ComS 331 **Spring 2025** **Name: Mobina Amrollahi**

CheatSheet

1 Regular Languages

1. Give a regular expression, simplified to the best of your abilities, for the language of all strings of a's, b's, and c's that contain an even number of b's.
   Answer: $(a^*c+b)^*a^*$

2. Give a regular expression, simplified to the best of your abilities, for the language of all strings of a's, b's, and c's that contain an even number of b's.
   Answer: $(a+b(a+c)^*b+c)^*$

3. Simplify (if possible) the expression $(a+b)^*c^*(a+b)^*$, then describe as concisely as you can in English the language it defines.
   Answer: The set of all strings of a,b, and c's with at most one run of consecutive c's.

4. Prove or disprove: If A_k is a regular language for each $k \in \mathbb{N}$, then the language

$$A = \bigcup_{k=0}^{\infty} A_k$$

   is also regular.
   Answer: The statement is not true. Consider $A_k = \{0^k1^k \mid k \in \mathbb{N}\}$. Since $\forall k$ each of A_K has a single element, hence they're regular. However,

$$\bigcup_{k=0}^{\infty} A_k = \{0^k1^k \mid \forall k \in \mathbb{N}\}$$

   which is not regular.

5. Recall that a string $x \in \{0,1\}^*$ is a prefix of a string $y \in \{0,1\}^*$, and we write $x \sqsubseteq y$, if there exists $z \in \{0,1\}^*$ such that $xz = y$.

6. Prove or disprove: A finite language can be accepted by a DFA.

7. Prove or disprove: if L_1 is regular, and L_2 is not regular, then $L_1 \cup L_2$ is note regular.
   Answer: The statement is not true. Consider the following examples:
   1. $L_1 = (a+b)^*$, $L_2 = \{a^nb^n \mid n \in \mathbb{N}\}$. Then $L_1 \cup L_2 = (a+b)^*$.
   2. $L_1 = \epsilon$, $L_2 = \{a^nb^n \mid n \in \mathbb{N}\}$. Then $L_1 \cup L_2 = \{a^nb^n \mid \forall n \in \mathbb{N}\}$.

8. Prove or disprove: For each $y \in \{0,1\}^*$, the language

$$A_y = \{x \in \{0,1\}^* \mid x \sqsubseteq y \text{ or } |x| > |y|\}$$

is regular.

Answer: Consider the two following sets:

$$A_{y-prefix} = \{x \in \{0,1\}^* \mid x \sqsubseteq y\} \quad \text{and} \quad A_{long-y} = \{x \in \{0,1\}^* \mid |x| > |y|\}$$

Then we know:

$$A_y = A_{y-prefix} \bigcup A_{long-y}$$

$A_{y-prefix}$ is a finite set and all finite sets are regular. In addition $A_{long-y} = \Sigma^* \backslash (\bigcup_{k=0}^{|y|} \Sigma^k)$ which is the complement of $\bigcup_{k=0}^{|y|} \Sigma^k$. Since $\forall \quad 0 \leq k \leq |y|$, Σ^k is a finite set, they're all regular. Consequently, since regular languages are close under union, $\bigcup_{k=0}^{|y|} \Sigma^k$ is also regular. Hence its complement, which is $\Sigma^* \backslash (\bigcup_{k=0}^{|y|} \Sigma^k$, is also regular. Therefore, A_{long-y} is also regular, and so is A_y.

9. Sometimes, it might be easier to find a state diagram for the complement of such a language, and since regular languages are closed under the complement, we can figure out the state diagram. Examples include:

   a. $\{w \mid w$ doesn't contain the substring $110\}$

   b. $\{w \mid$ w is any string except 11 and $111\}$

   c. $\{w \mid$ w contains an even number of 0s, or contains exactly two $1s\}$

   c. $\{w \mid w$ doesn't contain the substring 0101 (i.e., $w = x0101y$ for some x and y)$\}$

10. Please make sure you understand whether $\epsilon \in L$ or not. Sometimes, it might get tricky. Examples that you need to consider ϵ include:

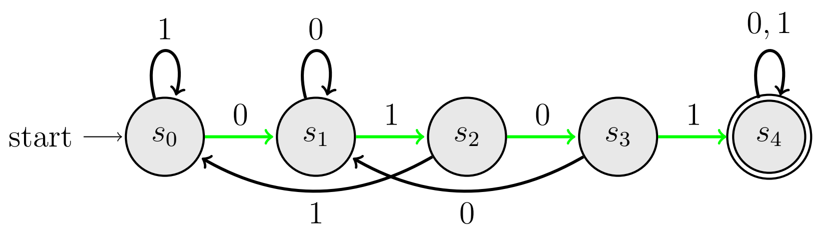
    a. $\{w \mid$ every odd position of w is a $1\}$

    b. $\{w \mid$ w contains an even number of 0s, or contains exactly two $1s\}$

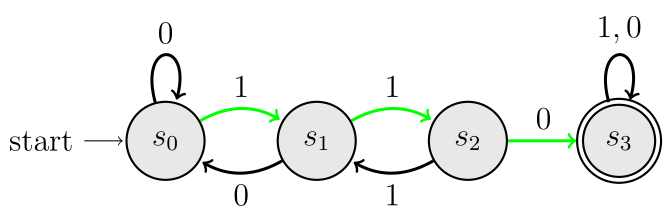
    c. $\{w \mid$ w is any string except 11 and $111\}$

11. Constructing DFAs for the following examples is easier to start with focusing on the substring that the language requires to contain:

    a. $\{w \mid w$ contains the substring 0101 (i.e., $w = x0101y$ for some x and y)$\}$



    b. $\{w \mid w$ contains the substring $110\}$



12. We constructing a DFA or NFA, follow this checklist to make sure you've doing everything right:

(a) Remember to specify the start and final states.

(b) Make the start state apparent, and not looking like an arrow.

13. Consider the n-bit binary representation of a natural number x:

$$(x_{n-1}x_{n-2}\dots x_1x_0)_2 \iff x = \sum_{i=0}^{n-1} x_i 2^i$$

where each bit x_i is a binary digit, either 0 or 1. For example, $(00000101)_2$ is the 8-bit binary representation of the number 5 since: $0{\cdot}2^7+0{\cdot}2^6+0{\cdot}2^5+0{\cdot}2^4+0{\cdot}2^3+1{\cdot}2^2+0{\cdot}2^1+1{\cdot}2^0 = 4+1 = 5$. This is the format normally employed by digital computers to store nonnegative integers. Now, consider the language:

$$\begin{aligned} L =& \{a_0b_0c_0\dots a_{n-1}b_{n-1}c_{n-1} \mid n \in \mathbb{N} \wedge \forall i, 0 \le i < n, a_i \in \{0,1\}, b_i \in \{0,1\}, c_i \in \{0,1\} \wedge \\ & (a_{n-1}\dots a_0)_2 + (b_{n-1}\dots b_0)_2 = (c_{n-1}\dots c_0)_2\} \end{aligned}$$

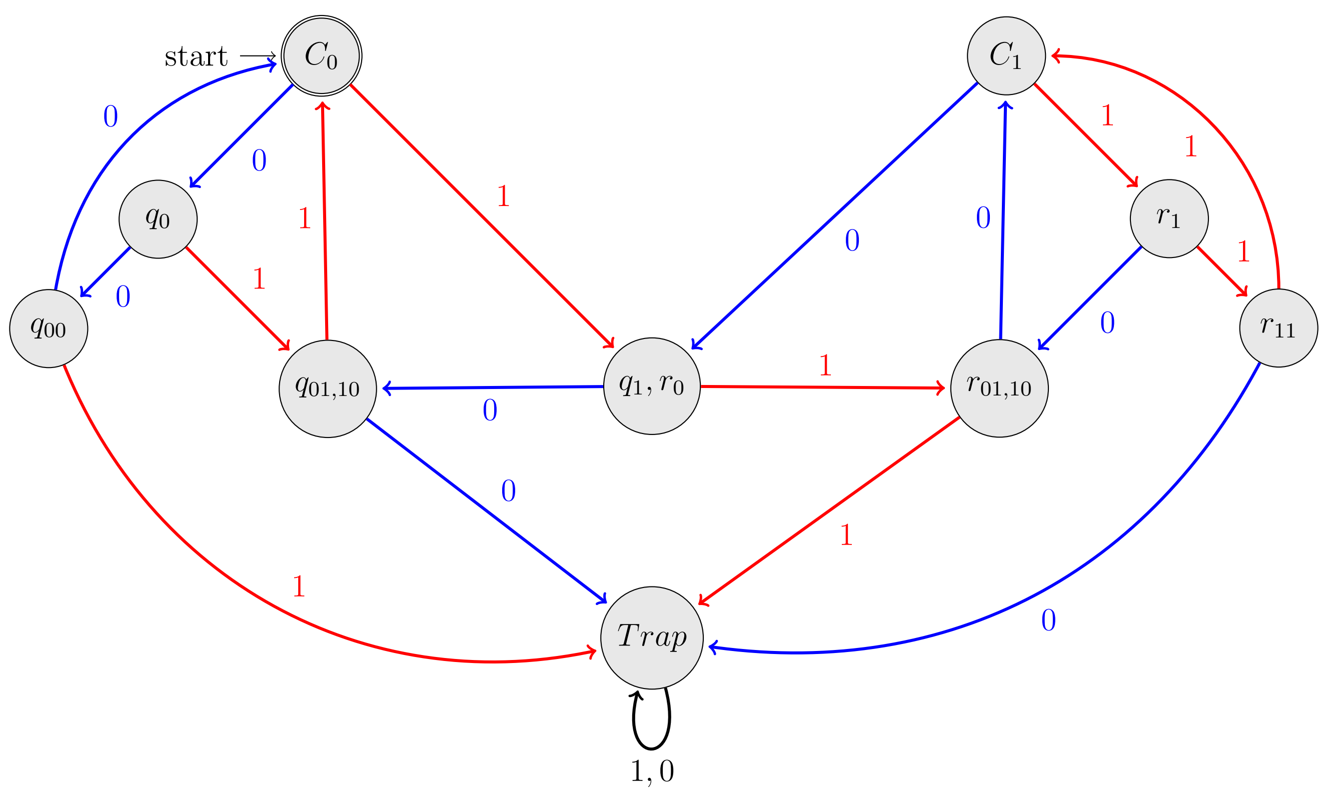
For example, since $5+3=8, 5=(000101)_2, 3=(000011)_2$, and $8=(001000)_2$ then:

$$110 \quad 010 \quad 100 \quad 001 \quad 000 \quad 000 \in L$$

(the string is spaced every three digits for readability's sake only).

Draw a DFA that recognizes L.

Answer



14. Suppose you want to convert a regular expression to an NFA that's not trivial. Firstly, remember the followings:

2 Context-Free Languages

1. Context-Free Languages (CFLs) are closed under union, concatenation, and Kleene star.

2. Every regular language is a CFL. With that being said, it's good to review some of the regular language we know with Context-Free Grammars (CFGs). For instance,

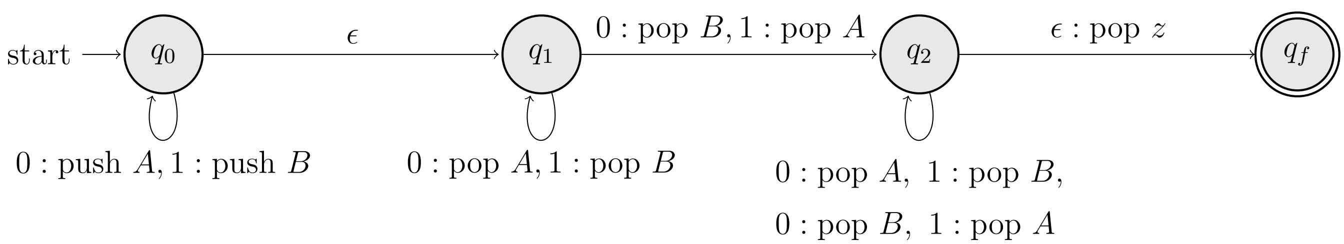
$$L = \{w \mid w \in \{a, b\}^*\}$$

   Consider the following grammar $G = (\{S\}, \{a, b\}, R, S)$ where the set of rules, R, is

$$S \to aS \mid bS \mid \epsilon.$$

3. To give a formal description of the NPDA asserting a CFL, we can use the following template: Let M_1 be $(Q, \Sigma, \Gamma, \delta, q_0, z, F)$, where

$$\begin{aligned} Q &= \{q_0, q_1, q_2, q_f\}, \\ \Sigma &= \{0, 1\}, \\ \Gamma &= \{A, B, z\}, \\ F &= \{q_f\}, \text{ and} \end{aligned}$$



4. To give a formal description of the pumping lemma for CFLs, you can use the following:
   Consider the language

$$L = \{w \in \{a, b\}^* : \text{the longest run of } a\text{'s in } w \text{ is longer than any run of } b\text{'s in } w\}.$$

   Prove that L is not context-free.
   Answer: Proof by contradiction.

   Proof. Assume the contrary that L is context-free. Let p be the given pumping length. Now, consider the string $s = b^p a^{p+1} b^p$. Since the longest run of a has length of $p + 1$, and the longest run of b has length p, hence the longest run of a is longer than any run of b's in s and $s \in L$. Additionally, $\mid s \mid \geq p$, and based on pumping lemma, s may be divided into five pieces s $= uvxyz$ satisfying the conditions:

   (a) for each $i \geq 0$, $uv^ixy^iz \in L$,

   (b) $\mid vy \mid > 0$, and

   (c) $\mid vxy \mid \leq p$.

The restriction $\mid vxy \mid \leq p$ guarantees that v and y can only span at most two out the three runs. That is, whether they span at the first run of b's and a's, and not the second run of b's or the span at the run of a's and second run of b's and not the first run of b's. Hence, no matter what, there will always we at least one of two run of b's that v and y don't span at. With this, there will be two possibilities:

(a) At least one of v or y spans at the run of a's: without the loss of generality, assume at lest v is spanning at the run of a's, and the second run of b's. Then:

$$\underbrace{b^p a^l}_{u} \underbrace{a^{p+1-l} b^p}_{vxyz}$$

In this case, consider the string uv^0xy^0z. Since v contained at least one a, $\mid uv^0xy^0z \mid_a \leq m$. However, the first run of b's has length m, which means the longest run of $a's$ in uv^0xy^0z is not longer than the longest run of b's in this string. Therefore, we arrive at a contradiction that $uv^0xy^0z \notin L$ and conclude that L is not context-free.

(b) v and y only span at a run of b's: without the loss of generality, suppose v and y only span at the second run of b's. Then:

$$\underbrace{b^p a^{p+1} b^l}_{u} \underbrace{b^i}_{v} \underbrace{b^j}_{x} \underbrace{b^k}_{y} \underbrace{b^{p-l-i-j-k}}_{z}$$

In this case, the string uv^2xy^2z is:

$$\underbrace{b^p a^{p+1} b^l}_{u} \underbrace{b^{2i}}_{v^2} \underbrace{b^j}_{x} \underbrace{b^{2k}}_{y^2} \underbrace{b^{p-l-i-j-k}}_{z} = b^p a^{p+1} b^{p+k+i}$$

If $uv^2xy^2z \in L$, then $\mid b^{p+k+i} \mid < \mid a^{p+1} \mid$. However, since $\mid vy \mid > 0$, $i+k > 0$. Consequently, $p+k+i \geq p+1$. Which means the longest run of $a's$ in uv^2xy^2z is not longer than the longest run of b's in this string. Therefore, we arrive at a contradiction that $uv^2xy^2z \notin L$ and conclude that L is not context-free.

$\square$

5. CFLs are not closed under intersection and complementation. However, we need to be cautious with both operations. If L_1 and L_2 are CFLs, the complement or the intersection of them might be a CFL—or might not.
As a simple example, consider a CFL L (over the alphabet Σ) as well as the regular language Σ^* (which is also a CFL since every regular language is a context-free language). Consider, then, that $L \cap \Sigma^* = L$ is once again context-free.
As a more interesting example, consider the complement of $L = \{a^i b^i : i \geq 0\}$, which can be produced by the following grammar:

$$\begin{aligned} S &\rightarrow aSb \mid bY \mid Ya \\ Y &\rightarrow bY \mid aY \mid \varepsilon \end{aligned}$$

is context-free.

6. An interesting problem that I came across:
A language is *prefix-closed* if the prefix of any string in the language is also in the language. Show that every *infinite prefix-closed context-free language* contains an *infinite regular* subset.

Solution.

Let L be an infinite prefix-closed context-free language. Since L is a CFL, the pumping lemma holds. Let p be the pumping length and let s be a string in L longer than p. Then s can be split into $uvxyz$ such that $uv^ixy^iz \in L$ for all $i \geq 0$ and $|vy| \geq 1$.

Since L is prefix-closed, all prefixes of s are also in L. Therefore, $uv^i \in L$ for all $i \geq 0$. Thus, the regular language $uv^* \subseteq L$. If $v \neq \varepsilon$, then uv^* is an infinite regular subset of L, which proves the statement.

If $v = \varepsilon$, then $y \neq \varepsilon$ (since we know $|vy| \geq 1$). In this case, uxy^* is an infinite regular subset of L.

3 Turing Machines

Definition

Call a language Turing-recognizable if some Turing machine recognizes it. It is also called a recursively enumerable language.
Call a language Turing-decidable or simply decidable if some Turing machine decides it. It is also called a recursive language.

1. Closure Properties of Decidable and Turing recognizable Languages:

Operation	Decidable	Turing Recognizable
Union	✓	✓
Intersection	✓	✓
Reversal		
Concatenation	✓	✓
Star	✓	✓
Homomorphism	×	✓
Complementation	✓	×

2. Turing machines and its reasonable variants all have the same power; they recognize the same class of languages.

3. **Enumerators**

   - A language is Turing-recognizable if and only if some enumerator enumerates it.
   - A language is decidable iff some enumerator enumerates the language in the standard string order.

4. Let $B = \{\langle M_1\rangle, \langle M_2\rangle, \ldots\}$ be a Turing-recognizable language consisting of Turing machine descriptions. Then there exists a decidable language C, also consisting of Turing machine descriptions, such that for every machine described in B, there is an equivalent machine in C, and vice versa. Basically, the language of Turing machine descriptions is decidable.

5. Context sensetive grammars:

- for a^{n^2}:

$$
\begin{aligned}
S &\to LAYR \\
ZA &\to aAZ \\
Za &\to aZ \\
ZR &\to AAYR \\
aY &\to Ya \\
AY &\to YA \\
LY &\to LZ \\
YR &\to X \\
aX &\to Xa \\
AX &\to Xa \\
LX &\to \varepsilon
\end{aligned}
$$

- for $a^{n!}$:

$$
\begin{aligned}
S &\to LaYR \\
YR &\to X \\
YR &\to DBR \\
CD &\to DB \\
aD &\to Da \\
aT &\to Ta \\
AT &\to TA \\
LT &\to LZ \\
ZA &\to AZ \\
Za &\to AaZ \\
ZB &\to TC \\
ZC &\to CZ \\
CT &\to TC \\
ZR &\to YR \\
CX &\to X \\
AX &\to Xa \\
aX &\to Xa \\
LX &\to \varepsilon \\
AD &\to Da \\
LD &\to LU \\
UA &\to AU \\
Ua &\to aU \\
UB &\to TC
\end{aligned}
$$

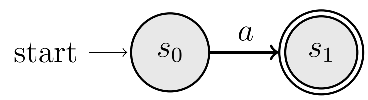


Figure 1: NFA recognizing $L(R) = \{a\}$

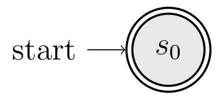


Figure 2: NFA recognizing $L(R) = \{\epsilon\}$

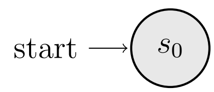


Figure 3: NFA recognizing $L(R) = \emptyset$

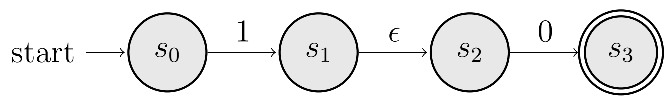


Figure 4: NFA recognizing 10

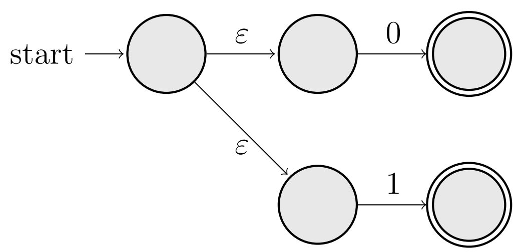


Figure 5: NFA recognizing $1 + 0$

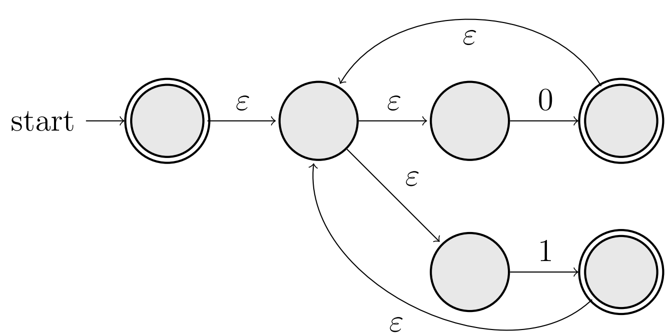


Figure 6: NFA recognizing $(1 + 0)^*$