1. (a) Recall the definition of the factorial function:

\[ n! = n \times (n - 1) \times \ldots \times 2 \times 1 \]

Prove that \( L! \) (as defined below) is not in the class of context free languages \( \mathcal{L}_2 \).

\[ L! = \{ a^{n!} \mid n \in \mathbb{N} \setminus \{0\} \} \]

(b) A concurrent transition grammar is similar to a context free grammar but has two sets of production rules. The first set of rules map single non-terminals to a string of non-terminals whereas the rules in the second map single non-terminals to strings of terminal symbols. Each set contains exactly one rule from each non-terminal.

A concurrent transition grammar \( G \) can be represented as a 5-tuple \( \langle \Sigma, N, S, P_1, P_2 \rangle \), where:

- \( \Sigma \) = the set of terminal symbols
- \( N \) = the set of non-terminal symbols
- \( S \) = the start symbol (\( S \in N \))
- \( P_1 \in N \rightarrow N^* \)
- \( P_2 \in N \rightarrow \Sigma^* \)

i. We say that \( \beta \) is immediately concurrently derivable from \( \alpha \) (\( \alpha, \beta \in N^* \)), written as \( \alpha \Rightarrow \beta \) if each non-terminal in \( \alpha \) is replaced according to the rules in \( P_1 \).

For example, if \( P_1 = \{ A \rightarrow AB, B \rightarrow A \} \), \( AA \Rightarrow ABAB \) and \( BA \Rightarrow AAB \).

Formally define \( \Rightarrow \).

ii. We say that \( \alpha \) concurrently resolves to \( \beta \) (\( \alpha \in N^*, \beta \in \Sigma^* \)), written as \( \alpha \Rightarrow_r \beta \) if each non-terminal in \( \alpha \) is replaced according to the rules in \( P_2 \).
For example, if $P_2 = \{ A \rightarrow a, B \rightarrow ba \}$, $AA \Rightarrow_r aa$ and $BA \Rightarrow_r baa$.

Formally define concurrent resolution.

iii. A string $x \in \Sigma^*$ is said to be in the language generated by grammar $G = \langle \Sigma, N, S, P_1, P_2 \rangle$ if we can perform a number of immediate concurrent derivations starting from $S$ followed by a single concurrent resolution to reach $x$. Formally define $L(G)$, the class of all such strings.

2. Forced termination non-deterministic finite state automata (FT-NFSA) are different from standard non-deterministic finite state automata (N FSA) in that termination is forced once a terminal state is reached. In other words, final states may have no transitions going out of them.

   (a) Formalise FT-NFSA and define the language of strings accepted by this class of automata.

   (b) Give a construction to show that FT-NFSA are equivalent to NFSA, provided that $\varepsilon$ is not in the language recognised by the automata.

   (c) Hence or otherwise prove that there is no FT-NFSA which accepts the following language:

   $$ \{ww \mid w \in \{a, b\}^+ \} $$

3. (a) Prove that the following problem is $NP$-complete:

   **Longest Path (LP)**

   **Instance:** Graph $G = (V, E)$, positive integer $k \leq |V|$.

   **Question:** Does $G$ contain a simple path (a path encountering no vertex more than once) with $k$ or more edges?

   (b) Consider the following problem:

   *Given a set of web-pages and links between pages, is there a subset of not more than $n$ pages such that each web-page in the original set is reachable in not-more than $k$ clicks from this set of pages?*

   Formalise the problem and prove that it is $NP$-complete. You may assume that **Dominating Subset** (See section 10.4 of the notes) is $NP$-complete.

   Clearly state any results proved in the course notes which you use.