1. Systemic Analysis

Objective:

The aim of this project is to generate an artificial database of nucleotide sequences, identify motifs within these sequences, and apply Shannon entropy as a measure of chaos to filter out sequences with excessive repetitions. The process involves:

1. **Database Creation:** Generating an artificial dataset consisting of nucleotide sequences.
2. **Motif Detection:** Developing an algorithm to identify the most frequent motifs of a given size.
3. **Entropy Filtering:** Using Shannon entropy to filter sequences to maintain diversity.

System Overview:

4. **Database Creation:**
    - Generate sequences of length m with nucleotides A, C, G, T.
    - The probability of each nucleotide is parameterized.
    - Save the sequences in a .txt file.
5. **Motif Detection:**
    - Develop an algorithm to find motifs of length s.
    - Evaluate all possible motifs of length s and select the most frequent one, prioritizing those with the highest consecutive repeated bases.
6. **Entropy Filtering:**
    - Apply Shannon entropy to measure and filter sequences based on their diversity.
    - Define an optimal entropy threshold to remove low-diversity sequences.

Implementation Strategy:

7. **Divide and Conquer Strategy:**
    - Split the dataset into smaller chunks.
    - Generate sequences in parallel using a ForkJoinPool for efficiency.
8. **Motif Detection Algorithm:**
    - Use a sliding window approach to iterate through each sequence.
    - Store occurrences of each motif and track the most frequent motif.
    - Optimize by avoiding redundant calculations.
9. **Entropy Filtering:**
    - Compute Shannon entropy for each sequence.
    - Filter sequences based on a predefined entropy threshold.

2. Complexity Analysis

Database Creation:

- **Time Complexity:**
  - Generating each sequence takes *O(m)O(m)*O(m), and generating n sequences results in *O(n×m)O(n \times m)*O(n×m).
  - Saving the database to a file involves writing each sequence, resulting in a similar complexity, *O(n×m)O(n \times m)*O(n×m).
- **Space Complexity:**
  - Storing n sequences of length m requires *O(n×m)O(n \times m)*O(n×m) space.

Motif Detection:

- **Time Complexity:**
  - For each sequence, detecting motifs involves checking all possible substrings of length s. The complexity for a single sequence is *O(m×4s)O(m \times 4^s)*O(m×4s), where *4s4^s*4s represents all possible motifs of size s.
  - For n sequences, the overall complexity becomes *O(n×m×4s)O(n \times m \times 4^s)*O(n×m×4s).
- **Space Complexity:**
  - Storing motifs and their occurrences requires *O(n×m×4s)O(n \times m \times 4^s)*O(n×m×4s) space.

Entropy Filtering:

- **Time Complexity:**
  - Computing Shannon entropy for each sequence involves iterating through each nucleotide base, resulting in *O(m)O(m)*O(m) per sequence. For n sequences, this results in *O(n×m)O(n \times m)*O(n×m).
- **Space Complexity:**
  - Storing entropy values for each sequence requires *O(n)O(n)*O(n) space.

3. Chaos Analysis

Entropy as a Chaos Measure:

Shannon entropy quantifies the randomness or disorder within a sequence. A higher entropy value indicates greater diversity and less predictability, while a lower value suggests more repetition and less diversity.

Entropy Calculation:

For each sequence, compute Shannon entropy using:
*H(S)=−∑i=1kP(Si)log⁡2(P(Si))H(S) = -\sum_{i=1}^{k} P(S_i) \log_2 (P(S_i))*H(S)=−∑i=1k P(Si)log2 (P(Si)) where *P(Si)P(S_i)*P(Si) is the probability of nucleotide *SiS_i*Si in the sequence.

Filter Threshold:

Define a threshold to filter out sequences with low entropy, ensuring the dataset contains only those with a desirable level of diversity.