \lambda.
\]

a) Show that \( 3n_a \geq n_b \).

We will prove by induction. Every sentential form has a prefix consisting only of \( a \)'s, at most a single variable, and a suffix consisting only of \( b \)'s, and \( 3n_a \geq n_b \).

The sentential form \( S \) has \( n_a = n_b = 0 \), so the statement is true in the base case.

Suppose the statement is true for \( w \). Every rule adds exactly one \( a \) to the prefix, and at most 3 \( b \)'s to the suffix, so the form is preserved, and the new form \( w' \) has \( n_a(w') = n_a(w) + 1 \) and \( n_b(w) + 1 \leq n_b(w') \leq n_b(w) + 3 \).

So \( n_b(w') \leq n_b(w) + 3 \leq 3n_a(w) + 3 = 3(n_a(w) + 1) = 3n_a(w') \), as required.

So the statement is true by induction.

b) Show whether or not the grammar is ambiguous.

The grammar is not ambiguous. The elements of the language are either of the form \( a^i b^i \), which can only be derived by \( i \) applications of the \( A \) rule, or \( a^i b^{2i} \), which can only be derived from \( i \) applications of the \( B \) rule, or \( a^i b^{3i} \), which can be derived from \( i \) application of the \( C \) rule.

c) Use the Pumping Lemma to show that there is no regular expression for \( L(G) \).

All sentential forms are of the form \( a^i V b^i \), for some variable \( B \), and satisfy \( n_a \leq n_b \).
This is true of the empty string, and is preserved under every rule application.

Let \( K \) be the number of states in an automaton \( M \) with \( L_M = L \). Consider the string \( a^K b^K \). This string is in \( L \) derived using only \( A \) rules. The pumping lemma says that \( a^K b^K \) can be factored as \( uvw \) with \( \text{length}(uw) \leq K \) \( v \) 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. **(5 pts)** 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 one incoming and two outgoing edges, so there are two edges to be added, one one which is a loop at $q_0$.

$q_3$ 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

$$(b^2)^*(cb \cup bc)a(a^2)^*$$
8. (10 pts) Let $G$ be the grammar given by

$$
G : S \rightarrow AB \mid BCS \\
A \rightarrow aA \mid C \\
B \rightarrow bB \mid b \\
C \rightarrow cC \mid \lambda
$$

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

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

$$
G'' : S \rightarrow AB \mid BCS \mid B \mid BS \\
A \rightarrow aA \mid C \mid a \\
B \rightarrow bB \mid b \\
C \rightarrow cC \mid c
$$

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.

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

So

$$
G'' : S \rightarrow AB \mid BCS \mid Bb \mid b \mid BS \\
A \rightarrow aA \mid cC \mid c \mid a \\
B \rightarrow bB \mid b \\
C \rightarrow cC \mid c
$$