Sparse Dynamic Programming for RNA Folding

Prisha Priyadarshini

During Spring 2025, I conducted undergraduate research focused on RNA secondary structure prediction using sparse dynamic programming (DP). This project aimed to explore classical RNA folding algorithms and more recent ML-assisted techniques, while connecting biological insight with algorithmic efficiency. The experience challenged me to balance research alongside a heavy semester of Operating Systems, Natural Language Processing, Calculus B, and Logic.

Biological Foundation

RNA is a single-stranded molecule that can fold upon itself to form base-paired helices, resulting in a secondary structure. Understanding transcription (DNA to RNA), translation (RNA to protein), and codon mapping was critical groundwork for this project. Key processes include alternative splicing, intron removal, and the role of ribosomes and tRNA in protein synthesis.

Classical Algorithms (with Explanations)

• **Needleman-Wunsch:**
This global sequence alignment algorithm uses dynamic programming to find the optimal match between two sequences. It constructs a matrix where each cell represents the best alignment score up to that point, allowing for matches, mismatches, and gaps. Traceback from the bottom-right reveals the optimal alignment.

• **Smith-Waterman:**
A local alignment algorithm similar to Needleman-Wunsch, but it only aligns the most similar regions between sequences. It sets negative scoring cells to zero and starts traceback from the highest-scoring cell, yielding the most relevant local alignment.

• **Edit Distance (Levenshtein):**
Calculates how many operations (insertions, deletions, substitutions) are needed to turn one string into another. Useful for measuring sequence similarity. Operates with a dynamic programming matrix using cost functions for each operation.

• **Ukkonen's Algorithm:**
An efficient suffix tree-based algorithm used for pattern matching and identifying common substrings. It incrementally builds suffix trees for each character in the input string, running in linear time.

• **Nussinov's Algorithm:**
Predicts RNA secondary structure by maximizing the number of non-crossing base pairs. It uses a dynamic programming matrix to compute the best folding between each pair of nucleotides. Fills the matrix bottom-up and uses a traceback procedure to determine the pairing.

• **Nussinov-Jacobson:**
Extends Nussinov by assigning energy scores to base pairs, better modeling real RNA

folding thermodynamics. The goal is to minimize the total free energy, producing more biologically accurate structures.

- **Zuker Algorithm:**
  Also predicts RNA secondary structure but focuses on minimizing free energy using the nearest-neighbor model. It builds and traces back a matrix that incorporates energy contributions from stacking pairs, dangling ends, and loops, resulting in more nuanced predictions.

Modern & Sparse Approaches

I explored recent improvements in RNA folding:
• Sparse DP: Focuses computation only on valid base pairings to reduce complexity.
• MCTS & LORNA-Fold: Combined reinforcement learning with tree search for folding decisions.
• LinearFold: Uses beam search to reduce runtime from $O(n^3)$ to $O(n)$.
These methods offer scalable alternatives to cubic-time algorithms while sacrificing minimal accuracy.

Learning & Reflection

I used YouTube, research papers, and online tools like GeeksforGeeks and PubMed to guide my learning. Despite the initial difficulty, I successfully learned how to trace back matrices and implement basic folding logic. I'm proud of having managed both advanced coursework and independent research this semester. This project deepened my interest in algorithms and bioinformatics.