Algorithms in Structural Biology

Homework 1 - RNA Folding

MSc Data Science and Information Technologies

Andrinopoulou Christina (DS2200013)

Introduction

This exercise is focused on *RNA Folding*. We use as RNA the sequence AAUACUCCGU-UGCAGCAU and we try to find the optimal secondary structure for this RNA. An RNA structure is optimal when many Watson-Crick bonds are established. In this case, the structure is stable and the energy is minimized. However, there are other bonds that may be created in RNA.

We utilize a simplified version of Zuker's algorithm in order to create an optimal folding.

RNA Folding

The Zuker algorithm is a well-known algorithm for RNA folding. It calculates the energy W of the globally optimal structure and the energy V of different structures. The algorithm creates two tables, one for the W and one for the V of shape *length_of_sequence* $\times$ *length_of_sequence*. It assigns values in these tables based on the formulas below:

$$W(i,j) = \min \begin{cases} W(i-1,j) \\ W(i,j+1) \\ V(i,j) & \text{if i is a structure} \\ \min_k\{W(i,k) + W(k-1,j) : j+1 \leq k \leq i\} \end{cases}$$

$$V(i,j) = \min \begin{cases} s(i,j) + h(i-1,j+1) & \text{hairpin} \\ s(i,j) + W(i-1,j+2) & \text{match} \end{cases}$$

The original version of Zuker's algorithm contains 2 more cases in the formula for the V, but for this implementation, we used a more simple model. The calculation of the hairpin energy $h(i,j)$ is based on the formula

$$h(i,j) = 2(i-j+5)$$

and the calculation of the stem energy $s(i,j)$ is based on the formula

$$s(i,j) = \begin{cases} -4 & \text{Watson-Crick bonds} \\ 0 & \text{GU bonds} \\ 4 & \text{otherwise} \end{cases}$$

Also, the algorithm takes into consideration a constraint about the curvature. This constraint is

$$j+5 > i \Rightarrow V(i,j) = W(i,j) = \infty, i > j$$

The implementation of this algorithm is included in the python notebook *RNA_folding.ipynb*. We created a class with the name *Zuker*, which contains all the functions that implement the aforementioned formulas and combine them. The user should create a *Zuker* object with an RNA sequence and an integer that corresponds to the "distance" in the constraint for the curvature. Then, he/she should call the function *Zuker_algorithm* of the class that activates the full pipeline and prints the matrices W, V and the backtracking.

For the backtracking, we created another matrix that contains the kind of moves that the algorithm followed in every step. This matrix contains the characters: 'L', which corresponds to a left move, 'D', which corresponds to a down move, 'M', which corresponds to a match and 'H', which corresponds to a hairpin. The matrix is constructed while the Zuker algorithm is running and for each case of the W and the V formula, the corresponding character is assigned to the matrix for the path. After the creation, we traverse the matrix starting from the up right cell and we move right or left if the corresponding cell contains the character 'R' or 'L' or we move down and left if the corresponding cell contains the character 'M'. If the cell contains the character 'H', we stop, because we reached a hairpin loop.

We also, create another kind of representation of the folding, using dots and parentheses in order to utilize a tool for visualization of the secondary structure of the RNA. The tool is called *PseudoViewer* and it is an online tool (`http://wilab.inha.ac.kr/pseudoviewer/`).

The W matrix is

	A	A	U	A	C	U	C	C	G	U	U	G	C	A	G	C	A	U
A	inf	inf	inf	inf	inf	12.0	12.0	12.0	12.0	12.0	10.0	10.0	10.0	8.0	4.0	4.0	0.0	-4.0
A	inf	inf	inf	inf	inf	inf	20.0	20.0	18.0	14.0	10.0	10.0	10.0	8.0	4.0	4.0	0.0	-4.0
U	inf	inf	inf	inf	inf	inf	inf	20.0	18.0	14.0	14.0	12.0	12.0	8.0	4.0	4.0	0.0	0.0
A	inf	inf	inf	inf	inf	inf	inf	inf	20.0	14.0	14.0	12.0	12.0	8.0	4.0	4.0	4.0	0.0
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	20.0	12.0	12.0	8.0	4.0	4.0	4.0	4.0
U	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	12.0	12.0	8.0	8.0	8.0	8.0	8.0
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	12.0	12.0	12.0	12.0	12.0	12.0	12.0
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	20.0	16.0	12.0	12.0	12.0
G	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	16.0	12.0	12.0	12.0
U	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	16.0	16.0	14.0	14.0
U	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	14.0	14.0
G	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	18.0
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0
A	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
G	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
A	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
U	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf

The V matrix is

	A	A	U	A	C	U	C	C	G	U	U	G	C	A	G	C	A	U
A	inf	inf	inf	inf	inf	12.0	22.0	24.0	24.0	14.0	10.0	14.0	14.0	14.0	12.0	8.0	8.0	-4.0
A	inf	inf	inf	inf	inf	inf	20.0	22.0	24.0	14.0	10.0	18.0	16.0	16.0	12.0	8.0	8.0	-4.0
U	inf	inf	inf	inf	inf	inf	inf	20.0	18.0	24.0	18.0	14.0	16.0	8.0	8.0	8.0	0.0	8.0
A	inf	inf	inf	inf	inf	inf	inf	inf	20.0	14.0	16.0	24.0	16.0	16.0	12.0	8.0	8.0	0.0
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	22.0	16.0	16.0	16.0	4.0	12.0	12.0	12.0
U	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	18.0	16.0	8.0	12.0	16.0	8.0	16.0
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	12.0	22.0	24.0	16.0	20.0	16.0	16.0
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	22.0	16.0	20.0	16.0	16.0
G	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	22.0	12.0	20.0	14.0
U	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	16.0	22.0	16.0	18.0
U	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	14.0	24.0
G	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0	18.0
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	20.0
A	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
G	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
C	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
A	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf
U	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf	inf

For the backtrack paths and the matrix with the moves, please check the python notebook. One of the foldings that can be formed based on these results is shown in the figure below.

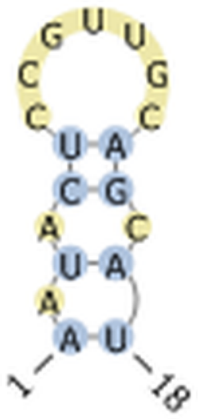

Figure 1: RNA optimal fold based on simplified Zuker's algorithm