

## 5.1 Introduction

A RNA (Ribonucleic acid) molecule is a single-stranded chain of the nucleotides A, C, G, U. RNA has the properties of both DNA and protein. Like the DNA, RNA can store and transfer information. An example of RNA being used as information storage can be seen in viruses such as the infamous HIV. In fact, most of the known viruses are RNA based.

Unlike double-stranded DNA, RNA is a single stranded molecule. It has an extra OH attaching to the 2' carbon which allows it to form extra hydrogen bond and enables it to have 3D structure. This difference is shown in Figure 5.1(a) and Figure 5.1(b) below.

Due to the ability to have 3D structure, RNA is able to function as a catalyst and perform functions such as generating amino acid from codon.

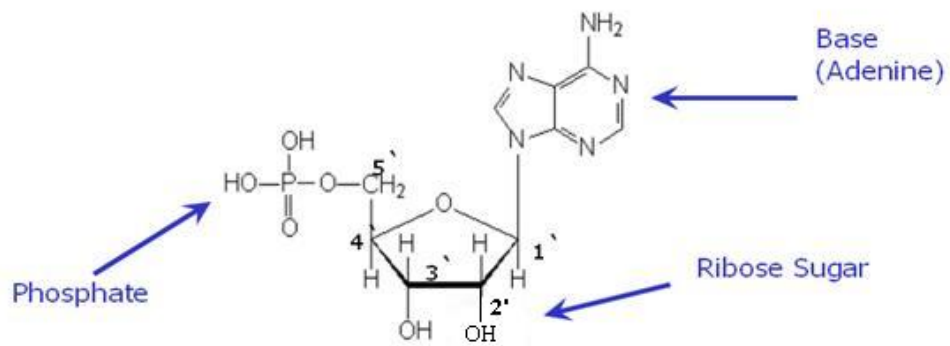
### 5.1.1 RNA Structure

Secondary structure of a RNA is a list of base pairs that are close together in its three dimensional structure. According to the theory of thermodynamics the optimal foldings of an RNA sequence are those of minimum free energy, and thus the native foldings i.e. the foldings encountered in the real world should correspond to the optimal foldings. Information on the secondary structure of a RNA molecule can be used as a stepping stone to model the full structure of a RNA molecule which in turn relates to the biological function. Therefore, it is important to study the structure of the RNA as well as ways to predict it[8].

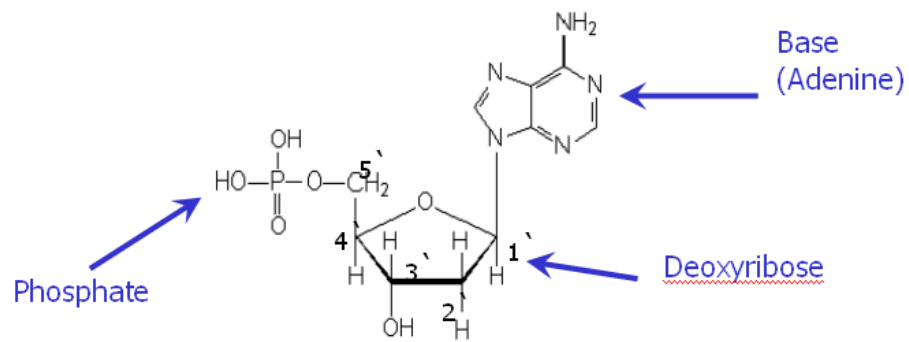
In fact, due to the higher degree of freedom, the number of structures the RNA can produce is more than the number of structures the protein can produce. The RNA structure can be described in three levels: primary structure, secondary structure and tertiary structure.

#### 1 Primary structure

The primary structure of a RNA is just a sequence of nucleotides describing the RNA. An example of the primary structure of a RNA is shown below.



(a)



(b)

Figure 5.1: The difference between (a)RNA and (b)DNA is the extra OH in the 2' carbon

**GGGAUUUAGCUCAGUJGGGAGAGCGCCAGACUGAAGAUCUGGA**  
**RNA Primary Structure For Phenylalanyl-tRNA**

## 2 Secondary structure

The secondary structure of a RNA is a list of base-pairs that are close together in its three dimensional structure, see Figure 5.2 for the secondary structure of phenylalanyl-tRNA.

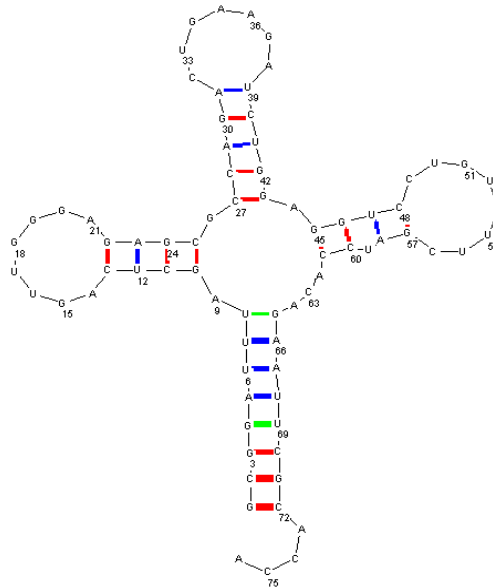


Figure 5.2: Secondary structure for Phenylalanyl-tRNA

As we can see, the secondary structure consists of a set of nucleotides which form the base of the RNA. Each base can pair with at most 1 other base which produces the base-pairs. The secondary structure consists of various elements such as loops and pseudoknots, which will be discussed later.

## 3 Tertiary structure

The tertiary structure is the 3-dimensional structure of the RNA. Figure 5.3 shows the tertiary structure for phenylalanyl-tRNA.

### 5.1.2 Definition of RNA secondary structure

A secondary structure of a RNA is defined as a set of base-pairs such that each base is paired at most once[10].

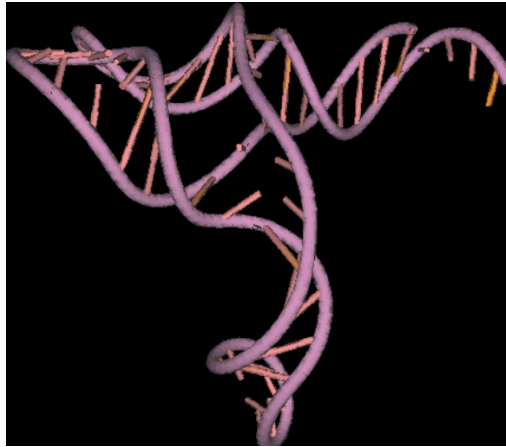


Figure 5.3: Tertiary structure for Phenylalanyl-tRNA

- RNA is a polymer composed of a combination of four nucleotides: adenine (A), cytosine (C), guanine (G), and uracil (U).
- G-C and A-U form complementary hydrogen bonded base pairs, with the GC base pairs being more stable since they form three hydrogen bonds as opposed to the two hydrogen bonds formed by AU base pairs.
- In addition to the canonical Watson-Crick GC and AU base pairs, non-canonical pairs can occur in RNA secondary structure as well. The most common of these is the non-canonical GU base pair.
- RNA is typically produced as a single stranded molecule (unlike DNA) which folds upon itself to form base pairs. This structure is referred to as the secondary structure of the RNA.

RNA secondary structure can be viewed as an intermediary between a linear molecule and a three-dimensional structure. It is mainly composed of double-stranded RNA regions formed by folding the single-stranded RNA molecule back on itself. Several elements are also defined in the secondary structure. There are a number of different secondary structures that can be formed from this base-pairing, including:

### **Pseudoknot**

To define a pseudoknot, let us consider a RNA sequence which consists of bases with index 1 to  $n$ . We also define a base-pair to be  $(i, j)$  if a base with index  $i$

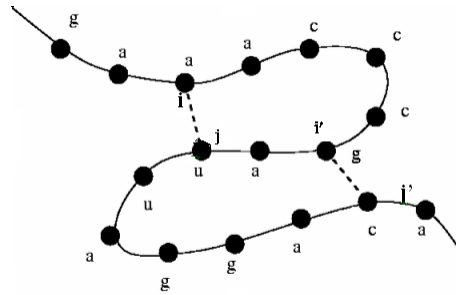


Figure 5.4: Pseudoknot

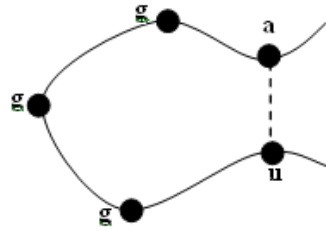


Figure 5.5: Hairpin loop

forms a pair with another base with index  $j$  in the same sequence. A pseudoknot is then defined as two base-pairs  $(i, j)$  and  $(i', j')$  such that  $0 < i < i' < j < j' < n$  where  $0$  and  $n$  are the indices to the two ends of the RNA. In other words, if the base-pairs overlap each other, we have a pseudoknot as shown in Figure 5.4.

### Loops

Loops are defined as regions enclosed by a backbone and some base-pairs. In our definition of the loops we assume that the RNA subsequence does not contain any pseudoknots. A loop closed by base pair  $i$  and  $j$  is named according to the number of interior base pairs and unpaired bases. Loops can be further classified into five different types as shown in Figure 5.5[9]:

- **Hairpin loop**

Hairpin loop is a loop that contains exactly one base-pair or zero interior base pairs. This can happen when the RNA strand folds into itself with a base-pair holding the fold together as shown in Figure 5.5.

- **Stacked pair**

A stacked pair is a loop formed by two base-pairs that are adjacent to each other. In other words, if a base-pair that forms a loop is  $(i, j)$ , the other

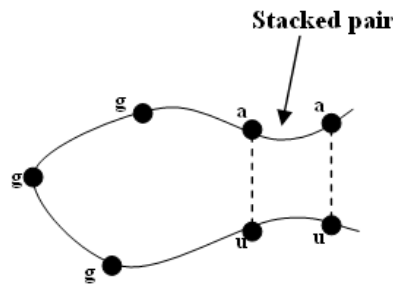


Figure 5.6: Stacked pair

base-pair that form the stacked pair could be  $(i+1, j-1)$ . A stretch of consecutive stacking base pairs is called a helix. Figure 5.6 shows an example of a stacked pair.

- **Internal loop**

Internal loop consists of two base-pairs like the stacked pair. The difference is that internal loop consists of at least one unpaired base on each side of the loop between the two base-pairs. In short, the length of the two sides of the RNA between the two base-pairs must be greater than 1, i.e. if  $i$  and  $j$  form a pair then  $i'$  and  $j'$  that may form another pair must satisfy  $i' > i + 1$  and  $j' < j - 1$ . This is shown in Figure 5.7.

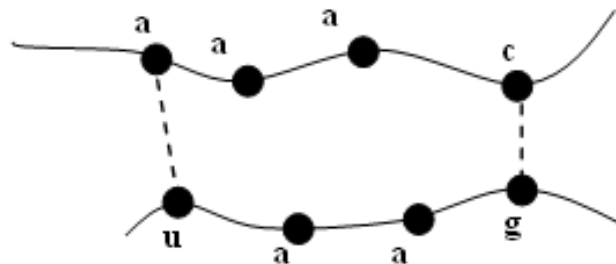


Figure 5.7: Internal loop

- **Bulge**

A bulge has two base-pairs like the internal loop. However, only one side of the bulge has unpaired bases, that is, if  $i' = i + 1; j' < j - 1$  or if  $i' > i + 1; j' = j - 1$ . The other side must have two base-pairs adjacent to each other as shown in Figure 5.8.

- **Multi-loop**

Any loop with 3 or more base-pairs are multi-loop. Usually, one can view a multi-loop as a combination of multiple double-stranded regions of the RNA. Figure 5.9(a) and Figure 5.9(b) below show examples of the various loop types.

### 5.1.3 Obtaining RNA Secondary structure

There are multiple ways of obtaining RNA secondary structure.

- **Experimental Methods**

The experimental methods include X-Ray Crystallography and NMR Spectroscopy.

- **Phylogenetic Approach**

Another way of obtaining the RNA secondary structure is through phylogenetic approach. This approach tries to find the relationship among different organisms and study the causes behind the difference. To do this, it needs a sufficiently large number of related RNA sequences. With these sequences, the approach tries to find some sequences with common functionality and performs an alignment to infer the structure of the RNA.

- **Structure Prediction**

The third way of obtaining RNA secondary structure is by prediction, either based on energy minimization methods or some other methods. Current technology can correctly predict up to 73% of known base-pairs for a secondary structure when a sequence of fewer than 700 bases are folded. The focus of this lecture is on RNA secondary structure prediction. Due to the complex nature of this problem, it was broken into two stages to ease the understanding of the solution. The two stages are:

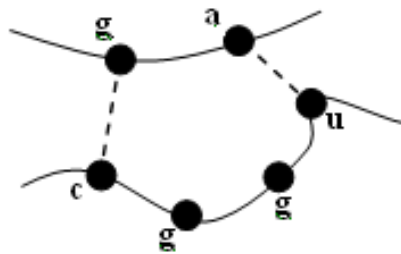
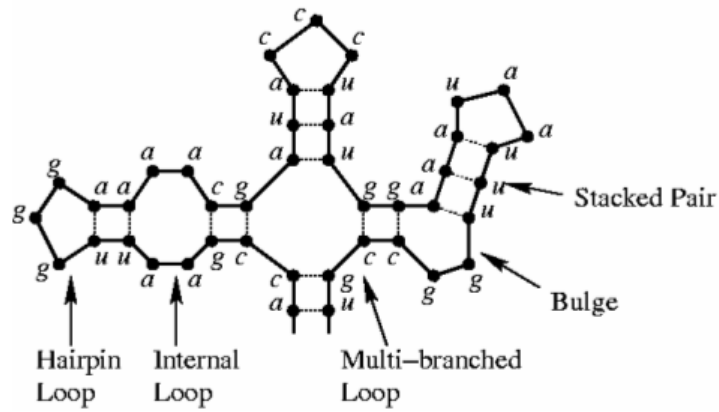
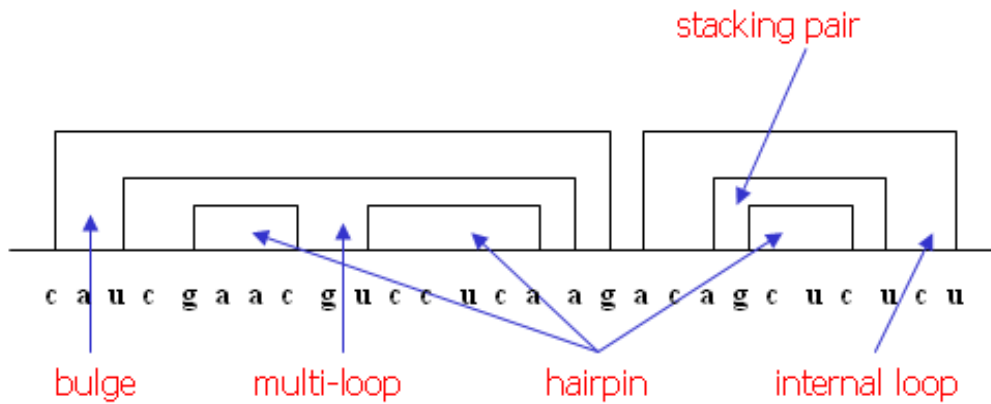


Figure 5.8: Bulge



(a)



(b)

Figure 5.9: Different loop types

- RNA secondary structure prediction *without pseudoknot*: In this stage, we ignore all pseudoknots and do a prediction based on a thermodynamic model. The reason for ignoring pseudoknots is because of the high computational complexity of predicting a structure with pseudoknots. In this stage, we will look at several dynamic programming solutions as well as speed-ups.
- RNA secondary structure prediction *with pseudoknot*: In this stage, we look at ways of predicting the secondary structure of a RNA with a particular type of pseudoknot known as the simple pseudoknot. Predicting RNA secondary structure with pseudoknots is a NP-hard problem.

## 5.2 RNA Secondary Structure Prediction without Pseudoknots

### 5.2.1 Nussinov Folding Algorithm

In 1978 Ruth Nussinov, George Pieczenik, Jerrold Griggs and Daniel Kleitman [16] used Dynamic Programming to find an efficient solution to the RNA secondary structure prediction problem. The algorithm tries to find the configuration with the largest number of paired bases.

Consider a RNA sequence,  $S[1..n]$ . Let  $V(i,j)$  be the maximum number of base pairs of any secondary structure of  $S[i..j]$ .

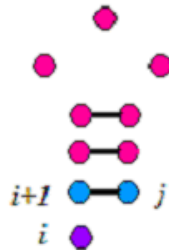


Figure 5.10: Unpaired base  $i$

- When  $i = j$ , we have the base case,  
 $V(i, i) = 0$  since the sequence has only one base.
- When  $i < j$ , then consider the four cases:

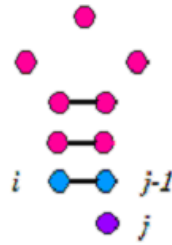


Figure 5.11: Unpaired base  $j$

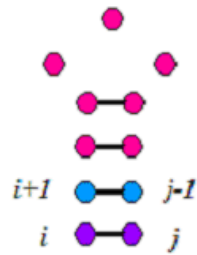


Figure 5.12: Paired  $i, j$

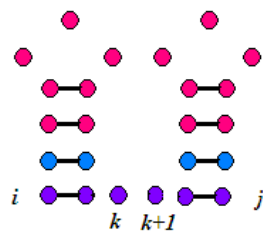


Figure 5.13: Bifurcation

1. No base pair attached to  $i$ :(See Figure 5.10 for an example)  
 $V(i, j) = V(i + 1, j)$
2. No base pair attached to  $j$ :(See Figure 5.11 for an example)  
 $V(i, j) = V(i, j - 1)$
3.  $(i, j)$  form a base pair :(See figure 5.12 for an example)  
 $V(i, j) = V(i + 1, j - 1) + \delta(S[i], S[j])$  where,  
 $\delta(x, y) = 1$ , if  $\delta(x, y) \in \{(a, u), (u, a), (c, g), (g, c), (g, u), (u, g)\}$   
 $= 0$  otherwise
4. Both  $i$  and  $j$  are attached to some base pairs and  $(i, j)$  is not a base pair :(See Figure 5.13 for an example)  
 $V(i, j) = \max_{i \leq k < j} \{V(i, k) + V(k + 1, j)\}$

- Note that cases 1 and 2 are sub-cases of 4

- For 1, put  $k = i$  in 4:  
 $V(i, j) = V(i, i) + V(i + 1, j)$   
 $= 0 + V(i + 1, j)$   
 $= V(i + 1, j).$
- For 2, put  $k = j - 1$  in 4:  
 $V(i, j) = V(i, j - 1) + V(j - 1 + 1, j)$   
 $= V(i, j - 1) + V(j, j)$   
 $= V(i, j - 1) + 0$   
 $= V(i, j - 1).$

- Therefore, we have the following recursive function:

- Base case ( $i=j$ ):  
 $V(i, j)=0$
- Recursive case ( $i < j$ ):

$$V(i, j) = \max \begin{cases} V(i + 1, j - 1) + \delta(S[i], S[j]) \\ \max_{i \leq k < j} \{V(i, k) + V(k + 1, j)\} \end{cases}$$

### 5.2.2 Time analysis

There are  $O(n^2)$   $V(i, j)$  entries. Each  $V(i, j)$  entry can be computed in  $O(n)$  time. Thus, Nussinov algorithm can be solved in  $O(n^3)$  time.

### 5.2.3 Drawbacks

The Nussinov algorithm was developed in 1978. Therefore, it is not state of the art today. However, it is a good starting point in the comprehension of modern algorithms. It returns one optimal structure based on the sole criterion of base-pairing. However, base-pairing is not enough. Some of the modern techniques are based on minimization of Gibb's free energy  $\Delta G$ . This includes scoring parameters for base-pair stacking, single dangling nucleotides, terminal mismatches and the lengths of hairpin loops, bulge loops, interior loops and multi branched loops. Furthermore, results from wet-lab experiments can be used to apply constraints on secondary structure, ionic conditions of the solvent can be set, and a number of further parameters may be fine-tuned.

### 5.2.4 Improvements

A further possibility of improving on the Nussinov algorithm is to use SCFGs (Stochastic Context Free Grammars) to generate a probabilistic model. The original algorithm changes only slightly to allow for the various probabilities in scoring and may alternatively be regarded as an adapted CYK algorithm [11].

### 5.2.5 Structure Prediction by Energy Minimization

The premises for this method of prediction is from the thermodynamic model. More the base pairs present, more stable is the structure. If base pairs are not bonded, they tend to form loops which destabilize the structure. Dynamic programming applied to this model provides a mechanism for RNA secondary structure prediction.

Two softwares which use this dynamic programming solution for predicting RNA folds are

- Zuker MFOLD algorithm <http://www.bioinfo.rpi.edu/~zukerm/rna/>
- Vienna package <http://www.tbi.univie.ac.at/~ivo/RNA/>

As stated before, this model assumes that no pseudoknots are present. In addition, this model assumes that

- Secondary structure has the lowest possible energy.
- Every loop's energy is independent of the other loops. This is not entirely true but is used for simplification.
- Energy of a secondary structure is the sum of the energies of all loops.

The values of the free energies of loops used in the algorithm come from experimentally measured values on oligonucleotides (Turner).

In this section, the details of the procedure are presented along with two methods to speed up the dynamic programming. The time complexity of the algorithm is also analysed.

## 5.2.6 Main Algorithm

### 5.2.6.1 Notations

The notations used to calculate the energy of the optimal structure for a sequence  $S$  are given below.

$eS$ ,  $eH$ ,  $eL$  and  $eM$  are pre-specified free energy functions which are used to calculate the entries of the arrays given below.

- $eS(i, j)$ :  
free energy of the stacking pair consists of base pairs  $(i, j)$  and  $(i+1, j-1)$ . No free base pairs are included, as a result, stacking pair is the only loop which can stabilize the secondary structure of RNA sequence. So the stacking pair has a negative energy. Negative energy indicates stability whereas positive energy indicates instability
- $eH(i, j)$ :  
free energy of the hairpin closed by the base pair  $(i, j)$ .
- $eL(i, j, i', j')$ :  
free energy of an internal loop or bulge enclosed by  $(i, j)$  and  $(i', j')$ .
- $eM(i, j, i_1, j_1, \dots, i_k, j_k)$ :  
free energy of a multi-loop enclosed by  $(i, j)$  and consists of  $k+1$  base pairs  $(i_1, j_1), \dots, (i_k, j_k)$ .

Four arrays -  $W$ ,  $V$ ,  $VBI$  and  $VM$  - used to hold minimum free energies of certain restricted structures of subsequence of  $S$ .

- $W(j)$ :  
Energy of the optimal secondary structure
- $V(i, j)$ :  
Energy of the optimal secondary structure for  $S[i..j]$  when  $(i, j)$  forming a base pair
- $VBI(i, j)$ :  
Energy of the optimal secondary structure for  $S[i..j]$  when  $(i, j)$  form a bulge or an internal loop.

- $VM(i, j)$ :  
Energy of the optimal secondary structure for  $S[i..j]$  when  $(i, j)$  form a multi loop

### 5.2.6.2 Detailed description

1. To compute array  $W(j)$  is to find the free energy of the optimal secondary structure for  $s[1..j]$ :

- **Initialization:**  $W(0) = 0$
- **For**  $j > 0$ :

$$W(j) = \min \begin{cases} W(j-1), & j \text{ is free;} \\ \min_{1 \leq i < j} V(i, j) + W(i-1), & j \text{ pairs with } i. \end{cases}$$

If  $j$  is a free base (Figure 5.14(a)), then no penalty needs to be added in which case  $W(j)$  and  $W(j-1)$  are the same. If  $j$  pairs with  $i$  (Figure 5.14(b)), the penalty for the new loop needs to be added. Since  $i$  varies from 1 to  $j-1$ ,  $j-1$  possible pairs need to be considered to determine the structure with the lowest energy. The energy is divided into two parts :  $W(i-1)$  and  $V(i, j)$

$W(i-1)$  is the energy of the subsequence whose energy is not affected by the new pair  $(i, j)$ .  $V(i, j)$  is the energy of the subsequence whose energy is affected by the pair  $(i, j)$ .



Figure 5.14: illustration of the calculation of  $W(j)$

2.  $V(i, j)$  is to find the free energy of the optimal secondary structure when  $S[i..j]$  with  $(i, j)$  forms a base pair.
  - **If**  $i \geq j$ ,  $V(i, j)$  is undefined.

$$\bullet \text{ If } i < j, V(i, j) = \min \begin{cases} eH(i, j), & \text{Hairpin loop;} \\ eS(i, j) + V(i + 1, j - 1), & \text{Stacking loop;} \\ VBI(i, j), & \text{Bulge/Internal loop;} \\ VM(i, j), & \text{Multi-loop.} \end{cases}$$

Once a new base pair is generated, a new loop will be formed by the base pair and the energy of the new loop should be added into the original energy sum. The new base pair,  $(i, j)$ , can form one new loop belonging to one of four loop types, hairpin loop, stacking loop, internal loop and multi-loop, so  $V(i, j)$ , the energy of the optimal secondary structure for  $s[i..j]$  with  $(i, j)$  forms a base pair, can be divided into four parts:

- $eH(i, j)$ :  $(i, j)$  forms a hairpin loop. According to the definition of hairpin loop, there is no nest loop in the hairpin loop formed by  $(i, j)$ . So the energy merely includes the free energy of the hairpin closed by the base pair  $(i, j)$ .
- $eS(i, j) + V(i + 1, j - 1)$ :  $(i, j)$  forms a stacking loop. In this loop, there could be nest loops closed by base pair  $(i + 1, j - 1)$ . So the energy of the stacking loop closed by pair  $(i, j)$  should be the sum of the stacking loop's free energy,  $eS(i, j)$ , and the sum of the nest loops's free energy,  $V(i + 1, j - 1)$ .
- $VBI(i, j)$ :  $(i, j)$  forms an internal loop. In this case,  $V(i, j)$  is equal to  $VBI(i, j)$ , which will be described in the following.
- $VM(i, j)$ :  $(i, j)$  forms a multi-loop. In this case,  $V(i, j)$  is equal to  $VM(i, j)$ , which will be described in the following.

3.  $VBI(i, j)$  represents the free energy of the optimal secondary structure for  $S[i..j]$  when  $(i, j)$  closes a bulge or internal loop (see figure 5.15).

$$VBI(i, j) = \min_{i < i' < j' < j} \{eL(i, j, i', j') + V(i', j')\}$$

Once pair  $(i, j)$  forms an internal loop (because the bulge loop belongs to the internal loop, we just discuss the internal loop in the following), we can find the base pair  $(i', j')$  ( $i < i' < j' < j$ ) with which pair  $(i, j)$  forms the internal loop. The loop's free energy is the sum of the new internal loop's free energy,  $eL(i, j, i', j')$ , and the free energy of the nest loops closed by pair  $(i', j')$ ,  $V(i', j')$ . We should find the structure with the internal loop

closed by pair  $(i, j)$  and one possible pair  $(i', j')$ , which has the minimal free energy.

4.  $VM(i, j)$  represents the free energy of the optimal secondary structure for  $S[i..j]$  with  $(i, j)$  closes a multi-loop.

$$VM(i, j) = \min_{i < i_1 < j_1 \dots < i_k < j_k < j} \{eM(i, j, i_1, j_1, \dots, i_k, j_k) + \sum_{h=1}^k V(i', j')\}$$

Calculating of  $VM(i, j)$  is akin to the calculation of  $VBI(i, j)$ . When  $k$  is equal to 1,  $VM(i, j)$  becomes  $VBI(i, j)$ .  $VM(i, j)$ 's calculation is generalized form of  $VBI(i, j)$ .

The four arrays  $W$ ,  $V$ ,  $VBI$  and  $VM$  are filled using the formulae given above. The free energy  $W(n)$  of the optimal structure of  $S[1..n]$  and the optimal structure corresponding the lowest free energy can be determined through backtracking.

### 5.2.6.3 Time Complexity

4 matrixes  $W(i)$ ,  $V(i, j)$ ,  $VBI(i, j)$  and  $VM(i, j)$  are to be filled.

- $W(i)$ :  $n$  entries are present and each entry requires finding the minimum of  $n$  terms.  $V(i, j) + W(i - 1)$  for  $i$  varying from 1 to  $j-1$ . So, each entry needs  $O(n)$  time. As a result, it costs  $O(n^2)$  time in total.
- $V(i, j)$ : It is an array with  $n^2$  entries and each entry requires finding minimum of 4 terms,  $eH(i, j)$ ,  $eS(i, j) + V(i + 1, j - 1)$ ,  $VBI(i, j)$  and  $VM(i, j)$ . So each entry needs  $O(1)$  time. As a result, it costs  $O(n^2)$  time in total.
- $VBI(i, j)$ : It is an array with  $n^2$  entries and each entry requires finding minimum of  $n^2$  terms:  $eL(i, j, i', j') + V(i', j')$  ( $i < i' < j' < j$ ), in which  $i'$  and  $j'$  both vary from 1 to  $n$  at most. So, each term needs  $O(n^2)$  time. As a result, it costs  $O(n^4)$  time in total.

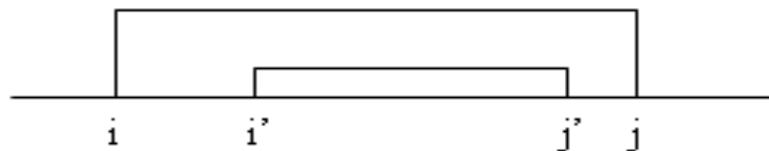


Figure 5.15: Illustration of VBI calculation

- $VM(i, j)$ : It is an array with  $n^2$  entries and each entry requires finding the minimum of exponential terms:  $eM(i, j, i_1, j_1, \dots, i_k, j_k) + \sum_{h=1}^k V(i', j')$  ( $i < i_1 < j_1 \dots < i_k < j_k < j$ ).

So in total, it costs exponential time.

- **So in total, the time cost is exponential!**

The time taken to evaluate possible multi-loops and internal loops are not acceptable.

The major problem for the algorithms is the time required to evaluate possible multi-loops and internal loops. For multi-loops, using affine linear function, the execution time can be reduced from exponential time to  $O(n^3)$  time. For internal loops, by using ninio equation, the overhead of  $VBI$  can be reduced to  $O(n^3)$  time.

Thus the overall complexity can be reduced to  $O(n^3)$  time from the original exponential time.

The two speed-up method will be discussed in detail respectively in the following.

## 5.2.7 Speed up multi-loops

### 5.2.7.1 Assumption

#### Approximating free energy for multi-loop:

To make it tractable to predict the optimal secondary structure, the following simplified assumption is made here. The energy of multi-loops can be decomposed into linear contributions from the number of unpaired bases in the loop, the number of base pairs in the loop and a constant, that is

$$eM(i, j, i_1, j_1, \dots, i_k, j_k) = a + bk + c((i_1 - i - 1) + (j - j_k - 1) + \sum_{h=1}^{k-1} (i_h + 1 - j_h - 1))$$

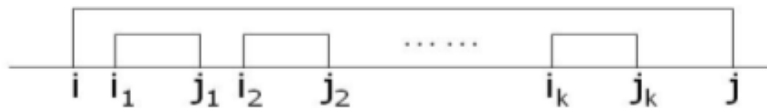


Figure 5.16: Approximating free energy for multi-loop

where  $a$ ,  $b$ ,  $c$  are constant. “ $k$ ” is the number of base pairs in the loop and  $((i_1 - i - 1) + (j - j_k - 1) + \sum_{h=1}^{k-1} (i_h + 1 - j_h - 1))$  is the number of unpaired bases in the loop.

### 5.2.7.2 Detailed description

An extra array is introduced which stores the energy of an optimal structure of the subsequence from  $i$  through  $j$  that constitutes part of a multi-loop structure, that is, where unpaired bases and external base pairs are penalized according to equation :

$$WM(i, j) = \min \begin{cases} WM(i, j - 1) + c, & \text{j is free base;} \\ WM(i + 1, j) + c, & \text{i is free base;} \\ V(i, j) + b, & \text{(i, j) is a pair;} \\ \min_{i < r \leq j} \{WM(i, r - 1) + WM(r, j)\}, & \text{i and j are not free} \\ & \text{and (i, j) is not a pair;} \end{cases}$$

- $WM(i, j)$ : When  $j$  is a free base,  $WM(i, j - 1)$  adds  $c$ , the penalty for the new free base, to become  $WM(i, j)$ .
- $WM(i, j - 1) + c$ : It is similar with  $WM(i, j - 1) + c$ .
- $V(i, j) + b$ : When  $i$  pairs with  $j$ , a nest loop closed by  $(i, j)$  is generated. So  $WM(i, j)$  should be the sum of the energy of the nest loop and  $b$ , the penalty of the pair  $(i, j)$ .
- $\min_{i < r \leq j} \{WM(i, r - 1) + WM(r, j)\}$ : When  $i$  and  $j$  are neither free nor a pair, we can find 'r' between  $i$  and  $j$  which divides the subsequence  $S[i..j]$  into two parts:  $S[i..r - 1]$  and  $S[r..j]$ . The sum of the two parts' energy give the minimal energy for the subsequence  $S[i..j]$ . Then the calculation of  $WM(i, j)$  can be reduced into the calculation of  $WM(i, r - 1)$  and  $WM(r, j)$ .

The calculation of the energy of the optimal multi-loop is now

$$VM(i, j) = \min_{i+1 < r \leq j-1} \{WM(i + 1, r - 1) + WM(r, j - 1) + a\}$$

A value 'r' can be found between  $i+1$  and  $j-1$  which divides the  $S[i..j]$  into 2 parts. The sum of the two parts' energy should be minimal. Then the sum adds the energy penalty of the multi-loop,  $a$ , equals  $VM(i, j)$ .

### 5.2.7.3 Time complexity

There are 5 arrays to be filled in as follows:  $W(i)$ ,  $V(i, j)$ ,  $VBI(i, j)$ ,  $VM(i, j)$  and  $WM(i, j)$ .

The time complexity of  $W(i)$ ,  $V(i, j)$  and  $VBI(i, j)$  is the same with the above analysis. They cost  $O(n^2)$ ,  $O(n^2)$  and  $O(n^4)$  time respectively. The array of  $WM(i, j)$  has  $n^2$  entries and each requires finding minimum of 4 terms:  $WM(i, j - 1) + c$ ,  $WM(i + 1, j) + c$ ,  $V(i, j) + b$  and the minimum of  $WM(i, k - 1) + WM(k, j)$  for  $i < k \leq j$ . To find the minimum of  $WM(i, k - 1) + WM(k, j)$  for  $i < k \leq j$  needs  $O(n)$  time. Then each entry costs  $O(n)$  time. So in total the array needs  $O(n^3)$  time.

The array of  $VM(i, j)$  has  $n^2$  entries and each requires finding minimum of  $n$  terms:  $WM(i + 1, k - 1) + WM(k, j - 1) + a(i + 1 < k \leq j - 1)$ . Then each entry will cost  $O(n)$  time. So in total the array needs  $O(n^3)$  time.

**So in total, the time cost is  $O(n^4)$  time.**

## 5.2.8 Speed up internal loops

### 5.2.8.1 Assumption

1. Assumption for internal loop/ bulge free energy

$$eL(i, j, i', j') = \text{size}(n_1 + n_2) + \text{stacking}(i, j) + \text{stacking}(i', j') + \text{asymmetry}(n_1, n_2)$$

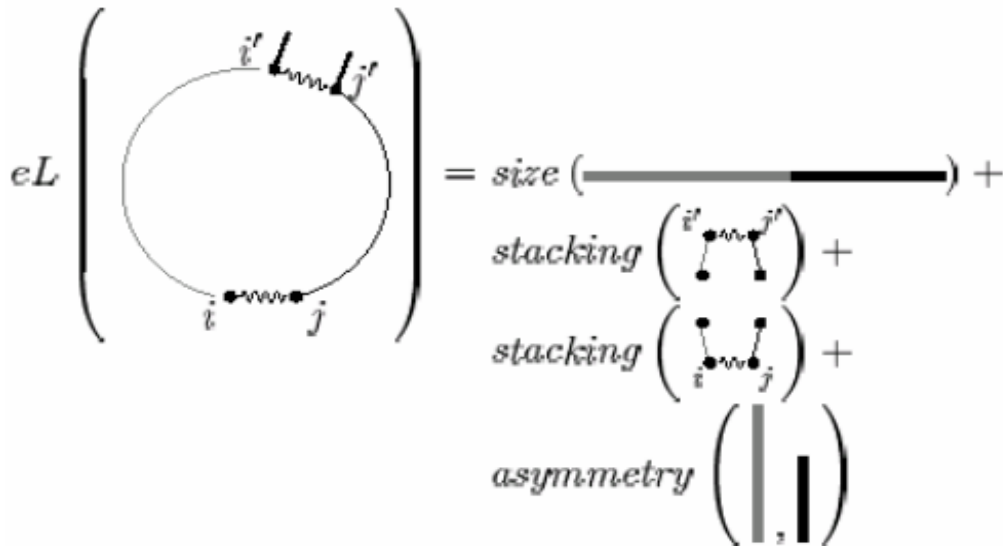


Figure 5.17: Assumption for internal loop/ bulge free energy

Currently used energy rules for internal loop stability split the contributions into three parts:

- **size( )**: energy depends on loop size
- **stacking( )**: energy for the mismatched base pair adjacent to the base pair
- **asymmetry( )**: asymmetry penalty

## 2. Asymmetry Function Assumption:

The change of the asymmetry function when varying the size while maintaining lopsidedness thus only depends on the size of the loop.

$$asymmetry(k+1, l+1) = asymmetry(k, l) + g(k+l)$$

$$asymmetry(k, l) = lopsidedness(|k-l|) + size'(k+l)$$

$$asymmetry(k+1, l+1) = lopsidedness(|k-l|) + size'(k+l+2)$$

$$\Rightarrow g(k+l) = size'(k+l+2)$$

The size-dependence of the asymmetry function,  $g(k+l)$ , can be moved into the size function,  $size(n_1+n_2)$ , of the overall internal loop energy function. So we can get the assumption that  $asymmetry(n_1, n_2)$  is only depend on the difference of  $n_1$  and  $n_2$ . More precisely, we assume that, when  $n_1, n_2 > c$ ,

$$asymmetry(n_1, n_2) = asymmetry(n_1-1, n_2-1)$$

Currently by using the Ninio equation, which is

$$asymmetry(n_1, n_2) = \min\{K, |n_1 - n_2|f(m)\}$$

where  $m = \min\{n_1, n_2, c\}$ ,  $K$  and  $c$  are constants, and  $f(m)$  is any function. Note that  $asymmetry(n_1, n_2)$  satisfies the above assumption and  $c$  is proposed to be 1 and 5 in two literatures [8].

### 5.2.8.2 Detailed description

Based on the two assumptions,  $VBI(i, j)$  for all  $i < j$  can be found in  $O(n^3)$  time. The difference in destabilizing energy when extending a loop from being closed by pair  $(i, j)$  to being closed by pair  $(i-1, j-1)$  is determined only by the size of the loop and the change in stacking stability of the closing base pair. We can thus reuse the comparisons between different choices of base pairs, e.g. pair  $(i', j')$  and  $(i'', j'')$ . (see Figure 5.18)

**lemma 1** Consider  $i < i' < j' < j$ . Let  $n_1 = i' - i - 1, n_2 = j - j' - 1$  and  $l = n_1 + n_2$ . For  $n_1 > c$  and  $n_2 > c$ , we have

$$eL(i, j, i', j') - eL(i+1, j-1, i', j') = size(l) - size(l-2) + stacking(i, j) - stacking(i+1, j-1)$$

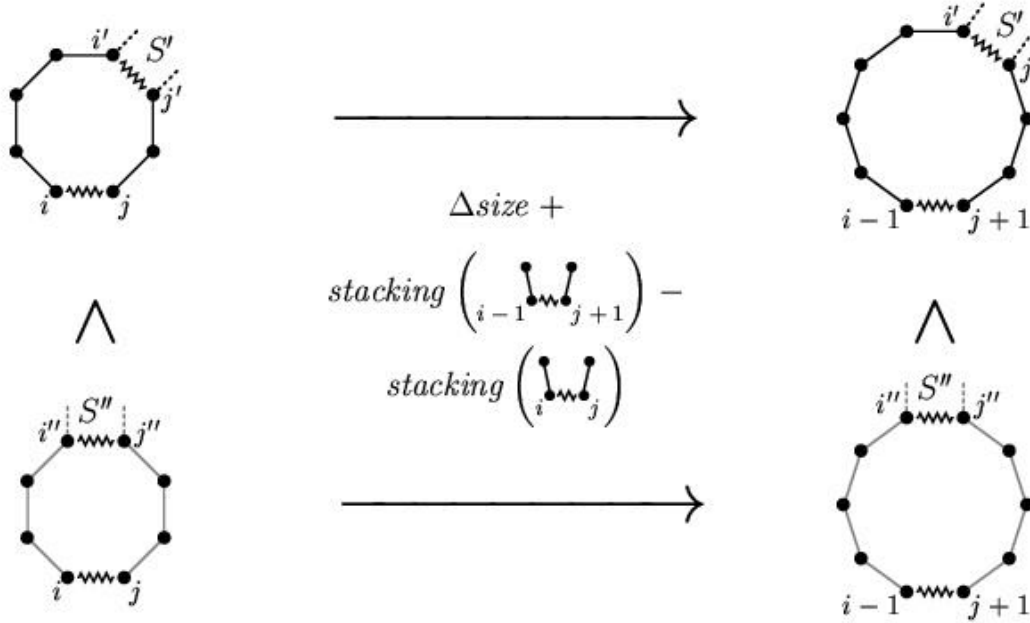


Figure 5.18: Illustrations of Base Pairs

*Proof.*

$$\begin{aligned}
 & eL(i, j, i', j') - eM(i+1, j-1, i', j') \\
 &= [size(l) + stacking(i, j) + stacking(i', j') + asymmetry(n_1, n_2)] \\
 & - [size(l-2) + stacking(i+1, j-1) + stacking(i', j') + asymmetry(n_1-1, n_2-1)] \\
 &= size(l) - size(l-2) + stacking(i, j) - stacking(i+1, j-1)
 \end{aligned}$$

A new array  $VBI'$  is defined such that  $VBI'(i, j, l)$  is the minimal energy of an internal loop of size  $l$  closed by pair  $(i, j)$ .  $VBI''$  is also a new array which is the minimal energy of an internal loop for size  $l$  closed by pair  $(i, j)$ . Moreover,  $VBI''$  requires the number of the bases between  $i$  and  $i'$  and the number of the bases between  $j$  and  $j'$ , excluding  $i, i', j$  and  $j'$ , are both more than a constant  $c$ . For example, if  $c$  is equal to 1,  $VBI''$  stores the energies of internal loops excluding bulge loops.

$$VBI'(i, j, l) = \min_{i < i' < j' < j \wedge i' - i - 1 + j - j' - 1 = l} \{eL(i, j, i', j') + V(i', j')\}$$

$$VBI''(i, j, l) = \min_{i < i' < j' < j \wedge i' - i - 1 + j - j' - 1 = l \wedge i' - i - 1, j - j' - 1 > c} \{eL(i, j, i', j') + V(i', j')\}$$

By definition, we have

$$VBI(i, j) = \min_l \{VBI'(i, j, l)\}$$

Based on the lemma 1, we have

$$VBI''(i, j, l) - VBI''(i+1, j-1, l) = size(l) - size(l-2) + stacking(i, j) + stacking(i+1, j-1)$$

$$VBI'(i, j, l) =$$

$$\min \begin{cases} \text{close } VBI''(i+1, j-1, l) + size(l) - size(j-2) + stacking(i, j) - stacking(i+1, j-1) \\ \min_{1 \leq d \leq c} V(i+d, j-l-d) + eL(i, j, i+d, j-l+d-2) \\ \min_{1 \leq d \leq c} V(i+l+d, j-d) + eL(i, j, i+l-d+2, j-d) \end{cases}$$

The last two entries of the above equation handle the cases where this minimum is obtained by an internal loop, in which  $d$  is less than a constant  $c$ , especially a bulge loop when  $c$  is equal to 1, that is at  $j' = i + 1$  or  $j' = j - 1$ .

### 5.2.8.3 Time analysis

The array of VBI has  $O(n^3)$  entries. Each entry can be computed in  $O(c)$  time. Thus, this table can be filled in  $O(cn^3)$  time. The array of VBI has  $O(n^3)$  entries. Each entry can be computed in  $O(1)$  time. Thus, this table can be filled in  $O(n^3)$  time. **In total, it takes  $O(n^3)$  time to compute VBI(i,j) for all  $i < j$ .**

## 5.3 RNA Secondary Structure Prediction with Pseudoknots

### 5.3.1 Introduction

Up to now, there is no good way to predict RNA secondary structure with pseudoknots. In fact, this problem is an NP-hard problem. Many approaches and research has been done to address this problem. Heuristic search procedures are adopted in most RNA folding methods that are capable of folding pseudoknots. For example, quansi-Monte Carlo searches by Abrahams et al. [2], genetic algorithms by Gulyaev et al. [3], method based on Hopfield network by Akiyama and Kanehisa [4] and another method based on the stochastic context-free grammar by Brown and Wilson [5]. These approaches are not able to guarantee the best structure can be found and unable to say how far a given prediction is from optimality [6].

A dynamic programming algorithm for predicting optimal pseudoknotted RNA structures was first introduced by Rivas et al. [6]. The algorithm has a worst case of  $O(n^6)$  time and  $O(n^4)$  space. Although the demands of this algorithm are admittedly high, it demonstrates that using certain approximations consistent with the accepted Turner thermodynamic model, the prediction of pseudoknotted structures is a problem of polynomial complexity.

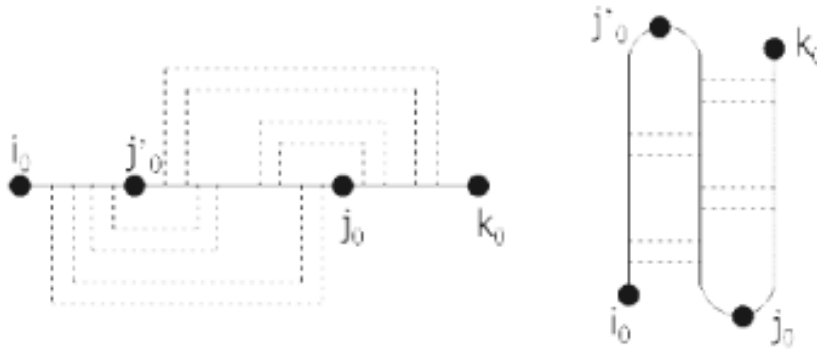


Figure 5.19: An illustration of simple pseudoknot

In this lecture, a dynamic programming algorithm for RNA secondary structure prediction with a particular kind of pseudoknot - simple pseudoknot is discussed based on an algorithm proposed by Akutsa [7]. The algorithm has  $O(n^4)$  time complexity.

## 5.3.2 Definition

### 5.3.2.1 Simple Pseudoknot

Let  $S[i_0..k_0]$  be a substring of an RNA sequence  $S$  where  $i_0$  and  $k_0$  are arbitrary chosen positions. A set of base pairs  $M_{i_0,k_0}$  is a simple pseudoknot if there exist  $i_0 < j'_0 < j_0 < k_0$  such that

- Each endpoint  $i$  appear in  $M_{i_0,k_0}$  once.
- Each base pair  $(i, j)$  in  $M_{i_0,k_0}$  satisfies either  $i_0 \leq i < j'_0 < j \leq j_0$  or  $j'_0 \leq i < j_0 < j \leq k_0$
- If pairs  $(i, j)$  and  $(i', j')$  in  $M_{i_0,k_0}$  satisfies  $i < i' < j'_0$  or  $j'_0 \leq i < i'$ , then  $j > j'$ .

The sequence  $(i_0, k_0)$  is divided into three parts using  $(j'_0, j_0)$  as shown in Figure 5.19. For each base pair in  $M_{i_0,k_0}$ , if it starts in the section  $(i_0, j'_0)$ , it must ends with a point in segment  $(j'_0, j_0)$  and if it starts in the segment  $(j'_0, j_0)$ , it must ends with a point in the segment  $(j_0, k_0)$ . Besides, one base must not exist in two different base pairs and base pairs in one segment must not intersect with each other.

### 5.3.2.2 RNA Secondary Structure with Simple Pseudoknot

With the definition of simple pseudoknot, we define RNA secondary structure with simple pseudoknot as follow.

A set of base pairs  $M$  is called an RNA secondary structure with simple pseudoknots if the following conditions are satisfied:

- $M = M' \cup M_1 \cup M_2 \dots M_t$  where  $t$  is a non-negative integer,
- Each  $M_h$  is a simple pseudoknot for  $S[i_h..k_h]$  where  $1 \leq i_1 < k_1 < i_2 < k_2 < \dots < i_t < k_t \leq n$ ,
- $M'$  is a secondary structure without pseudoknots for string  $S'$  where  $S'$  is obtained by deleting all  $S[i_h..k_h]$ .

If after removing all the simple pseudoknots in  $M$ , the remaining sequence is still not a usual RNA secondary structure without pseudoknots, then we conclude  $M$  is not a RNA secondary structure with pseudoknot.

### 5.3.3 Formulation of Problem

Now the problem can be defined as follow:

**Input:** an RNA sequence  $S[1..n]$

**Output:** an RNA sequence secondary structure with simple pseudoknots.

The score function used in this problem is different from that of the RNA secondary structure without pseudoknots. The score function here is the number of the base pairs in  $S[1..n]$ . Our aim is to maximize the number of base pairs.

### 5.3.4 Main Algorithm

Let  $S[i..j]$  be a subsequence of  $S[1..n]$ . Let  $V(i, j)$  be the optimal score for  $V[i..j]$  and  $V_{pseudo}$  be the optimal score for a pseudoknot in  $V[i..j]$ .

$$V(i, j) = \max \begin{cases} V_{pseudo}(i, j) \\ V(i+1, j-1) + \delta(S[i], S[j]) \\ \max_{i < k \leq j} \{V(i, k-1) + V(k, j)\} \end{cases}$$

With

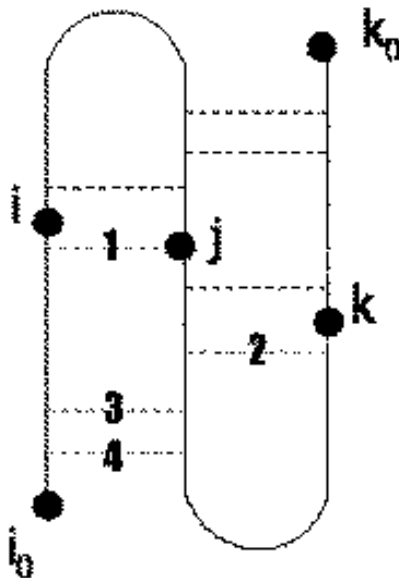
$$\delta(S[i], S[j]) = 1 \text{ if } \{S[i], S[j]\} = \{a, u\}, \{c, g\} \text{ or } \{g, u\}$$

Otherwise,

$$\delta(S[i], S[j]) = -\infty$$

Besides,

$$V(i, i) = 0 \text{ for any } i$$

Figure 5.20: An illustration of Triplet  $(i, j, k)$ 

There is  $n^2$  entries for  $V(i, j)$  and each can be computed in  $O(n)$  time if for all  $i, j$ ,  $V_{pseudo}(i, j)$  are available. Then  $V$  can be filled in using  $O(n^3)$  time. The crux of the algorithm is to compute  $V_{pseudo}(i, j)$ .

### 5.3.5 Sub Algorithm

#### 5.3.5.1 To compute $V_{pseudo}(i, j)$

We use dynamic programming to calculate  $V_{pseudo}(i, j)$ . First we give some terminology.

Given a set of base pairs in simple pseudoknot for  $S[i_0, k_0]$ , a base pair is said to be below the triplet  $(i, j, k)$  if they are the edges with numbers. (see Figure 5.20)

For a triplet  $(i, j, k)$  there are 3 possible cases,  $(i, j)$  is a base pair,  $(j, k)$  is a base pair or both  $(i, j)$  and  $(j, k)$  are not base pairs. To separate these, we define for  $i_0 < i < j < k < k_0$ ,

- $V_L(i, j, k)$  be the maximum number of base pairs below the triplet  $(i, j, k)$  in a pseudoknot for  $S[i_0..k_0]$  with  $(i, j)$  is a base pair
- $V_R(i, j, k)$  be the maximum number of base pairs below the triplet  $(i, j, k)$  in a pseudoknot for  $S[i_0..k_0]$  with  $(j, k)$  is a base pair

- $V_M(i, j, k)$  be the maximum number of base pairs below the triplet  $(i, j, k)$  in a pseudoknot for  $S[i_0..k_0]$  with both  $(i, j)$  and  $(j, k)$  are not a base pair

Note that  $\max\{V_L(i, j, k), V_R(i, j, k), V_M(i, j, k)\}$  is the maximum number of base pairs below the triplet  $(i, j, k)$  in a pseudoknot for  $S[i_0..k_0]$ . Now  $V_{pseudo}(i_0, j_0)$  can be calculated.

$$V_{pseudo}(i_0, k_0) = \max_{i_0 \leq i < j < k \leq k_0} \{V_L(i, j, k), V_M(i, j, k), V_R(i, j, k)\}$$

### 5.3.5.2 To compute $V_L(i, j, k)$ , $V_R(i, j, k)$ , $V_M(i, j, k)$

Based on the definition of  $V_L(i, j, k)$ ,  $V_R(i, j, k)$ ,  $V_M(i, j, k)$ , we calculate them using the following formula.

$$V_L(i, j, k) = \delta(S[i], S[j]) + \max \left\{ \begin{array}{l} V_L(i-1, j+1, k) \\ V_M(i-1, j+1, k) \\ V_R(i-1, j+1, k) \end{array} \right\}$$

In this case,  $(i, j)$  is a pair. Thus  $V_L(i, j, k)$  is equal to  $\delta(S[i], S[j])$ , which is the cost of a base pair, plus the maximum number of base pairs below  $(i-1, j+1, k)$ . Similarly we have,

$$V_R(i, j, k) = \delta(S[i], S[j]) + \max \left\{ \begin{array}{l} V_L(i, j+1, k-1) \\ V_M(i, j+1, k-1) \\ V_R(i, j+1, k-1) \end{array} \right\}$$

$$V_M(i, j, k) = \max \left\{ \begin{array}{l} V_L(i-1, j, k), V_M(i-1, j, k) \\ V_L(i, j+1, k), V_M(i, j+1, k), V_R(i, j+1, k) \\ V_M(i, j, k-1), V_R(i, j, k-1) \end{array} \right\}$$

### 5.3.5.3 To compute basis

The basis values for  $V_L(i, j, k)$ ,  $V_R(i, j, k)$ ,  $V_M(i, j, k)$  is as follow.

$$\left\{ \begin{array}{l} V_R(i_0 - 1, j, j+1) = \delta(S[j], S[j+1]) \quad \forall j \\ V_R(i_0 - 1, j, j) = 0 \quad \forall j \end{array} \right.$$

$$\left\{ \begin{array}{l} V_L(i, j, j) = \delta(S[i], S[j]) \quad \forall i_0 \leq i < j \\ V_L(i_0 - 1, j, k) = 0 \quad \forall k = j+1 \vee k = j \end{array} \right.$$

$$V_M(i_0 - 1, j, k) = 0 \quad \forall k = j+1 \vee k = j$$

The explanation for basis(3) is showed in Figure 5.21(a), basis(1) in Figure 5.21(b), basis(2) in Figure 5.21(c), basis(4) and basis(5) are trivial because of their out of range.

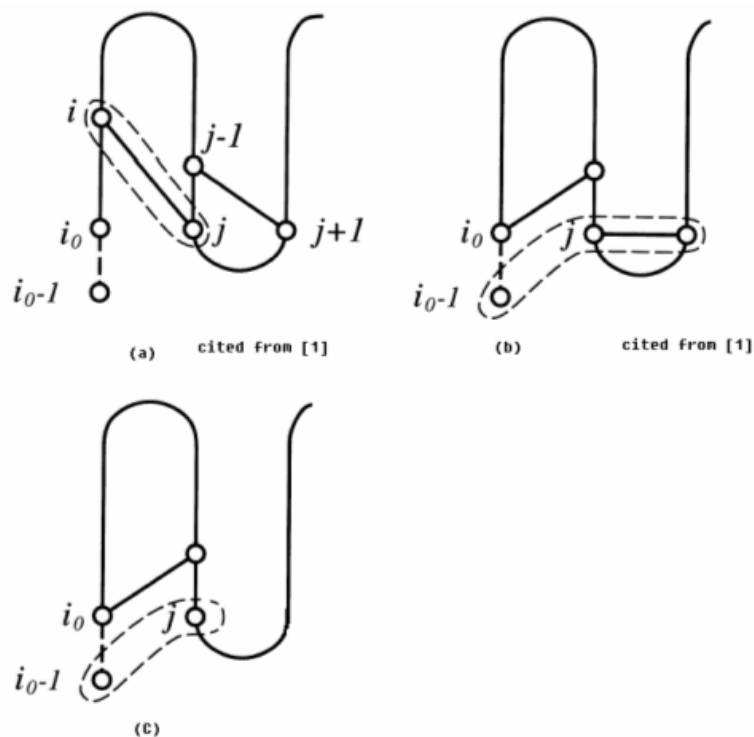


Figure 5.21: Basis

#### 5.3.5.4 Time Complexity for computing $V_{pseudo}(i, j)$

The intuitive time complexity for computing all  $V_{pseudo}(i, j)$  is  $O(n^5)$ . The proof is as follows.

- For a fixed  $i_0, k_0$ , the basis can be computed in  $O(n^2)$ . Each of  $V_L, V_R, V_M$  has 3 variables, but when calculating the basis at least one of the variables is fixed. So at most 2 variables change in the range from 1 to  $n$  when the basis is calculated. The time complexity for computing basis with fixed  $i_0, k_0$  is then  $O(n^2)$ .

- $V_L, V_R, V_M$  can be computed in  $O(n^3)$  time since each of  $V_L, V_R, V_M$  has  $n^3$  entries and each entry can be computed in  $O(1)$  time if the basis has been computed. Then the time complexity for computing  $V_L, V_R, V_M$  is  $O(n^3)$ .

- For every  $i_0, k_0$ , the computation for  $V_{pseudo}(i_0, k_0)$  must compare  $3n^3$  values and each comparison needs  $O(1)$  time if  $V_L, V_R, V_M$  have been computed. Then the time complexity for computing  $V_{pseudo}(i_0, k_0)$  with fixed  $i_0, k_0$  is  $O(n^3)$ .

- It takes  $O(n^5)$  time to compute  $V_{pseudo}(i, j)$  for all  $i_0 < k_0$  since  $V_{pseudo}(i, j)$  has  $n^2$  entries and each of them need  $O(n^3)$  time. So the total time complexity is  $O(n^5)$ .

Actually this algorithm can run in  $O(n^4)$  time. Note that the basis only depends on  $i_0$  and  $V_L, V_R, V_M$  only depends on the basis. Thus for a fixed  $i_0$  for any  $k_0$  the values of table  $V_L, V_R, V_M$  are the same. Then we can compute  $V_{pseudo}(i_0, k_0)$  for a fixed  $i_0$  and for any  $k_0$  in  $O(n^3)$  time.

Then when we want to compute the table  $V_{pseudo}(i,j)$ , we only need to change the value of  $i$  from 1 to  $n$ , for each  $i$  it takes  $O(n^3)$  time and the total time complexity will be  $O(n^4)$ .

### 5.3.6 Conclusion

The table  $V_{pseudo}$  can be filled using  $O(n^4)$ . Then table  $V$  can be filled using  $O(n^4)$  after table  $V_{pseudo}$  has been filled. Thus the RNA secondary structure problem with simple pseudoknots can be solved in  $O(n^4)$  time.

## References

- [1] TEN DAM, E., PLEIJ, K. and DRAPER, D., "Structural and functional aspects of RNA pseudoknots." *Biochemistry* 31, 11665-11676, 1992.
- [2] ABRAHAMS, J. P., VAN DER BERG, M., VAN BATENBERG, E. and PLEIJ, C. W. A., "Prediction of RNA secondary structure, including pseudoknotting, by computer simulation.", *Nucl. Acids Res.*, 3035-3044, 1990.
- [3] GULTYAEV, A. P., VAN BATENBURG, F. H. and PLEIJ, C. W. A., "The computer simulation of RNA folding pathways using a genetic algorithm.", *J. Mol. Biol.*, 250, 37-51, 1995.
- [4] AKUTSU, T., "Approximation and exact algorithms for RNA secondary structure prediction and recognition of stochastic context-free languages", *J. Combin. Optim.*, 3, 321-336, 1999.
- [5] BROWN, C. W., "RNA pseudoknots modeling using intersections of stochastic context free grammars with applications to database search", *Pacific Symposium on Biocomputing'96*, 109-125, 1996.
- [6] RIVAS, E. and Eddy, S., "A dynamic programming algorithm for RNA structure prediction including pseudoknots.", *Journal of Molecular Bio.*, 285, 2053-2068, 1999.
- [7] AKUTSU, T., "Dynamic programming algorithm for RNA secondary structure prediction with pseudoknots.", *In Discrete Applied Mathematics*, 45-62, 2000.

- [8] RUNE B. LYNGSO, MICHAEL ZUKER, CHRISTIAN N.S. PEDERSEN, “An Improved Algorithm for RNA Secondary Structure Prediction”, *BRICS Report Series, RS-99-15*, 1999.
- [9] RUNE B. LYNGSO, “Computational Biology : Ph.D. dissertation”, *BRICS Dissertation Series, DS-00-5*, 2000.
- [10] ERIC ROUCHKA, “RNA Secondary Structure Prediction: Lecture Notes”, <http://kbrin.a-bldg.louisville.edu/~rouchka/CECS694/Lecture11.htm>, 2003.
- [11] DURBIN, EDDY, KROGH, AND MITCHISON, “Biological sequence analysis”, *Cambridge University Press* 1998.
- [12] ZUKER, “Calculating nucleic acid secondary structure”, *Current Opinion in Structural Biology* , 2000.
- [13] JUAN ET AL., “RNA Secondary Structure Prediction Based on Free Energy and Phylogenetic Analysis”, *Journal of Molecular Biology* , 1999.
- [14] ZUKER, “On Finding All Suboptimal Foldings of an RNA Molecule”, *Science* , 1989.
- [15] ZUKER , “Prediction of RNA secondary structure by energy minimization”, *Methods Molecular Biology* , 1994.
- [16] R. NUSSINOV, G. PIECZNIK, J. R. GRIGG AND D. J. KLEITMAN, “Algorithms for loop matchings”, *SIAM Journal on Applied Mathematics* , 1978.