BMS 422, Algorithmic Foundations of Computational Biology, Spring 2021

Single RNA Secondary Structure Prediction using Nussinov-Jacobson Algorithm: Dynamical programming-based algorithm.

Muhammad Elsadany[1]

[1]Biomedical Sciences Program, Computational Biology and Genomics Concentration.

[1]University of Science and Technology, Zewail City, Giza, Egypt, 12578.

Abstract

Motivation: RNA is a complex molecule that has an important role in cellular protein synthesis and is defined to be the second important element after the DNA. RNA secondary structure was discovered by X-Ray methods, which were extremely hard and expensive to be performed for all possible RNA sequences. Thus, the problem of computationally predicting the secondary structure of RNA molecules was first introduced more than thirty years ago and yet continues to be an area of active research and development. The basic RNA-folding problem of finding a maximum cardinality, non-crossing, matching of complimentary nucleotides in an RNA sequence of length n, has an $O(n^3)$ time dynamic programming solution that is widely applied. It is known that an $O(n^3)$ worst-case time solution is possible. This algorithm is known as the Nussinov-Jacobson Algorithm. In this study, the Nussinov Algorithm was implemented using Python programming to predict the secondary structure of 4 different RNA sequences. **Results:** The algorithm implementation code was able to predict the secondary structures of entered RNA sequences with a time complexity of $O(n^3)$. Previous literature showed different comparisons between Nussinov Algorithm and another improved algorithm from it, known as Nussinov-Four Russians Algorithm. The results from literature showed less time complexity in the computation of Four Russians algorithm than Nussinov as it takes $O(n^3/\log(n))$ time complexity. There are several limitations reported to Nussinov Algorithm like that it cannot predict crossing structures and it ignored pseudoknots.

Contact: s-muhammadelsadany@zewailcity.edu.eg

Code Availability: https://github.com/melsadany/Single-RNA-Secondary-Structure-Prediction-using-Nussinov--Jacobson-Algorithm-Dynamical-programming

1. Introduction

RNA is a complex molecule that has an important role in cellular protein synthesis and is defined to be the second important element after the DNA. The RNA structure is used in encoding and decoding genes, as well as regulation of their expression in living organisms. It is constructed of ribose nucleotides connected with phosphodiester bonds, forming strands of varying lengths. The RNA consists of four nitrogenous bases, which are Adenine (A), Guanine (G), Cytosine (C), and Uracil (U). The RNA 3D structure is critical for its stability and function, since bases could be modified in different ways by cellular enzymes and manipulation of groups to the chain. RNA could be defined as a structure which features are between linear molecule and 3-D structure. If secondary structure is taken in consideration, the RNA is composed of double stranded regions, when the single linear RNA is folded upon itself.

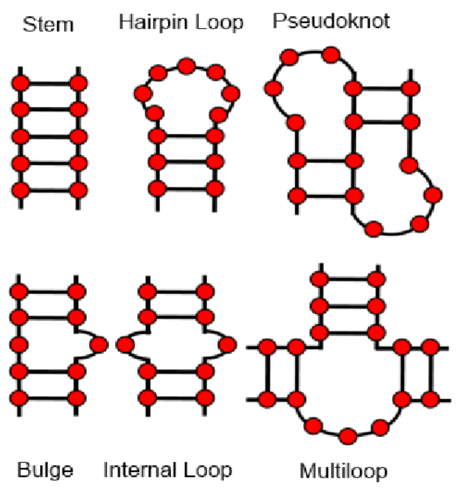


Figure 1 Different secondary RNA structures [1]

Unlike DNA, RNA is produced as single stranded molecule that folds intramolecularly. Base pairs almost always occur in a nested fashion in secondary structures (non-crossing matches) [1]. Shown on Figure 1 [1], different structures that can be observed in a secondary RNA structure.

2. Problem Definition

RNA secondary structure was discovered by X-Ray methods, which were extremely hard and expensive to be performed for all possible RNA sequences. Thus, a new concept for faster and efficient formation of a secondary structure was invented, known as computational prediction of RNA secondary structure. One type of computational prediction is by dynamical programming (DP) which is a useful technique for complex problems like the RNA structure. With the help of DP, the efficient prediction of RNA structure could be performed.

3. Related Work and Survey

With recent advances in bioinformatics and computer science, there are various methods to solving the challenge of predicting RNA secondary structures. The dynamic programming algorithms include: Nussinov-Jacobson Algorithm, Minimal-free Energy Algorithm (Zucker's Algorithm), Maximum Expected Accuracy Algorithm (MEA), and Pseudoknot Algorithm.

There is a speedup improvement to the Nussinov Algorithm, known as Nussinov-Four Russians Algorithm or simply Four Russians Algorithm. Shown below is the pseudocode for the Nussinov Algorithm that is explained in the Proposed Method section.

```
for j =2 to n do
    [Independent] Calculations below do not depend on the current column j.
    for i =1 to j − 1 do
        S (i, j) =max (S (i+1, j-1) +B (i, j), S (i, j-1)) (Rules a, b)
    End for
    [Dependent] Calculations below depend on the current column j.
    for i =j − 1 to 1 do
        S (i, j) =max (S (i+1, j), S (i, j)) (Rule c)
        for k = j − 1 to i+1 do {The loop below is called the Rule d loop}
            S (i, j) =max (S (i, j), S (i, k-1) +S (k, j)) (Rule d)
        End for
    End for
End for
```

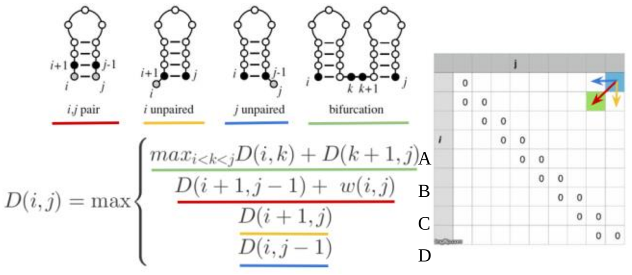
In the Speedup Four Russians Algorithm, each execution of the loop labeled “independent” takes O(n) time and is inside a loop that executes only O(n) times, so the independent loop takes O(n^2) time in total and does not need any improvement. The cubic-time behavior of the algorithm comes from the fact that there are three nested loops, for j, i and k respectively, each incrementing O(n) times when entered. The speedup will be due to reducing the work in the Rule d loop. Instead of incrementing k through each value from j − 1 down to i + 1, indices will be combined into groups of size q (to be determined later) so that only constant time per group will be needed. With that speedup, each execution of that Rule d loop will increment only O(n/q) times when entered. However, there is a need to do some preprocessing, which takes time that increases with q. Setting q = log3(n) will yield an O (n^3/ log n) overall worst-case time bound [5].

4. Proposed Method

The goal of this project is to implement the first developed DP algorithm, Nussinov-Jacobson Algorithm. It is defined as an algorithm for secondary RNA structure prediction based on the folding principle, when the RNA strand is folding onto itself, without taking in consideration formations like pseudoknots which is tackled in one of the other 3 DP prediction algorithms [2]. This algorithm takes in considerations two cases: if leftmost base is unpaired or paired with another base. It is constructed by following 3 main stages: initialization, recursion, and traceback.

The initialization step is achieved by scoring of matching elements present on the main diagonal and the diagonal below it, where the rules respected are M[i][i] = 0 for i=1 to L, and M[i][i-1] = 0 for i=2 to L, where L is the length of the RNA.

The second step, recursion, considers filling the matrix based on the four major conditions explained in Figure 2 [2]. Rule A covers all matchings that can be decomposed into two non-crossing matchings in the interval i: k, and the interval k + 1: j. In the case of rule A, the matching is called a bipartition, and the interval i: k is called the head of bipartition, and the interval k + 1: j is the called the tail of the bipartition. Rule B covers all matchings that contain a possible (i, j) match. Rule C covers all matchings when site i is not in any match. Rule D covers all matchings when site j is not in any match. Recursion furnishes a DP-Algorithm for computing the Nussinov matrix in O(n^3) time and O(n^2) space.



$$D(i,j) = \max\begin{cases} \max_{i<k<j} D(i,k) + D(k+1,j) & \text{A} \\ D(i+1,j-1) + w(i,j) & \text{B} \\ D(i+1,j) & \text{C} \\ D(i,j-1) & \text{D} \end{cases}$$

Figure 2 Recursion step of Nussinov-Jacobson Algorithm [2]

The final step, traceback, considers the formation of actual secondary structure of RNA sequence based on the trace-back from the given scores in the matrix, which are filled in by the previous steps. Shown below is the pseudocode for the traceback function implemented in Nussinov Algorithm. Complexity of traceback is O(n^2) time.

```
Procedure traceback (i, j)
    if $j \le i$ then
        return
    else if $N_{ij} = N_{ij-1}$ then
        traceback (i, j − 1)
        return
    else
        for all k: $i \le k < j$, $S_k$ and $S_j$ complementary do:
            if $N_{ij} = N_{i\,k-1} + N_{k+1\,j-1} + 1$ then
                print (k, j)
                traceback (i, k − 1); traceback (k + 1, j − 1)
                return
            end if
        end for
end if
```

Practically, the overall algorithm has a time complexity of $O(n^3)$ as it must look at all O(n) previous cells in the same line/column for each $O(n^2)$ cells.

5. Evaluation

To evaluate this algorithm, 4 different RNA sequences were entered to predict their secondary structure that vary in length and the time taken was reported. The RNA sequences entered were as shown in Table 1. The transcript RefSeq of each of these RNAs was retrieved from NCBI [3].

Table 1 Evaluation rounds for the algorithm

Round	Sequence Length	Gene Name
Round 1	994 bp	Toy example
Round 2	2094 bp	NPTN
Round 3	2704 bp	NOP2 nuclear protein
Round 4	3800 bp	BRCA1

[1] http://rna.tbi.univie.ac.at/forna/

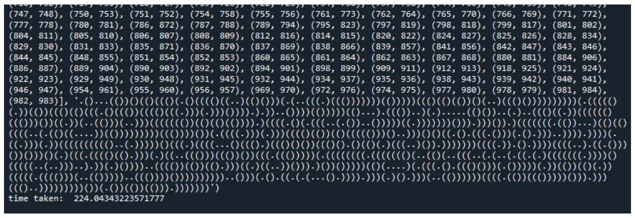
The algorithm is expected to give a dot-bracket representation for the predicted secondary structure of the entered RNA sequence along with the indices of binding pairs.

The time taken of Nussinov Algorithm to predict the secondary structure of each entered RNA sequence was as reported in Table 2.

Table 2 Nussinov algorithm results

Round	Sequence Length	Time taken (in sec.)
Round 1	994 bp	224
Round 2	2094 bp	2095
Round 3	2704 bp	4364
Round 4	3800 bp	13499

An example of the algorithm output for an RNA sequence of 995 bp is shown below.



```
(747, 748), (750, 753), (751, 752), (754, 758), (755, 756), (761, 773), (762, 764), (765, 770), (766, 769), (771, 772),
(777, 778), (780, 781), (786, 872), (787, 788), (789, 794), (795, 823), (797, 819), (798, 818), (799, 817), (801, 802),
(804, 811), (805, 810), (806, 807), (808, 809), (812, 816), (814, 815), (820, 822), (824, 827), (825, 826), (828, 834),
(829, 830), (831, 833), (835, 871), (836, 870), (837, 869), (838, 866), (839, 857), (841, 856), (842, 847), (843, 846),
(844, 845), (848, 855), (851, 854), (852, 853), (860, 865), (861, 864), (862, 863), (867, 868), (880, 881), (884, 906),
(886, 887), (889, 904), (890, 903), (892, 902), (894, 901), (898, 899), (909, 911), (912, 913), (918, 925), (921, 924),
(922, 923), (929, 949), (930, 948), (931, 945), (932, 944), (934, 937), (935, 936), (938, 943), (939, 942), (940, 941),
(946, 947), (954, 961), (955, 960), (956, 957), (969, 970), (972, 976), (974, 975), (977, 980), (978, 979), (981, 984),
(982, 983)], '.()...(())()(()((()(.()(((()((..)(()()))(.(..(((.)((()))))))(()))))((()(()(()()(..)((()()))))))))(.(((((()
(.))(())((()(()(((.()((())((((()(((.)))(.)))()))).).))..()))(()))))(()....).((())..)(.)......(()())..(.)..((()((.))((((()
((())()((.))(..(()))(..)))((((((()(()(()(()))).)((((.()(.(((..(.())..())))((.)))))))())).)))()).)(((((((.(())...)(()(()
((((..(.(()((.....))(()))))))))(((()))())(.((((.)))(.)))(((()(())(()((((()))()..)))()()((.().(((.()))(.().)))..)))).))))(.
((.)))(.))((((((((((()..(.))))()(((.)((((...()((().)((()()())((()().()(()(.)(((..)()).)))))))((((.)).().)))((((..).((.()))
())()))()(.)(((.((((()(().)))(.)((..(())))((())())(((.((())))(.((((((((.((((((()(..(()(..(((..(.(..(.((.(.)((((((.)))))()
(((((..(..)))..).))(.)()))..(((())(())(().)))((.)((..))()).)()()))))(()(.....)(.(((.().((()()))(.())))(.))(())(()(.))
(((((.(((()))(..(()))))..(((())))()))))))))))..()))(.().((.(.(...().)))).)))(.)().)))(..(())))))((((.(())((())))))()).)))
((()..))))))))))())(.())(())(())).)))))))')
time taken:  224.04343223571777
```

The output gives the user two main formats. The first one is a list of paired indices that indicate matching, in other words binding, pairs. The second output is a dot-bracket representation for these binding pairs. Each pair is represented by two brackets ().

The dot-bracket representation can be used for visual illustration of the predicted secondary structure. This output format can be used as an input for the online tool in ViennaRNA Web Services site[1] [6]. Shown below in

Figure 3 a visual representation output from this tool for the toy example of 995 bp.

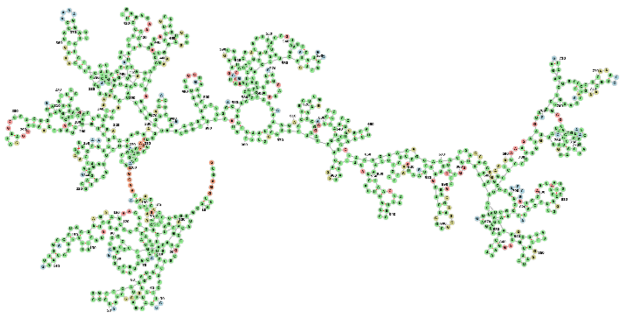


Figure 3 Predicted secondary structure of 1 kbp RNA seq using online visualization tool in [6]

The RNA secondary structures can fold into shapes and patterns, one being a pseudoknot, as shown in Figure 4. Pseudoknots are ignored by the Nussinov algorithm as they break down dynamic programming algorithms due to being too complicated to handle and perform backtracking on [4]. To enforce this, a minimal loop length was set in the code so that it does not result in nonsensical base-pairs and loops.

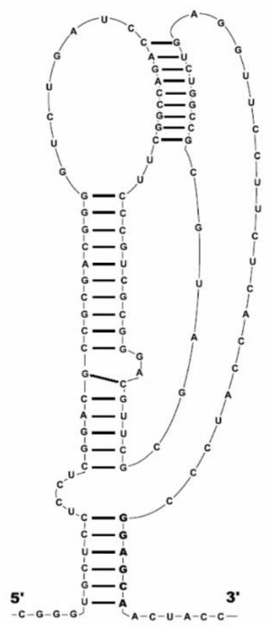


Figure 4 RNA pseudoknot example [4]

The Nussinov algorithm based on base pair maximization does not yield biologically relevant structures:

- no stacking of base pairs considered.
- loop sizes not distinguished
- no special scoring of multi-loops

Also, it gives only one structure predicted as it cannot differentiate structures sufficiently well: possibly many optima. Also, crossing structures cannot be predicted. However, it shows pattern of RNA structure prediction by DP (simple + Instructive).

A comparison between Nussinov Algorithm and Four Russians Algorithm was reported in [5]. Based on this comparison, The computation time of Nussinov Algorithm is much higher than that of the Four Russians Algorithm. The results of this study were reported as follow in Table

3. This difference in time is due the algorithm complexity itself.

Table 3 Computation time (seconds) of Nussinov Algorithm ($O(n^3)$) and Four Russians Algorithm ($O(n^3/log(n))$) [5]

Size (in bp)	$O(n^3)$ Algorithm	$O(n^3/log(n))$ Algorithm	Ratio
1000	3.2	1.43	2.23
2000	27.10	7.62	3.55
3000	95.94	26.90	3.55
4000	241.45	55.11	4.38
5000	470.16	97.55	4.82
6000	822.79	157.16	5.24

Another comparison of computation time between Nussinov Algorithm and Nussinov Four Russians Algorithm was reported in [7] as shown below in Table 4.

Table 4 Computation time of Nussinov Algorithm and Four Russians Algorithm [7]

Seq (in bp)	Nussinov Algorithm	Four Russians Algorithm
Example (994)	3s 237ms	1s 327ms
BRCA1 (3800)	1m 8s 89ms	20s 861ms
BCL2 (6492)	21m 46s 760ms	6m 57s 328ms

The reported computation time from [7] was plotted in a graph as shown below in Figure 5.

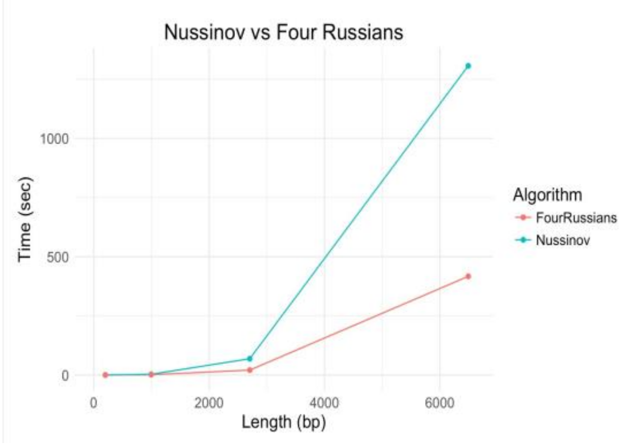
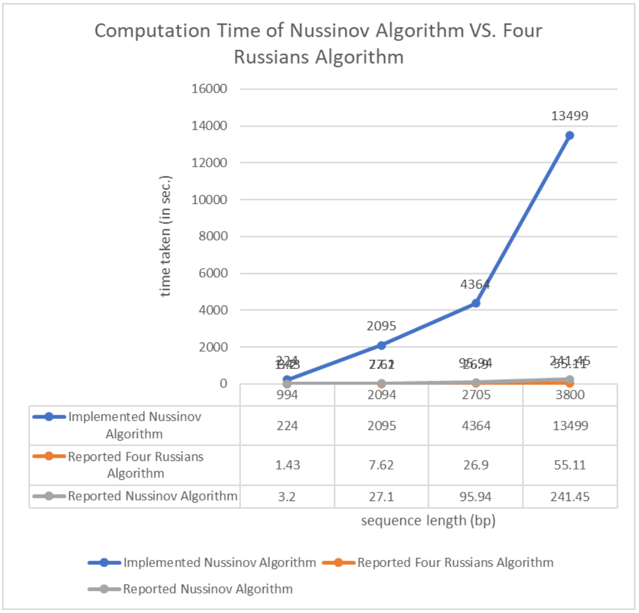


Figure 5 Computation time difference between Nussinov and Four Russians [7]

6. Conclusion

Based on the algorithm coding results, the algorithm code fulfilled its goal by predicting the secondary structure. Although the literature reported and implemented Nussinov Algorithm do not match in computation time, they still take time more than the Four Russians Algorithm. This difference in time taken can be due to the differences in hardware between the used one in the reported literature and the one used for implementing the algorithm in this project. Figure 6 shows a graphical plot for computation time in implemented Nussinov, literature reported Nussinov, and literature reported Four Russians.

Although the Nussinov Algorithm can predict the secondary structure of RNA sequence, there are still several limitations. These limitations include that it cannot predict crossing structures and it ignores pseudoknots.



	994	2094	2705	3800
Implemented Nussinov Algorithm	224	2095	4364	13499
Reported Four Russians Algorithm	1.43	7.62	26.9	55.11
Reported Nussinov Algorithm	3.2	27.1	95.94	241.45

Figure 6 Computation time of Nussinov Algorithm VS. Four Russians Algorithm

7. Suggestions

Such limitations reported to Nussinov Algorithm can be tackled by other dynamic programming algorithms. The fastest algorithm of all is the Four Russian's algorithm or so known as the new generation of the Nussinov's algorithm, but not always the fastest algorithm means the most accurate algorithm. The most reported accurate of all DP algorithms is the pseudoknotted algorithm, as the most advanced which is in capability to predict any kind of structure. But as a cost for this advantage, this algorithm is the slowest.

8. References

[1] Chheda, Nilay & Gupta, Manish. (2014). RNA as a Permutation.

[2]"RNA Structure and RNA Structure Prediction", Math.mit.edu, 2021. [Online]. Available: https://math.mit.edu/classes/18.417/Slides/rna-prediction-nussinov.pdf.

[3]"Home - Nucleotide - NCBI", Ncbi.nlm.nih.gov, 2021. [Online]. Available: https://www.ncbi.nlm.nih.gov/nuccore.

[4] "Nussinov algorithm to predict secondary RNA fold structures", BAYESIAN NEURON, 2019. [Online]. Available: https://bayesianneuron.com/2019/02/nussinov-predict-2nd-rna-fold-structure-algorithm/. [Accessed: 05- Jun- 2021].

[5] Y. Frid and D. Gusfield, "A simple, practical and complete O(n^3/logn) -time Algorithm for RNA folding using the Four-Russians Speedup", Algorithms for Molecular Biology, vol. 5, no. 1, 2010. Available: 10.1186/1748-7188-5-13 [Accessed 5 June 2021].

[6] S. Hammer and P. Kerpedjiev, "TBI - forna: RNA Secondary Structure Visualization Using a Force Directed Graph Layout", Rna.tbi.univie.ac.at, 2021. [Online]. Available: http://rna.tbi.univie.ac.at/forna/.

[7] L. Cho, Y. Dubois and A. Kwon, "Nussinov VS. Four Russians: Comparing and Implementing Current RNA Folding Algorithms", GitHub, 2017. [Online]. Available: https://github.com/YannDubs/FourRussiansRNA/blob/master/OnlyThreeRussians.pdf.