Figure 1: Finite automaton $M_1$ (on the left) and finite automaton $M_2$ (on the right).

1. Define:

$$A_{NFA} = \{ < M, w > \mid M \text{ is an NFA, } w \text{ is a string and } w \in L(M) \}$$

$$EQ_{DFA,REX} = \{ < M, R > \mid M \text{ is a DFA, } R \text{ is a regular expression, and } L(M) = L(R) \}.$$

Recall that DFA (respectively, NFA) stands for deterministic (respectively, nondeterministic) finite automaton. Let $M_1$ and $M_2$ be the finite automata given in Figure 1 and let $R_1$ be the regular expression $a^*b(a^*b)^*$

Answer the following questions and give reasons for your answers.

(i) Is $< M_1, aba > \in A_{NFA}$? Yes. $M_1$ accepts $aba$.

(ii) Is $< M_1, bab > \in A_{NFA}$? No. $M_1$ does not accept $aba$.

(iii) Is $< M_2, abb > \in A_{NFA}$? Yes. $M_2$ accepts $abb$.

(iv) Is $< M_2, abab > \in A_{NFA}$? Yes. $M_2$ accepts $abab$.

(v) Is $< M_1, R_1 > \in EQ_{DFA,REX}$? No. $bb \notin L(R_1) - L(M_1)$

(vi) Is $< M_2, R_1 > \in EQ_{DFA,REX}$? No. $M_2$ is not a DFA

(vii) Is $< M_1, M_1 > \in EQ_{DFA,REX}$? No. The second input should be a regular expression.

(viii) Is $< M_1, M_2 > \in EQ_{DFA,REX}$? No. The first input should be a DFA.
2. Give a complete construction, that is, a state transition diagram of a **single-tape deterministic** Turing machine $M$ that decides the language

$$\{ b^i c^i \mid i \geq 0 \}$$

- Give also the sequence of configurations that your Turing machine $M$ enters when started on input string $bbc$. (Note: The string $bbc$ should be rejected.)

```plaintext
All undefined transitions go to the reject state. Sequence of configurations:

1. $b b b c \rightarrow 3$  
2. $b c 2 b c \rightarrow 2$  
3. $b c 2 b \rightarrow 2$  
4. $b c 2 \rightarrow 2$  
5. $b c \rightarrow 2$  
6. $b \rightarrow 2$  
7. $\rightarrow$ reject
```
3. (i) (3 marks) Consider the function \( f : \mathbb{N} \to \mathbb{N} \) defined by \( f(n) = (2 \cdot n) - 1 \). (Here \( \mathbb{N} = \{1, 2, 3, \ldots\} \) is the set of positive integers.) Answer the following questions and give reasons for your answers.

(a) Is \( f \) onto?

\[ \text{No. 2 is not an image of any element of } \mathbb{N}. \]

(b) Is \( f \) one-to-one?

\[ \text{Yes. } 2x - 1 = 2y - 1 \text{ implies } x = y. \]

(c) Is \( f \) a correspondence?

\[ \text{No because it is not onto.} \]

(ii) (7 marks) Let \( T = \{(i, j, k) \mid i, j, k \in \mathbb{N}\} \). Show that \( T \) is countable.

The following list contains all elements of \( T \):

- **Stage 1**: List triples \((i, j, k)\) where \( i + j + k = 3 \)
  - (only \((1, 1, 1)\))

- **Stage 2**: List triples \((i, j, k)\) where \( i + j + k = 4 \)
  - \((1, 1, 2), (1, 2, 1), (2, 1, 1)\)

\[ \vdots \]

- **Stage \( x \)**: List triples \((i, j, k)\) where \( i + j + k = x + 2 \)

\[ \vdots \]
4. Give an implementation-level description of a deterministic Turing machine $M$ that decides the following language $A$ over the alphabet $\Sigma = \{ b, c \}$.

$$ A = \{ w \in \Sigma^* \mid |w|_b = 2^{(|w|_c)} \} $$

Above the number of occurrences of symbol $b$ (respectively, $c$) in a string $w$ is denoted as $|w|_b$ (respectively, $|w|_c$). "$2^{(|w|_c)}$" denotes "$2$ to the power $|w|_c$".

If you wish, your deterministic TM $M$ can use more than one tape.

The TM $M$ uses 3 tapes.

$M =$ "On input $w \in \Sigma^*$:

1. Write $\$ \$ to tape 2.

2. Repeat as long as input tape has unmarked $c$'s:
   - Mask the next unmarked $c$ on input tape.
   - Copy contents of tape 2 two times to tape 3.
   - Replace tape 2 contents by tape 3 contents and erase tape 3.

3. After 2. Finisher tape 2 will have 2 symbols $\$'$.

3. By scanning tape 1 and tape 2 in parallel, check
   that the number of $b$'s on tape 1 equals the number of $\$$
   on tape 2. If yes, accept. If no reject."
5. (i) (4 marks) Does the following instance of the Post Correspondence Problem have a match (that is, a solution). Justify your answer.

\[ \{ \frac{baab}{ab}, \frac{bab}{baab}, \frac{baab}{baba}, \frac{ab}{abba} \} \]

Match:

\[
\begin{align*}
\frac{ab}{abba} & \quad \frac{baab}{ab} \\
\end{align*}
\]

(ii) (6 marks) We define the following language
\[
PCP = \{ <P> \mid P \text{ is an instance of the Post Correspondence Problem with a match} \}
\]

Answer the following questions and justify your answers.

(a) Does there exist a decidable language \( A \) such that \( PCP \leq_m A \)?

No. \( PCP \) is undecidable and an undecidable language cannot be mapping reduced to a decidable language.

(b) Does there exist an undecidable language \( B \) such that \( PCP \leq_m B \)?

Yes. \( PCP \leq_m PCP \) via the identity function.

(c) Does there exist a decidable language \( C \) such that \( C \leq_m PCP \)?

Yes. Choose \( C = \{ b \} \) and let \( P_o \) be the PCP instance from (i), and \( P_i \) is the PCP instance \( \frac{ab}{abba} \). Define \( f(b) = \langle P_o \rangle \) and \( f(x) = \langle P_i \rangle \) when \( x \neq b \). Function \( f \) is computable and \( C \leq_m PCP \) via function \( f \).
6. Let

\[ B_{TM} = \{ \langle M \rangle \mid M \text{ is a deterministic Turing machine with input alphabet } \{a, b\}, \text{ such that all strings accepted by } M \text{ begin with } b \text{ and } L(M) \neq \emptyset \} \].

**Without using** Rice’s theorem show that \( B_{TM} \) is undecidable.

We reduce \( A_{TM} \) to \( B_{TM} \). Suppose \( TM \ P \) decides \( B_{TM} \).

Construct a decider \( Q \) for \( A_{TM} \):

\[
Q = \text{"On input } \langle M, w \rangle \text{ where } M \text{ is a TM}
\]

1. Construct \( TM \ M_w \) with input alphabet \( \{a, b\} \)

   \[ M_w = \text{"On input } x \text{:
   1. If } x = b, \text{ accept. Else continue.
   2. Run } M \text{ on } w \text{ and accept if } M \text{ accepts"}
   \]

2. Run \( P \) on \( \langle M_w \rangle \). If \( P \) accepts, reject.
   If \( P \) rejects, accept.

Why this works:

- If \( M \) does not accept \( w \), \( L(M_w) = \{b\} \) and \( \langle M_w \rangle \in B_{TM} \).
- If \( M \) accepts \( w \), \( L(M_w) = \{a, b\}^* \) and \( \langle M_w \rangle \notin B_{TM} \).
7.  (i) (4 marks) In each part circle the correct answer.
   (a) $n^2 = O((\log n)^5 \cdot n)$  \hspace{1cm}  TRUE \hspace{1cm}  FALSE
   \hspace{1cm}  (c) $2^n = o(3^n)$  \hspace{1cm}  TRUE \hspace{1cm}  FALSE
   \hspace{1cm}  (d) $1 = O(n^2)$  \hspace{1cm}  TRUE \hspace{1cm}  FALSE

   (ii) (6 marks) We define
   \[ \text{ALL}_{DFA} = \{ < A > | A \text{ is a DFA with alphabet } \Sigma \text{ and } L(A) = \Sigma^* \}. \]
   Show that \text{ALL}_{DFA} \text{ is in P. In your solution you can assume known that the reachability problem on directed graphs (PATH) is in P.}

   \[ Q \text{ deci} \text{d} \text{ by TM } Q:\]
   \[ Q = " \text{ On input } < A > \text{ where } A \text{ is a DFA} \]
   \[ 1. \text{ Compute the set of states } R \text{ of } A \text{ that are reachable from the start state} \]
   \[ 2. \text{ If } R \text{ contains a non-final state or a state with some transition undefined, reject.} \]
   \[ 3. \text{ If didn't reject yet, accept.} \]
   \[ Q \text{ de} \text{c} \text{is} \text{ ALL}_{DFA} \text{ because } L(A) \neq \Sigma^* \text{ iff some reachable state is non-final or has an undefined transition.} \]
   Step 1 can be done in polynomial time because PATH EA.
   Step 2 involves going through elements of $R$ (subset of states of $A$) and is also polynomial time.
8. Give an example of a non-context-free language $A$ such that $A$ is in the class L ($=$ \(\text{SPACE}(\log n)\)). You should briefly explain how a logarithmic space deterministic TM decides your language $A$. You do not need to prove that $A$ is non-context-free.

$$A = \{a^ib^ic^i \mid i \geq 0\}$$

Deterministic TM for $A$, $Q$ = "On input $w$:

1. Scan $w$ and check that it is in $a^*b^*c^*$
2. Return the tape head to left end.
3. Count the number of $a$'s, $b$'s, and $c$'s, respectively, and store the values in binary on the work tape.
4. By zig-zagging on the work tape, check that all the count values are equal.
(Q stores on work tape 3 binary numbers, each having length at most $\log |w|$.)

9. Let $B$ be NP-complete and assume that $B \leq_p \overline{B}$. (Here $\overline{B}$ is the complement of $B$ and $\leq_p$ is polynomial time reducibility.)

Show that the above implies $\text{NP} = \text{coNP}$.

$B \leq_p \overline{B}$ implies $\overline{B} \leq_p B$ (via the same reduction function)

Let $A \in \text{NP}$ be arbitrary. Since $B$ is NP-complete, we have $A \leq_p B$ which implies $\overline{A} \leq_p \overline{B}$. By transitivity $\overline{A} \leq_p \overline{B}$, we have $\overline{A} \leq_p B$. Since $B \in \text{NP}$, it follows that $\overline{A} \in \text{NP}$, that is, $A \in \text{coNP}$. We have shown that $\text{NP} \subseteq \text{coNP}$.

Converse inclusion:

$A \in \text{coNP}$ implies $\overline{A} \in \text{NP} \subseteq \text{coNP}$ implies $A \in \text{NP}$,

shown above
10. Recall that $\leq_L$ denotes the logarithmic space reducibility. Answer the following questions and justify your answers. Note: To show that an implication does not hold, one should give a counter-example.

(i) Let $B$ be an NP-complete language and $A$ is a language such that $A \leq_L B$. Does this imply that $A$ is in NP?

Yes. $A \leq_L B$ implies $A \leq_P B$ and $\text{NP}$ is closed under $\leq_P$.

(ii) Let $B$ be a context-free language and $A$ is a language such that $A \leq_L B$. Does this imply that $A$ is in NP?

Yes. Context-free languages are in $P \subseteq \text{NP}$ and $\text{NP}$ is closed under $\leq_L$.

(iii) Let $B$ be a context-free language and $A$ is a language such that $B \leq_L A$. Does this imply that $A$ is in NP?

No. Let $M_1$ be a TM s.t. $L(M_1) = \emptyset$ and $M_2$ a TM s.t. $L(M_2) = \{a^n \mid n \geq 1\}$. Define $f$ by setting

$$f(x) = \begin{cases} 1 & x = a \\ 0 & x \neq a \end{cases}$$

$f$ is computable in log-space and $\{a^n \mid n \geq 1\} \leq_L \text{ETM}$ via function $f$. $\text{ETM}$ is undecidable and, hence, not in $\text{NP}$.
11. What is the relationship (equal "="; strict inclusion "⊊" or "⊋"; inclusion that is not known to be strict "⊊" or "⊋") between the following pairs of complexity classes. \textbf{Justify} your answers.

(i) \text{TIME}(n^3) \text{ and } \text{TIME}(n^3 \cdot \log n)

\[ n^3 \text{ is not } o\left(\frac{n^3 \cdot \log n}{\log (n^3 \cdot \log n)}\right) \text{ and time hierarchy theorem does not apply.} \]

(ii) \text{SPACE}(n^3 + n^2 \cdot (\log n)^3) \text{ and } \text{SPACE}(n^3 \cdot \log n)

\[ \text{SPACE}(n^3 + n^2 \cdot (\log n)^3) \nRightarrow \text{SPACE}(n^3 \cdot \log n) \text{ because } n^3 + n^2 \cdot (\log n)^3 = o(n^3 \cdot \log n) \text{ and use space hierarchy theorem.} \]

(iii) \text{SPACE}(2^n) \text{ and } \text{SPACE}(2^{n+5})

\[ \text{SPACE}(2^n) = \text{SPACE}(2^{n+5}) \text{ because } 2^{n+5} = 32 \cdot 2^n \]

(iv) \text{TIME}(2^n) \text{ and } \text{TIME}(3^n)

\[ \frac{3^n}{\log 3^n} = \frac{3^n}{(\log 3)^n} \text{ and } \lim_{n \to \infty} \frac{2^n}{3^n} = 0 \text{ and time hierarchy theorem gives strict inclusion} \]

(v) \text{SPACE}(n^5) \text{ and } \text{NSPACE}(n^2 \cdot \log n)

\[ \text{NSPACE}(n^2 \cdot \log n) \leq \text{SPACE}(n^4 \cdot (\log n)^2) \nRightarrow \text{SPACE}(n^5) \text{ switch space hierarchy theorem} \]