UNIVERSITY OF TECHNOLOGY, JAMAICA

SCHOOL OF COMPUTING AND INFORMATION TECHNOLOGY

COMPUTER SCIENCE MAJOR
THEORY OF COMPUTATION (CIT3006)

GROUP PROJECT (20%)

Documentation Due Date: April 4, 2025 Interview: Week 13

Group size of 3-4 students

DNA Pattern Detection Using Finite Automata

Objective: Students will work in groups to design and implement a program that uses deterministic finite automata (DFA) to find specific patterns in DNA sequences. The project aims to demonstrate the practical application of finite automata in solving real-world problems such as those encountered in bioinformatics.

Note: This project is a simulation intended for educational purposes and does not represent actual diagnostic tools or medical advice.

Background Information: DNA sequences are strings composed of four nucleotides: Adenine (**A**), Thymine (**T**), Cytosine (**C**), and Guanine (**G**). These sequences encode genetic information through triplets of nucleotides called codons, where each codon corresponds to an amino acid or a signal during protein synthesis. Key patterns in DNA sequences, such as motifs, start codons, or regulatory sequences, play essential roles in biological processes. For example, the codon **ATG** signals the start of translation, while specific sequences like **CAG repeats** and certain mutations can indicate diseases. Finite automata provide an efficient way to detect such patterns by constructing a state machine that matches the desired sequence.

Project Description: Each group will develop a program that implements a DFA to search for specific patterns in a DNA sequence:

1. **Start Codon Detection**: The program must locate the codon **ATG**, which marks the beginning of protein synthesis. If **ATG** is not found, the program should output a message stating, "***Start codon not found.***"

2. **Huntington's Disease Gene Detection**: After detecting **ATG**, the program must search for three consecutive repetitions of **CAG**, signalling a Huntington's disease-related gene. Output: "***Huntington's disease gene found.***"

3. **Possible Cancer Mutation Detection**: If the codon **GGT** is followed by **GAT**, the program should output: "***Possible cancer mutation found.***"

4. If neither **CAG repeats** nor the **GGT-GAT** pattern is found after detecting **ATG**, the program should output: "***No significant patterns found.***"

Requirements:

1. **DFA Design**:
    - **States**: Create a state for each character in the target patterns and one start state.
    - **Transitions**: Define transitions for each nucleotide (A, T, C, G) based on the current state and character.
    - **Accept State**: Define states where the full pattern is matched.
2. **Algorithm**:
    - Traverse the DNA sequence character by character.
    - Transition between DFA states based on the input nucleotide.
    - Record the position whenever the DFA reaches an accept state.
3. **Implementation**:
    Use any programming language to:
    - Input the DNA sequence and the patterns.
    - Construct the DFA dynamically based on the input pattern.
    - Traverse the sequence and output the results.
4. **Testing**:
    - Test the program with various DNA sequences and edge cases.
5. **Optimization and Analysis**:
    - Analyse the time complexity of the DFA-based pattern matching (O(n)).
    - Compare it to naive string matching (O(n * m), where m is the length of the pattern).

Deliverables:

1. **Source Code**: Fully commented and functional program code.

2. **Report**: (1) Explanation of the DFA design. (2) Sample input/output and screenshots of the program running. (3) Performance analysis comparing DFA to naive string matching.

3. **Interview and Presentation**: A brief presentation showcasing the DFA design, test results, and program functionality.

Students are reminded to consult the UTech, Ja. Student Handbook for details on academic regulations, including those governing academic misconduct (see Regulation 5).

Late Submission Deductions

- Deduction of 5% of total marks per day for late submission.
- Submissions more than 7 days late will not be graded and will receive a score of 0.

Rubric

Evaluation Criteria	Description	Marks	Breakdown
Correctness (50%) The program accurately detects the required DNA patterns and handles all specified cases.	Accurately detects the start codon (**ATG**) and outputs the correct message if not found.	10	10: Perfect detection; 5-9: Partial detection; 0-4: No detection or incorrect implementation.
	Identifies and outputs the message for three consecutive **CAG repeats**.	15	15: Fully correct; 8-14: Partial functionality; 0-7: No or incorrect results.
	Detects **GGT** followed by **GAT** and outputs the correct message.	15	15: Fully correct; 8-14: Partial detection; 0-7: No or incorrect implementation.
	Outputs "No significant patterns found" or "Start codon not found" when applicable.	10	10: Fully correct; 5-9: Minor issues; 0-4: Incorrect or missing outputs.
Design (20%) DFA is logically designed with appropriate states and transitions.	States and transitions are correctly defined for each pattern.	10	10: All states and transitions are accurate; 5-9: Minor errors; 0-4: Significant design flaws.
	Handles all required patterns effectively.	10	10: Handles all edge cases; 5-9: Partial handling; 0-4: No handling of edge cases.
Efficiency (10%) Program executes efficiently for both small and large DNA sequences.	Demonstrates and implements efficient DFA traversal (O(n)).	7	7: Fully optimized; 4-6: Some inefficiencies; 0-3: Significant inefficiencies.
	Includes analysis of DFA vs. naive string matching (O(n * m)) in the report.	3	3: Thorough analysis; 2: Partial; 0-1: No or incorrect analysis.
Documentation (10%) Clear explanation of the DFA design, implementation, and testing.	Includes DFA diagrams, sample input/output, test results, and performance analysis.	6	6: Comprehensive; 3-5: Adequate; 0-2: Missing or insufficient details.
	Code is well-documented with meaningful comments explaining logic.	4	4: Fully documented; 2-3: Somewhat clear; 0-1: Poor or no documentation.
Interview and Presentation (10%)	Effectively communicates the problem, solution, DFA design, and results with clarity.	10	10: Outstanding; 5-9: Adequate; 0-4: Poor or unclear explanation.