Applied Mathematical Sciences, Vol. 6, 2012, no. 54, 2695 - 2702

An Integer Linear Programming Problem for RNA Structures

G. H. Shirdel

Department of Mathematics, Faculty of Basic Sciences
University of Qom, Qom, Iran
shirdel81math@gmail.com

N. Kahkeshani

Department of Mathematics, Faculty of Basic Sciences
University of Qom, Qom, Iran

Abstract

We suggest a method for finding a $k-$noncrossing RNA structure with arc-length $\geq \lambda$, stack-length ≥ 2 and the maximum number of arcs. This method is base of solving an integer linear programming problem. Also, an integer linear programming model is presented for obtaining a $k-$noncrossing core structure with arc-length $\geq \lambda$ and the maximum number of arcs. We solve these problems by Maple. Our method is important for prediction of RNA secondary structures. One main result of this paper is the enumeration of a type from RNA structures.

Mathematics Subject Classification: 92B05 and 90C10

Keywords: RNA structure, Core structure, Integer programming problem

1 Introduction

An RNA secondary structure is a sequence of four nucleotides A, G, U and C together with the Watson-Crick base pairs (A-U), (C-G) and Wobble base pair (U-G), according to [1]. RNA structures are represented as graphs such that nucleotides are drawn on a horizontal line as vertices and hydrogen bonds as arcs in the upper-half plan. Then integers are given to vertices. Graphs are categorized according to the three parameters: the maximum number of mutually crossing arcs, $k-1$, the minimum arc-length, λ, and minimum stack-length, σ. The length of arc (i, j) is $j-i$. A stack is a sequence of parallel arcs

and the length of stack is the number of these arcs, i.e., a set of arcs as follows: $((i-\sigma+1, j+\sigma-1), \ldots, (i,j))$. The crossing of arcs in graph correspond to existence of pseudoknot in RNA structure. In pseudoknot RNA structures, k is greater or equal to three. If $k=2$, then there isn't pseudoknot in RNA structure. A type of $k-$noncrossing RNA structure is $k-$noncrossing core structure. In these structures, there exist no stack with length ≥ 2, according to [2]. Each RNA structure has a core structure where obtain as follows: (1) All arcs in stack identify by a single arc, i.e., $((i-\sigma+1, j+\sigma-1), \ldots, (i,j)) \mapsto (i,j)$; (2) Isolated vertices keep in structure, i.e., if i is isolated, then $i \mapsto i$; (3) Vertices are relabeled.

Suppose c_n is a $k-$noncrossing core structure having following properties:(1) Arc-length is greater or equal to λ; (2) The number of stacks is equal to α; (3) The number of vertices is equal to n; (3) Structure has maximum number of arcs over n vertices. Now, we denote the set and number of $k-$noncrossing RNA structures over $m+n$ vertices in which their core structure is c_n by $\mathrm{R}_{m+n}(c_n)$ and $\mathsf{R}_{m+n}(c_n)$, respectively. Each element of $\mathrm{R}_{m+n}(c_n)$ has following properties: (1) Arc-length is greater or equal to λ; (2) Stack-length is greater or equal to 1; (3) The number of stacks is equal to α; (4) m is even. If m be odd, then $\mathsf{R}_{m+n}(c_n)=0$; (5) Structure has maximum number of arcs over $m+n$ vertices.

The remainder of the paper is organized as follows. In section 2, we represent an integer programming model for obtaining a $k-$noncrossing RNA structure with arc-length $\geq \lambda$ and stack-length ≥ 2. Also an integer programming model is represented for finding a $k-$noncrossing core structure with arc-length $\geq \lambda$. We obtain the number of RNA structures having core structure c_n. In section 3, the proposed models is illustrated by examples. Conclusions and directions of future works are discussed in section 4. Finally, in Appendix A, a program is given for solving of models.

2 Integer programming model for RNA and core structures

We introduce decision variable as follows:

$$x_{ij} = \begin{cases} 1, & \text{if arc } (i,j) \text{ exists;} \\ 0, & \text{otherwise.} \end{cases}$$

The RNA and core structures can be formulated mathematically as integer linear programming problems to be shown as problems A and B, respectively. The constraints of problem A are introduced as follows: The set of constraints (1) ensure that the degree of each vertex is at most 1. An RNA structure is $k-$noncrossing if

$$\nexists (i_1, j_1), (i_2, j_2), \ldots, (i_k, j_k);$$
$$i_1 < i_2 < i_3 < \cdots < i_k < j_1 < j_2 < \cdots < j_k.$$

The set of constraints (2) guarantee that RNA structures are $k-$noncrossing. The constraint (3) guarantees all variables to be either 0 or 1. For problem B, we add the constraint (4) to foregoing constraints. The set of constraints (4) say that length of each stack in core structures is equal to 1. The formation of Watson-Crick base pairs and Wobble base pair stabilizes the molecule by lowering its free energy. We can suppose that if the number of hydrogen bonds increase, then free energy decrease. Therefore, we consider the objective function as sum of decision variables and maximize it.

Problem A:

$$\text{Max} \quad z = \sum_{j-i \geq \lambda} x_{ij}$$

$$\text{s.t} \quad \sum_{j=1}^{i-\lambda} x_{ji} + \sum_{l=i+\lambda}^{n} x_{il} \leq 1 \qquad i = 1, \ldots, n \tag{1}$$

$$x_{i_1 j_1} + x_{i_2 j_2} + \cdots + x_{i_k j_k} \leq k-1 \qquad i_1 = 1, \ldots, n-(k+\lambda-1) \tag{2}$$

$$j_1 = i_1 + \lambda, \ldots, n-(k-1) \quad i_l = i_{l-1}+1, \ldots, j_1 - k + l - 1$$
$$j_l = j_{l-1}+1, \ldots, n-k+l \quad l \in \{2, \ldots, k\}$$
$$x_{ij} = 0, 1 \quad i, j \in \{1, \ldots, n\},\ j - i \geq \lambda \tag{3}$$

Problem B:

$$\text{Max} \quad z = \sum_{j-i \geq \lambda} x_{ij}$$

$$\text{s.t} \quad \sum_{j=1}^{i-\lambda} x_{ji} + \sum_{l=i+\lambda}^{n} x_{il} \leq 1 \qquad i = 1, \ldots, n$$

$$x_{i_1 j_1} + x_{i_2 j_2} + \cdots + x_{i_k j_k} \leq k-1 \qquad i_1 = 1, \ldots, n-(k+\lambda-1)$$

$$j_1 = i_1 + \lambda, \ldots, n-(k-1) \quad i_l = i_{l-1}+1, \ldots, j_1 - k + l - 1$$
$$j_l = j_{l-1}+1, \ldots, n-k+l \quad l \in \{2, \ldots, k\}$$
$$x_{i-1,j+1} + x_{ij} \leq 1 \qquad i = 2, \ldots, n-\lambda-1 \tag{4}$$
$$j = 2+\lambda, \ldots, n \qquad x_{ij} = 0, 1 \quad i, j \in \{1, \ldots, n\},\ j - i \geq \lambda$$

We know that stack-length in RNA structures is greater or equal to 2, i.e., if there is arc (i, j) in RNA structure, then there is arc $(i-1, j+1)$ or $(i+1, j-1)$ in the structure. Now, we add this condition to problem A. Therefore, the one of following conditions should establish for each (i, j) exactly.

$$x_{ij} + x_{i+1,j-1} = 0$$
or
$$x_{ij} + x_{i+1,j-1} = 2$$
or

$x_{ij} + x_{i-1,j+1} = 2$ and $x_{i+1,j-1} = 0$
or
$x_{i+1,j-1} + x_{i+2,j-2} = 2$ and $x_{i,j} = 0$

These set of constraints can be written as follows:

$$
\begin{aligned}
&x_{ij} + x_{i+1,j-1} \leq My_1\\
&x_{ij} + x_{i+1,j-1} \leq 2 + My_2\\
&-x_{ij} - x_{i+1,j-1} \leq -2 + My_2\\
&x_{ij} + x_{i-1,j+1} \leq 2 + My_3\\
&-x_{ij} - x_{i-1,j+1} \leq -2 + My_3\\
&x_{i+1,j-1} \leq My_3\\
&x_{i+1,j-1} + x_{i+2,j-2} \leq 2 + My_4\\
&-x_{i+1,j-1} - x_{i+2,j-2} \leq -2 + My_4\\
&x_{i,j} \leq My_4\\
&y_1 + y_2 + y_3 + y_4 = 3\\
&y_1, y_2, y_3, y_4 = 0, 1
\end{aligned}
$$

where M is a big number. When these constraints are added to problem A, we obtain problem A$'$. We solve the problems A$'$ and B using Maple program for $k = 3$, see Appendix A.

Theorem 2.1. *Let $m, n, \alpha \in \mathbb{N}$ and m is even. Then $\mathsf{R}_{m+n}(c_n) = \binom{\alpha+\frac{m}{2}-1}{\frac{m}{2}}$.*

Proof. Let $\beta = \frac{m}{2}$. We know that any structure of $\mathrm{R}_{m+n}(c_n)$ has a unique core c_n. For obtaining of any element of $\mathrm{R}_{m+n}(c_n)$, we add m nonisolated vertices to c_n, i.e., β arcs are added to structure c_n. On the other hand, with adding β arcs to structure c_n, the number of stacks, α, mustn't increase. Then the number of ways for adding of m vertices to structure c_n with these conditions is equal to the number of integer solutions of the equation $x_1+x_2+\cdots+x_\alpha = \beta$ where $x_i \geq 0$ for all $1 \leq i \leq \alpha$. This completes the proof. □

3 Numerical example

Example 1. Let $n = 15$, $\lambda = 4$ and $k = 3$. The optimal solution of problem A with this conditions is as follows: $x_{15} = x_{2,13} = x_{3,12} = x_{4,11} = x_{6,10} = x_{8,15} = x_{9,14} = 1$, and other variables are zero. Its is shown in Figure 1.

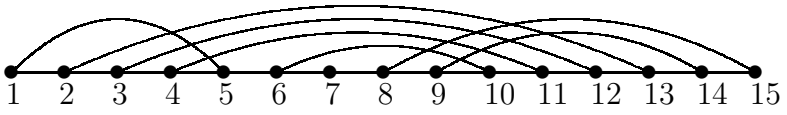

Figure 1: A 3-noncrossing RNA structure with $\lambda = 4$.

Table 1: Elements of $\mathrm{R}_{14}(c_{10})$.

number	the arcs of the structures
1	(1,13),(2,12),(3,11),(4,14),(5,9),(6,10)
2	(1,11),(2,14),(3,13),(4,12),(5,9),(6,10)
3	(1,13),(2,14),(3,9),(4,12),(5,11),(6,10)
4	(1,13),(2,14),(3,11),(4,10),(5,9),(6,12)
5	(1,13),(2,14),(3,10),(4,9),(5,12),(6,11)
6	(1,12),(2,11),(3,14),(4,13),(5,9),(6,10)
7	(1,13),(2,12),(3,14),(4,9),(5,11),(6,10)
8	(1,13),(2,12),(3,14),(4,10),(5,9),(6,11)
9	(1,12),(2,14),(3,13),(4,9),(5,11),(6,10)
10	(1,12),(2,14),(3,13),(4,10),(5,9),(6,11)

The elements of $\mathrm{R}_{14}(c_{10})$ are shown in Table 1. The core structure of them is c_{10} in Figure 3.

Example 2. Let $n = 10$, $k = 3$ and $\lambda = 4$. The optimal solution of problem A$'$ is as follows: $x_{2,8} = x_{3,7} = x_{4,10} = x_{5,9} = 1$, and other variables are zero. In Figure 2, we display corresponding structure with this solution.

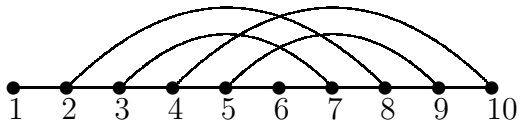

Figure 2: A 3-noncrossing RNA structure with $\lambda = 4$, $\sigma = 2$.

Example 3. The optimal solution of problem B with same conditions of example 2 is as follows: $x_{19} = x_{2,10} = x_{3,7} = x_{4,8} = 1$, and other variables are zero. The corresponding structure with this solution is shown in Figure 3. In this structure, $\alpha = 4$. Let $m = 4$. Then by Theorem 1, we have $\mathsf{R}_{14}(c_{10}) = 10$.

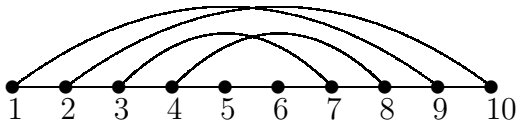

Figure 3: A 3-noncrossing core structure with $\lambda = 4$, (c_{10}).

4 Conclusion

We proposed an integer linear programming model for presentation of RNA secondary structures. The optimal solution of this model is an RNA structure with maximum number of hydrogen bonds. We can tell that this optimal structure has the minimum free energy. Therefore, this method can use for prediction of RNA structures.

There is a direction for our future research. We are examining how can obtain the number of optimal solutions of problems A′ and B.

5 Appendix A

```
> restart;
> with(LinearAlgebra); with(linalg); with(Optimization);
> n:=readstat("please enter a positive integer number");
> lambda:=readstat("please enter a positive integer number");
> #------Program A^\prime:
> for i to n do  if i-lambda<=0 then P[i]:=sum(x[i,k],k=i+lambda..n) end if;
   if i+lambda>n then P[i]:=sum(x[j,i],j=1..i-lambda) end if;
   if i-lambda>=1 and i+lambda<=n then
    P[i]:=sum(x[j,i],j=1..i-lambda)+sum(x[i,k],k=i+lambda..n)  end if;  end do;
> for i to n do
   s:=[coeffs(P[i])]; if s=[1] then P[i]:=0; end if; end do;
> a:=0; b:=0; j:=1;
> for i to n do if P[i]<>0 then
    a:=a+1; Q[j]:=P[i]; j:=j+1; else b:=b+1; end if; end do;
> d:=0; e:=0; l:=1;
> for i to n do  for j to n do
    if j-i>=4 then d:=d+1; end if; end do; end do;
> d:=d; > C:=Matrix(a,d);
> for i to n do for j to n do  if j-i>=4 then
     for k to a do   C[k,l]:=coeff(Q[k],x[i,j]);
     end do; l:=l+1; end if; end do; end do;
> for i to n-lambda-2 do  for j from i+lambda to n-2 do
    for k from i+1 to j-2 do  for l from j+1 to n-1 do
      for v from k+1 to j-1 do for w from l+1 to n do
        if j-i>=lambda and l-k>=lambda and w-v>=lambda then
         e:=e+1; R[e]:=x[i,j]+x[k,l]+x[v,w]; end if;
       end do; end do; end do; end do; end do; end do;
> T:=Matrix(e,d); l:=1;
> for i to n do for j to n do if j-i>=4 then   for k to e do
      T[k,l]:=coeff(R[k],x[i,j]);   end do; l:=l+1; end if; end do; end do;
```

```
> l:=0; k:=0; W:=Vector(n*n); t:=1; h:=0;
> for i to n do  for j to n do
    if j-i>=lambda+2 then   l:=l+1; k:=k+1;
     P[l]:=x[i,j]+x[i+1,j-1]-5*y[k]; W[l]:=0;
     P[l+1]:=x[i,j]+x[i+1,j-1]-5*y[k+1]; W[l+1]:=2;
     P[l+2]:=-x[i,j]-x[i+1,j-1]-5*y[k+1]; W[l+2]:=-2;
     Q[1]:=y[k]+y[k+1]; l:=l+2; k:=k+1  else Q[1]:=0;  end if;
    if j-i>=lambda+2 and i>=2 and j<=n-1 then
     l:=l+1; k:=k+1; P[l]:=x[i,j]+x[i-1,j+1]-5*y[k]; W[l]:=2;
     P[l+1]:=-x[i,j]-x[i-1,j+1]-5*y[k]; W[l+1]:=-2;
     P[l+2]:= x[i+1,j-1]-5*y[k]; W[l+2]:=0; Q[2]:=y[k]; l:=l+2;
    else Q[2]:=0;  end if;  if j-i>=lambda+4 then
     l:=l+1; k:=k+1; P[l]:=x[i+1,j-1]+x[i+2,j-2]-5*y[k]; W[l]:=2;
     P[l+1]:=-x[i+1,j-1]-x[i+2,j-2]-5*y[k]; W[l+1]:=-2;
     P[l+2]:=x[i,j]-5*y[k]; W[l+2]:=0;
     Q[3]:=y[k]; l:=l+2; else Q[3]:=0;  end if;
    R[t]:=Q[1]+Q[2]+Q[3]; t:=t+1;  end do;  end do;
> for i to t-1 do if R[i]<>0 then h:=h+1; V[h]:=R[i]; end if; end do;
> beta:=Vector(h);
> for i to h do beta[i]:=nops(V[i])-1; end do;
> A4:=Matrix(h,k); alpha:=1;
> for i to k do  for z to h do
    A4[z,alpha]:=coeff(V[z],y[i]);  end do;  alpha:=alpha+1; end do;
> A4:=augment(Matrix(h,d),A4);
> P[l+1]:=x[1,5]; P[l+2]:=x[1,6]; P[l+3]:= x[n-4,n]; P[l+4]:=x[n-5,n];
> V:=Vector(l+4);
> for i to l do V[i]:=W[i]; end do;
> V[l+1]:=0; V[l+2]:=0; V[l+3]:=0; V[l+4]:=0;
> F:=Matrix(l+4,d+k); t:=1;
> for i to n do  for j to n do if j-i>=4 then
     for z to l+4 do  F[z,t]:=coeff(P[z],x[i,j]);
     end do; t:=t+1; end if; end do; end do;
> t := d+1;
> for i to k do  for z to l+4 do
    F[z,t]:=coeff(P[z],y[i]);  end do;  t:=t+1; end do;
> A1:=convert(stackmatrix(C,T),Matrix);
> A2:=Matrix(RowDimension(A1),k);
> A3:=augment(A1,A2);
> A:=convert(stackmatrix(A3,F,A4),Matrix);
> b:=Vector([Vector(a,1),Vector(e,2),V,beta]);
> c:=Vector([Vector(d,1),Vector(k,0)]);
> X:=Optimization[LPSolve](c,[A,b],assume=nonnegint,maximize);
```

```
> l:=0; M:=Matrix(n,n); s:=0;
> for i to d do  X[2][i]:=round(X[2][i])
  end do;
  > for i to n do  for j from i+lambda to n do
    l:=l+1;  if X[2][l]=1 then M[i,j]:=1; s:=s+1; end if;
   end do;  end do;
> First:=Vector[row](s); End:=Vector[row](s); l:=1;
> for i to n do  for j to n do
    if M[i,j]=1 then First[l]:=i; End[l]:=j; l:=l+1; end if;
   end do; end do;
> First; End;
> #------Program B:
> l:=0; v:=1;
> for i from 2 to n-lambda-1 do  for j from 2+lambda to n-1 do
    if j-i>=lambda then l:=l+1; P[l]:=x[i,j]+x[i-1,j+1] end if;
   end do; end do;
> E:=Matrix(l,d);
> for i to n do for j to n do if j-i>=4 then
     for k to l do   E[k,v]:=coeff(P[k],x[i,j]);
     end do; v:=v+1; end if; end do;  end do;
> A:=convert(stackmatrix(A1,E),Matrix);
> b:=Vector([Vector(a,1),Vector(e,2),Vector(l,1)]); c:=Vector(d,1);
> X:=Optimization[LPSolve](c,[A,b],assume=nonnegint,maximize);
> l:=0; M:=Matrix(n,n); s:=0;
> for i to d do  X[2][i]:=round(X[2][i]);  end do;
> for i to n do   for j from i+lambda to n do
    l:=l+1;   if X[2][l]=1 then M[i,j]:=1; s:=s+1; end if;
   end do;  end do;
> First:=Vector[row](s); End:=Vector[row](s); l:=1;
> for i to n do   for j to n do  if M[i,j]=1 then First[l]:=i;
End[l]:=j; l:=l+1; end if; end do; end do;
> First; End;
```

References

[1] R. T. Batey, R. P. Rambo and J. A Doudna, *Tertiary motifs in RNA structure and folding*, Journal of Angewandte Chemie International Edition, **38**(1999), 2326-2343.

[2] E. Y. Jin, and C. M. Reidys, *Combinatorial design of pseudoknot RNA*, Journal of Advances in Applied Mathematics, **42**(2009), 135-151.

Received: November, 2011