Report for the Workshop

Universidad Distrital Francisco José de Caldas

UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Student:

Alejandro mauricio junco Oviedo 20231020129

Faculty of Systems Engineering

Systems Analysis

Teacher: Carlos Andrés Sierra

13/09/2024

1. System Analysis

1.1 System Description

In this workshop, a system was developed to manage and analyze genetic sequences. The system was implemented using the Model-View-Controller (MVC) pattern and consists of several classes that interact to provide complete functionality.

- **Model (GeneticSequenceManager)**: Responsible for generating, storing, and analyzing sequences. It allows for generating random sequences, calculating their entropy, filtering sequences, and detecting frequent motifs.
- **View (VistaConsola)**: Manages user interaction through the console. It provides methods for displaying messages, reading user input, and showing sequences with their entropy.
- **Controller (Controller)**: Coordinates the interaction between the model and the view. It manages the application's flow, handles user input, and calls appropriate methods from the model and view.

1.2 Objectives of the System

- Generate DNA sequences with random probabilities for bases A, C, G, and T.
- Calculate the entropy of sequences and allow filtering based on an entropy threshold.
- Detect the most frequent motif in the sequences.
- Provide data persistence through file storage.

2. Complexity Analysis

2.1 Algorithmic Complexity

- **Sequence Generation**: Time complexity is O(n·m)O(n \cdot m)O(n·m), where nnn is the number of sequences and mmm is the length of each sequence. This is because for each sequence of length mmm, random selection is performed mmm times.
- **Entropy Calculation**: The complexity is O(m)O(m)O(m) for each sequence, where mmm is the length of the sequence. The entropy calculation requires traversing each character of the sequence to count its frequency.
- **Filtering by Entropy**: The complexity is O(n)O(n)O(n), where nnn is the number of sequences. Each sequence is evaluated to determine if its entropy meets the specified threshold.

2.2 Space Complexity

- **Database**: Uses a HashMap to store sequences, with a space complexity of O(n·m)O(n \cdot m)O(n·m), where nnn is the number of sequences and mmm is the length of each sequence.
- **Entropy and Motifs**: Additional memory is required for storing entropies and motifs, but overall memory usage is dominated by the size of the sequences and the database.

3. Chaos Analysis

3.1 Sensitivity to Parameters

The system shows sensitivity to input parameters such as base probabilities and entropy thresholds. Small variations in probabilities can lead to significant changes in sequence composition and, consequently, in entropy analysis and motif detection.

3.2 System Behavior

The system is deterministic given a set of parameters. However, the generation of sequences and detection of motifs are influenced by random probabilities, which can cause results to vary significantly between runs.

4. Results

4.1 Sequence Generation

The system was able to generate DNA sequences based on random probabilities for bases A, C, G, and T. The sequences were generated consistently with the given specifications.

4.2 Entropy Calculation and Filtering

The entropy calculation and sequence filtering worked as expected. Sequences with entropy lower than the specified threshold were correctly removed from the database.

4.3 Motif Detection

The system correctly detected the most frequent motifs in the analyzed sequences, demonstrating its capability to identify significant patterns in large datasets.

5. Discussion of Results

5.1 System Efficiency

The system demonstrated good efficiency in sequence generation and analysis, although space complexity can be a concern with large data volumes. Filtering and motif detection were also performed efficiently.

5.2 Robustness

The robustness of the system was tested through input validation and data persistence in files. Validations ensure that input parameters are correct, and persistence allows for data recovery in future runs.

5.3 Limitations

The system depends on user input for sequence generation and parameter configuration. Incorrect or improperly formatted inputs can lead to errors, though mechanisms have been implemented to handle these errors.

6. Conclusions

6.1 Achievements

The system successfully met the established objectives, including random sequence generation, entropy calculation, sequence filtering, and frequent motif detection. Data persistence through file storage enhances the system's functionality and utility.

6.2 Recommendations

- **Optimization**: Explore optimization techniques to handle large data volumes more efficiently.
- **Validations**: Continue improving input validations to handle all possible invalid input cases.