

## Project report: Nussinov Folding Algorithm for RNA Secondary Structure Prediction

Sanket Achari UFID: 71096329  
Rushikesh Gawali UFID: 83304331

Project Link: <https://github.com/sanketachari15/Nussinov>

### Algorithm:

Nussinov's algorithm compares given sequence of RNA to itself in order to predict the secondary structure. Dynamic programming is used to calculate the maximum number of matching pairs in the sequence and predict the structure which will be 2D representation of RNA.

Allowed pairs: **A – U & G – C**

### Objective:

Identify maximum number of non-crossing pairs in a given RNA sequence and trace back the sequence to predict the secondary structure of RNA (folding of a RNA sequence)

### Dynamic Programming:

Let S be a given RNA sequence consisting characters A, C, G, U

Let  $\delta(i, j) = 1$  if  $x_i$  and  $x_j$  are complementary i.e. they form a pair  
= 0 otherwise

Initialization:

$dp(i, i-1) = 0$  for  $i = 2$  to  $N$   
 $dp(i, i) = 0$  for  $i = 1$  to  $N$

Recursion:

$$dp(i, j) = \max \left\{ \begin{array}{l} dp(i+1, j), \\ dp(i, j-1), \\ dp(i+1, j-1) + \delta(i, j) \\ \max(dp(i, k) + dp(k+1, j)) \text{ where } i < k < j \end{array} \right\}$$

### Result:

$dp(1, N)$  gives the maximum number of paired bases in the given RNA sequence.

### Backtracking:

We backtrack from  $dp(1, N)$  position till we get diagonal position at which  $dp(i, i)$  is zero.  $Backtrack(1, N)$  will give the secondary structure of RNA sequence.

*Backtrack(i, j):*

```
if i == j
    return S(i)
if i > j
    return ""

if dp(i, j) == dp(i + 1, j)
    return S(i) + Backtrack(i + 1, j)

if dp(i, j) == dp(i, j - 1)
    return Backtrack(i, j - 1) + S(j)

if S(i) == S(j) && dp(i, j) == dp(i + 1, j - 1) +  $\delta(i, j)$ 
    return "(" S(i) + Backtrack(i + 1, j - 1) + S(j) + ")"

if (dp(i, j) == dp(i, k) + dp(k + 1, j)) for every  $i < k < j$ 
    return Backtrack(i, k) + Backtrack(k + 1, j);

return "Backtracking failed"
```

### Example

1. Input RNA sequence: GGGAAAUCC  
Output: Matched pairs: 3  
Secondary Structure after folding: G(G(GAA(AU)C)C)
2. Input RNA sequence: AAUCCCAGGA  
Output: Matched pairs: 3  
Secondary Structure after folding: AA(UC(C(CAG)G)A)
3. Input RNA sequence: AAUCCCAGGAU  
Output: Matched pairs: 4  
Secondary Structure after folding: AA(A(UC(C(CAG)G)A)U)

**Run Instructions:**

Compile: `javac -cp . Nussinov.java`

Run: `java -cp . Nussinov AAAUCCCAGGA`

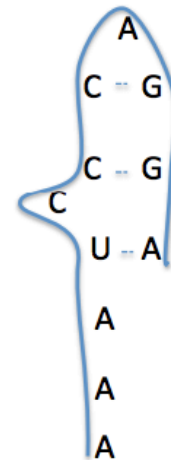
**Ouput:**

Matched pairs: 3

Secondary Structure after folding: AAA(UC(C(CAG)G)A)

Visualization of above example:

	A	A	A	U	C	C	C	A	G	G	A
A	0	0	0	1	1	1	1	1	2	3	3
A	0	0	0	1	1	1	1	1	2	3	3
A		0	0	1	1	1	1	1	2	3	3
U			0	0	0	0	0	1	1	2	3
C				0	0	0	0	0	1	2	2
C					0	0	0	0	1	2	2
C						0	0	0	1	1	1
A							0	0	0	0	0
G								0	0	0	0
G									0	0	0
A										0	0

**Complexity:**

Space complexity:  $O(n^2)$

Time complexity along with backtrack:  $O(n^2)$

**Workload Distribution:**

1. Both of us worked on understanding the algorithm, testing our implementation and the final project report.
2. Rushikesh worked on shortlisting the project idea and the project proposal.
3. Sanket worked on the implementation of the algorithm.

**Conclusion:**

1. We have primarily focussed on the linear(single-core) implementation of the Nussinov algorithm.
2. During our implementation we noticed that the algorithm takes a lot of time especially if the RNA sequence is too long, since it is of the order  $O(n^2)$
3. As a result, we feel a multi-core implementation will help to accelerate the Nussinov algorithm which we will try to implement in the future.

**References:**

1. [A new method for predicting RNA secondary structure](#)
2. [Accelerating Nussinov RNA secondary structure prediction with systolic arrays on FPGAs](#)
3. [Comparison of dynamic programming and evolutionary algorithms for RNA secondary structure prediction](#)
4. <https://www.cise.ufl.edu/~jl3/publications/nussinov>
5. <https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4120147/>