21BIO201 Intelligence of Biological Systems-3
II-B.Tech-CSE-AI (Semester 3)
Batch (2021-25)
End Semester Project

RNA FOLDING USING ILP

Submitted by:

Reddy Hema Radhika (CB.EN.U4AIE21050)

Karthic Raj A (CB.EN.U4AIE21022)

S Anirudh (CB.EN.U4AIE21053)

Hema B (CB.EN.U4AIE21017)

Under the supervision of
Dr.K.P.Soman Dr.HarishChandar

DEPT. OF COMPUTATIONAL ENGINEERING AND NETWORKING
AMRITA SCHOOL OF ENGINEERING
25th january 2023

I. Abstract

Almost all of the problems in bioinformatics are usually solved using dynamic programming and a few using the graph algorithms, However, in current trends a variety of modern biological and bioinformatics problems are deciphered using Computational methods, this calls for the need for a multidisciplinary interventions in bioinformatics for advancements in Modern Biology.The brand-new methodologies of applying Computational Algorithms like Integer linear programming will completely transform the fields of modern biology forever. In this work, a simple attempt has been made to bind the fields of Bioinformatics with Integer Linear Programming. This paper focuses on the RNA folding problem giving insights about the reason for RNA folding, a detailed description about the various components or loops present in the folded structure of an RNA and the formulation of RNA folding as an ILP problem.

II. Introduction

In recent years, the availability of vast amounts of biological data has entered a new phase in genomic analysis and structure prediction of DNA, RNA, and proteins. There are many types of RNA like tRNA (transfers specific amino acids that correspond to the mRNA sequence into the growing polypeptide chain at the ribosome during translation), mRNA (carries the genetic code from DNA in a form that can be recognized to make proteins), ribosomal RNA (combines with protein components to form a nucleoprotein complex called the ribosome which binds mRNA and synthesizes proteins, also called as translation), signal recognition particle RNA (recognizes the signal peptide and binds to the ribosome, halting protein synthesis). As a result, RNAs serve a critical function in regulating a variety of metabolic and signalling processes in cells.

The biological significance of completely sequenced genomes, ribonucleic acids (RNAs), and proteins data has led to development of specialised tools to explore, examine, and interpret the data. Researchers have been focused on discovering the structure of RNA for several decades since it is one of the most important concerns in understanding hereditary illnesses and developing new medications and helps to understand the role of the molecule in a cell.

III. LITERATURE Review

The biological significance of completely sequenced genomes, ribonucleic acids (RNAs), and proteins data has led to development of specialised tools to explore, examine, and interpret the data. Researchers have been focused on discovering the structure of RNA for several decades since it is one of the most important concerns in understanding hereditary illnesses and developing new medications and helps to understand the role of the molecule in a cell.

IV. OBJECTIVES

BIOLOGICAL FOUNDATIONS FOR PREDICTING RNA SECONDARY STRUCTURE An RNA molecule is made up of a lengthy chain of monomers called nucleotides, each of which has a base (any of adenine (A), cytosine (C), guanine (G), and uracil (U), as well as a phosphate group and a sugar group. The Watson-Crick base pairs are formed when two complementary bases, A-U and C-G, form hydrogen bonds and create stable base pairs. While the A-U pairings form two hydrogen bonds, the C-G pairs form three hydrogen bonds and are thus more stable. So, the secondary structure of RNA molecules is decided in terms of substructures by examining whether each base is paired or not, and structure creation is based on Watson-Crick base pairs, wobble base pairs, and stacking of nearby base pairs.

V. THEORETICAL BACKGROUND

In RNA, there are seven identified secondary structural elements:

1) hairpin loop
2) bulge loop
3) internal loop
4) multi-branched loops
5) single-stranded regions,
6) helix and
7) pseudo-knots

Many portions of RNA, as well as hairpin loops, helices, and pseudo knots, have a similar sub substructure known as stem. A helical stem refers to a stem that is part of a helix. Stems also help to assess the beginning and ending of most substructures. An RNA sequence, which is the fundamental structure of RNA, is represented by a four-letter alphabet. If we take a sequence $(S) = s_1, s_2, \ldots..,_s n$ which constitutes the base $s_i A, G, C, U$ where $1in$. Let a sub-sequence $S(i, j) = si, si + 1, \ldots.sj$ be a segment of the sequence S where $1ijn$ and s_i and s_j belongs to any pair of AU, CG, UG, they make a base pair. Each base can only be in one base pair at a time.

1) IN-BRIEF ABOUT VARIOUS LOOPS:

1) Single stranded region: If all the bases are not paired to any other base in a segment of the RNA sequence, then we say that it's a single stranded region.
2) Hairpin loop: If $(i, j) \in Si$ and none of the $i+1 \ldots j-1$ are paired to any other bases then we call this as Hairpin loop.
3) Bulge loop: When the bases between the base pairs (i, j) and $(i + 1, q)$ are not paired, then these unpaired bases form a bulge loop.
4) Internal loop: If we take two base pairs (i, j) and (p, q) where, $i + 1p < qj1$, then bases between i and p are unpaired and the bases between j and q are unpaired then it is called as internal loop.
5) Multi-branched loop: A multi-branched loop is formed when there are three base pairs $(i, j), (k, l), (p, q)$ and none of the bases from these indices lie within each other. So, this loop can radiate three or more loops.
6) Helical stem: Single-stranded RNA folds back on itself, generating helical sections interspersed with single-stranded portions that are unpaired, which usually helps in stability of the RNA. Other secondary structures are implicitly defined in the different bulges and loops that remain outside of the stacked pairs since the formation of a helix ends at the first mismatched base pair.
7) Pseudo-knots: Unpaired bases from one substructure bond to unpaired bases from another substructure to produce a stem, which is known as a pseudo-knot. The new development is called coaxial stacking between two stems with a Quasi continuous helix if the subsequent stem, created from this sort of bonding, stacks on top of an existing stem. This structure also helps to stabilise the RNA structure by forming a pseudo-knot. Watson-Crick base pairs, Wobble base pairs, and stacking of nearby base pairs are the key driving forces of structure creation in pseudo-knots as well.

VI. METHODOLOGY

The INTEGER-LINEAR PROGRAMMING Linear programming is a mathematical modelling technique, where the entire problem statement whatever it may be is boiled to maximization or minimization (basically optimization) of an objective function which must be linear i.e., the degree of the expression must be one, along with a set of linear constraints on the variables which can be equality constraints or inequality constraints. Basically, linear programming in a simple format can be expressed as follows,

$$\begin{aligned} &\text{objective function} - \text{maximize/minimize} \quad A^T x \\ &\text{subject to} : //www.overleaf.com/project/62394dd548f802c3ef95bc76mconstraints \quad \text{Cx} \leq b \ \ \text{(for maximization)} \\ &\qquad\qquad\qquad\qquad \text{Cx} \geq b \ \ \text{(for minimization)} \\ &\qquad\qquad\qquad\qquad \text{Dx} = e \end{aligned} \quad l \tag{1}$$

where, ‘A’ is a column vector of size nx1 (let’s say) of constants or weights corresponding to each variable in the objective function, ‘x’ is variable vector also of size nx1, ‘D’ and ‘E’ are constant square matrices of size nxn.

Integer linear programming, deals with the same optimization of an objective function over the given constraints, but the the major difference is that, in linear programming the variables need not be integers, but in the case of integer linear programming all the variables must be integers. Hence, in simple words integer linear programming is the optimization of an linear objective function, given a set of constraints on integer variables. So any integer linear programming problem can be expressed in the following format.

$$\begin{array}{ll} \text{objective function} - \text{maximize/minimize} & A^T x \\ \text{subject to constraints} & \text{Cx} \leq b \quad \text{(for maximization)} \\ & \text{Cx} \geq b \quad \text{(for minimization)} \\ & \text{Dx} = e \\ & x \geq 0 \\ & x \in Z^n \end{array} \tag{2}$$

Here, ‘A’ is a column vector of size nx1 (let’s say) of constants or weights corresponding to each variable in the objective function, ‘x’ is variable vector also of size nx1, ‘D’ and ‘E’ are constant square matrices of size nxn.

VII. Formulation of RNA folding problem as ILP problem

As mentioned, RNA has folded structure of existence naturally, where a few of the nucleotide are bonded with each other in an optimized manner, in such a way that the number of binds or interactions present in a particular RNA sequence is maximized to the extent possible, along with abiding by a set of biological constraints. In order to solve the RNA folding problem using ILP, let us consider a binary variable ‘P’ corresponding each and every pairs of nucleotide present in the given sequence, in such a way that the value of P(a,b) will be ‘1’ if and only if there is an interaction possible between the nucleotide at ‘a’ and nucleotide at ‘b’ of the given nucleotide sequence. Let us also consider a matrix ‘P’ in order to store all the P(i,j) in an organised manner so as to make all P(i,j) values easily accessible. So ‘P’ will an upper triangular matrix without a diagonal if, only the necessary variables are stored. All these following formulations can also be generated and solved using the MATLAB script. Let us understand this with an example and code the same simultaneously. Let us consider an RNA sequence ‘seq’ which shall be as follows,

$$\text{seq} = \text{“UGACU”} \tag{3}$$

After assuming the sequence, a matrix P is to be created in order to store all the P(i,j). since there are 5 nucleotide present in the sequence a 5x5 matrix is to be considered.

$$P = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} & P_{15} \\ P_{21} & P_{22} & P_{23} & P_{24} & P_{25} \\ P_{31} & P_{32} & P_{33} & P_{34} & P_{35} \\ P_{41} & P_{42} & P_{43} & P_{44} & P_{45} \\ P_{51} & P_{52} & P_{53} & P_{54} & P_{55} \end{bmatrix} \tag{4}$$

1) Implementation: In order to perform this definition in MATLAB, an object corresponding to an optimization problem is created. For defining the variables corresponding to this, the datatype ‘optimvar’ is used in order to reduce the time complexity of optimization and also for increasing the efficiency. As per our requirement, a variable square matrix with size as length of the input string is supposed to be initiated.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]n= length(seq); rna$_fold$ = $optimproblem(); rna_fold.ObjectiveSense$ = ‘$maximize’vars$ = $optimvar(\text{“}vars\text{”}, [n, n)]$, “$LowerBound$”, 0, “$Type$”, “$integer$”, “$UpperBound$”, 1);

On observing the variables present in the matrix, one of the major finding is that the matrix is populated with a set or unnecessary and redundant variables. In the case of RNA folding problem, a nucleotide cannot be paired with itself, hence, the variable ‘P(i,j)’ is considered for all ß $\neq j$.

Also, it can be observed that there is a redundancy in the variables. For example, the variables P(1,2) and P(2,1) both represent the bond or pairing interactions between nucleotide at position 1 and 2. So, in order to avoid this redundancy, the upper triangular part of the variable matrix is only considered, all the other elements are equated to zero. Hence, the matrix ‘P’ after updating turns out to be,

$$P = \begin{bmatrix} 0 & P_{12} & P_{13} & P_{14} & P_{15} \\ 0 & 0 & P_{23} & P_{24} & P_{25} \\ 0 & 0 & 0 & P_{34} & P_{35} \\ 0 & 0 & 0 & 0 & P_{45} \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{5}$$

While scripting this part, instead of directly making the corresponding variables zero in the matrix, a constraint is defined corresponding to our optimization problem, in which, variables corresponding to the above mentioned logic are made zero.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]
r=[]; c=[]; for i=1:n for j=1:n if i>=j r = [r,i]; c = [c,j]; end end end list1 = []; for i = 1:length(r) eqn = vars(r(i),c(i))==0; list1 = [list1;eqn]; end rna$_fold.Constraints.neccesary = list1$;

A. OBJECTIVE FUNCTION

Now, the main aim of RNA folding is stability, so, in order to increase the stability of a particular RNA sequence the number of pairings must be maximised to a large possible extent, which implies that the sum of all variables present in the matrix 'P' is supposed to be maximized, hence the objective function turns out to be,

$$\sum P\left(i,j\right) \ \forall \ \ i<j \text{ and } i,j < \text{length}\left(\text{RNA sequence}\right) \tag{6}$$

The objective function of the sample sequence 'seq' turns out to be as follows,

$$\begin{aligned} \text{objective function} : \text{maximize} - \ & P_{12}+P_{13}+P_{14}+P_{15} \\ & +P_{23}+P_{24}+P_{25}+P_{34}+P_{35}+P_{45} \end{aligned} \tag{7}$$

In order to script the same in MATLAB, a nested 'for' loop has been used to iterate through each and every variable present in the matrix and sum it up, finally the sum is assigned to the objective function attribute corresponding to the optimisation problem.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]
sum = optimexpr; for i=1:n for j=i:n sum = sum + vars(i,j); end end rna$_fold.Objective = sum$;

B. SUBJECT TO CONSTRAINTS

The constraints which the pairings are supposed to obey are as follows.

1) Constraint-1:

a) A nucleotide present in RNA sequence can only interact with it's complement present in the sequence.: Hence, an 'A' present in the sequence can be paired only with 'U' and vice-versa, in the same way, a 'C' can only be paired with 'G' and vice-versa. Here, it is to be noted that the above mentioned are just the possibilities of pairing, and pairing is not compulsory for the above mentioned cases, the nucleotide may or may not be paired and that depends upon the forthcoming constraints. So, in-order to fulfill this constraint the first step is to eliminate all the pairs among which pairing is not possible, which means that the P(i,j) corresponding to all the nucleotides between which pairing is not possible are supposed to be equated to zero.

$$P\left(i,j\right)=0 \ \ \forall \, \text{seq}\left(i\right), \text{seq}\left(j\right) \notin \left[\text{“AU”}, \text{“UA”}, \text{“CG”}, \text{“GC”}\right] \tag{8}$$

Henceforth, the following set of inequalities are generated for the sequence 'seq' on applying the constraint-1,

$$\begin{aligned} P\left(1,2\right)&=0 \quad [\text{seq}\left(1\right)=U, \ \text{seq}\left(2\right)=G] \\ P\left(1,4\right)&=0 \quad [\text{seq}\left(1\right)=U, \ \text{seq}\left(4\right)=C] \\ P\left(2,3\right)&=0 \quad [\text{seq}\left(2\right)=G, \ \text{seq}\left(3\right)=A] \\ P\left(2,5\right)&=0 \quad [\text{seq}\left(2\right)=G, \ \text{seq}\left(5\right)=A] \\ P\left(3,4\right)&=0 \quad [\text{seq}\left(3\right)=A, \ \text{seq}\left(4\right)=C] \\ P\left(3,5\right)&=0 \quad [\text{seq}\left(3\right)=A, \ \text{seq}\left(5\right)=A] \\ P\left(4,5\right)&=0 \quad [\text{seq}\left(4\right)=C, \ \text{seq}\left(5\right)=A] \end{aligned} \tag{9}$$

In order to make the calculations simple the matrix 'P' and objective function can be updated by substituting the value of the above variables as zero. The resultant 'P' matrix and objective function turns out to be as follows,

$$P = \begin{bmatrix} 0 & 0 & P_{13} & 0 & P_{15} \\ 0 & 0 & 0 & P_{24} & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \tag{10}$$

$$\text{objective function} : \text{maximize} - \ P_{13} + P_{15} + P_{24} \tag{11}$$

The above mentioned constraint is scripted in MATLAB but using a nested 'for' loop, using which valid pairs of nucleotides are found. If a pair of nucleotide which has no possibility of interacting with each other is found, the positions corresponding to these nucleotides are stored in a list and finally a constraint is generated which states that the variables corresponding these positions are equal to zero.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]
pairs = ["AU","UA","CG","GC"]; eqnns1 = []; for i=1:n for j=1:n if i<j con = strcat(seq(i),seq(j)) if any(strcmp(con,pairs)) eqn = vars(i,j) == 0; eqnns1 = [eqnns1;eqn]; end end end end rna$_fold.Constraints.set1 = eqnns1$;

2) Constraint-2:

a) A nucleotide present in RNA sequence can be paired with only one other nucleotide present in the sequence.: Which means that even if there are multiple complements corresponding to a particular nucleotide, it must be bonded with only one of them. The complement to which the nucleotide is to bonded is governed by the forthcoming set of constraints, the major purpose of this constraint is to limit the number of pairings corresponding to a particular nucleotide to one. In this case also it is to be noted that, pairing is not mandatory, but if a nucleotide is being paired it must be paired to only one other nucleotide.

$$\begin{array}{l} \sum P\left(a\right) + \sum P\left(b\right) \leq 1 \\ \text{where, } a = (i,j) \,\forall\, i < j \text{ and } b = (j,i) \,\forall\, i > j \end{array} \tag{12}$$

So, on taking sequence 'seq' as an example, the following inequalities are obtained corresponding to the constraint-2,

$$\begin{array}{l} P\left(1,2\right) + P\left(1,3\right) + P\left(1,4\right) + P\left(1,5\right) \leq 1 \\ P\left(1,2\right) + P\left(2,3\right) + P\left(2,4\right) + P\left(2,5\right) \leq 1 \\ P\left(1,3\right) + P\left(2,3\right) + P\left(3,4\right) + P\left(3,5\right) \leq 1 \\ P\left(1,4\right) + P\left(2,4\right) + P\left(3,4\right) + P\left(4,5\right) \leq 1 \\ P\left(1,5\right) + P\left(2,5\right) + P\left(3,5\right) + P\left(4,5\right) \leq 1 \end{array} \tag{13}$$

Now, on updating these constraint with the equations obtained in the first constraint, the following inequalities are procured,

$$\begin{array}{l} P\left(1,3\right) \ + \ P\left(1,5\right) \leq 1 \\ P\left(1,3\right) \leq 1 \\ P\left(2,4\right) \leq 1 \\ P\left(1,5\right) \leq 1 \end{array} \tag{14}$$

The MATLAB scripting corresponding to this constraint is as follows,

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]
eqnns2 = []; for i=1:n eqn = optimexpr; for j=1:n if i =j if i<j eqn = eqn+vars(i,j); elseif i>j eqn = eqn+vars(j,i); end end end eqns = eqn<=1; eqnns2 = [eqnns2;eqns]; end rna$_fold.Constraints.set2 = eqnns2$;

3) Constraint-3:

a) No two pairings or bonds present in an RNA sequence must cross each other, all the interactions must be in a nested manner.: The main reason for this is that crossing or non-nested interactions may disturb each other and the net result may reduce the stability of the entire molecule. Adding to this, the compactness of a molecule also increases when it is nested, which in turn increases the stability of an RNA molecule. In order to understand the formulation of this constraint let's assume four numbers a, b, c and d in such a way that and all these four numbers are less than the length of RNA sequence. So, in order to ensure that no two bonds are crossing, it is to be ensured that there are no simultaneous interactions between seq(a),seq(c) and seq(b),seq(d), so only either of P(a,c) or P(b,d) can be one, for all values of a, b, c and d with the properties mentioned above.

$$\begin{array}{l} P\left(i,j\right) + P\left(i',j'\right) \leq 1, \text{ where} \\ i < i' < j < j' \text{ and } i, i', j, j' < \text{length (RNA sequence)} \end{array} \tag{15}$$

On applying the constraint-3 to the sequence 'seq' the following inequalities are obtained,

$$\begin{array}{l} P\left(1,3\right) + P\left(3,5\right) \leq 1 \\ P\left(1,4\right) + P\left(3,5\right) \leq 1 \\ P\left(1,3\right) + P\left(2,5\right) \leq 1 \\ P\left(2,4\right) + P\left(3,5\right) \leq 1 \\ P\left(1,4\right) + P\left(2,5\right) \leq 1 \end{array} \tag{16}$$

Now, on updating these constraint with the equations obtained in the first constraint, the inequalities are updated as follows,

$$\begin{array}{l} P\left(1,3\right) + P\left(2,4\right) \leq 1 \\ P\left(1,3\right) \leq 1 \\ P\left(2,4\right) \leq 1 \end{array} \tag{17}$$

Figure 1: Graphical folded structure of the RNA sequence UGACA using 'bioinformatics' toolbox or MATLAB and optimised solution generated.

The MATLAB scripting corresponding to this constraint is done by using the concept of combinations. Using a list of all numbers from 1 till the length of the sequence, combination of 4 numbers are generated, and the sum of variables at indices (first number, third number) and (second number, fourth number) must be less that or equal to one.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]

nums = nchoosek([1:length(seq)],4); eqnns3 = []; b = size(nums); for i=progress(1:b(1)) eqn = vars(nums(i,1),nums(i,3))+ vars(nums(i,2),nums(i,4))<=1; eqnns3 = [eqnns3;eqn]; end rna$_fold.Constraints.set3 = eqnns3$;

At the end on compiling all the valid constraints the following inequalities are obtained and on applying them to the objective function and optimizing, the following is the optimized solution obtained.

$$\begin{matrix} \mathrm{P}\,(1,5) = 1 \\ P\,(2,4) = 1 \end{matrix} \quad (18)$$

That is the folded structure of the sequence is in such a way that, the first nucleotide interacts with the fifth one and the second nucleotide interacts with the fourth one simultaneously. On plotting the links between the nucleotide, a structure similar to the figure **??**.

In order to solve the objective function and constraints using the 'optimization' toolbox of MATLAB, the following command can be used.

C. SEASONAL AND TREND DECOMPOSITION USING LOESS (STL)

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]

[sol,fval] = solve(rna$_fold$);

In order to solve the objective function and constraints by integrating 'gurobi' optimizer with MATLAB, the 'intlinprog.m' script must be downloaded and stored in the path of execution, after which the following command is supposed to be used.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]

options = optimoptions('intlinprog'); sol = solve(rna$_fold, Options = 'options'$);

On pairing the corresponding nucleotide obtained in the solution, one of the major observations is that all the constraints have been followed perfectly. Using this solution the dot-bracket notation of an RNA sequence can also be generated. In the dot-bracket notation, every nucleotide which is not paired or which is not interacting with any other nucleotide must be represented as a dot and the nucleotide which are paired are represented by brackets out of which the nucleotide which is closer to the beginning of a sequence is represented with open bracket '(' and the other nucleotide is represented using a closed bracket ')'. The dot-bracket notation corresponding the example sequence 'seq' will be given as,

$$\text{dot bracket} = \text{“}((.))\text{”} \quad (19)$$

So, for scripting the same in MATLAB, a string comprising of only dots is to be created and using a 'for' loop the respective positions can be replaced with opening brackets and closing brackets respectively.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]

dot$_bracket$ = ''; $for i = 1 : length(seq) if ismember(i, begin) dot_bracket = [dot_bracket, '('];$ $else if ismember(i, end1) dot_bracket = [dot_bracket, ')'];$ $else dot_bracket = [dot_bracket, '.']; end$

Later, this dot-bracket structure can be passed into the 'Bioinformatics' toolbox of MATLAB to be plotted. The structure corresponding to the example sequence 'seq' can be seen in figure. 1.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]

rnaplot(dot$_bracket, 'Format', 'Diagram', 'Sequence', seq$)

The script so written simply accepts the RNA sequence as an input and prints it's folded structure. The 'bioinformatics' toolbox of MATLAB is actually used in-order to print the folded structure obtained as a result of optimization which is performed. After performing optimization the solution is then used to generate the dot-bracket notation which is then used by the bioinformatics toolbox to print the structure.

VIII. A FEW STEPS CLOSER TO REALITY

In reality instead, if the RNA sequence is given as an input to 'Bioinformatics' toolbox, the major observation is that, the sequence is not folded at all.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]

seq = 'UGACA'; ss = rnafold(seq); rnaplot(ss)

The main reason for this is that there are a couple more constraints that are supposed to be followed while folding an RNA sequence, which will be discussed in this section.

Figure 2: Graphical folded structure of the RNA sequence UGACA optimised by 'bioinformatics' toolbox-MATLAB.

1) Additional Constraint-1:

a) A nucleotide present in RNA sequence can interact with another nucleotide which is at least 4 nucleotides away: If two complementary nucleotides are present consecutively, they are not supposed to be paired. The minimum distance between two interacting nucleotides must be 4 nucleotides i.e., there must be at least 4 nucleotides present in between two interacting nucleotides.

$$P(i,j) = 0 \quad \forall \quad j - i < 4 \tag{20}$$

For scripting this in MATLAB, a nested for loop can be used to navigate through all the variables and check the difference between 'i' and 'j'.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]
r = []; c = []; for i=1:n for j=1:n if j-i<=4 r = [r,i]; c = [c,j]; end end end list1 = []; for i = 1:length(r) eqn = vars(r(i),c(i))==0; list1 = [list1;eqn]; end $rna_fold.Constraints.neccesary = list1$;

2) Additional Constraint-2:

a) The interactions between the pair of nucleotides 'C' and 'G' is a little more stronger than the interactions between the pair 'A' and 'U'.: While framing the objective function in the previous section, it was considered the strength interaction between 'AU' and 'CG' are exactly equal. But, in reality the strength of interactions between 'C' and 'G' is much higher than that of 'A' and 'U', the major reason behind this is that the G-C pair has three hydrogen bonds on the other hand, A-T pair has only two. This difference in number of hydrogen bond makes the 'CG' bond roughly 33 percentage more stronger than the 'AU' bond. In order to keep up to this, all the P(i,j) where the sequence elements correspond to 'G' and 'C' or 'C' and 'G' are supposed to be given 33 percentage higher weightage in the objective function i.e., such P(i,j) must be multiplied 1.33 before summing into the objective function.

$$\begin{aligned} &\text{objective function } : \sum P(i,j) \ + \sum 1.33 P(i',j') \\ &\text{where, sequence } (i,j) \in \ [\text{AU}, \text{UA}] \\ &\qquad\qquad \text{sequence } (i',j') \in [\text{CG}, \text{GC}] \end{aligned} \tag{21}$$

The scripting of this part slightly modifies the objective function. Using the same procedure, by using a nested 'for' loop the sum of all variables is calculated, in this process before adding the variables corresponding to 'CG' or 'GC' bond pairs are given a weightage of 1.33.

language=Matlab [frame=single,framexleftmargin=-1pt,framexrightmargin=-17pt,framesep=12pt,linewidth=0.98]
$st_pairs = [\text{“}CG\text{”}, \text{“}GC\text{”}]; for i = 1 : n for j = i : n seq1 = strcat(seq(i), seq(j)); if \ any(strcmp(seq1, st_pairs)) sum = sum + 1.33 * vars(i,j); else sum = sum + vars(i,j); end end end$

IX. Conclusion

a) W: e have developed an ILP formulation for predicting the secondary structure of an RNA molecule, which is a significant yet initially simple biological challenge. The biological model and ILP formulation were then enhanced to include more realistic biological elements of the situation. The goal of the RNA-folding problem is to predict an RNA molecule's secondary structure using merely its nucleotide sequence, for which integer linear programming is used.