6-27-2007

Structure Pattern Analysis Using Term Rewriting and Clustering Algorithm

Xuezheng Fu

Structure Pattern Analysis Using Term Rewriting and Clustering Algorithm

By

XUEZHENG FU

Under the Direction of Robert W. Harrison

ABSTRACT

Biological data is accumulated at a fast pace. However, raw data are generally difficult to understand and not useful unless we unlock the information hidden in the data. Knowledge/information can be extracted as the patterns or features buried within the data. Thus data mining, aims at uncovering underlying rules, relationships, and patterns in data, has emerged as one of the most exciting fields in computational science. In this dissertation, we develop efficient approaches to the structure pattern analysis of RNA and protein three dimensional structures. The major techniques used in this work include term rewriting and clustering algorithms. Firstly, a new approach is designed to study the interaction of RNA secondary structures motifs using the concept of term rewriting. Secondly, an improved K-means clustering algorithm is proposed to estimate the number of clusters in data. A new distance descriptor is introduced for the appropriate representation of three dimensional structure segments of RNA and protein three dimensional structures. The experimental results show the improvements in the determination of the number of clusters in data, evaluation of RNA structure similarity,

RNA structure database search, and better understanding of the protein sequence-structure correspondence.

INDEX WORDS:

Data mining, Knowledge discovery, Term rewriting, K-means clustering algorithm, Validation measure, Stability, and Bioinformatics.

Structure Pattern Analysis Using Term Rewriting and Clustering Algorithm

by

XUEZHENG FU

A Dissertation Submitted in Partial Fulfillment of Requirements for the Degree of

Doctor of Philosophy

In the College of Arts and Sciences

Georgia State University

2007

Structure Pattern Analysis Using Term Rewriting and Clustering Algorithm

by

XUEZHENG FU

Major Professor: Robert W. Harrison
Committee: Rajshekhar Sunderraman
Saeid Belkasim
Markus W. Germann

Electronic Version Approved:

Office of Graduate Studies

College of Arts and Sciences

Georgia State University

August 2007

Acknowledgments

I would like to express my gratitude to my advisor, Dr. Robert W. Harrison for his understanding, encouragement, and insightful advice during the process of my Ph.D. dissertation. In each phase of this research, he provided invaluable support. The dissertation would not have been possible without his helps.

I want to acknowledge my committee members, Dr. Rajshekhar Sunderraman, Dr. Saeid Belkasim and Dr. Markus W. Germann for their helpful comments and advice.

I would also like to thank the Department of Computer Science for the financial support, and the opportunity to complete this degree.

Thank you my family and friends for your support and beliefs.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

RNA: Ribonucleic acid

HFSS: High Frequent Structure Segment

MTRG: Mersenne Twister Random Generator

DM: Distance Matrix

HSSP: Homology-derived Secondary Structure of Proteins

DSSP: Definition of Secondary Structure of Proteins

NMR: Nuclear Magnetic Resonance

RMSD: Root Mean Square Deviation

Chapter 1

Introduction

Biological data is produced and collected at a growing pace (Berman et al., 2000; Berman et al., 1993). However, raw data are generally difficult to understand and not useful unless we unlock the information hidden in the data. Knowledge/information can be extracted as the patterns or features buried in the data. Thus data mining, aims at uncovering underlying rules, relationships, and patterns in data, has emerged as one of the most exciting fields in computational science. In this dissertation, we develop new efficient algorithms for mining the structure patterns of RNA and protein data. The major techniques used in our work include term rewriting (Baader and Nipkow, 1999; Clavel et al., 2003) and clustering algorithms (Jain et al., 1999). The application results show the improvements in the determination of the number of clusters in data, evaluation of RNA structure similarity, RNA structure database search, classification of RNA structures, and better understanding of the protein sequence-structure correspondence.

1.1 Motivation

Recently, due to the development of X-ray and NMR techniques, the number of ribonucleic acid (RNA) and protein three dimensional structures is increasing at a growing pace (Berman et al., 2000; Berman et al., 1993). RNA plays a critical role in mediating every step of cellular information transfer from genes to functional proteins (Batey et al., 1999), and proteins are essential parts of all living organisms and participate

in every process within cells (Branden et al., 1998). However, experimental methods to determine RNA or protein structures are still highly labor intensive and expensive. Therefore, systematic approaches to automatically identify the structure patterns of RNA and protein structures are highly desirable. Though much effort has been made in this research field, there are still many problems under discussion and no one method is widely accepted, especially for RNA structure analysis. In this dissertation, we decide to uncover the structure patterns of RNA and protein structures using term rewriting and K-means clustering algorithm. The hard problems and even open problems we have to solve include selection, transformation, and representation of RNA secondary motifs, three-dimension structure segments, and protein secondary structures; definition of terms, equations, and rules in term rewriting system; determination of the number of clusters in data; validation measure of clustering, and evaluation of structure similarity.

1.2 Challenges

When mining structure patterns from RNA and protein data, it is a big challenge to make the algorithms and uncovered hidden information meet the following criteria:

To an algorithm:

- Efficient: The algorithm must have low computational complexity.
- Reliable: The independency of the algorithm on certain parameters or initialization conditions should be minimized. The algorithm should generate stable results for similar data.
- Extensible: The algorithm is easy-understanding and flexible to be extended.

To the uncovered hidden information:

- New: The uncovered hidden information must be irredundant.
- Correct: Correct and meaningful results are highly dependent on appropriate selection or representation of data. Thus specific domain experts are needed for verification of the data and the uncovered information.
- Applicable: The uncovered information should be useful.

In this dissertation, we try to reach the above criteria from every step of data mining process: data collection, data preprocessing, data mining, and information interpretation. The objective of data mining is to uncover underlying rules, relationships, and patterns in data (Hand et al., 2001). The diagram of data mining process is shown in Figure 1.1 and further explanations of each phase are presented.

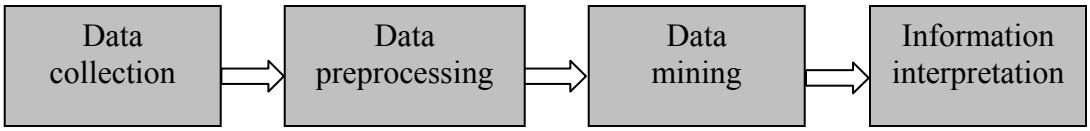


Figure 1.1. Data mining process.

- Data collection: Sometimes huge amounts of raw data are collected from different sources using different techniques. But most of them are not that useful. Inappropriate selection or representation data will lead to incorrect results. So the fields/attributes directly related to the target problem should be carefully selected from the raw data.
- Data preprocessing: Preprocessing is an important step in data mining. It includes removal of erroneous data and transformation of data. Erroneous data will lead to incorrect results. Transformation of data to a suitable representation, such as

encoding into a vector and reducing the dimensions, will decrease the computational complexity and data mining effort.

- Data mining: At this step, algorithms and models are designed for the data. Suitable techniques should be chosen according to a certain task.
- Information interpretation: The novelty, correctness, and usefulness of the information extracted from the data must be interpreted by domain experts.

1.3 Contributions

The contributions of our work are broadly summarized as follows:

- Transformation and representation of data: appropriate transformation and representation of sequence and structure information of RNAs and proteins needs specific domain knowledge and tools. Correct and useful results are dependent on the choice of suitable transformation and representation of data. We show how to efficiently interpret the data for further successful process in this dissertation.
- A new implementation of term rewriting system: To study the relationship between the RNA secondary structures and RNA pseudoknots, we define terms, equations, and rules for a new implementation of term rewriting using Maude system (Clavel et al., 2003). To our knowledge, it is the first time that term rewriting is used to study problems related to RNA three dimensional structures.
- Determination of the number of clusters in data: we propose a new algorithm to estimate the reliability of the number of clusters in data based on K-means clustering algorithm. Stability with respect to bootstrap sampling is adapted as the cluster validation measure for estimating the reliable clustering.

- Converting a three dimensional structure to a string: we uncover the high frequent structure segments (HFSS) and their similarity from RNA with known structures. Thus an RNA three dimensional structure can be converted into a string of HFSS. This not only reveals the RNA three dimensional structure elements but also simplify many originally hard problems if we try to solve them using graph algorithms.
- Rapid RNA structure database search: we simplify the evaluation of similarity between RNA three dimensional structures by converting an RNA three dimensional structure into a string of HFSS. It makes the rapid RNA structure database search feasible.

1.4 Organization

The organization of this dissertation is as follows:

In chapter 2, a new approach is proposed to uncover the relationship between RNA secondary structures and RNA pseudoknots using term rewriting. The Maude term rewriting system is chosen for the implementation. The effectiveness of our approach is tested on a set of RNA data from PseudoBase (Batenburg et al., 2000).

Chapter 3 provide an introduction to K-means clustering algorithm and discussion of some general problems on K-means, including the discovery of suitable number of clusters in data, initialization of K-means, and evaluation of cluster similarity. An improved K-means algorithm is presented for determination of the suitable number of clusters in a data.

In Chapter 4, the improved K-means clustering algorithm is applied to the problem of structure pattern analysis of RNA three dimensional structures. The suitable window size for the representation of RNA structure segments is detected. Based on this window size, the high frequent structure segments (HFSS) and the similarity between different types of HFSS are found. It makes it possible to convert an RNA three dimensional structure into a string of HFSS. Many originally hard problems like similarity evaluation of RNA three dimensional structures and RNA structure database search can be simplified based on this work.

In Chapter 5, the improved K-means clustering algorithm is used to cluster the protein three dimensional structures, including structure segments and sequence segments. Six protein structure datasets are tested in the experiment, and the experimental results are validated by both biological measure and statistical measure. Though the true number of clusters in current available protein structures in unknown, our algorithm combined with bootstrap sampling is helpful to estimate the range of number of clusters.

Chapter 6 summarizes this dissertation and discusses the future works.

Chapter 2

Pattern Analysis Using Term Rewriting

RNA plays a critical role in mediating every step of cellular information transfer from genes to functional proteins. Pseudoknots are functionally important and widely occurring structural motifs among most prevalent RNA structures. It is widely recognized that RNA secondary structures play an important role in RNA folding. In this dissertation, a new method is proposed to uncover the relationship between RNA secondary structures and RNA pseudoknots using the concept of term rewriting. The method is implemented using the Maude system (Clavel et al., 2003), a formal language and tool set based on term rewriting logic. In our implementation, RNA structures are treated as terms and equations. Rules are discovered for indicating the relationship between RNA secondary structure and RNA pseudoknots. Our method is tested on RNA pseudoknots in PseudoBase and the experimental results show that methods based on term rewriting have capacity of description of data and properties, representation of general rules, and execution of queries using logical inference, and is fit for the simulation of complex biological system.

2.1 Introduction

2.1.1 RNA Structure

RNA primary structure is the nucleotide sequence of four bases A(Adenine), C(cytosine), G(guanine), U(Uracil). The pattern of base pairing determines the secondary structure of RNA. Watson-Crick(A=U and G≡C) and Wobble(G=U) are widely occurring

stable base pairs in RNAs while other base pairs are possible but less stable and often ignored. The secondary structure can be decomposed into a few types of secondary structural motifs: stem, hairpin loop, bulge, internal loop, multi-branched loop, start sequence and external sequence, which are shown in Figure 2.1. Three dimensional folding of an RNA molecule under appropriate conditions forms the RNA tertiary structure. The comprehensive introduction to RNA structures can be seen in (Saenger et al., 1984; Batey et al., 1999; Staple and Butcher, 2005).

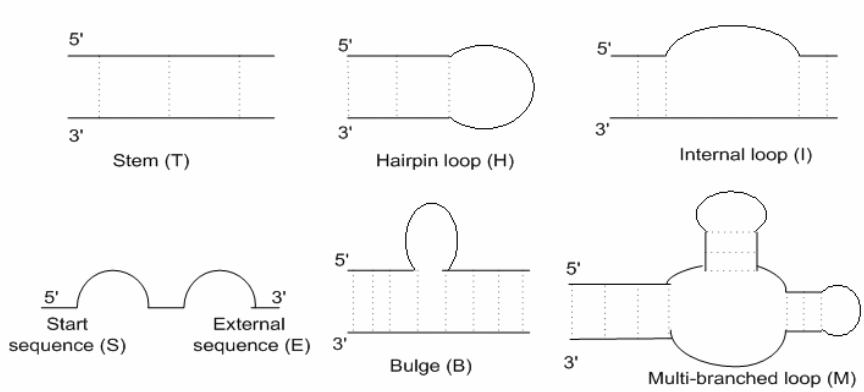


Figure 2.1. RNA secondary structural motifs. T: stem, H: hairpin loop, I: internal loop, S: start sequence, E: external sequence, B: bulge, M: multi-branched loop.

An RNA pseudoknot is a tertiary structural element that occurs in all classes of RNA and has shown to be important in many biological functions (Staple and Butcher, 2005). The topology of pseudoknot can be defined as follows.

Let S be an RNA sequence $S = a_1 a_2 a_3 a_4 ... a_n$ where n is the number of bases in the RNA sequences. (a_i, a_j) and (a_h, a_k) are two distinct base pairs in the sequence, where $i < j$ and $h < k$. i, j, h, and k are integers between 1 and n.

Pseudoknots are formed by base pairing between a secondary loop structure and

compliment bases outside the loop. According to the work of Cornelis et al. (Chastain and Tinoco Jr., 1991), there are three main types of pseudoknots: I-type pseudoknot, H-type pseudoknot, and B-type pseudoknot. I-type pseudoknot is formed between the secondary motif internal loop and the bases outside the loop (see Figure 2.2(a)); H-type pseudoknot is established between the secondary motif hairpin loop and the bases outside the loop (see Figure 2.2(b)); B-type pseudoknot is a structure constructed by the secondary motif bulge loop and the bases outside the loop (see Figure 2.2(c)). The structures of different types of pseudoknots are shown in Figure 2.2.

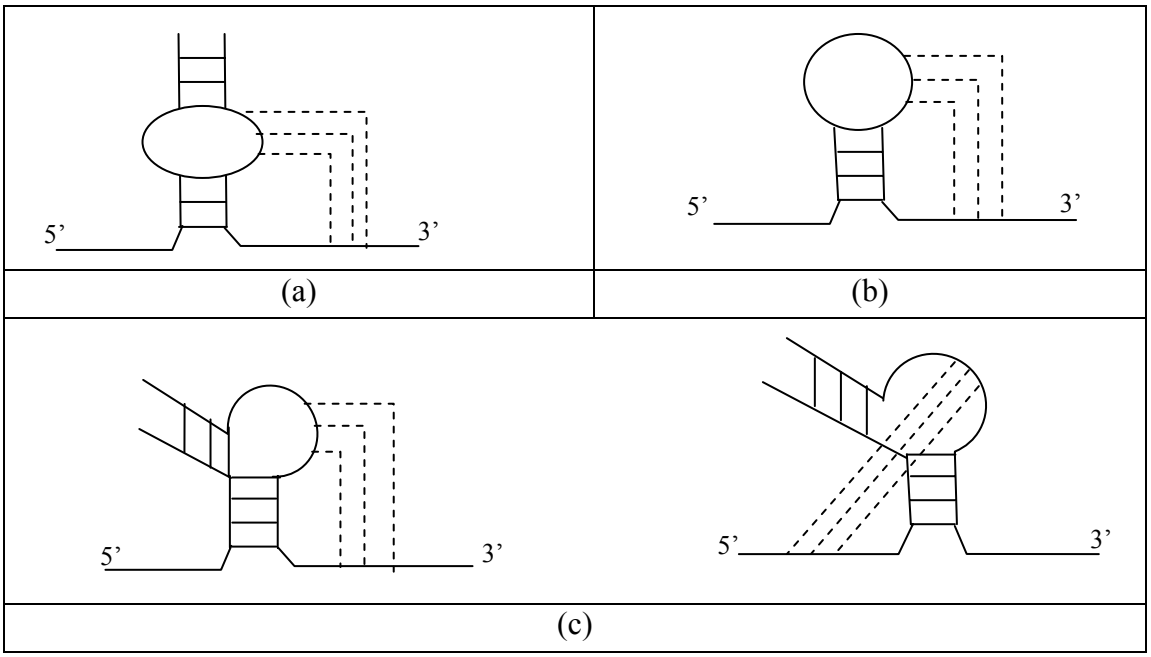


Figure 2.2. Types of pseudoknots (Chastain and Tinoco Jr., 1991). (a) I-type pseudoknot. (b) H-type pseudoknot. (c) B-type pseudoknot.

From the above definition of pseudoknots, we can see that the base pairing in pseudoknots is not well nested. Since standard dynamic programming algorithms use a recursive scoring system to identify paired stems, non-nested base pairs can not be

detected. It has been proven that pseudoknot prediction problem is NP-complete using minimum free energy model (Lyngsø and Pedersen, 2000).

2.1.2 Term Rewriting

Term rewriting is a logical formalism (Baader and Nipkow, 1999; MARTI-OLIET and Meseguer, 2002). It consists of two key ideas: states and the behavior of a system. The states of a system are represented as elements of an algebraic data type; and the behavior of a system is described by local transitions between states according to a predetermined set of rewrite rules. A rewrite rule (also called rewrite law) declares the relationship between the states and the transitions between them.

Term rewriting has a long history in theoretical Computer Science (Baader and Nipkow, 1999; MARTI-OLIET and Meseguer, 2002; Robinson and Voronkov, 2001). Recently it found a place in bioinformatics applications (Eker et al., 2002, Talcott et al., 2004) as well. In Carolyn et al., work (Talcott et al., 2004), a model called pathway logic was built to simulate the overall state of proteins and protein functional domains and their interactions by using a rewriting tool Maude (Clavel et al., 2003).

2.1.3 Maude Term Rewriting System

Maude, as a term rewriting language, supports both equational and rewriting logic computation for a wide range of applications with high performance (Clavel et al., 2003). Its high performance is achieved by compiling the rewrite rules into efficient matching and replacement automata (Eker, 1996). The advantage of this technique is to make it possible to trace every single rewriting step in Maude system. Besides high performance,

the naturalness and simplicity of Maude for concurrent computation perfectly match the requirements of our application where the computation of concurrent interaction of RNA structure motifs is required.

The following is a brief introduction to the reserved words for the definition of terms, equations and rewrite laws in the Maude language.

- *sort*: sort can be considered as a type of collection. *subsort* term can be used to indicate a belonging relationship between sorts.
- *op*: is used to define an operator. It enables the user-definable syntax in Maude. Operator declarations may include attributes that provide additional information about the operator, like *associativity*, *commutativity* et al.
- *eq:* stands for equation. It demonstrates a bidirectional equivalent relationship between two sorts. Equation can be used to deploy reduction and conversion in rewriting logic language by defined rules.
- *ceq:* is the shortage of conditional equation. It declares a conditional equation.
- *rl:* is used to define a rewrite law. The rewrite laws are different from equations in that rewrite laws define the concurrent features while equations are used to define the non-concurrent features of the language. Equations are two-directional but rewrite laws are irreversible. One state can transition back to the previous state but any particular rewrite rule cannot go backwards.
- *crl:* stands for conditional rewrite law. It is used for definition of conditional rewrite laws.

In this work, we develop a new implementation using the Maude term rewriting system to study the relationship between RNA secondary structures and RNA pseudoknots. The detailed definitions of terms, equations, and rewrite rules in the implementation are introduced in the following sections.

2.3 Method

Our method has four steps (see Figure 2.3). In the first step, the RNA secondary structure is predicted from RNA sequence. Step 2 parses the secondary structure using term rewriting to retrieve motifs. Step 3 performs motif-motif interactions by certain rules and a score function is applied to evaluate each motif-motif interaction. Step 4 outputs the predicted structure.

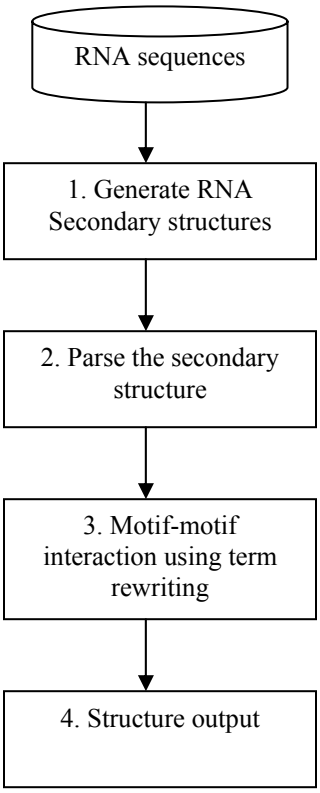


Figure 2.3. Term-rewriting based method for pattern analysis of RNA structures.

Here we explain our model step by step by giving an example of Viral 3'-UTR RNA pseudoknot (PKB116) (Batenburg et al., 2000). Figure 2.4 shows the procedure of pattern analysis. In Figure 2.4, the left part is our method and the right part is the results corresponding to each step of the method.

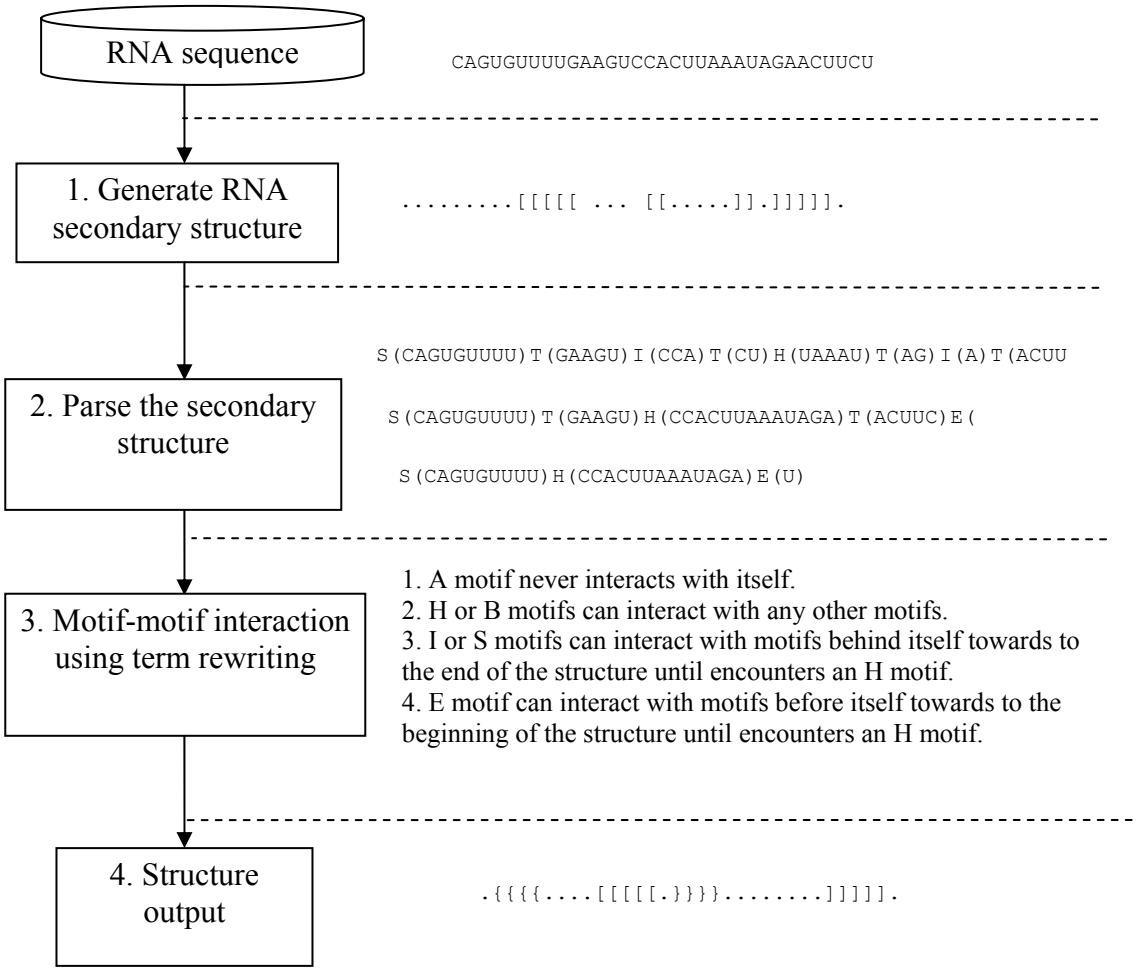


Figure 2.4. Example of pattern analysis of RNA structures.

Step 1:

The step 1 generates pseudoknot-free secondary structure from an RNA sequence. The secondary structure of RNA molecules can be predicted by calculating the minimum free energies structure for all possible combinations of base pairs. There are many tools for RNA secondary structure prediction, in which Mfold (Zuker, 2003) and RNAfold (Hofacker et al., 1994; Hofacker, 2003) package are most widely used in literature. In our practice, Mfold is used with default parameters. The output secondary structure of step 1 is a dot-bracket string in which corresponding brackets stand for base pairs of nucleic bases.

```
.........[[[[[ ... [[.....]].]]]]].
```

Step 2:

Step 2 retrieves secondary structural motifs (see Figure 2.1) from the dot-bracket string. Here, multi-branched loop is treated as independent internal loops. The motifs in the example of Figure 2.4 are as follows:

```
S(CAGUGUUUU)T(GAAGU)I(CCA)T(CU)H(UAAAU)T(AG)I(A)T(ACUUC)E(U)
```

Where S(CAGUGUUUU) is a start sequence; T(GAAGU), T(CU), T(AG), and T(ACUUC) are stems; I(CCA) and I(A) are internal loops; H(UAAAU) is a hairpin loop; E(U) is an external sequence.

Additional modifications on the stems are necessary for pseudoknot prediction because nucleic bases in a stem may be involved in the pseudoknot establishing. Hence, the base pairs in a stem whose length is less than a predefined value will be separated.

After separating certain stems, motifs need to be parsed again. It is noticeable that the bases pairs in stems T(CU) and T(AG) are separated. Now the dot-bracket string is:

```
.........[[[[[.............]]]]].
```

Parsing this string, we get motifs:

```
S(CAGUGUUUU)T(GAAGU)H(CCACUUAAAUAGA)T(ACUUC)E(U)
```

Definition:

- ‘ and ’: to facilitate retrieving motifs from the dot-bracket string, we add ‘ and ’ symbols into the dot-bracket string to label the beginning and ending of the string.
- *sorts:* sorts defined in our model.(see Figure 2.5, Figure 2.6)

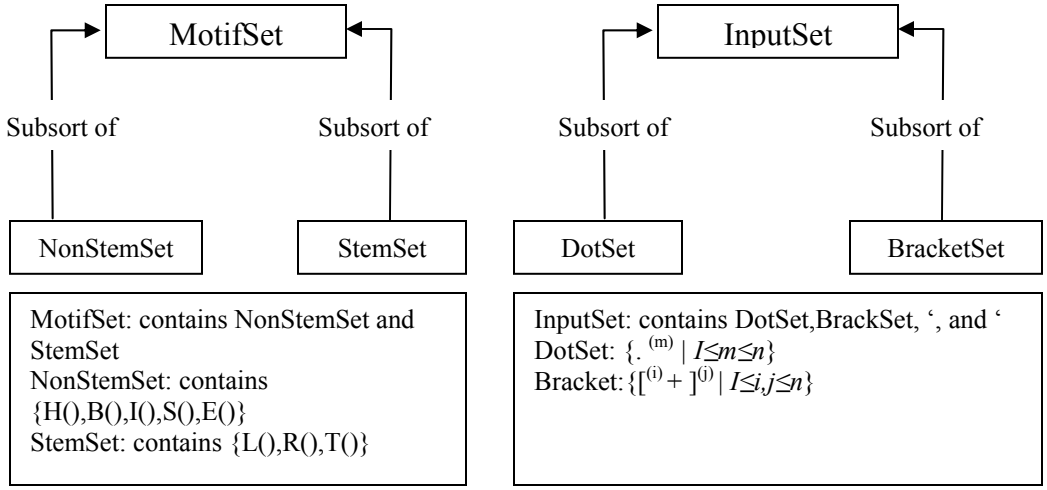


Figure 2.5. Definition of sorts used in the Step 2.

In this step, motifs are retrieved as follows:

- Find start motif and external motif, i.e. S(), E().

The S motif has a pattern that it must begin with a start symbol (‘), followed by one or more dot (.) and ended with a left bracket ([ ). This pattern can be outlined by the following Maude code:

```
op .:→DotSet .
ops [ ]:→BracketSet .
op __:DotSet DotSet→DetSet .
op ‘_[:DotSet→MotifSet .
vars D D1:DotSet .
eq ‘D[=S(D)[ .
```

The pattern of E motif is that it must begin with a right bracket ( ]), followed by one or more dot (.) and ended with a end symbol (’). It can be deployed in Maude as:

```
op ]_’:DotSet→MotifSet .
eq ]D’=]E(D) .
```

- Find hairpin loop motif, i.e. H().

The hairpin loop motif begins with a left bracket

([ ), followed by one or more dot (.) and ended with a right bracket ( ]).

```
op [_]:DotSet→MotifSet .
eq [D]=[H(D)] .
```

- Find bulge motif and internal loop motif, i.e. B(),I().

These two motifs have something in common. They must contain a part having a pattern as either $[.^{(m)}[$ or $].^{(m)}]$ (excluding multi-branched case, which we will discuss below), where $m \geq 1$. For a bulge loop motif, a right scans of $[.^{(m)}[$ part will encounter the first

right bracket immediately followed by another right bracket, whereas an internal loop motif will see the first right bracket directly followed by something but a right bracket symbol. The difference distinguishes a bulge motif from an internal loop motif. The same thing holds for a] . $^{(m)}$] part if some direction changes are taken.

Hence, the Maude code for bulge motif is:

```
var M M1:MotifSet .
op [_[_]]:DotSet MotifSet→MotifSet .
eq [D[M]]=L([)B(D)L([)MR(])R(]) .
```

And the code for internal loop motif is:

```
op [_[_]_:DotSet MotifSet DotSet→MotifSet .
eq [D[M]D1=[I(DL([)MR(])D1 .
op [_[_]_:DotSet MotifSet MotifSet→MotifSet .
eq [D[M]M1=[I(DL([)MR(])M1 .
```

The code for] . $^{(m)}$] part is skipped here. An internal loop motif in multi-branched introduce another pattern which is:] .$^{(m)}$ [. It can be recognized by the following code:

```
op _]_[_:MotifSet DotSet MotifSet→MotifSet .
eq M]D[M1=M]I(D)[M1 .
```

- Other reduction steps for the parsing

Besides parsing individual motif, more operators and equations are necessary to reduce brackets such that the dot-bracket string can be correctly parsed into motifs we need.

(1) Bracket reduction

Nested continuous pairs of brackets can be described in a pattern like [[MotifSet]]. The inner pair of brackets will not have any impact on the motif determination. Thus, they can be reduced.

```
30 op [[_]]:MotifSet →MotifSet .
31 eq [[M]] =[L()MR()] .
```

Similarly, the brackets in a pattern like '[MotifSet]'can be reduced as follows:

```
32 op '[_] ':MotifSet →MotifSet .
33 eq '[M] ' ='L()MR()' .
```

(2) Stem Motif concatenation

Nested continuous pairs of stem motifs can be further concatenated if they fall into certain pattern like L()L()MotifSetR()R(). The Maude code is:

```
34  op  _  _  _  _  _:MotifSet  MotifSet  MotifSet  MotifSet
    MotifSet → MotifSet .
35 eq L(loop)L(loop1)MR(loop2)R(loop3)=
        L(looploop1)MR(loop2loop3) .
36  op _ _ :MotifSet MotifSet→MotifSet (commu) .
```

Finally, convert all L() and R() motifs into T() motifs using the following code:

```
37  var t : StemSet .
38  eq t( )M = T()M .
```

The code from line1 to line28 parses the dot-bracket string into motifs. The code for re-

parsing the dot-bracket string after separating stems is skipped here. Finally we get the motifs used in next step by removing all stems:

```
S(CAGUGUUUU)H(CCACUUAAAUAGA)E(U)
```

Step 3:

This Step performs *permissible* motif-motif interaction based on our rules. A score function is applied to evaluate each motif-motif interaction.

- ***Permissible* motif-motif interactions**

We enforce motif-motif interactions by the following rules;

(a) A motif never interacts with itself.

(b) Each H or B motif can interact with all other motifs.

(c) Each I or S motif can interact with the motifs behind itself towards the end of the structure until encounters an H motif.

(d) The E motif can interact with the motifs before itself towards the beginning of the structure until encounters an H motif.

- **Score function**

Each motif-motif interaction is scored by taking the weight of base pair region and the distance penalty of the base pair region into consideration. The weights in the score function are chosen to reflect the physical chemistry of nucleic acids (Bloomfield et al., 1999). A base pair region is defined as:

$$region = \{(i, j), (i-1, j+1), \ldots (i-m, j+m)\} \tag{3.1}$$

where $i \in motif\ 1, j \in motif\ 2, i < j, m = length\ (region)$ and each base pair in this region belongs to the set of $\{CG, GC, AU, UA, GU, UG\}$.

The weight of *region* is defined as:

$$W_{region} = \sum_{(i,\ j) \in region} weight(i,\ \ j) \tag{3.2}$$

Where weight(CG/GC):weight(AU/UA):weight(GU/UG) =3:2:1. The weights are approximate values that indicate the trends in base pair energy.

The distance penalty of the base pair region is defined as

$$Dis_{region} = \ i\text{-}j \tag{3.3}$$

where (i, j) is the closest base pair in the region and i<j. Then, score function can be defined as:

$$Score_{motif\text{-}motif} = Max(\alpha \times W_{region} + (1-\alpha) \times Dis_{region}) \tag{3.4}$$

where α is a constant and $\alpha \in [0,1]$. The constant α is a heuristic parameter adjusting the significance of W_{region} and Dis_{region} in the score function. In our experiment, α is set to 0.8.

- ***Format of motifs***

A motif has a format like MotifType(seq:String,max:Int)*, where MotifType∈{H,I,B,S,E}; seq* is a variable storing the nucleotide sequence of a motif; *max* is a variable storing the

maximal score of each motif when it interacts with other motifs. Each motif has an initial max score 0. The motifs we will deal with are:

```
S(CAGUGUUUU,0)H(CCACUUAAAUAGA,0)E(U,0)
```

Figure 2.6 shows the sorts defined in Step3.

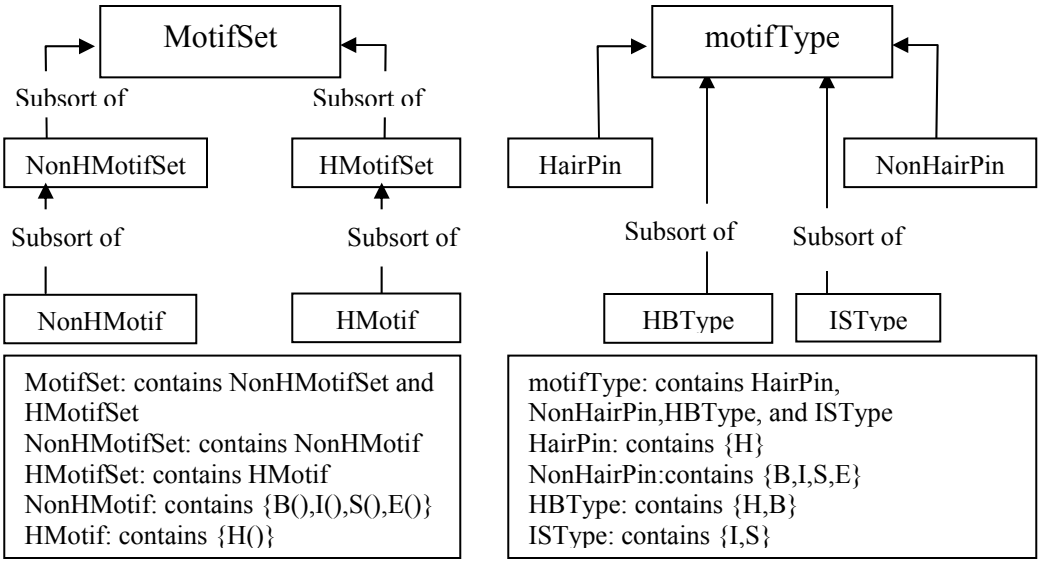


Figure 2.6. Definition of sorts used in the Step 3.

Maude code for motif-motif interaction is as follows:

***(a) Operators and variables

```
1  op _(_,_):Hairpin String Int→HMotif .

2  op _(_,_):NonHairPin String Int→NonHMotif .

3  op _ _:NonHMotifSet NonHMotif→NonHMotifSet .

4  op _ _: MotifSet MotifSet→MotifSet .

5  op _ _ _:Motif MotifSet Motif→MotifSet .

6  op _ _ _:NonHMotif NonHMotifSet NonHMotif→NonHMotifSet .
```

Line1 defines the operator to recognize hairpin loop motifs. Line2 defines the operator to recognize non-hairpin loop motifs. Line3 to line6 define the operators which deal with multiple continuous motifs.

```
7  vars seq seq1:String .          ***sequence of a motif

8  vars max max1:Int .             *** maximal score

9  vars mSet:MotifSet NonHSet:NonHMotifSet .

10 var M:MotifType .               ***M is H,I,S,E, or B

11 var Z:NonHairpin .              ***Z is I,S,E, or B

12 var X:HBtype .                  ***X is H or B

13 var Y:IStype .                  ***Y is I or S
```

Line7 to line13 define variables used by line14 - 21.

***(b) H and B motifs

```
14 ceq X(seq,max)M(seq1,max1)=
     X(seq,MAX(seq,seq1))M(seq1,max1)if MAX(seq,seq1)>max .


15 ceq X(seq,max)mSetM(seq1,max1)=
X(seq,MAX(seq,seq1))mSetM(seq1,max1) if MAX(seq,seq1)>max .
```

Line14 and line15 let the X motif interact with the motifs towards to the end of the structure.

```
16 ceq M(seq1,max1)X(seq,max)=
    M(seq1,max1)X(seq,MAX(seq,seq1)) if MAX(seq,seq1)>max .


17 ceq M(seq1,max1)mSetX(seq,max)=
 M(seq1,max1)mSetX(seq,MAX(seq,seq1))if MAX(seq,seq1)>max .
```

Line16 and line17 let the X motif interact with the motifs towards to the beginning of the structure.

***(c) I and S motifs

```
18  ceq Y(seq,max)Z(seq1,max1)=
    Y(seq,MAX(seq,seq1))Z(seq1,max1) if MAX(seq,seq1)>max .


19  ceq Y(seq,max)NonHSetZ(seq1,max1)=
    Y(seq,MAX(seq,seq1)) NonHSet Z(seq1,max1)
    if MAX(seq,seq1)>max .
```

Line18 and line19 let the Y motif interact with the motifs towards to the end of the structure.

***(d) E motif

```
20 ceq Z(seq1,max1)E(seq,max)=
 Z(seq1,max1)E(seq,MAX(seq,seq1)) if MAX(seq,seq1)>max .


21 ceq Z(seq1,max1)NonHSetE(seq,max)=
 Z(seq1,max1)NonHSetE(seq,MAX(seq,seq1))
 if MAX(seq,seq1)>max .
```

Line20 and line21 let the Y motif interact with the motifs towards to the beginning of the structure.

In the above code, MAX(seq:String, seq1:String) is a module which calculates the score of each motif-motif interaction by implementing the score function. seq and seq1 are variables storing the nucleotide sequences of two motifs. Maude provides convenient string processing operators such as *find*, *substr* et al. It is easy to implement the MAX module in Maude.

Finally, the global maximal score of all motif-motif interactions is found. The potential pseudoknot is located between the two motifs with the global maximal score.

Step 4:

The motifs with the highest score are considered as the candidates forming the potential Pesudoknot. The stems separated in step 2 may or may not be recovered before outputting the final structure. The effect of this recovery operation will be discussed in the data analysis section. The pseudoknot in the final output structure is labeled with '{' and '}'

```
.{{{{....[[[[[.}}}}........]]]]].
```

Figure 2.7 shows three structures of the example. (A) is the secondary structure predicted by Mfold, (B) is our prediction, and (C) the experimental structure. Comparing our prediction with the experimental structure, the accuracy we get is 93.94%.

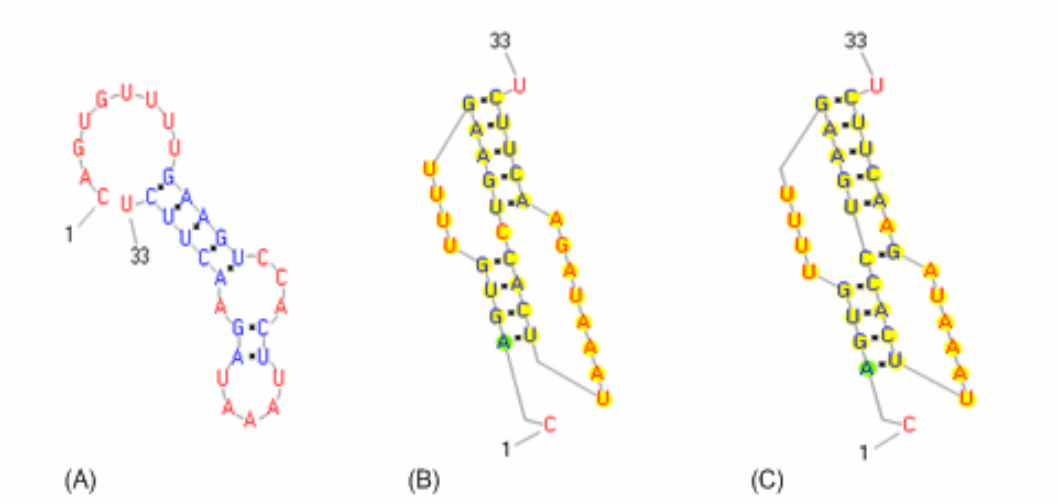


Figure 2.7. Final output results. (A) Mfold predicted structure (B) Our predicted structure. (C) Experimental structure.

2.4 Experiment results

2.4.1 Evaluation Criteria

For each nucleic base a_i in the sequence, assume a_j and a_k are the partners of a_i in the predicted structure and reference structure respectively with $0 \le j, k \le n$ and $1 \le i \le n$. If the partner of a_i is a_0, a_i is unpaired. For a_i, our prediction has two possible results:

(1) Correct if $j == k$ (2) Wrong if $j \ne k$

Accuracy is defined as follows:

$$Accuracy = 100\% \times \frac{\#Correct}{n} \tag{3.5}$$

2.4.2 Data Analysis

Our method is tested on 211 single-strand pseudoknots in PseudoBase (Batenburg et al., 2000) with length varies from 21 to 137. These pseudoknots are classified into 13 classes by PseudoBase according to Tabaska's work (Tabaska et al., 1998). The classification is shown in Table 2.1.

Table 2.1. Data in PseudoBase.

Classification	#RNA	Length
1. Viral ribosomal frameshifting signals	15	39-73
2. Viral ribosomal readthrough signals	6	61-63
3. Viral tRNA like structures	54	37-137
4. Other viral 5'-UTR	1	35
5. Other viral 3'-UTR	80	21-96
6. Viral others	20	24-77
7. rRNA	3	46-51
8. mRNA	7	28-120
9. tmRNA	10	30-90
10. Ribozymes	3	73-89
11. Aptamaers	6	33-57
12. Artificial molecules	1	26
13. Others	4	35-121

In Table 2.1, the first column is the names of 13 classes of RNA pseudoknots in PseudoBase (Batenburg et al., 2000). The second column is the number of RNA pseudoknots in each class. The third column is the length range of RNA pseudoknots in each class. From Table 2.1, We can see that the number of RNA pseudoknots in each class of 1, 3, 5, 6, and 9 is no less than 10. Since these classes cover most of the data, more attention should be paid to these classes when analyzing the accuracy of the results.

During the procedure of pattern analysis of pseudoknots, the bases of the stems are separated according to the length of stems in Step 2 (See Figure 2.4). To specify the effect of stems on pseudoknots, we test the 211 pseudoknots by adjusting two parameters

in our method*: stem-length(L)* and *recovery*. If a stem with a length less than L, the base pairs in the stem will be separated. When *recovery* is 'yes', any stem separated in step 2 will be recovered if no base in this stem is involved in pseudoknot. The experimental results on the effect of stem-length and recovery are shown in Table 2.2.

Table 2.2. Effect of stem-length and recovery

Case	Stem-length(L)	Recovery	Average Accuracy (%)
(a)	3	yes	72.412
(b)	3	no	72.707
(c)	4	yes	73.277
(d)	4	no	74.085
(e)	N/A	N/A	70.878

From the Table 2.2, we can see that each case (a)-(d) has higher accuracy than case (e) in which no stem is separated. This indicates that small stems have effect on pseudoknots. Comparing case (a) with (b) and case (c) with (d), we can see that simply recovering stems based on the secondary structure does not contribute to the pseudoknot prediction. The highest accuracy is obtained in (d). Other combinations of the two parameters are tested but cannot result in as good prediction as that when L is 3 or 4 and *recovery* is 'no'.

Testing the 211 pseudoknots using parameters that *stem-length* is 4 and *recovery* is 'no', our model achieves an average accuracy of 74.085%. 36 pseudoknots reach 100% accuracy. Only six pseudoknots have accuracy lower than 30%. The Figure 2.8 shows the accuracy distribution.

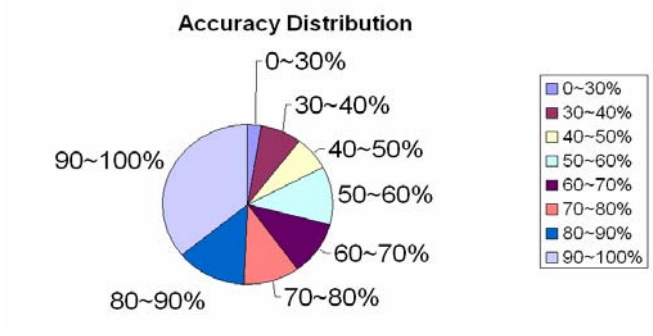


Figure 2.8. Accuracy Distribution.

We compare our method with the algorithm introduced in (Ruan et al., 2004). Their method was implemented in their web server (http://cic.cs.wustl.edu/RNA/) which supports thermodynamic and comparative analysis for prediction of RNA secondary structure with pseudoknots. We tested the 211 pseudoknots on this web server by using their default parameters. The result is described in Table 2.3.

In Table 2.3, there are 211 pseudoknots in 13 classes. The accuracies of class 1, 3, 5, 6, and 9 are more indicative than other classes because they cover most of the data in the *PseudoBase*. In Table 2.3, the first column is the classification of pseudoknots in PesudoBase. The second column is the number of pseudoknots in each class. The third column is the accuracy of different methods, including Mfold (Zuker, 2003), our method (Fu et al., 2005), and Ruan's method (Ruan et al., 2004). According to accuracy described 2.4.1, the accuracy of each method on 211 pseudoknots are demonstrated in Table 2.3.

Table 2.3. Comparison of accuracy

Classification	#RNA	Accuracy (%)		
		Mfold (Zuker, 2003)	**Our method**	**Ruan's server** (Ruan et al., 2004)
1	15	54.983	63.136	56.317
2	6	31.388	38.1	51.89
3	54	59.459	63.745	53.014
4	1	71.429	100	37.143
5	80	66.94	88.257	31.822
6	20	65.745	77.13	36.552
7	3	51.161	55.843	36.676
8	7	51.362	60.04	40.66
9	10	56.316	61.849	36.356
10	3	45.02	57.94	36.94
11	6	70.976	79.443	40.029
12	1	79.923	100	38.462
13	4	65.715	70.036	46.248
Average accuracy (%)		**61.6107**	**74.085**	**41.264**

From Table 2.3, we can see that our method obtains much higher average accuracy than both Mfold (Zuker, 2003) and Ruan's server (Ruan et al., 2004). Mfold is specially designed to predict RNA pseudoknot-free secondary structures, and the base pair accuracy is therefore an example of a random expectation.

2.5. Conclusion

Exploring the rules and patterns of RNA structures can be treated as a problem in logic programming under constraint of limited knowledge. We have shown how to make the implementation using Maude rewriting langue. Our method was tested on 211 pseudoknots in PseudoBase and achieves an average accuracy of 74.085% compared to the experimentally determined structure. A paper (Fu et al., 2005) is published based on this work. The experimental results show that Maude is effective for building and analyzing a complex biological system, defining new data and rules, and executing reduction and queries using logical derivation. The combination of simple rules and rigorous logical derivation appears to be a powerful tool for predicting complex structures.

Chapter 3

Improved K-means Algorithm

Clustering refers to the task of partitioning the similar samples of a dataset into meaningful groups (clusters). It is a useful tool in data mining processes for revealing hidden patterns and underlying knowledge from a dataset which is poorly understood. Clustering algorithms have been widely used in various fields, such as information retrieval, text classification, image segmentation, and bioinformatics.

The purpose of our work is to explore the structure patterns of both RNA and protein structures using K-means clustering algorithm. Our work centers on finding solutions to the following hard problems:

- How to initialize the clustering algorithms;
- How to determine the suitable number of clusters for a specific dataset;
- How to speed up K-means;

The chapter is arranged as follows. In section 3.1, the traditional K-means algorithm and its drawbacks are introduced. Section 3.2 gives the definition of cluster similarity which is used for finding the most reliable clustering for a dataset. The pseudocode of the improved K-means clustering algorithm is given in section 3.3. The concept of bootstrap sampling, cluster similarity and reliable clustering are explained in detail. Section 3.4 introduces the random number generator used in our implementation. In section 3.5, we discuss how to initialize the K-means algorithm. Section 3.6 gives the definition on how to speed up K-means by triangle inequality. To verify the effectiveness of our algorithm and show how to use it, an experiment on synchronized data is done in

section 3.7. The discussion and conclusion are given in section 3.8.

3.1 K-means Clustering Algorithm

K-means (MacQueen, 1967) is a widely used unsupervised learning algorithm that solves the clustering problem. Given a dataset, the traditional K-means algorithm works as follows. It begins with defining K centroids, at random or using some heuristic data, as the initial centers of K clusters, one for each cluster. The second step is to assign each point of the dataset to the cluster with the nearest centroid. The third step is to take each point in the dataset and compute its distance from the centroids of al clusters. If a point is not currently in the cluster with the closest centroid, switch this point to that cluster and update the centroid of the cluster. The third step is repeated until convergence is achieved, that is no point assignment occurs.

K-means algorithm is a very popular clustering method widely used in scientific and industrial applications (Jain et al., 1999; Xu and Wunsch, 2005). However, its robustness is heavily affected by the initial number of clusters K, and in general, there is no reliable algorithm for predicting K. A variety of methods have been proposed on how to initialize the K or on how to let the clustering depend less on the initial K. All these efforts try to discover the suitable number of clusters for a dataset. Since stability has been widely used as a validation measure and proved to have good performance (Ben-Hur et al., 2002; Bryan et al., 2004; Dudoit and Fridlyand, 2002), in this work we investigate the stability-based measure formulated by Ben-hur et al., (Ben-Hur et al., 2002) for reliable clustering of protein structures. We proposed a new descriptor for representation of the protein structures and compared it with the descriptor used by Chen

et al., (Chen et al., 2006).

3.2 Cluster Similarity

Cluster similarity is the estimation of similarity between any two clustering solutions on the same dataset. It is given by

Let X is the dataset to be clustered, $X=\{x_1, x_2,\ldots, x_N\}$ where x_i is R^d ,N is the number of samples in the dataset. X is clustered into k clusters. We use an NxN matrix C to indicate whether any two samples in X are in the same cluster. If x_i and x_j are in the same cluster and i is not equal to j, C_{ij} is 1. Otherwise C_{ij} is 0.

For two datasets P and Q, their matrix representations of clustering solutions are respectively C^P and C^Q . The number of common joined pairs in clustering solutions of P and Q can be described the following dot product

$$\mathrm{r} = < \mathrm{C}^{\mathrm{P}}, \mathrm{C}^{\mathrm{Q}} > = \sum_{i,j} C_{i,j}^{P} C_{i,j}^{Q} \tag{3.1}$$

The cluster similarity proposed by Fowlkes and Mallows (Fowlkes and Mallows, 1983) is then defined as

$$S(P,Q) = \frac{r}{\sqrt{|C^P| \times |C^Q|}} \tag{3.2}$$

where $|C^P| = < C^P, C^P >$.

According to Cauchy-Schwartz inequality: $< C^P, C^Q > \leq \sqrt{< C^P, C^P > \times < C^Q, C^Q >}$, the similarity is between 0 and 1. 1 represents that two clustering solutions are identical. Equation (3.2) is the cluster similarity used in this experiment.

3.3 Detecting the Number of Clusters

Our method of determining the suitable number of clusters in a dataset relies on the stability of clustering. The meaning behind of the stability is that a meaningful cluster should appear on all bootstrap samplings of the data. In bootstrap sampling data are drawn from the dataset with replacement thus preserving the underlying distribution. The introduction to bootstrap sampling is as follows.

3.3.1 Bootstrap Sampling

The bootstrap is a resampling method for statistical inference. It is generally used to estimate confidence intervals, but it can also be used to estimate bias and variance of an estimator. The literature on the bootstrap is extensive. Major survey papers on the bootstrap and its applications can be found in (DiCiccio and Romano, 1988; Young et al., 1994; DiCiccio and Efron, 1996; Davison et al., 1997). Bootstrap methods are based on two main assumptions. First, the sample is a valid representative of the population. Second, bootstrap method will take sampling with replacement from the original sample. Each sub-sampling is independent and the subsamples come from the same distribution of the population. To determine the number of clusters in data, in this dissertation we decide to use the concept of bootstrap sampling in clustering analysis.

The most significant strength of bootstrap sampling is that it needs minimum assumptions. It is most useful when the sample distribution is unknown. But several problems must be carefully considered in this procedure, such as the random number generator and initialization of the clustering algorithm. We will explain how to solve these problems in the following sections.

3.3.2 Algorithm of Detecting Stable Clustering

Estimating the number of clusters in data is an ill-posed problem in cluster analysis (Jain and Dubes, 1988) because the definition of what a cluster is is unclear. Various algorithms have been proposed and can be roughly categorized into two groups: the approaches based on the dispersion of clusters (Tibshirani et al., 2001; Maulik and Bandyopadhyay, 2002; BelMufti et al., 2005) and approaches based on the concept of stability of the clustering (Levine and Domany, 2001; Monti et al., 2003; Law and Jain, 2003; Lange et al., 2004).

The approaches based on the dispersion of clusters reply on validation measures to decide the stable clustering. The widely used validation measures include compactness, isolation, and within-cluster and between-cluster dispersion. However, each validation measure favors certain shape of clusters. If clusters are in very different shapes, the validation measure is not that useful. So to use these kinds of methods to determine the suitable number of clusters, we have to assume the shape of the clusters. On the contrary, the approaches based on stability of clustering do not imply any assumption on the shape or size of clusters. By considering the data we have to deal with, especially the RNA tertiary structures, we have little information available. Approaches based on stability of clustering are more appropriate for our problems. The algorithm of determining the suitable number of clusters is outlined in Table 3.1.

As it is shown in Table 3.1, the algorithm requires two parameters Kmax and Smax. Kmax is the user-defined maximum number of clusters and Smax is the user-defined maximum number of bootstrapping. Kmax gives a range where the reliable

clustering possibly exit. Smax defines the number of subsets of the original dataset used for bootstrapping.

Table 3.1. Algorithm of detecting stable clustering

Input:

Kmax{user-defined maximum number of clusters},

Smax{user-defined maximum number of bootstrap sampling}

Output: Clustering results with stable K

Requirements: K-means algorithm, cluster similarity measure

```
1. Generate the average cluster similarity for each k
    For k=2 to Kmax do
       For s=1 to Smax do
              1.1 Bootstrap two subsets from the original dataset;
              1.2 Cluster points in each subset into k clusters;
              1.3 Compute cluster similarity of two subsets;
         End For
         1.4 Compute the average cluster similarity for each k;
    End For
2. Determine the suitable number of clusters
         2.1 The maximum average similarity occurs at k1;
         2.2 The second maximum average similarity occurs at k2;
         2.3 Detect the stable k
            If isStable(k1) is true, K=k1
            Elseif isStable(k2) is true, K=k2
            Else K=0,return
3. Cluster the points in original dataset into K clusters.
```

The algorithm includes three stages. The first two stages are responsible to

generate the distribution of average cluster similarity of each k. Bootstrap sampling, clustering of each subset using K-means algorithm, and cluster similarity evaluation are done during the two stages. The K which gives the most stable clustering is detected from the distribution of average cluster similarity. In the third stage, the original dataset is clustering into K clusters.

The explanation of each stage is given as follows:

Stage 1:

The stage is to generate the average cluster similarity for each k (k is from 2 to Kmax). The step 1.1 is to obtain two subsets from the original dataset by bootstrap sampling which is introduced in section 3.3.1. With the subsets available, K-means algorithm is applied onto the subsets in step 1.2. In step 1.3, the similarity between two clustering results of subsets is calculated. These first three steps are repeated Smax times. The average of the similarities is computed as the stability of the clustering under k in step 1.4. The distribution of average similarities is further used to determine the reliable clustering in the stage 3.

Stage 2:

Stage 2 clarifies the criteria of determining the reliable clustering and the reliable k for the dataset. This stage consists of three steps. Since a meaningful cluster should appear on all bootstrap samplings of the data, the most reliable clustering should results in the maximum average cluster similarity. So the maximum and the second maximum average cluster similarities are found in step 2.1 and 2.2. The third step is to identify the reliable k which is shown in the table. In step 2.3, `isStable(k)` is a function to judge whether

the clustering at k is reliable. Given the k, if the distribution of average cluster similarities before k is in increasing trend while the distribution after k is in decreasing trend, `isStable(k)` is true. If the return value is 0, it indicates that there is no structure in the dataset and the algorithm returns. The cluster similarity mentioned in the Table 3.1 is defined in section 3.2. If the K is not 0 after stage 2, the algorithm precedes to stage 3.

Stage 3:

From the stage 2, we find the K which is believed to yield the most reliable clustering in the range of k=2 to K_{max}. In this stage, all the samples in the original dataset are clustered using K-means algorithm with the stable K, and the clustering results are returned.

3.4 Random Number Generator

In bootstrap sampling, data points are randomly drawn with replacement from the original dataset. The quality of randomness directly affects the clustering results and further the determination of suitable number of clusters. So a good random number generator is required for bootstrap sampling.

There are two major methods of random number generation. One obtains the random numbers by measuring some physical phenomena which is expected to be random such as sound samples taken in a noisy environment and radioactive decay. The other is called pseudo-random number generators which use computational algorithms to produce long sequences of random results.

Usually, library routines of random number generators can be found in most of the computer programming languages. However, such library functions often have poor

statistical properties. They are often initialized using a computer's real time clock as the seed or key. These functions are unsuitable for statistics or numerical analysis when high-quality randomness is required. There is a tradeoff between the reliability and speed when designing random generators. The more reliable the generator is, the more slowly it runs. By taking these two factors into consideration, we choose the Mersenne Twister random generator (MTRG) (Matsumoto and Nishimura et al., 1998; Matsumoto and Kurita, 1992; Knuth, 1992; Eddelbuettel, 2007) which has been proved to be superior to most of random number generator algorithms. It has an astronomical period of $2^{19937} - 1$ while no other generator has achieved this. Its output is free of long-term correlations when considered from a viewpoint of 623 dimensions. It is essentially a large linear-feedback shift register, and thus the output has excellent statistical properties. There are a number of implementations of MTRG in different programming languages like C, C++, java et al. It facilitates the implementation of bootstrap sampling.

3.5 Initialization of K-means

The initialization of K-means is to set k cluster centroids for a dataset. It has been reported that results obtained from the K-means are dependent on the initialization of the cluster centroids (Pena et al., 1999).

Two simple approaches to initialize K-means can be seen in various applications. One is to set the initial centroids randomly. The other is to choose the first k samples of the dataset. Since the two methods do not make use of any information about the data, it is highly possible for them to choose inappropriate initial centroids such that the lead to results far from suitable. An alternative method is to select multiple sets of initial

centroids first, then to choose the set which is closet to the suitable. However, testing multiple initial sets is sometime impractical, especially for large dataset.

In this work, we initialize the K-means using an algorithm which tries to make the initial centroids separated as far as possible. Thus the coverage of these centroids is extended as large as possible.

The method works as follows. Let X is the dataset with size NxM. N is the number of samples and M the number of attributes for each sample. Let K be the number of clusters.

(1) Select a sample from the dataset randomly as the first initial centroid;

(2) Search for the second sample which is farthest from the first centroid and take it as the second centroid;

(3) Search the next sample which is farthest from the previous centroids;

(4) Repeat step (3) till K centroids are found.

3.6 Speed up K-means

More and more data is produced and accumulated at a fast pace with the development of new techniques. It is required that the standard K-means clustering algorithm should be as fast as possible to deal with large datasets without losing any clustering quality. In this dissertation, large datasets are used in our experiments with high dimensions. So it is necessary to speed up the standard K-means algorithm. K-means algorithm is an iterative procedure. At each iteration, a lot of calculations are needed to assign every data point to the cluster whose centroid is nearest to the data point. However, according to triangle inequality, many calculations are redundant. Based

on the triangle inequality, we give the following definition:

Let x be the data point and let c1 and c2 be the two different centroids. If the distance between x and c1 is no greater than half of the distance between c1 and c2, then x is closer to c1 than to c2.

From the above the definition, if the distance between x and c1 is no greater than half of the distance between c1 and c2, it is unnecessary to compute the distance between x and c2. Otherwise, the calculation has to be done.

The proof is given as follows:

To prove that if $d(x,c1) \leq \frac{1}{2} * d(c1,c2)$, then $d(x,c1) \leq d(x,c2)$.

From triangle inequality, we know

$$d(c1,c2) \leq d(x,c1) + d(x,c2) \qquad \text{(a)}$$

Then $$d(c1,c2) - d(x,c1) \leq d(x,c2) \qquad \text{(b)}$$

if $d(x,c1) \leq \frac{1}{2} * d(c1,c2)$, that is $d(c1,c2) \geq 2d(x,c1)$, then

$$d(c1,c2) - d(x,c1) \geq 2d(x,c1) - d(x,c1) = d(x,c1) \qquad \text{(c)}$$

From (b) and (c), we get $d(x,c1) \leq d(x,c2)$ (d)

3.7 Validation Using Synchronized Data

The purpose of algorithm is to discover the suitable number of clusters in a dataset. This suitable number reveals the structure of dataset and the high-quality hidden structural patterns could be further extracted based on the understanding of the dataset. To validate the effectiveness of the algorithm, we set up an experiment to test our algorithm on synchronized data. Two datasets are used in the experiment. One has no structure (only one cluster) while the other contains four clusters. By observing the

distribution of the mean similarities of two different datasets, we are supposed to detect the suitable number of clusters. This experiment is also helpful for us to see the relationship between the number of clusters (K) and the mean similarity of each clustering with number K. The discussion about the experimental results and observations is given in this section.

3.7.1 Dataset

Dataset 1:

This dataset has 200 points which are in uniform distribution. Thus no structure exists in the dataset.

Dataset 2:

This dataset also has 200 points which are in four obviously isolated clusters.

The datasets are shown in Figure 3.1.

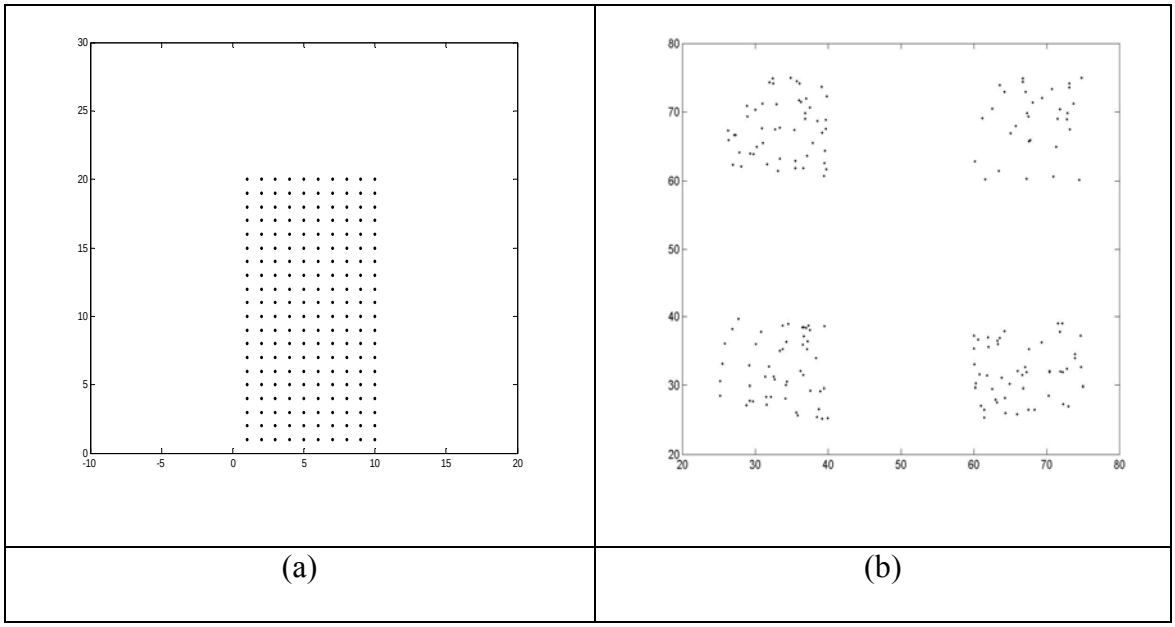


Figure 3.1. Datasets. (a) Dataset without structure. (b) Dataset with four clusters.

3.7.2 Experiment Setup

3.7.2.1 Environment

We implement the algorithm and perform the experiment on a PC with a P4-2.8MHz CPU and 512M memory in Matlab (Matlab, 1994) and C languages. We use Matlab for data preprocessing and visualization. The K-means clustering algorithm is written in C for better computational speed. The simulations in Chapter 4 and 5 are also done under the same environment. We will not explain it again in the following Chapters.

3.7.2.2 Experiment Parameters

The major parameters in this experiment are the number of clusters (K) and the number of similarities to be calculated for each clustering with number K. In this experiment, we set the number of clusters K change from 2 to 50 and 20 similarities are calculated for each clustering with number K.

3.7.2.3 The Suitable Number of Clusters

The change between adjacent average cluster similarities and the overall distribution of average cluster similarities demonstrate the stability of the clustering. As it is mentioned above, the average cluster similarity is between 0 and 1 and larger average cluster similarity value indicates higher similarity clustering with k. We assume the suitable number is K for a distribution. This suitable number should simultaneously satisfy all of the three criteria:

(1) The average cluster similarity at K is the largest value of all average cluster similarities.

(2) The average cluster similarities before K and after K follow an increasing distribution and a decreasing distribution respectively.

(3) K is unique.

The three criteria above are equally important and must be satisfied. If no average cluster similarity in a distribution meets the requirements, we can come to two conclusions. One is that it fails in discovering the structure of the dataset. The other is that there is no structure inside the dataset.

3.7.3 Experimental Results

The distribution of average cluster similarities of dataset without structure is shown in Figure 3.2.

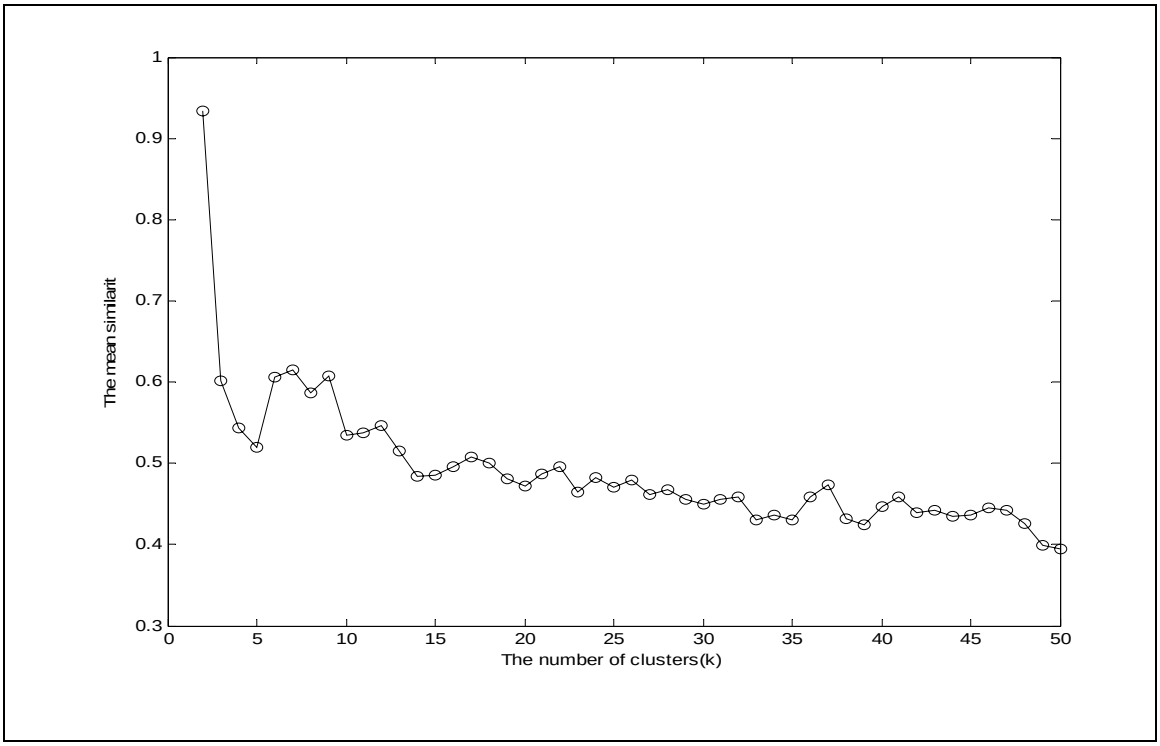


Figure 3.2. Experimental results of dataset without structure.

In Figure 3.2, the number of clusters changes from 2 to 50. The average cluster similarity of each k is marked with a circle. We can see that the maximum average cluster similarity is obtained when k is 2. Other average cluster similarities are far from the largest one. The overall trend of the distribution is that the average cluster similarities decrease with the increasing of k without big change. From the distribution, no average cluster similarity satisfies the requirements described in section 3.6.2.3. So we come to the conclusion that the dataset has no structure.

The distribution of average cluster similarities of dataset with four clusters is shown in Figure 3.3.

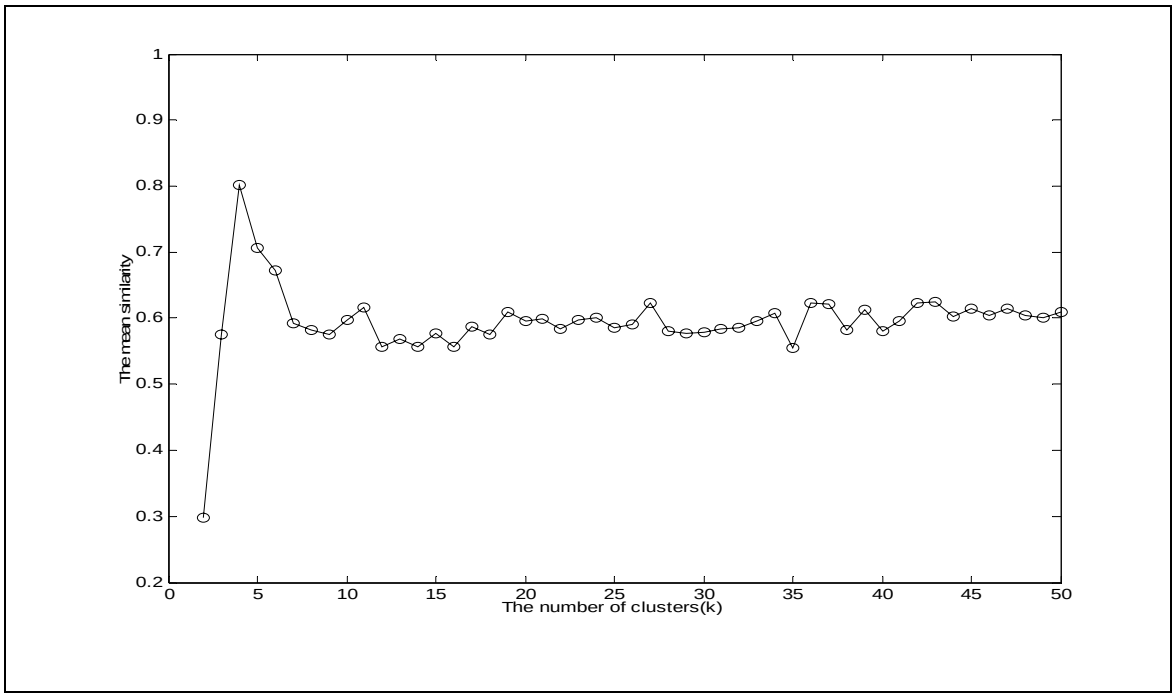


Figure 3.3. Experimental results of dataset with four clusters.

The distribution of average cluster similarities in Figure 3.3 is much different of that in Figure 3.2. It is obvious to see that the largest average cluster similarity occurs

when k is 4. The average cluster similarities before k=4 is in increasing distribution while the average cluster similarities after k=4 is in decreasing distribution with little fluctuation. What is more, the value at k=4 is much larger than others. From these observations, we can draw a conclusion that the suitable number of clusters is 4.

3.7.4 Summary

In this experiment, our algorithm is tested on two simulated datasets, one without structure and the other with four clusters. The purpose of the experiment is to see whether our algorithm is able to handle both datasets with and without structures. The experimental results show much difference which further verifies the ability of our algorithm. It not only can detect the suitable number of clusters in a dataset which indeed has a structure but also can identify that a dataset has no structure. This is the major advantage of our algorithm over traditional K-means algorithm. Another valuable advantage is its reliability of performance. Unlike model-based methods, the suitable number is determined by our algorithm is based on the distribution of average cluster similarities instead of predefined models. Therefore, it is more suitable for the data with little available knowledge.

3.8 Conclusion

In this Chapter, an improved K-means clustering algorithm is introduced for the determination of the suitable number of clusters in data using bootstrap sampling. Additionally, other major issues are also discussed, such as how to set the initial centroids of K-means, random number generator, and cluster similarity. We demonstrate how to

use the proposed algorithm to analyze the suitable number of clusters in a synchronized dataset. The experimental results show the effectiveness of our algorithm. Further applications will be made by applying our algorithm to the structure pattern analysis of RNA and protein data in Chapter 4 and 5.

Chapter 4

Pattern Analysis on RNA structures

Structure analysis of the RNA complexes is essential to the complete understanding of RNA folding and interaction, thus contributes to the development of new drugs. With the increasing number of RNA structures available, a systematic approach to automatically identify the RNA three dimensional structure patterns is required. Since the mechanism of RNA three dimensional folding is poorly understood to date and the RNA structure patterns is still under discussion (Batey et al., 1999), we decide to utilize clustering algorithms in our work. Clustering is an unsupervised learning technique, which learns the structure in data without using labeled class, feedback signal or any prior knowledge but the raw data and group principles. This is exactly what the RNA structure pattern analysis requires.

In order to cluster the three dimensional structures, we propose a new descriptor to represent three dimensional structure segments. This descriptor is tested on both RNA and protein datasets. During the pattern analysis of RNA structures, we find the HFSS and their similarity. The definition of HFSS is given in section 4.2.5. Based on these HFSS, an RNA three dimensional structure can be converted into a string of HFSS. It is well known that comparison of molecule structure similarity is a problem of searching the maximum common substructure (also called graph Isomorphism problem) which is NP-complete if the molecule structure is described as a graph. By converting a three dimensional molecule structure into a string, many problems become practically solvable, such as the comparison of molecule structure similarity, mining known and novel motifs,

rapid structure database search.

This chapter is organized as follows. Section 4.1 explains the objective of this experiment. Section 4.2 demonstrates all the experimental procedures and results of RNA pattern analysis, including large-sized and small-sized types of structure segments, the types of HFSS in the RNA dataset, and the similarities among different types HFSS. Our clustering algorithm and experimental results can be used for rapid RNA structure database search. The framework of rapid RNA structure database search is introduced in Section 4.3. The discussion and conclusions are given in section 4.4.

4.1 Objective

It is important to assess the structure similarity of RNA molecules and identify the known and novel tertiary motifs in RNA three dimensional structures. With more and more RNA structures available, efficient algorithms are highly desired for solving these problems. To date, the classification of RNA tertiary structures or motifs is still under discussion. The representation of RNA structures is the basis of mining structure patterns from them. To study the structure patterns of RNA three dimensional structures, the first task is to develop new method of appropriate representation of RNA structures. Based on the suitable representation of RNA structures, structure patterns are extracted using K-means clustering algorithm.

4.2 Experiments on RNA Structures

In this section, we demonstrate the experiments of pattern analysis using clustering algorithms on RNA three dimensional structures. The flowchart of pattern

analysis on RNA structures is firstly given in section 4.2.1. Then the dataset chosen for our experiment is explained in section 4.2.2. The details about experiment parameters and setup are introduced in section 4.2.3 and section 4.2.4. In section 4.2.5, we show how to find the appropriate window size for the description of RNA structure segments. Our experimental results prove that window size 4 is most suitable for RNA structure segments description. Section 4.2.6 contains all details about the definitions of HFSS, the types of the HFSS, and the similarities of different types of HFSS. The improved K-means algorithm described in Chapter 3 is used in this section to find different types of HFSS. The summary of this chapter is given in section 4.2.7.

4.2.1 Flowchart of RNA Structure Pattern Analysis

The flowchart of pattern analysis of RNA structures used in our experiment is composed of 4 steps. It is shown in Figure 4.1. The purpose and function of each step is explained as follows:

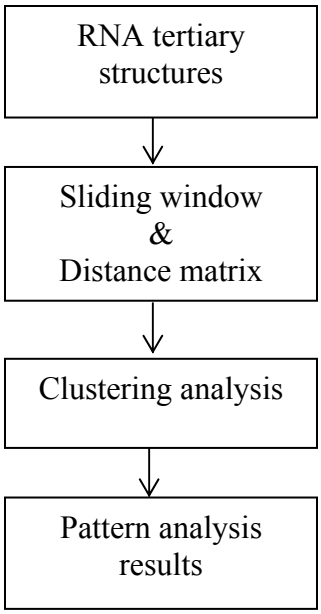


Figure 4.1. The flowchart of RNA structure pattern analysis.

Step 1:

Step 1 is data collection. At present, the PDB (Berman et al., 2000) and NDB (Berman et al., 1993) serve as a repository for raw structural data of molecules, including RNA. However, neither of these databases contains a complete representation of all available RNA structures. What is more, the data in them is in essentially raw form with few annotations and structures in PDB files often have exceptions like missing data in sequence or in the structure. Therefore manual work is required to select the suitable data for study.

Step 2:

Step 2 is data preprocessing including sliding window and distance matrix. To transform the raw data into suitable representations for further processing, we use a sliding window method to separate an RNA three dimensional structure into structure segments by moving the sliding window along the backbone of the sequence. These structure segments are the targets of pattern analysis. The structure segments are three dimensional. To facilitate the calculation of similarity between two structure segments, we use a distance matrix to describe a structure segment. The distance matrix contains the Euclidean distances between any two adjacent nucleic acids in a sliding window.

Step 3:

Step 3 is the core part of our pattern analysis experiment. In our work, an improved K-means clustering algorithm is introduced in Chapter 3. It is used to group structure segments obtained in step 2 into clusters in this experiment. The structure segments in the

same cluster are regarded to share common characteristics.

Step 4:

Step 4 is the interpretation of pattern analysis results produced by step 2. In this step, the features of different structure patterns are analyzed, including consensus distance matrix and deviation. Based on the consensus distance matrix of each structure pattern, the similarity of any two different structure patterns is calculated. This knowledge can be used to convert an RNA three dimensional structure into a string of structure patterns, and the similarity between RNA three dimensional structures can be rapidly computed by the evaluation of similarity of their strings of HFSS.

4.2.2 Dataset

Numerous studies have shown that metal ions play a crucial role in stabilizing RNA three dimensional folding (Pyle, 1993; Lilley, 1999; Hanna and Doudna, 2000; DeRose, 2003). That is why RNA-metal binding structures are relatively numerous in the RNA structures available now. In order to study the RNA three dimensional structure patterns, RNA-metal binding structures with more than 4 nucleic acids and complete structures are extracted from PDB and NDB. Currently there are 256 RNA-metal binding structures in PDB formats. However, not all these 256 structures are fit for our pattern analysis. We first remove the structures that have less than 4 nucleic acids. We then eliminate the structures whose three dimensional structures are incomplete. There are also some duplicate sequences in these 256 structures. 115 out of 256 structures are suitable for study.

In these 115 structures, there are 48 structures with a single chain folding to itself and 67 structures that have multiple chains with each chain folding with another chain. When a chain folds with another chain, almost all of such structures form a long stack (continuous base pairs). In such simple structures, there are no three dimensional motifs we are interested in. So the single-chain-folding structures become the targets of our research. The names of single-chain-folding structures are shown in Table 4.1.

Table 4.1. RNA structures with single-chain folding

1b23	1duh	1f7y	1gax	1hq1	1jbr	1jj2	1q2r	1s03	333d	1c0o	1jtw
1cx0	1ehz	1ffy	1gid	1i9v	1jbs	1kog	1q93	1u8d	486d	1drz	1u9s
1d4r	1f1t	1fir	1h4q	1j1u	1jbt	1l3z	1qf6	283d	1a1t	1f6u	1xst
1dk1	1f27	1g2j	1hc8	1j2b	1jid	1m5p	1qu3	2bbv	1ajf	1f78	1xsu

In the above 48 RNA structures, 1xst and 1xsu have the same sequences and structures. To avoid redundancy, 1xsu is removed from the final dataset used for our research. So the final dataset we use in the following experiment has 47 independent single-chain-folding RNA structures.

4.2.3 Data Preprocessing

An RNA structure is divided into structural segments by moving a sliding window along the backbone of an RNA sequence. Each time the sliding window moves one base. The sequence in the sliding window is called subsequence and the structure in the window is called a structure segment. These structure segments are to be used for pattern analysis using clustering algorithms. The structure segment in a sliding window is

described using distance matrix. The stability of RNA three dimensional structures is determined by the base pairing interaction of nucleic acids. Thus base pairing, as one of the most important information related to RNA three dimensional structures, is indispensable for pattern analysis of RNA structures. The distance matrix and classification of RNA base pairs are defined as follows.

4.2.3.1 Distance Matrix

In this work, structural clustering is to group similar structure segments obtained from RNA three dimensional structures into clusters. How to evaluate the similarity of structure segments is the key point. Since structure segments are three dimensional, we can see the structure segment as a three dimensional graph. There are various algorithms available for three dimensional graph comparison. However, three dimensional graph comparison is computationally expensive and unsuitable for efficient processing of large dataset. We can also convert a structure segment into a tree or two dimensional graph. Most of the research efforts have been made for tree comparison and graph similarity evaluation. However, some information is missed during the process of converting three dimensional to three dimensional tree or graph. Tree-based algorithms are still complex and computationally expensive. In this work, we prefer a descriptor which can represent the structure segment without putting too much overhead onto the clustering. We find that the distance matrix is a suitable descriptor meeting our requirements. It represents the Euclidean distance between any two adjacent nucleic acids in a sliding window. The definition of distance matrix is described as follows:

Let the sliding window size is w. The subsequence in the sliding window is

$S=X_1X_2\ldots Xw$. The coordinates of an RNA backbone are known. The distance matrix (DM) is

$$DM = \begin{cases} d_{ij}, & if\ i < j <= w \\ 0, & if\ i >= j \end{cases} \tag{4.1}$$

where d_{ij} is the Euclidean distance between nucleic acids i and j; w is the sliding window size.

The DM can be further simplified into a vector which only contains d_{ij} where i<j<=w. Consequently, a structure segment is finally described as a vector as follows:

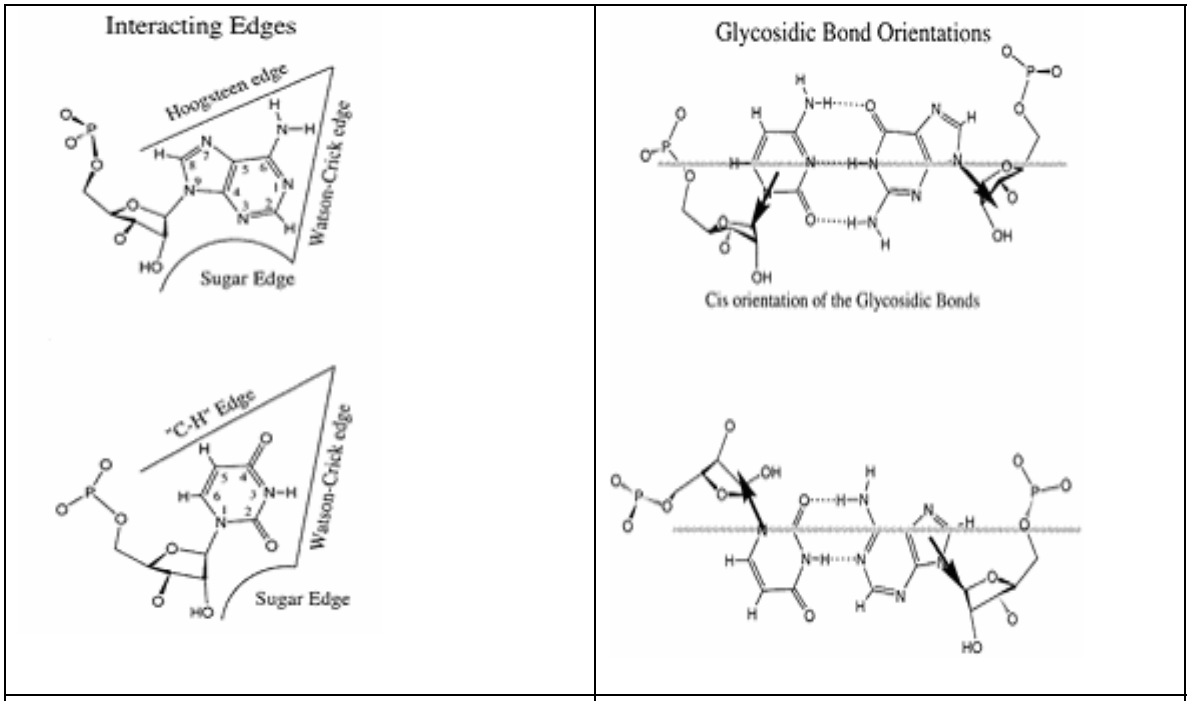
$$V = \{d_{ij} \mid 0 < i < j <= w\} \tag{4.2}$$

4.2.3.2 RNA Base Pair Classification

The RNA base pair information of each structure segment must be considered in pattern analysis. The major reason is that an RNA sequence folds itself to form secondary structures by base pairing interaction. RNA secondary structure is an important transitional step to the formation of a functional three dimensional structure. Consequently, the base pair information should be taken into consideration in structure pattern analysis of RNA three dimensional structures.

The available RNA structures to date show a great diversity of base pairing interaction (Batey et al., 1999; Nagaswamy et al., 2002). Leontis and Westhof (Leontis and Westhof, 2001) gave the criteria of base pairs classification. This classification is adapted by NDB (Berman et al., 1993) and widely used in the literature. The classification is based on two major requirements: (a) the planar edge-edge hydrogen bond interactions between two bases involve one of the three distinct edges: Watson-

Crick edge, Hoogsteen edge, and Sugar edge (see Figure 4.2). (b) a base pair has at least two hydrogen bonds. The relative orientation of the two bases is also considered in the classification as *trans* and *cis*. A line is drawn parallel to and between the two connecting H-bonds. The relative orientation of the two bases is called *trans* if the glycosidic bonds of the interacting nucleotides lie on opposite sides of the line. Otherwise it is called *cis*. This is described in Figure 4.2.



(a) (b)

Figure 4.2. Base pair classification. (a) Hytrogen bond edges in RNA bases. (b) Cis versus trans orientation of glycosidic bonds. The three edges are Waston–Crick, Hoogsteen and Sugar (Leontis and Westhof, 2001).

In this dissertation, we use the classification defined by Leontis and Westhof (Leontis and Westhof, 2001) when analyzing RNA structure segments. The classification

gives 12 classes of base pairs with the base pair orientation as in Table 4.2 shows.

Table 4.2. 12 families of base pairs.

Base pair relative orientation	Interaction edges
cis	Watson-Crick/ Watson-Crick
	Watson-Crick/Hoogsteen
	Watson-Crick/sugar
	Hoogsteen/ Hoogsteen
	Hoogsteen/sugar
	Sugar/sugar
trans	Watson-Crick/ Watson-Crick
	Watson-Crick/Hoogsteen
	Watson-Crick/sugar
	Hoogsteen/ Hoogsteen
	Hoogsteen/sugar
	Sugar/sugar

In Table 4.2, the first column is the relative orientations of the glycosidic bonds of the interaction bases and the second column is the interaction edges.

4.2.4 Experiment Setup

We have introduced the major parameters of the improved K-means algorithm and the criteria on how to determine the suitable number of clusters in section 3.6.2.3. In this experiment, the same criteria are used to identify the suitable number of clusters. The

number of clusters (K) is set from 2 to 50 to observe the changes of mean similarities with K. For each K, 20 iterations of similarity calculation are performed. We further compute the mean of the 20 similarities which indicates the stability of the clustering under cluster number K. In this experiment, F, the fraction of dataset for sampling, is 0.7. The window sizes of 3 to 20 are tested.

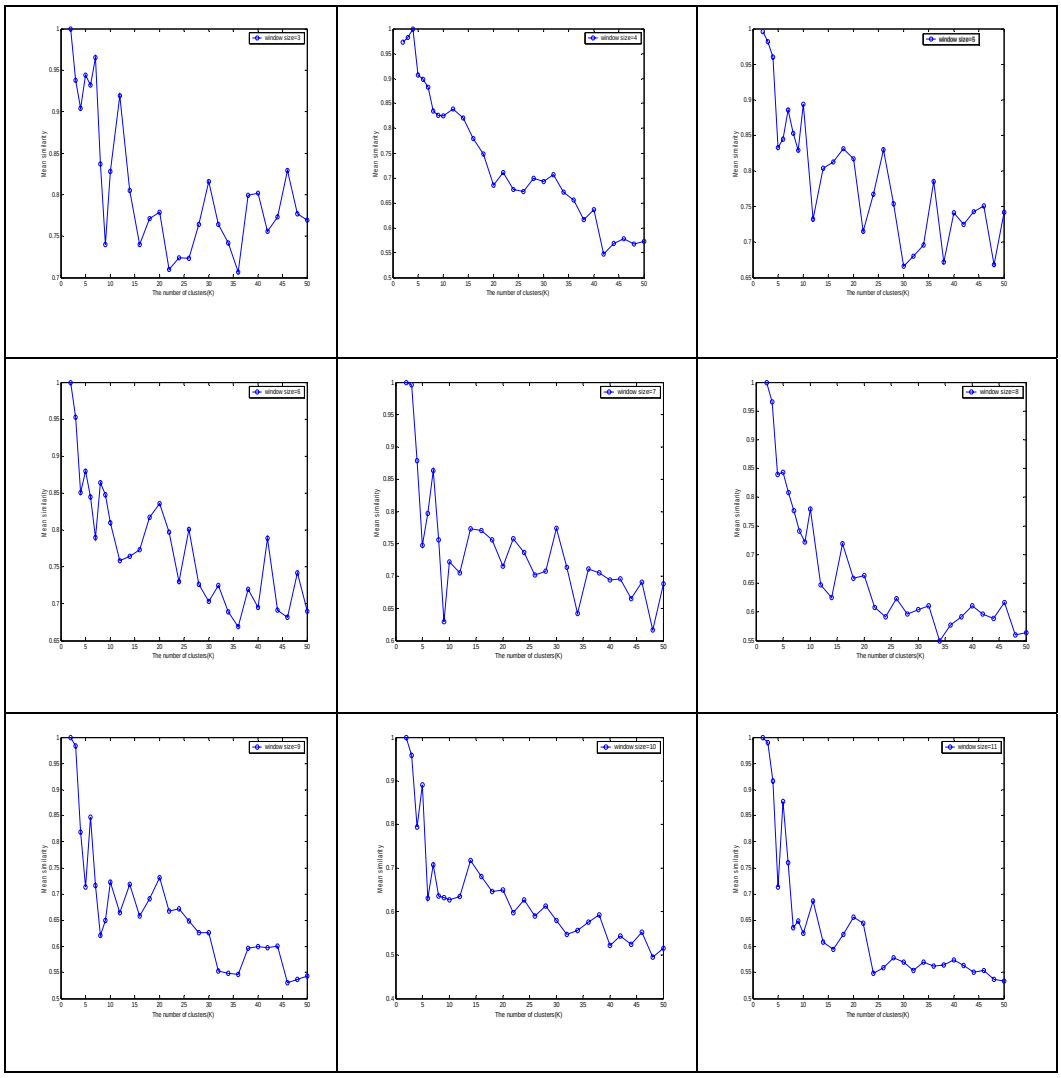
4.2.5 Appropriate Window Size

Our objective is to discover the most appropriate window size for the description of structure segments. The window size is important because it directly determines the clustering results of our pattern analysis. With an appropriate window size, the pattern of structure segments can be detected with ease. On the contrary, a too small window size or a too large window size may lead to the failure of clustering that no structure can be found. To find the most suitable window size, in this work we try window sizes from 3 to 20 using the single-chain-folding structures. The number of structure segments of the dataset with different window sizes is shown in Table 4.3.

Table 4.3. The number of structure segments with different window size 3 to 20

Window size	# of structure segments	Window size	# of structure segments
3	2183	12	1771
4	2136	13	1728
5	2089	14	1686
6	2042	15	1644
7	1995	16	1603
8	1948	17	1562
9	1902	18	1522
10	1858	19	1482
11	1814	20	1443

To find the most appropriate window size for RNA structure segment representation, in this experiment we test the window sizes from 3 to 20 with K from 2 to 50. The distribution of mean similarity under different window sizes is shown in Figure 4.3.



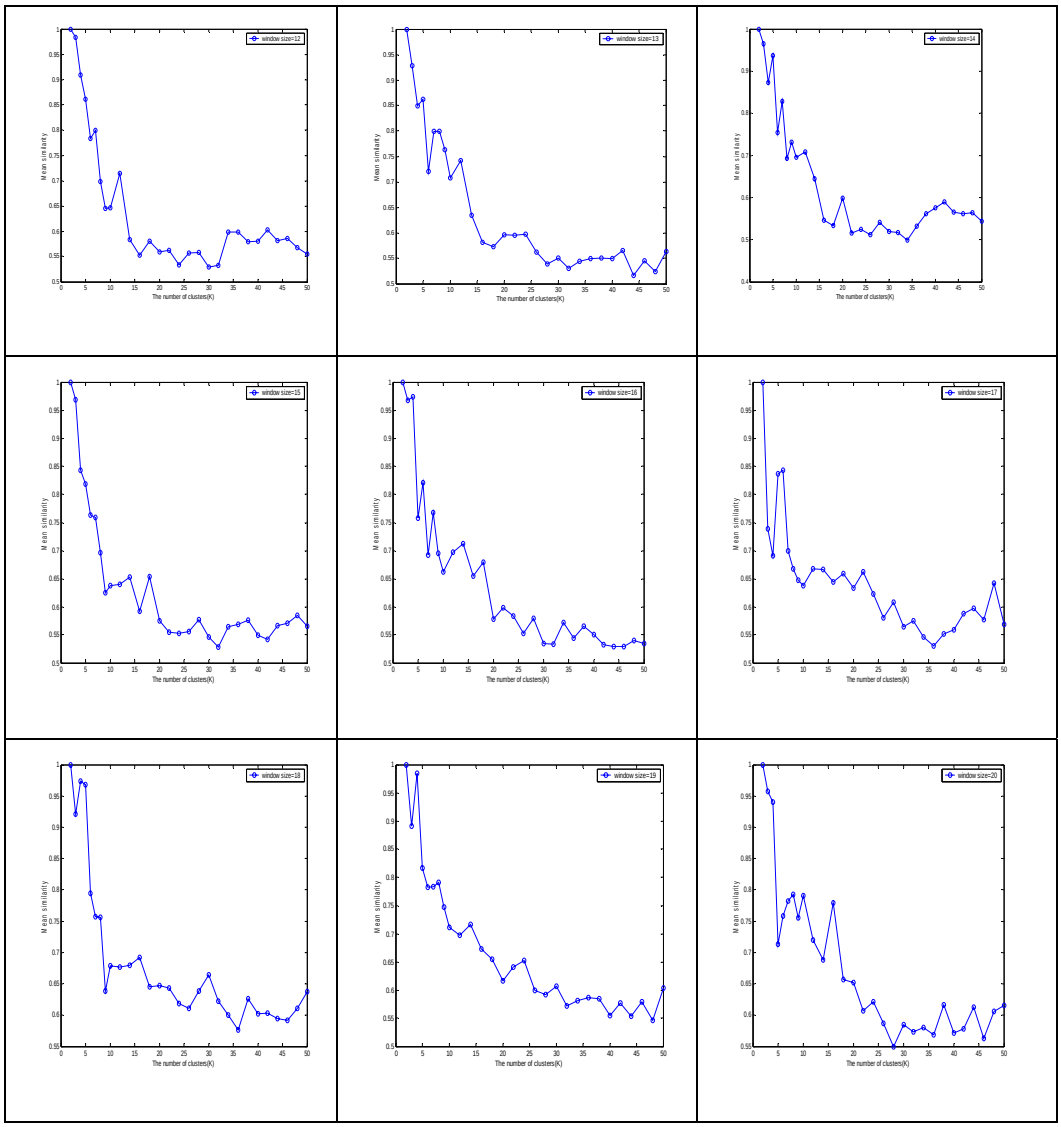


Figure 4.3. Distribution of average cluster similarities with different window sizes.

In Figure 4.3, there are eighteen figures showing the distribution of average cluster similarities under window sizes of 3 to 20. In each figure, the horizontal axis is the number of clusters (K) and the vertical axis is the average cluster similarity.

According to the criteria of the suitable number of clusters determination (see section 3.6.2.3), we expect to see a distribution that has a unique maximum average

cluster similarity value and the average cluster similarity values before the maximum are in an increasing distribution and the average cluster similarity values after the maximum are in an decreasing distribution. From Figure 4.3, we can see that only the distribution whose window size is 4 meets our requirements. When the window size is 4 and K is 4, the maximum average cluster similarity is 1 and the distribution before K=4 is in an increasing trend and the distribution after K=4 is a decreasing distribution.

Besides window size 4, the distribution at window size 17 shows some desiring characteristics that the second maximum average cluster similarity value is 0.985 at K=4 and the distribution before and after the second maximum average cluster similarity value are in increasing and decreasing trends respectively. It is worth mentioning that our algorithm generally gives relatively high average cluster similarity value at K=2. The maximum value at K=2 could be ignored if the second maximum average cluster similarity value is close to 1.

Compared with the distribution at window size 4 and 17, other distributions do not show any desiring characteristics matching our criteria of determining the suitable number of clusters. Firstly, the maximum average cluster similarity value occurs at K=2 and the second maximum value is much less than 1. Secondly, the distribution before or after the second maximum average cluster similarity value does not follow the trend we expect.

Although the distribution under window size 17 shows desiring characters, it is too large for description of a small RNA structural segment. What is more, the distribution of window size 4 is better than that of window size 17 according to our criteria. So we adopt 4 as the most appropriate window size to describe small RNA

structural segments.

Based on the above experimental results, we cluster the structure segments of window size 4 with setting K=4. Compactness of each cluster is a widely used measure for evaluation of clustering quality. The compactness of each cluster could be observed by the standard deviation of structure segments of each cluster. The standard deviation of structure segments in each cluster is show in Table 4.4.

Table 4.4. Clustering results (window size =4 and K=4)

Index	**Size**	Standard deviation					
1	1237	0.1029	0.2020	0.4207	0.1021	0.2262	0.1126
2	553	0.3003	0.6303	0.6267	0.1770	0.7894	0.3598
3	180	0.4100	1.4566	3.0631	0.8012	1.5766	0.7836
4	166	0.8596	1.9026	2.1140	0.5201	0.7084	0.2589

In Table 4.4, the first column is the index of clusters from 1 to 4. There are four clusters. The second column is the size of each cluster. The third column is the standard deviation of structure segments in each cluster. As the window size is 4, the standard deviation vector has six elements. From left to right, the elements in the standard deviation vector correspond to the pairs of bases (1,2), (1,3), (1,4), (2,3), (2,4), (3,4).

From Table 4.4, we can see that the first two clusters cover 83.80% of the whole dataset and have very low standard deviations. The third cluster has a larger standard deviation but acceptable. Only the forth cluster has relatively high standard deviation but it is a small-sized cluster with 7.77% of the whole data. Consequently, the clustering has good compactness. According to Duarte et al. (Duarte, 2003), the length of four-

nucleotide segments was successfully used to search RNA motifs from a set of RNA three dimensional structures. Our result matches the direct observations.

4.2.6 HFSS

To integrate the base pair information into the HFSS representation, we define three symbols for HFSS description. It is described in Table 4.5.

Table 4.5. Symbols for HFSS description

Symbols	Name	Definitions
(	left parenthesis	A base that is paired with another base after itself along the sequence
)	right parenthesis	A base that is paired with another base before itself along the sequence
:	colon	An unpaired base

The above clustering analysis reveals that window size 4 is most suitable for the description of structure segments. With window size 4 and 3 possible symbols at each position, there are totally 81 types of combinations. To find the HFSS, we statistically analyze all types of structure segments. The structure segments with quantity larger than 20 are shown in Table 4.6 while the structure segments with quantity no more than 20 are shown in Table 4.7. The types in Table 4.6 are called *large-sized types* while the types in Table 4.7 are called *small-sized types*. The standard deviation of each type of structure segments is also displayed in Table 4.6.

As we know, the window size 4 and 3 possible symbols at each position, there should be totally 81 types of structure segments. Actually, 53 types of structures segments are found in our dataset. That the structure segments do not exist in this dataset

does not mean that they could not occur beyond the dataset used in this experiment.

Table 4.6. Standard deviation of large-sized types

ID	Structure segment	Num	Standard Deviation					
1	((((	301	0.249	0.373	0.675	0.203	0.350	0.229
2	(((:	116	0.225	0.452	1.297	0.270	0.711	0.420
3	((:(	21	0.342	0.478	1.876	0.300	1.544	0.744
4	((::	104	0.274	0.695	1.459	0.466	0.924	0.483
5	(:::	81	0.463	0.970	2.179	0.495	1.473	0.753
6	))))	333	0.219	0.418	0.647	0.209	0.362	0.220
7	)))):	78	0.227	0.354	0.909	0.270	0.786	0.828
8	)))::	65	0.290	0.869	1.483	0.914	1.278	0.315
9	)::::	53	0.993	1.358	2.106	0.289	1.636	0.693
10	:(((	70	0.719	1.740	2.105	0.373	0.574	0.230
11	:)))	106	0.643	1.157	1.545	0.245	0.491	0.254
12	::((	58	0.696	2.036	3.104	0.736	1.837	0.404
13	::))	105	0.647	1.724	2.398	0.656	1.201	0.247
14	:::(	45	0.708	1.872	3.018	0.783	2.102	0.773
15	:::)	82	0.696	1.505	2.523	0.701	1.874	0.684
16	::::	322	0.670	1.635	2.475	0.751	1.637	0.750

In Table 4.6, the first column is the ID of structure segments of each type. There are 16 types of structure segments which are high frequent in this dataset. The second

column is the structure segments represented in the symbols defined by Table 4.5. The column Num is the number of structure segments of each type. The last column is the standard deviation of distance matrix of the same type of structure segments.

For each type of structure segment in Table 4.6, the standard deviation which is larger than 1 is underlined. From the standard deviation, we can get some clue about the variation of the structure segments in each type. Larger standard deviation indicates larger variation of the structure segments in the same type. If the types with 2 or more standard deviation values larger than 1 are considered as regular types, only type 1, 2, 4, 6, and 7 are regular while others are irregular, especially type 9, 12, 13, 14, 15, and 16.

Table 4.7. Consensus distance vectors of small-sized types

ID	Structure segment	Num	Consensus Distance Vector					
17	((()	4	6.180	11.826	16.636	6.316	11.788	6.203
18	(())	5	6.300	10.962	15.156	5.621	11.054	6.223
19	(():)	1	6.782	12.657	18.339	6.426	12.345	6.270
20	((:)	2	6.489	10.733	14.624	6.181	11.793	6.180
21	())	2	5.110	10.562	15.952	6.430	12.310	6.242
22	()):	3	5.962	11.383	15.605	6.086	11.216	6.416
23	()::)	3	5.882	11.002	13.517	5.553	9.690	5.951
24	(:((	18	6.382	10.551	14.726	5.753	10.597	6.281
25	(:(:	4	6.318	11.171	15.513	5.591	10.586	5.939
26	(:))	1	6.313	12.289	17.423	6.232	11.872	6.182
27	(:):	1	6.049	11.297	17.936	6.127	12.758	7.290
28	(::(	15	5.973	11.160	14.869	6.041	10.623	6.086
29	(::)	14	6.371	10.757	13.895	6.041	10.811	5.905
30	)(((	10	5.955	10.647	15.258	6.185	11.751	6.275
31	))((	7	6.186	11.668	16.079	6.531	12.056	6.160

32	)))(	7	6.218	11.743	15.736	6.186	11.668	6.531
33	)):(	3	6.076	11.583	16.016	6.113	11.693	6.253
34	)):)	11	6.276	11.591	16.380	6.122	11.852	6.185
35	):((	3	6.113	11.693	16.268	6.253	11.886	6.329
36	):)(	1	6.467	12.089	16.611	6.245	10.641	4.617
37	):))	9	5.988	11.788	16.482	6.257	11.762	6.277
38	):):	3	6.119	11.535	15.209	6.041	10.987	5.988
39	)::(	2	6.657	11.624	14.639	6.598	11.050	5.452
40	)::)	16	6.190	11.224	15.134	6.169	11.212	5.863
41	:(()	2	6.660	12.078	15.295	6.507	10.157	4.861
42	:((:	10	5.722	10.630	15.281	6.402	11.730	5.903
43	:():	2	6.771	11.849	15.701	5.610	10.330	5.194
44	:(:(	1	5.083	10.226	15.906	6.119	11.287	6.207
45	:(::	6	5.789	10.930	15.560	5.851	11.093	6.110
46	:)((	3	5.058	9.124	10.460	4.610	7.360	6.243
47	:)):	6	5.895	10.944	15.114	6.216	11.470	6.052
48	:):)	2	5.303	10.129	15.476	5.688	11.203	6.324
49	:)::	7	6.162	11.457	15.100	6.360	10.402	5.961
50	::()	2	4.377	10.455	14.858	6.771	11.849	5.610
51	::(:	3	5.725	10.420	14.542	5.819	11.154	5.823
52	::)(	2	6.273	10.036	12.822	4.465	8.366	4.607
53	::):	5	6.194	11.346	15.627	5.897	10.947	6.128

The layout of Table 4.7 is similar to Table 4.6, but the last column is the consensus distance vector instead of standard deviation. The consensus distance vector is the mean of all distance vectors whose structure segments belong to the same type. Since the each type of structure segments in Table 4.7 has a small quantity, all distance vectors

of the same type have to be considered. However, for a large-sized type, to emphasize the majority of the structure segments the minority should be ignored as noise. The steps to compute the consensus distance vector for a type with large quantity is explained below.

Consensus Distance Vector

In this work, the special features of a structure segment are described using a distance vector. So we use a consensus distance vector to interpret the special features of a specific type of structure segments. For a type with low standard deviation or small size, we directly calculate the mean of all distance vectors of structure segments belong to the same type. However, for a large-sized type with high standard deviation, we have to extract the consensus distance vector through clustering analysis. It includes two steps. In step 1, find the suitable number of clusters in the structure segments of a specific type. In step 2, partition the distance vectors of the structure segments and compute the consensus distance vector by only considering significant clusters. Based on the statistical results of Table 4.6, the types 3, 5, 9, 12, 13, 14, 15, and 16 should be analyzed by the two steps before calculating their consensus distance vectors. The results are demonstrated as follows.

Step1. Find the suitable number of clusters

The large-sized types of 3, 5, 9, 12, 13, 14, 15, and 16 have high standard deviations and should be analyzed to find the suitable number of clusters. The distribution of average cluster similarities of each type of structure segments is shown in Figure 4.4.

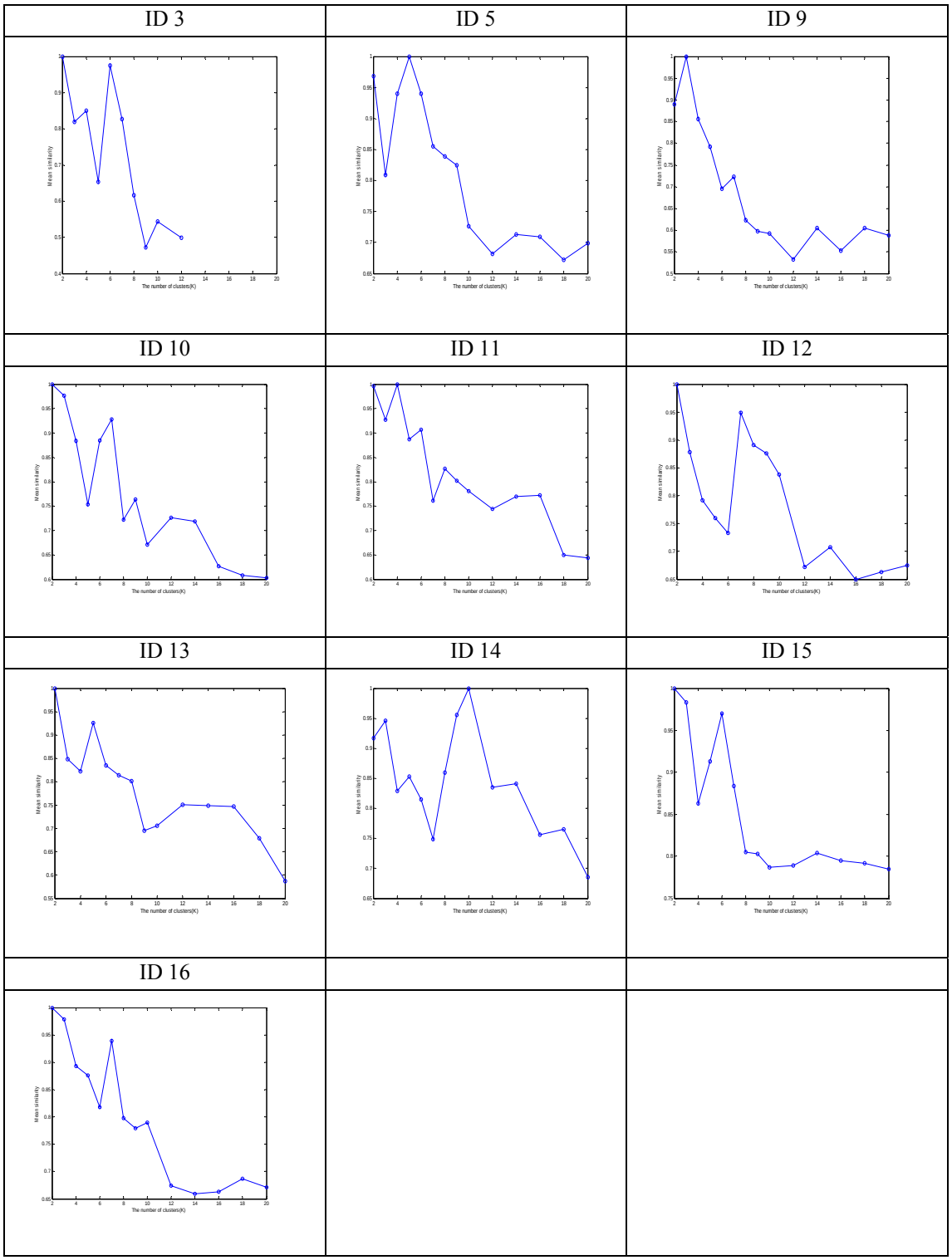


Figure 4.4. Distribution of average cluster similarities of structure segments with high standard deviation.

Figure 4.4 shows the distributions of average cluster similarities for the structure segment types 3, 5, 8, 9, 10, 12, 13, 14, 15, and 16. From the distribution, we can easily determine the suitable number of clusters of each type. The suitable numbers are 6, 5, 3, 5, 4, 7, 5, 10, 6, and 7 respectively.

Step 2. Calculate the consensus vector

In step 1, the suitable number of clusters for each type of structure segments has been discovered. Using the suitable numbers, we cluster the structure segments type by type. When calculating the consensus distance vectors, the clusters whose sizes are fewer than 10% of the whole set of structure segments are ignored. The consensus distance vectors of types with high standard deviations are shown in Table 4.8.

Table 4.8. Consensus distance vectors of large-sized types

ID	Structure segments	Num	Consensus Distance Vector					
1	((((	301	6.245	11.959	16.823	6.242	11.971	6.251
2	(((:	116	6.229	11.888	16.554	6.251	11.760	6.198
3	((:(	21	6.306	12.126	15.823	6.307	10.664	5.505
4	((::	104	6.255	11.713	16.154	6.132	11.276	6.057
5	(:::	81	6.150	11.590	15.768	6.167	11.133	5.953
6	))))	333	6.235	11.939	16.937	6.233	11.972	6.244
7	)))):	78	6.231	11.948	16.999	6.244	11.951	6.222
8	)))::	65	6.230	11.937	16.790	6.243	11.695	6.185
9	)::::	53	6.105	11.618	16.265	6.158	11.390	5.982
10	:(((	70	6.082	11.316	15.874	6.244	11.848	6.221
11	:)))	106	5.983	11.185	15.980	6.232	11.827	6.229
12	::((	58	6.069	10.353	13.629	5.886	10.588	6.239
13	::))	105	5.976	10.558	14.769	5.987	11.249	6.239
14	:::(	45	5.938	10.354	14.508	5.843	10.945	5.979
15	:::)	82	6.003	10.667	13.987	5.925	10.546	6.032
16	::::	322	5.984	10.563	14.065	5.931	10.445	5.919

Now we have found the consensus distance vectors of all 53 types of structure segments. The structural similarity of any two types of structure segments can be assessed using the distance between the corresponding consensus distance vectors. With the 53 types of structure segments and their similarities, given a three dimensional RNA structure and the base pair information, we can convert the structure into a string of structure segments. Using the similarities among different types of structure segments, we can efficiently evaluate the similarity of any two RNA molecules and rapid search known and novel motifs from the strings. These hard problems if representing an RNA as a graph becomes much easier to be solved by using structure segment definition and similarities of different types of structure segments.

If our results are used to serve for rapid substructure search or mining known and novel motifs, all the definitions and experimental results are good enough for solving these problems. If the objective is to make the comparison of RNA structure similarity as efficient as possible, more work needs to be done.

By observing the size and consensus distance vector of each type in Table 4.7 and 4.8, we find that there are many small-sized types whose consensus distance vectors are very close to those of some large-sized types, and many small-sized types have similar consensus distance vectors with each other. The small-sized types with similar consensus distance vectors with large-sized types are unnecessary to exist independently because even though they are independently treated they can not bring much performance to estimate the similarity of RNA structures. On the one side, they do not appear frequently. On the other side, it is highly possible for them to be compared with similar consensus distance vectors of different types. So we decide to simplify the 53 types into a smaller

set of types of structure segments through clustering analysis. Before clustering, we have to find the suitable number of clusters in the 53 consensus distance vectors. The distribution of average cluster similarities is shown in Figure 4.5.

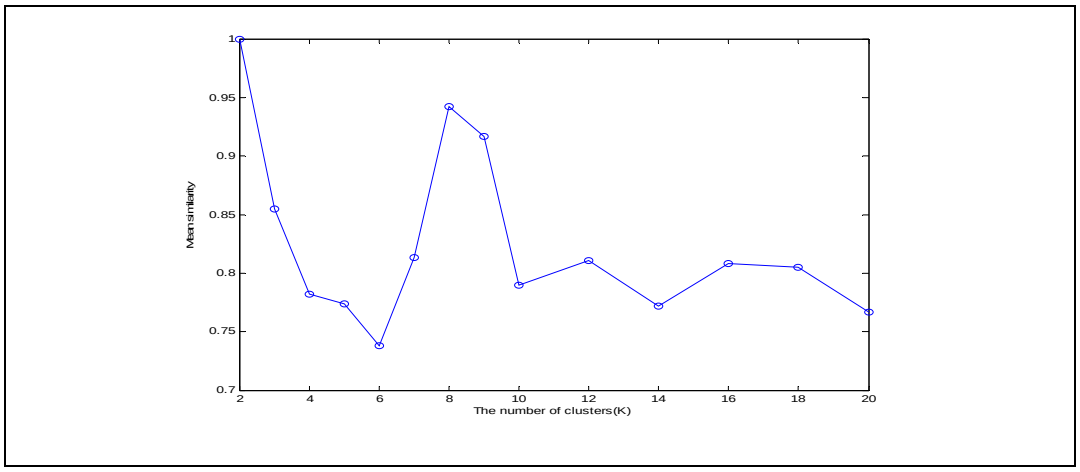


Figure 4.5. Distribution of average cluster similarities of clustering 53 types of structure segments.

From the Figure 4.5, we can see that the suitable number of clusters in the 53 types of structure segments is 8. Based on the result, we partition the 53 types into 8 clusters. The distribution of the 53 types is shown in Table 4.9.

Table 4.9. Eight clusters

Index	Num	SS Name	SS ID
1	3	A	19 26 27
2	4	B	21 44 48 50
3	7	C	12 14 15 16 23 24 29
4	16	D	1 2 4 6 7 8 9 10 11 17 31 32 33 34 35 37
5	4	E	3 36 41 43
6	11	F	5 18 22 25 28 38 39 40 45 49 53
7	6	G	13 20 30 42 47 51
8	2	H	46 52

In Table 4.9, the first column is the index of clusters from 1 to 8. The second

column is the number of types in each cluster. The third column is the names of each cluster. The names are used to represent the 8 types of structure segments. The last column is the SS ID which matches the column SS ID in Table 4.7 and 4.8.

The consensus distance vector and standard deviation of each cluster is shown in Table 4.10.

Table 4.10. Eight types of HFSS

SS Name	**Consensus distance vector and Standard Deviation**
A	6.381 12.081 17.899 6.262 12.325 6.581
	0.371 0.704 0.459 0.152 0.443 0.616
B	4.968 10.343 15.548 6.252 11.662 6.096
	0.406 0.200 0.507 0.461 0.518 0.328
C	6.090 10.607 14.047 5.847 10.517 6.044
	0.204 0.229 0.439 0.157 0.402 0.154
D	6.165 11.712 16.373 6.235 11.775 6.220
	0.094 0.223 0.391 0.097 0.206 0.117
E	6.551 12.036 15.858 6.167 10.448 5.044
	0.206 0.126 0.551 0.388 0.247 0.388
F	6.165 11.307 15.289 6.038 10.937 6.011
	0.225 0.239 0.353 0.297 0.275 0.242
G	5.960 10.655 14.931 6.132 11.525 6.079
	0.281 0.176 0.327 0.202 0.276 0.185
H	5.665 9.580 11.641 4.537 7.863 5.425
	0.859 0.645 1.670 0.102 1.157 0.711

In Table 4.10, the italic numbers are standard deviations of each cluster. Above each line of standard deviation, it is the consensus distance vector of each cluster. We can see

that the standard deviations of all clusters except of H are very small with most fewer than 0.5. Only the standard deviations of cluster H is a little large but acceptable. The results indicate that 8 types are good enough to represent 53 types. These eight clusters are the HFSS we want.

The similarity of any two different types of HFSS is measured by the distance between the corresponding consensus distance vectors. Smaller distance indicates higher similarity. The distances between different types of structure segments are shown in Table 4.11.

Table 4.11. Distances between eight different types of HFSS.

SS Name	A	B	C	D	E	F	G	H
A	0	3.350	4.563	1.716	3.177	3.124	3.454	8.376
B		0	2.249	2.004	2.838	1.734	1.223	5.844
C			0	2.898	2.577	1.501	1.378	4.021
D				0	1.915	1.457	1.825	6.783
E					0	1.482	2.313	5.838
F						0	0.977	5.347
G							0	5.334
H								0

In Table 4.11, the first row and the first column contain the eight types of HFSS. The eight types of HFSS are described in Table 4.9 and 4.10. With the similarity of different types of HFSS, the similarity of any two RNA three dimensional structures can be obtained by assessing the similarity of their corresponding HFSS strings.

4.3 Conclusion

The contribution of our work is that we discover the HFSS in RNA three dimensional structures by clustering algorithms. Using these HFSS, a complex RNA three dimensional structure can be converted into a string of HFSS. Since a string of HFSS describes a specific RNA structure, the subtle structural differences between molecules can be obtained by directly comparing their strings of HFSS. Our clustering algorithm and experimental results can be used for evaluation of similarity of RNA three dimensional structures and rapid RNA structure database search.

Chapter 5

Pattern Analysis on Protein Structures

5.1 Objective

The effectiveness of our improved K-means clustering algorithm and structure descriptor has been tested on RNA structures and successfully verified by the experimental results. In this chapter, we will test the algorithm and new descriptor on protein structures. The objective is to see whether our algorithm and structure description are useful for protein structure pattern analysis. We are also interested in the difference between RNA and protein structure patterns.

Since we propose a new structure descriptor in this work, the targets of clustering are the three dimensional structure segments instead of sequence segments. Through grouping similar structure segments into clusters, the protein sequence-structure correspondence is uncovered and is further used for protein structure research. We compare our new descriptor with the descriptor used in Chen's work (Chen et al., 2006). The experimental results of protein dataset show that our new descriptor is more effective for the protein pattern analysis.

This chapter is organized as follows. Section 5.2 gives an introduction to protein structures. Section 5.3 demonstrates all of the details and results of this experiment, including the six datasets used in this experiment, experiment setup, the suitable number of clusters of each dataset analyzed using the improved K-means clustering algorithm, experimental results validation by the measures of secondary structure similarity and the

population analysis. The discussion and conclusion are drawn in section 5.4.

5.2 Background Knowledge

Proteins are molecules performing a wide variety of functions in living systems. (Lesk, 2004). Proteins are composed of amino acids linked together by covalent peptide bonds. There are 20 different standard amino acids, also referred to as residues (Creighton, 1993). Proteins have multiple levels of structure, including primary structure, secondary structure, three dimensional structure, and quaternary structure (Branden and Tooze, 1998). The number of released protein structures is increasing at a growing pace. Thus many databases have been built for protein three dimensional structures repository and research. PDB (Berman et al., 2000) is the largest one. As of January 2007, PDB contains 37,537 protein structures with around 90% determined by X-ray crystallography and about 9% of the protein structures obtained by NMR. The protein structures used in our experiment are retrieved from PDB, which are determined by X-ray crystallography.

The protein functions are highly related to their structures. So obtaining protein structures is very important for understanding their functions. However, experimental determination of protein three dimensional structures is still time-consuming and expensive. As a result, computational approaches to the prediction of protein structures from their amino acid sequences are required (Han and Baker, 1995; Rost, 2001; Petrey, 2005; Zhong et al., 2005; Zhang et al., 2005). The methods those predict the structure for an unknown protein relying on the homologous proteins with known structures are called homology based methods. For homology based methods, uncovering the protein sequence-structure correspondence is crucial for the success of protein structure

prediction. Clustering algorithms have been successfully used to study the sequence-structure correspondence for protein secondary structure prediction problem (Han and Baker, 1995; Zhong et al., 2005; Chen et al., 2006). In this chapter, we show how to improve the performance of clustering by efficient representation protein structure segments and determination of the suitable number of clusters in a set of protein structures.

5.3 Experiments on Protein Structures

In this section, we describe the procedure of pattern analysis of protein structures and validate the experiment results using both biological and statistical measures. This section is organized as follows. The protein structures used for protein pattern analysis are introduced in Section 5.2.1. Section 5.2.2 explains the details of experiment setup. In Section 5.2.3, we show how to find the suitable number of clusters in each dataset (there are six datasets) using the improved K-means clustering algorithm. To show the effectiveness our algorithm, the experimental results are verified using the measures of secondary structure similarity analysis and population analysis respectively in Section 5.2.4 and 5.2.5.

5.3.1 Dataset

Six datasets are used in this experiment and each dataset includes 100 protein structures. The protein structures are randomly selected from Protein Sequence Culling Server (Wang and Dunbrack, 2003), a public server for culling sets of protein sequences from the PDB through certain sequence identity and structural quality criteria. In Chen's

work (Chen et al., 2006), the protein sequences are separated into segments using sliding window with a window size 9. To compare with their work, we keep using 9 as the window size. The protein structures in the six datasets are represented in two descriptors. Our descriptor generates structure segments while the descriptor (Chen et al., 2006) produces sequence segments. The definitions of structure segments and sequence segments are explained as follows.

5.3.1.1 Representation of structure segments

A good descriptor should contain enough information and keep as easy-understanding and each-using as possible for study. Keeping this goal in mind, we propose a new descriptor for the protein structure segments. The descriptor is based on a distance matrix which consists of the Euclidean distance between any two amino acids in a sliding window. The definition of distance matrix is described as follows:

Let the sliding window size is w. The subsequence in the sliding window is $S=X_1X_2\ldots Xw$. The coordinates of a protein backbone are known. The distance matrix (DM) is

$$DM = \begin{cases} d_{ij}, & if\ i < j <= w \\ 0, & if\ i >= j \end{cases} \tag{5.1}$$

where d_{ij} is the Euclidean distance between amino acids i and j; w is the sliding window size.

The DM can be further simplified into a vector which only contains d_{ij} where i<j<=w. Consequently, a structure segment is finally described as a vector:

$$V = \{d_{ij} \mid 0 < i < j <= w\} \tag{5.2}$$

Based on the above definition, a structure segment with window size is represented by a 36-demension vector in which each element is the Euclidean distance of any two amino acids in this sliding window.

5.3.1.2 Representation of sequence segments

We use the frequency profile from the HSSP (Sander and Schneider, 1991) to represent the sequence segments. HSSP is a database of homology-derived secondary structure of proteins. The frequency profile from HSSP is constructed based on the alignment of each protein sequence from the homologous sequences in PDB. Since there are 20 standard amino acids and the window size used in this experiment is 9, each position in the window has 20 possible frequencies corresponding to 20 different amino acids. Consequently, a sequence segment is represented by a 180-demension vector.

5.3.1.3 Secondary Structure

The secondary structures of each protein sequence are obtained from DSSP (Kabsch and Sander, 1983), a database of secondary structure assignments for all protein entries in the PDB. DSSP originally assigns the secondary structure to eight different classes. In this work, we combine the eight classes into three classes: H, G and I to H (Helices); B and E to E (Sheets); all others to C (Coils). The secondary structures of protein sequences are used to evaluate the clustering quality in our experiment.

5.3.2 Experiment Setup

In this experiment, we set the number of clusters (K) change from 2 to 50. For each K, 20 iterations of similarity calculation are performed. The average of the 20

similarities is used to indicate the stability of the clustering under cluster number K. F, the fraction of samples, is 0.7. The criteria mentioned in section 3.6.2.3 are used to determine the suitable number of clusters. According to Chen's work (Chen et al., 2006), window size 9 is appropriate size for sequence segment. To compare with their work, we keep using it in this experiment.

5.3.3 Detecting the Number of Clusters

We test our algorithm on both structure segments and sequence segments. The distribution of average cluster similarities of each dataset is shown as follows.

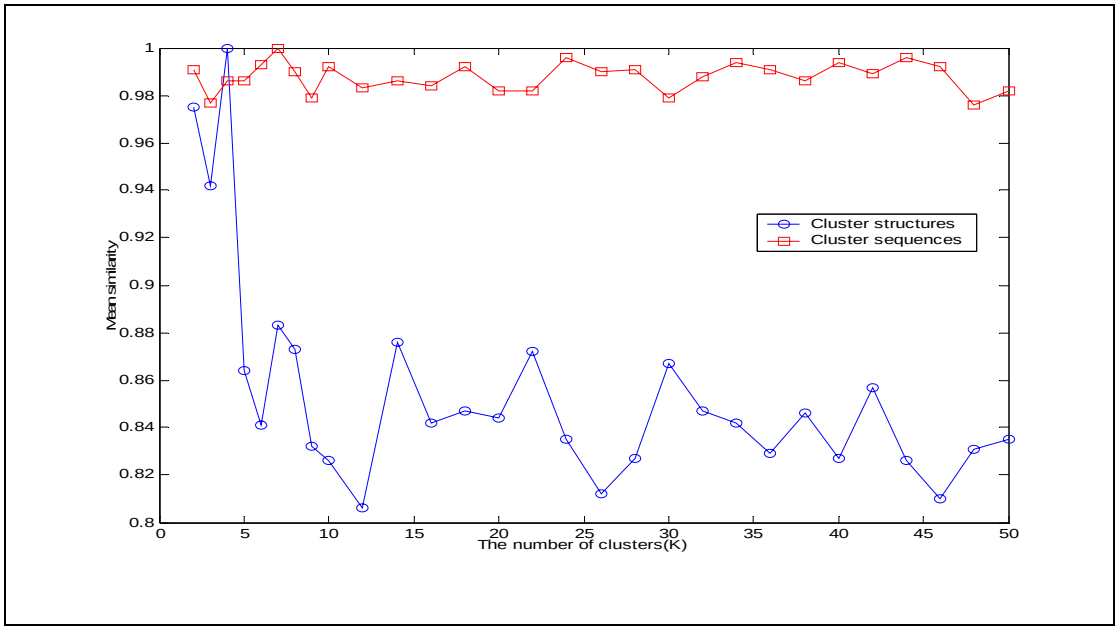


Figure 5.1. Distribution of average cluster similarities of dataset 1. Distribution of clustering on structures (in blue); (b) Distribution of clustering on sequences (in red)

From Figure 5.1, we can see that there is a unique maximum average cluster similarity in both (a) and (b). For (a), the maximum is at K= 4 and the maximum is at K=7 for (b). The distribution in (a) follows the increasing trend before K=4 and

decreasing trend after K=4. Thus it is clear that the suitable number of clustering of this dataset is 4 from (a). Compared with (a), the distribution in (b) has no decreasing trend after the maximum average cluster similarity value. In addition, most of the average cluster similarity values are very close to the maximum value with different less than 0.005. So there is no strong evidence to decide the suitable number of clusters.

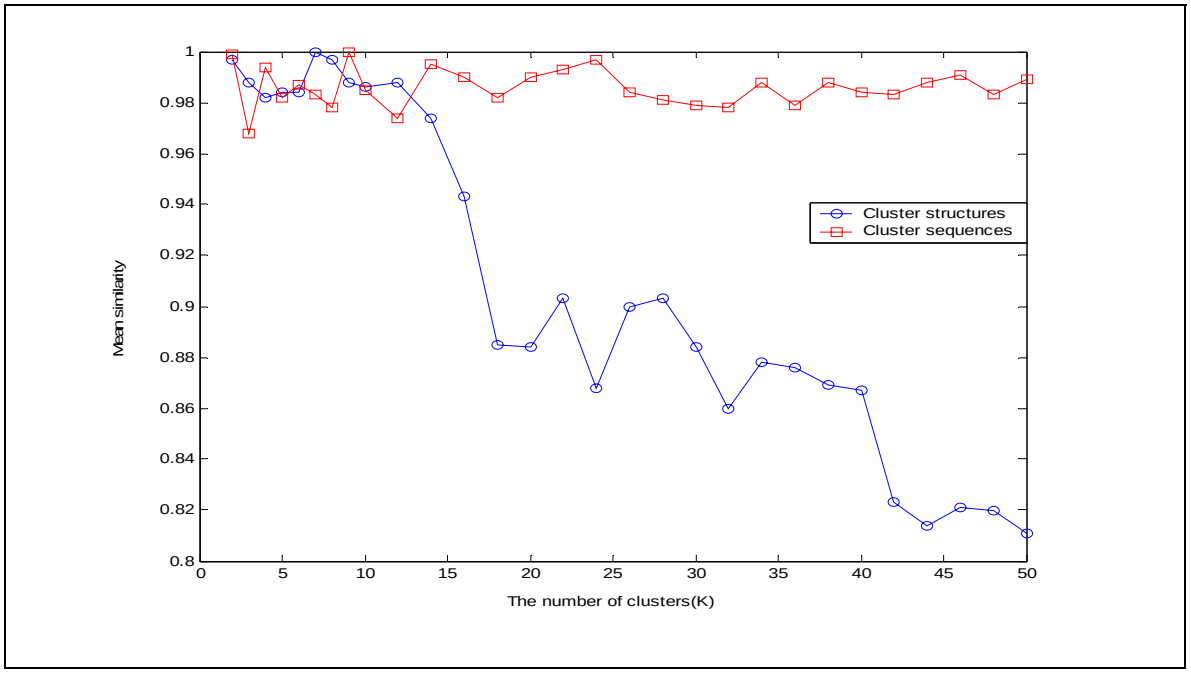


Figure 5.2. Distribution of average cluster similarities of dataset 2. Distribution of clustering on structures (in blue); (b) Distribution of clustering on sequences (in red)

From Figure 5.2, we can see that there is a unique maximum average cluster similarity in both (a) when K is 7 and (b) when k is 9. In (a), the average cluster similarities from K=2 to K=12 have no much change. Before the maximum value, there is no obvious increasing trend. However, after the maximum value the distribution follows a decreasing trend. It is hard to decide the definite suitable number but we can

give estimation that it is fewer than 12. Although (b) has unique maximum average cluster similarity value, there are many average cluster similarities very close to the maximum with difference less than 0.005. It is hard to decide the suitable number of clusters from (b).

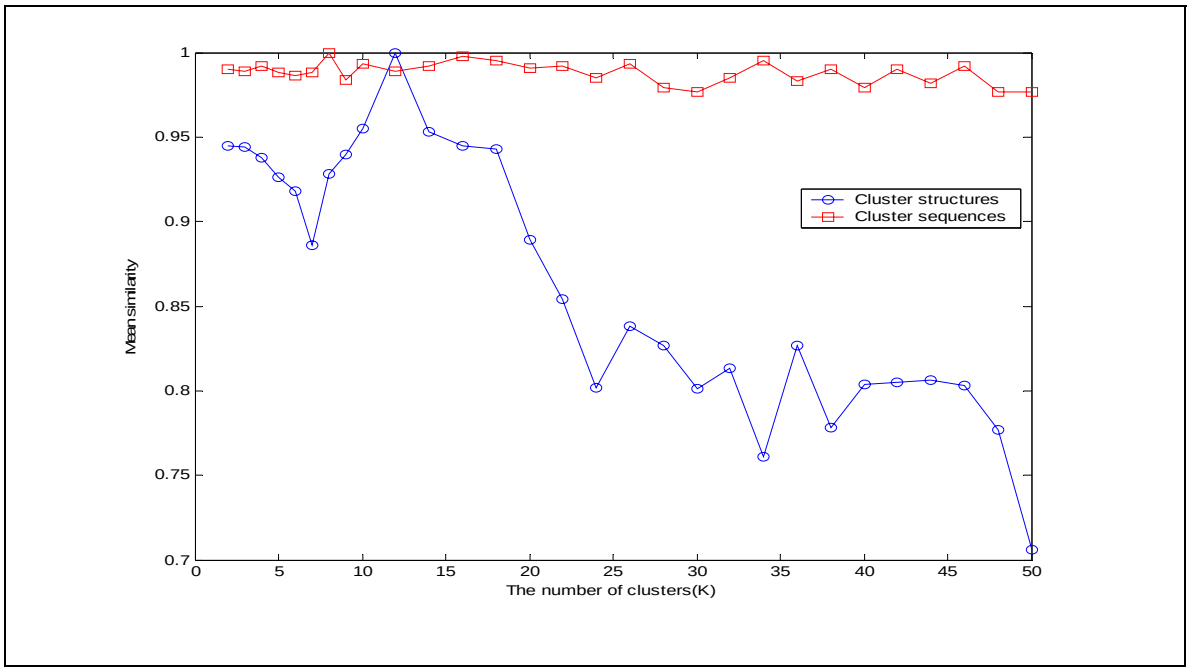


Figure 5.3. Distribution of average cluster similarities of dataset 3. (a) Distribution of clustering on structures (in blue); (b) Distribution of clustering on sequences (in red)

In Figure 5.3, there is a unique maximum average cluster similarity in both (a) when K is 12 and (b) when K is 8. The distribution in (a) matches our distribution requirements very well. So it is easy to determine the suitable number of clusters of dataset 3 is 4 from (a). Although (b) has unique maximum average cluster similarity value, there are many average cluster similarities very close to the maximum (the difference is less than 0.005) after K=8. The distribution in (b) does not meet our requirements of suitable number of clusters. So it is hard to determine the suitable

number of clusters from (b).

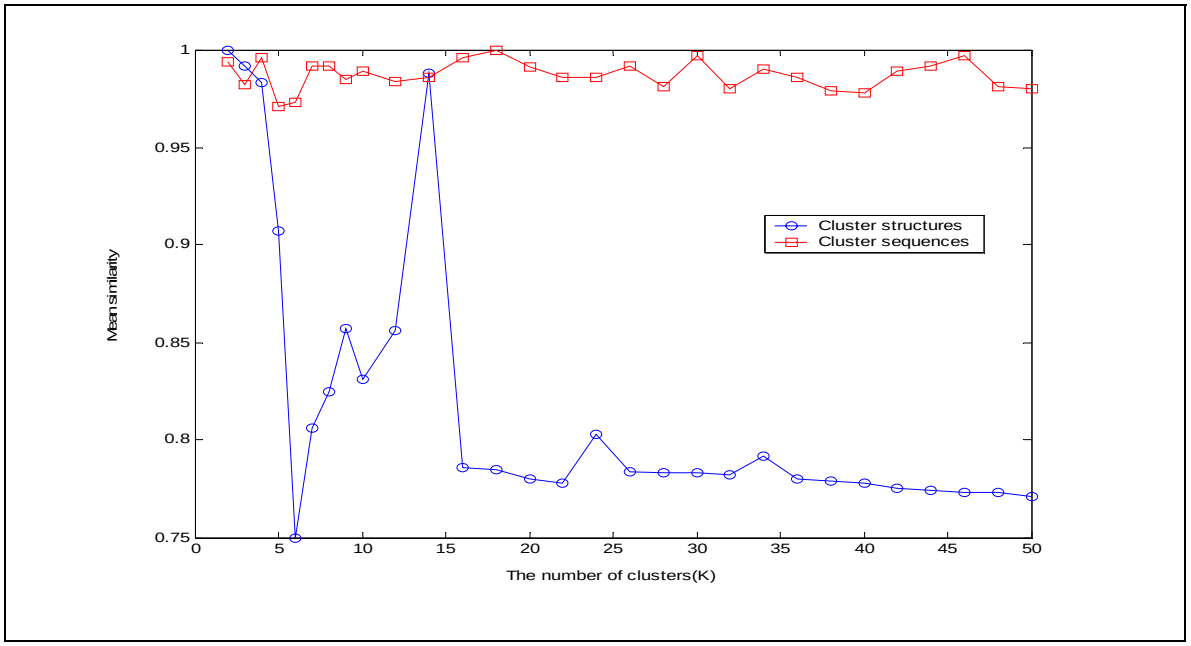


Figure 5.4. Distribution of average cluster similarities of dataset 4. (a) Distribution of clustering on structures (in blue); (b) Distribution of clustering on sequences (in red)

In Figure 5.4, there is a unique maximum average cluster similarity in (b) when k is 18. In (a), the average cluster similarity value at K=2 is larger than that at K=14. The value at K=2 could ignored since in general the average cluster similarity value is relatively high at K=2. It is resulted by the features of K-means algorithm and dataset. The distribution in (a) meets our distribution requirements pretty well. So the suitable number of clusters of dataset 3 is 14 In (b), there are many values very close to the maximum (the difference is less than 0.005) after K=18. Additionally, the distribution in (b) is far from our distribution criteria. So we cannot determine the suitable number of clusters from (b).

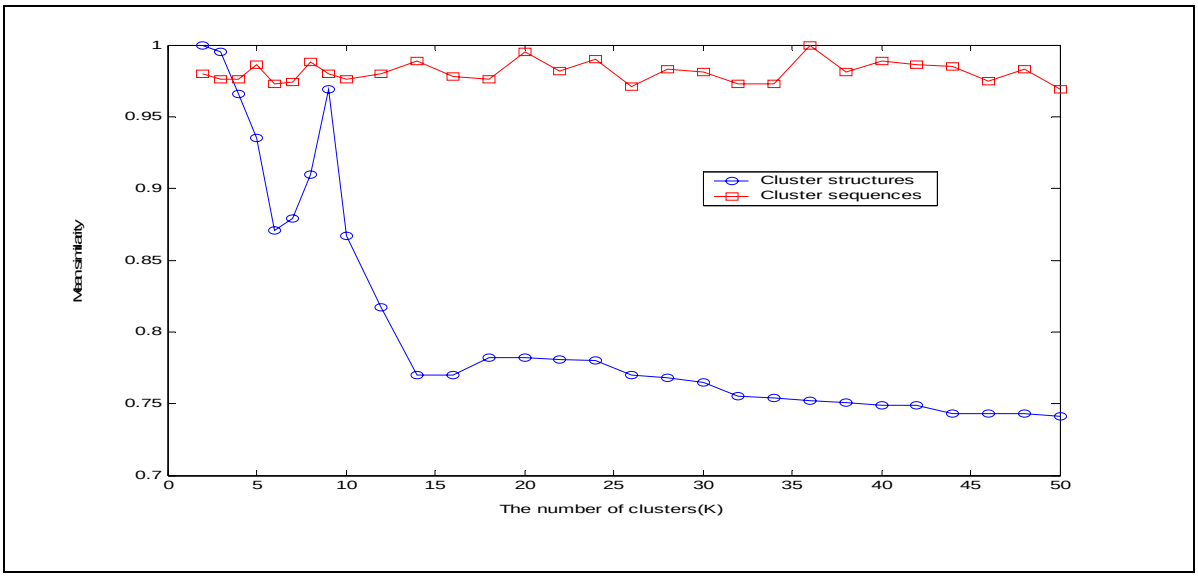


Figure 5.5. Distribution of average cluster similarities of dataset 5.(a) Distribution of clustering on structures (in blue);(b) Distribution of clustering on sequences (in red)

The distributions in Figure 5.5 are similar to those in Figure 5.1. From (a), the suitable number of clusters is 9. The distribution in (b) does not meet the requirements. Thus it is hard to determine the suitable number of clusters in dataset 5.

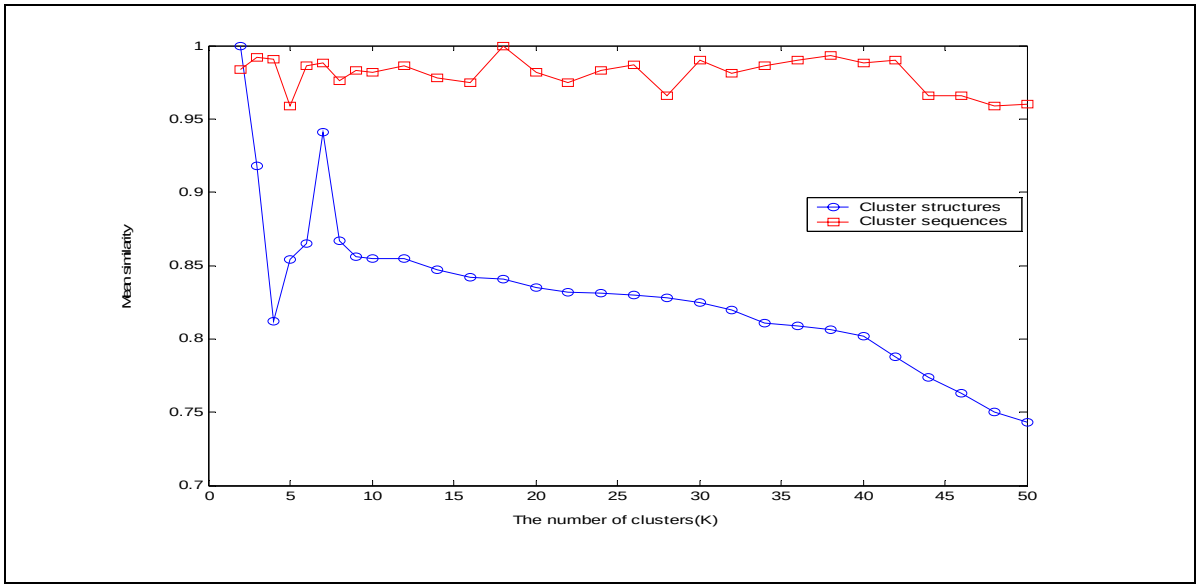


Figure 5.6. Distribution of average cluster similarities of dataset 6. (a) Distribution of clustering on structures (in blue); (b) Distribution of clustering on sequences (in red)

The distributions in Figure 5.6 are similar to those in Figure 5.5. From (a), we can

see that the suitable number of clusters is 7. The distribution in (b) does not meet the requirements. Thus it is hard to decide the suitable number in dataset 6.

We have analyzed the distributions of six protein datasets using both structure segments and sequence segments. We observe the following advantages of structure segments over sequence segments:

(1) Compared with average cluster similarities from structure segments clustering, the average cluster similarities from sequence segments clustering scatter in a small range. From dataset 1 to dataset 6, the average cluster similarities from sequence segments clustering range in [0.976, 1], [0.969, 1], [0.977, 1], [0.971, 1], [0.973, 1], [0.959, 1] respectively. By structure segments clustering, the average cluster similarities range in [0.806, 1], [0.811, 1], [0.761, 1], [0.750, 1], [0.741, 1], [0.743, 1] respectively. Larger range of average cluster similarities favors the determination of suitable number of clusters in data.

(2) Unique maximum average cluster similarity is found in every dataset no matter using sequence segments or structure segments.

(3) It is more reliable to find the suitable number of clusters using structure segments than using sequence segments. In our experiment, clustering on structure segments works well for all datasets while clustering on sequence segments works for none of the six datasets.

(4) Representation of structure segments needs fewer dimensions than that of sequence segments. For window size 9 in our experiment, the former requires 36 dimensions while the later needs 180 dimensions.

5.3.4 Secondary Structure Similarity Analysis

To verify the effectiveness of our clustering analysis, a biological evaluation criterion is used to evaluate the suitable number of clusters found by our algorithm. The suitable number of clusters for each protein dataset has been found for the clustering of structure segments. For the clustering of sequence segments, the K where the maximum average cluster similarity occurs is adapted as the suitable number of clusters. The biological evaluation criterion is called secondary structure similarity (Sander and Schneider, 1991) which is used in many literatures. For each cluster, the secondary structure similarity of a cluster is given by

$$\text{Secondary Structure Similarity} = \frac{\sum_{i=1}^{ws} \max(p_{i,H}, p_{i,E}, p_{i,C})}{ws} \qquad (5.3)$$

Where *ws* is the window size and $p_{i,H}$ shows the frequency of occurrence of helix among the segments of a cluster at position i. $p_{i,E}$ and $p_{i,C}$ are defined in a similar way.

According to Sander and Schneider (Sander and Schneider, 1991), if the secondary structure similarity of a cluster is larger than 70%, the cluster can be considered structurally identical. In Zhong's work (Zhong et al., 2005), if the secondary structure similarity of a cluster is over 60% and lower than 70%, the cluster can be considered weakly structurally homologous.

According to the definition of secondary structure similarity above, we calculate the secondary structure similarity of all segments of each dataset. It is shown in Table 5.1.

Table 5.1. Secondary structure similarity of each protein dataset

Datasets	**1**	**2**	**3**	**4**	**5**	**6**
# of segments (window size =9)	18797	16138	17866	16365	17454	16490
SSS of all segments	0.392	0.420	0.389	0.397	0.384	0.388

SSS: Secondary Structure Similarity

In Table 5.1, the fist row is the index of six datasets used in the experiment; the second row is the number of structure segments of each dataset with window size 9; the third row is the secondary structure similarity of all structure segments in each dataset. From Table 5.1, we can see that the secondary structure similarity of the six datasets ranges from 0.384 to 0.420.

We do experiments by clustering both structure segments and sequence segments. The average secondary structure similarity of each protein dataset is shown in Table 5.2.

Table 5.2. The average secondary structure similarity of each protein dataset

Datasets		**1**	**2**	**3**	**4**	**5**	**6**
Structure segments	Suitable K	4	7	12	14	9	7
	Arg. SSS	0.616	0.622	0.631	0.615	0.623	0.624
Sequence segments	Suitable K	7	9	8	18	20	20
	Arg. SSS	0.410	0.420	0.398	0.411	0.401	0.407

Arg. SSS: Average Secondary Structure Similarity

In Table 5.2, the fist row is the index of six datasets used in the experiment; the second row is the experimental results using the representation of structure segments; the third row is the experimental results using the representation of sequence segments. Suitable K indicates the suitable number of clusters found in each dataset. Arg.SSS represents the average secondary structure similarity of each dataset.

To obtain the average secondary structure similarity of each dataset, we first cluster the samples in the dataset using the suitable number of clusters which are shown in the row of suitable K. Then the secondary structure similarity of each cluster and the average secondary structure similarity are calculated. From the Table 5.2, we can see that the average secondary structure similarity of each dataset from structure segments clustering is above 0.6. However, clustering sequence segments gives an average secondary structure similarity about 0.4. According to Zhong's work (Zhong et al., 2005), the clusters found by structure segments clustering have meaningful secondary structural similarity. The experimental results show that the representation of structure segments is more effective than the representation of sequence segments. Our algorithm works well with the representation of structure segments for protein structure analysis.

5.3.5 Population Analysis

The total population can be estimated by extracting and tagging a subset of a population, releasing the tagged subset and then estimating the probability of recovering the tagged individual when a new sample is drawn from the population. Since the probability of recovering a tagged individual is:

$$P(Tagged \quad Individual) = \frac{N_{tagged}}{N_{total}} \approx \frac{N_{tagged \ \ in\ sample}}{N_{sample}} \qquad (5.4)$$

Where N_{tagged} is the number of tagged individuals in the total population and the N_{total} is the number of total population; N_{sample} is the number of individuals in a sample and $N_{tagged\ \ in\ \ sample}$ is the number of tagged individuals in a sample. That is, $N_{tagged\ in\ sample}$ is the number of coincidence between N_{tagged} and N_{sample}.

From the above equation 5.4, N_{total} can be derived as:

$$N_{total} = \frac{N_{tagged}}{P(Tagged\ \ Indivudal\ )} \approx \frac{N_{tagged}\, N_{sample}}{N_{tagged\ \ in\ sample}} \tag{5.5}$$

In our experiment, we have found the number of clusters for each protein dataset. Based on the information, we can estimate the range of how many clusters are in a larger or even the whole set of protein structures. The number of clusters in two different datasets can be seen as N_{tagged} and N_{sample} respectively. The number of coincidences between clusters of two different datasets is $N_{tagged\ in\ sample}$.

Consequently, individual estimates for the total population size can be made by counting the number of coincidences between clusters of different datasets. To determine the coincidence between clusters, we need to know the distance between clusters of different datasets. RMSD (Root Mean Square Deviation) is a widely-used measure of difference between structures. In this experiment, we use it to measure the distance of two clusters. The smaller the RMSD between any two clusters, the closer the two clusters is. Since the size of different clusters is different and the individual matching information between two clusters is unknown, the RMSD is derived from the centroids of clusters. The definition of RMSD is given by:

$$\mathrm{RMSD(U,V)} = \sqrt{\frac{\sum_{i=1}^{n}(U_i - V_i)^2}{n}} \tag{5.6}$$

Where U,V are the centroids of the two clusters. In this experiment, U and V are 36-dimension vectors. So n is 36.

The RMSD between any two clusters of different datasets is shown in Table 5.3-

5. 17. It is measured in unit Å. In each table, the clusters indexes of two datasets are in the first column and top two rows (in grey shade). The smallest distance from one dataset's cluster to all clusters in another dataset is underlined. The two clusters with the underlined RMSD distance are believed to matching each other in relative to the two datasets. Since different datasets may have different number of clusters in them, the matching of clusters is always from the dataset with fewer clusters to the dataset with more clusters.

Table 5.3. RMSD between any two clusters of datasets 1 and 2.

Dataset 2	Dataset 1			
	1	2	3	4
1	**2.127**	14.790	29.726	14.771
2	20.961	7.698	12.137	15.397
3	31.916	20.953	**2.378**	21.238
4	11.119	13.792	22.459	**3.721**
5	21.466	14.565	10.385	8.956
6	7.577	17.159	32.476	15.812
7	10.090	**5.105**	23.336	13.623

Table 5.4. RMSD between any two clusters of datasets 1 and 3.

Dataset 3	Dataset 1			
	1	2	3	4
1	**2.390**	15.278	30.225	15.208
2	15.572	17.892	23.257	5.490
3	14.002	10.159	16.660	**4.907**
4	6.230	10.942	24.189	8.601
5	34.452	23.474	4.933	23.570
6	7.845	8.969	24.813	13.521
7	9.088	18.942	32.121	14.157
8	23.897	16.762	9.308	11.139
9	13.719	**4.921**	23.232	16.247
10	20.361	6.967	13.027	15.288
11	9.683	14.991	32.370	18.483
12	27.380	15.926	**3.115**	17.725

Table 5.5. RMSD between any two clusters of datasets 1 and 4.

Dataset 4	Dataset 1			
	1	2	3	4
1	7.314	9.393	25.251	13.661
2	34.255	23.061	4.787	23.643
3	9.831	15.903	33.274	19.201
4	**2.390**	15.358	30.365	15.324
5	17.924	**4.017**	17.051	15.717
6	28.355	18.364	**2.524**	17.053
7	11.545	18.360	28.299	9.469
8	8.529	19.258	33.576	16.078
9	14.631	7.593	15.682	7.376
10	15.928	15.373	19.379	**2.693**
11	24.879	12.252	7.710	17.156
12	6.934	12.402	24.050	8.614
13	22.481	15.623	10.088	9.727
14	8.021	9.723	24.851	11.013

Table 5.6. RMSD between any two clusters of datasets 1 and 5.

Dataset 5	Dataset 1			
	1	2	3	4
1	10.093	8.929	19.914	6.073
2	16.442	16.326	20.036	**3.542**
3	33.654	22.558	**4.138**	22.953
4	9.675	18.587	30.687	12.269
5	9.416	15.736	32.796	18.424
6	22.384	9.313	10.624	15.991
7	12.317	**4.011**	22.674	14.990
8	**1.937**	14.670	29.677	14.706
9	24.187	16.065	7.536	12.014

Table 5.7. RMSD between any two clusters of datasets 1 and 6.

Dataset 6	Dataset 1			
	1	2	3	4
1	**2.020**	14.620	29.569	14.617
2	22.583	9.475	10.503	16.153
3	21.743	15.011	10.506	9.029
4	11.192	**3.870**	22.233	13.525
5	32.627	21.685	**3.083**	21.840
6	10.991	13.946	22.807	**4.058**
7	7.956	17.030	32.585	16.228

Table 5.8. RMSD between any two clusters of datasets 2 and 3.

Dataset 3	Dataset 2						
	1	2	3	4	5	6	7
1	**0.578**	21.662	32.489	12.355	22.233	9.712	11.317
2	16.374	20.372	25.399	**5.225**	12.968	15.940	17.318
3	14.515	12.037	18.917	6.879	8.175	16.452	11.327
4	6.817	16.387	26.523	6.253	15.668	10.389	8.525
5	34.518	16.599	**2.584**	27.114	14.758	37.279	28.125
6	8.269	15.636	27.006	12.130	18.408	11.892	5.548
7	10.864	25.074	34.408	10.657	22.822	**4.460**	15.183
8	24.126	13.201	11.223	14.821	**2.630**	26.368	20.011
9	14.438	11.761	25.393	15.808	18.870	15.731	**4.315**
10	20.586	**0.931**	15.066	17.193	12.276	23.446	12.026
11	11.478	22.461	34.650	15.868	25.191	6.366	10.277
12	27.508	9.166	5.092	20.971	9.835	30.297	20.624

Table 5.9. RMSD between any two clusters of datasets 2 and 4.

Dataset 4	Dataset 2						
	1	2	3	4	5	6	7
1	7.659	16.058	27.452	12.138	18.732	11.692	**5.843**
2	34.315	16.092	**2.440**	27.144	14.907	37.126	27.766
3	11.600	23.416	35.549	16.488	26.037	6.296	11.111
4	**0.696**	21.767	32.631	12.455	22.370	9.664	11.366
5	18.237	5.067	19.103	16.782	14.931	20.905	8.793
6	28.498	12.410	4.320	20.611	8.244	31.079	22.657
7	12.825	23.036	30.549	6.298	18.298	10.072	15.949
8	10.352	25.827	35.877	12.566	24.578	**2.956**	15.077
9	15.216	9.547	17.964	9.120	8.731	16.966	9.627
10	16.535	16.869	21.551	**5.149**	9.078	17.362	15.796
11	25.024	**4.809**	9.542	19.905	11.040	27.925	17.232
12	7.150	17.363	26.297	6.503	15.491	11.413	10.304
13	22.754	12.593	12.133	13.408	**1.195**	24.889	18.707
14	8.867	15.599	27.216	9.159	17.123	10.799	7.169

Table 5.10. RMSD between any two clusters of datasets 2 and 5.

Dataset 5	Dataset 2						
	1	2	3	4	5	6	7
1	10.631	13.093	22.218	5.772	11.672	13.274	8.492
2	17.089	17.825	22.174	**5.543**	9.725	17.588	16.636
3	33.724	15.656	**1.776**	26.463	14.203	36.499	27.231
4	11.304	24.239	32.970	8.765	21.090	6.392	15.286
5	11.281	23.165	35.075	15.673	25.397	**5.390**	11.009
6	22.570	**1.681**	12.591	18.325	11.481	25.454	14.356
7	12.969	11.567	24.860	14.506	17.890	14.852	**2.378**
8	**0.239**	21.082	31.946	11.886	21.707	9.450	10.708
9	24.407	11.911	9.564	15.651	**3.070**	26.750	19.652

Table 5.11. RMSD between any two clusters of datasets 2 and 6.

Dataset 6	Dataset 2						
	1	2	3	4	5	6	7
1	**0.214**	21.001	31.837	11.814	21.599	9.549	10.696
2	22.775	**1.871**	12.449	18.503	11.555	25.625	14.527
3	22.043	12.304	12.601	12.705	**0.513**	24.116	17.999
4	11.898	11.549	24.462	12.966	16.873	13.835	**1.286**
5	32.707	14.881	**0.755**	25.356	13.082	35.465	26.313
6	11.894	17.750	25.081	**0.495**	12.961	12.153	12.740
7	10.011	24.020	34.895	13.002	24.177	**0.895**	12.571

Table 5.12. RMSD between any two clusters of datasets 3 and 4.

Dataset 4	Dataset 3											
	1	2	3	4	5	6	7	8	9	10	11	12
1	8.14	16.91	11.63	7.836	29.96	**0.75**	13.59	21.20	9.917	15.50	11.06	22.74
2	34.81	27.75	21.28	28.91	**0.93**	29.20	36.77	13.51	27.43	17.00	36.87	7.19
3	11.70	20.46	18.64	12.97	38.05	11.53	10.69	28.54	13.05	22.68	**1.21**	30.74
4	**0.19**	16.78	15.16	7.361	35.15	8.838	10.95	24.72	14.86	21.18	11.55	28.15
5	18.70	20.42	12.50	14.27	21.54	12.67	22.64	16.73	**6.96**	4.212	18.44	14.10
6	28.99	21.12	15.07	22.80	6.539	23.91	30.46	6.973	22.89	13.24	31.37	**4.26**
7	13.04	6.794	12.84	9.558	32.87	14.94	6.76	20.32	19.12	22.66	15.11	26.87
8	10.38	15.52	17.21	11.27	38.33	13.71	**2.62**	26.98	18.23	25.25	9.210	31.55
9	15.74	12.63	**3.22**	9.729	20.46	10.47	17.02	11.30	11.61	9.437	17.29	13.51
10	16.93	**4.04**	7.317	10.45	23.77	15.78	15.32	10.80	18.25	16.81	20.33	18.44
11	25.52	22.05	13.96	19.91	11.90	19.55	28.45	11.57	16.45	**5.71**	26.77	4.734
12	7.583	11.39	8.130	**3.79**	28.76	8.403	11.03	17.88	13.95	16.99	13.94	22.13
13	23.22	13.38	9.288	16.64	14.27	19.52	23.69	**1.53**	19.90	13.05	26.25	9.87
14	9.242	13.21	11.12	4.688	29.77	9.143	12.02	19.39	9.415	14.91	11.09	22.56

Table 5.13. RMSD between any two clusters of datasets 3 and 5.

Dataset5	Dataset 3											
	1	2	3	4	5	6	7	8	9	10	11	12
1	11.14	10.57	**4.06**	4.93	24.73	8.18	13.12	14.21	11.64	12.76	14.66	17.86
2	17.47	**3.29**	8.24	11.13	24.35	16.55	15.33	11.30	19.11	17.77	20.77	19.20
3	34.22	27.05	20.63	28.28	**0.97**	28.66	36.11	12.81	26.95	16.56	36.30	6.67
4	11.45	10.75	14.51	9.78	35.36	14.05	**2.56**	23.32	18.55	23.77	12.19	28.97
5	11.40	19.62	17.94	12.40	37.57	11.28	9.72	27.91	13.14	22.45	**1.42**	30.31
6	23.07	20.95	12.68	17.64	15.00	17.06	26.29	12.56	13.40	**2.56**	23.99	7.64
7	13.40	18.92	12.38	10.55	27.36	7.35	17.40	20.18	**2.53**	10.79	11.75	19.85
8	**0.73**	16.31	14.44	6.73	34.47	8.13	10.73	24.08	14.29	20.50	11.28	27.45
9	24.88	15.92	10.78	18.44	11.73	20.62	25.79	**2.09**	20.53	12.52	27.71	7.56

Table 5.14. RMSD between any two clusters of datasets 3 and 6.

Dataset6	Dataset 3											
	1	2	3	4	5	6	7	8	9	10	11	12
1	**0.76**	16.24	14.34	6.65	34.36	8.12	10.80	23.97	14.28	20.42	11.38	27.34
2	23.27	21.12	12.81	17.85	14.84	17.24	26.47	12.61	13.54	**2.74**	24.14	7.51
3	22.51	12.86	8.47	15.93	14.79	18.77	22.95	**2.34**	19.28	12.73	25.48	10.08
4	12.36	17.48	10.94	9.12	26.99	6.18	16.12	19.24	**3.82**	10.82	11.36	19.48
5	33.20	25.93	19.57	27.22	**1.85**	27.72	35.04	11.70	26.11	15.78	35.34	5.82
6	12.28	**5.21**	7.16	6.32	27.45	12.11	10.25	15.18	15.86	17.46	15.61	21.31
7	10.11	16.51	16.68	10.78	37.38	11.87	**5.17**	26.65	15.44	23.38	5.64	30.36

Table 5.15. RMSD between any two clusters of datasets 4 and 5.

Dataset 4	Dataset 5								
	1	2	3	4	5	6	7	8	9
1	8.345	16.650	29.117	13.882	11.216	17.491	7.702	7.524	20.984
2	24.587	24.539	**2.719**	35.352	37.315	14.478	26.959	34.269	11.921
3	15.368	21.479	37.204	12.389	**1.164**	24.952	12.614	11.408	28.578
4	11.279	17.574	34.364	11.458	11.347	23.180	13.448	**0.814**	25.029
5	12.068	18.475	20.601	22.173	19.298	6.703	**7.153**	18.134	15.778
6	18.515	17.889	5.970	28.931	31.732	11.077	22.215	28.451	5.264
7	11.215	9.488	32.252	**4.222**	14.599	24.004	17.992	12.737	21.343
8	14.220	17.526	37.590	5.051	8.144	27.139	17.274	10.221	27.503
9	5.069	10.550	19.667	16.029	17.519	10.329	10.237	15.119	10.867
10	8.586	**1.081**	23.232	13.171	20.274	17.326	17.097	16.481	12.078
11	16.075	19.126	10.956	27.400	27.390	**3.152**	16.177	24.955	9.872
12	**4.892**	11.355	28.028	10.361	13.730	18.486	12.311	7.116	18.218
13	12.759	10.138	13.774	21.899	26.463	12.117	18.972	22.708	**2.594**
14	7.680	13.236	28.961	11.488	11.584	17.016	8.818	8.745	19.708

Table 5.16. RMSD between any two clusters of datasets 4 and 6.

Dataset 4	Dataset 6						
	1	2	3	4	5	6	7
1	7.519	17.680	19.097	**6.561**	28.174	12.119	11.708
2	34.160	14.319	14.970	26.622	**1.834**	27.490	37.216
3	11.522	25.100	26.328	12.220	36.247	16.224	5.554
4	**0.866**	23.385	22.653	12.422	33.346	12.380	10.063
5	18.076	6.850	15.390	7.598	19.824	16.988	20.768
6	28.341	10.996	8.305	21.625	4.850	20.960	31.232
7	12.725	24.190	18.323	16.564	31.147	5.886	10.751
8	10.310	27.314	24.747	16.126	36.537	12.194	**3.624**
9	15.039	10.453	9.123	8.969	18.649	9.352	17.059
10	16.391	17.490	9.016	15.703	22.111	**5.429**	17.833
11	24.862	**2.993**	11.422	16.033	10.244	20.228	27.922
12	7.011	18.657	15.750	10.793	26.976	6.597	11.830
13	22.598	12.185	**0.852**	17.976	12.655	13.764	25.174
14	8.711	17.236	17.413	7.815	27.930	9.183	11.013

Table 5.17. RMSD between any two clusters of datasets 5 and 6.

Dataset 5	Dataset 6						
	1	2	3	4	5	6	7
1	10.458	14.318	11.985	8.477	22.908	5.993	13.530
2	16.949	18.420	9.620	16.577	22.715	**5.761**	18.080
3	33.569	13.901	14.263	26.096	**1.127**	26.808	36.595
4	11.224	25.557	21.184	16.117	33.597	8.341	7.114
5	11.196	24.823	25.678	12.107	35.768	15.395	**4.612**
6	22.407	**0.314**	11.916	13.150	13.304	18.624	25.421
7	12.808	13.361	18.311	**1.695**	25.589	14.562	14.621
8	**0.161**	22.696	21.994	11.750	32.660	11.816	9.833
9	24.250	11.155	**3.072**	18.797	10.094	16.007	26.984

Only if the distance between two clusters from different datasets is less than certain threshold, the two clusters are called a coincidence. In protein structure similarity study, structures with distance no more than 1 or 2 angstroms are considered to be similar structures. Therefore, we use the same threshold in the determination of coincidence of

clusters. Table 5.18 shows the number of coincidences between different datasets when clusters are defined to be identical when the RMSD between them is less than 1Å. Table 5.19 displays the number of coincidences the number of coincidences between different datasets with a criterion of RMSD less than 2Å.

Table 5.18. The number of coincidences between clusters (less than 1 Å)

Datasets	**1**	**2**	**3**	**4**	**5**	**6**
1	4	0	0	0	0	0
2		7	2	1	1	5
3			12	3	2	1
4				14	1	2
5					9	2
6						7

Table 5.19. The number of coincidences between clusters (less than 2 Å)

Datasets	**1**	**2**	**3**	**4**	**5**	**6**
1	4	0	0	0	1	0
2		7	2	2	3	7
3			12	5	3	2
4				14	3	3
5					9	4
6						7

Comparing Table 5.18 and Table 5.19, we can see that there are more coincidences with the looser criteria (less than 2 Å), and therefore there will be a lower total population size when the looser criterion is used.

According to equation 5.5, we now calculate the N_{total} for each pair of different

datasets. Let's take the dataset 2 and 3 as an example. N_{tagged} is 7. N_{sample} is 12. $N_{tagged\ in\ sample}$ is 2 in Table 5.18. N_{total}=7*12/2=42. Computing in this way, we get the total estimate from each pair of datasets. Finding the mean and standard deviation of these individual estimates determines a statistical confidence interval. The mean value of total estimate is 62 and the standard deviation is 30.8 with the 1Å criterion. Thus the three-sigma estimate for the maximum number of 9-mers is 154. When the larger 2Å criterion is used the mean value of total estimate is 32.5 with a standard deviation of 12.7. Thus the three-sigma estimate for the maximum number of 9-mers is 71.

For comparison, Zhong et al (Zhong et al., 2005) found 211-253 clusters of 9-mers with >60% secondary structure similarity and 80-92 clusters with >70% secondary structure similarity when using different K-means algorithms. Thus our statistical estimates are consistent with direct observation.

5.4 Discussion and Conclusion

Clustering algorithms have been successfully used to study the sequence-structure correspondence for protein secondary structure prediction problem (Chen et al., 2006). They use sliding window method to separate protein sequences into sequences segments. Through grouping the similar sequence segments into clusters, consensus sequence segments and structures could be extracted from each cluster, which are used for the prediction of protein secondary structure. In this chapter, we present a new descriptor for the representation of protein structures and an algorithm for detecting stable clustering. To test our algorithm and descriptor, six datasets of protein three dimensional structures are randomly drawn from PDB. The algorithm converges in each case to a unique set of

stable clusters. Since these clusters are drawn randomly from the total current set of chains, counting the number of coincidences and using basic sampling theory provides a rigorous statistical estimate of the number of unique clusters in the dataset. The effectiveness of our algorithm with the new descriptor is verified by the secondary structure similarity analysis. The population estimates derived from sampling are consistent with the direct observation in Zhong's wrok (Zhong et al., 2005). Our algorithm, experiments results, and conclusions are to be published in paper (Fu et al., 2007).

Chapter 6

Conclusions and Future Work

In this dissertation, we explore the pattern analysis on RNA and protein structures using term rewriting and clustering algorithm. Our work can be summarized as two major parts. In the first part, we make a new implementation for the computation of concurrent interaction of RNA secondary structure motifs. This application is helpful for better understanding of the relationship between the RNA secondary structures and pseudoknots. The method propose in this dissertation can be used to model more complex RNA tertiary interactions. In the second part, we present an improved K-means clustering algorithm for structure pattern analysis of both RNA and protein three dimensional structures.

In regard to the specific area considered in this research, a possible way for continued work would be in RNA structure database search. RNA structure database search involves two major tasks: one is to search similar structures against RNA structure database; the other is the substructure search. The HFSS and their similarities uncovered using our improved K-means clustering algorithm from the RNAs with known structures can be used for rapid RNA structure database search. That is, given a structure, to find the structures which contain the given structure as substructures. Since the HFSS and their similarities information are available, an RNA three dimensional structure can be converted into a string of HFSS and the similarity between any two strings of HFSS can be obtained with ease. Combined with the process of uncovering high frequent structure segments and their similarities, we give the framework of rapid RNA structure database

search. It is shown in Figure 6.1.

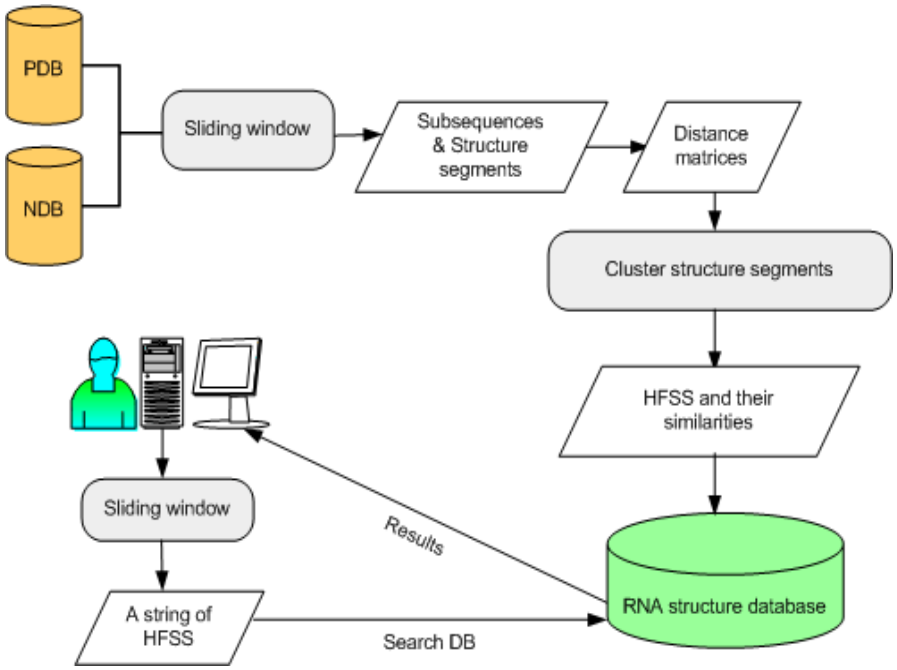


Figure 6.1. The framework of rapid RNA structure database search.

As it is shown in the Figure 6.1, the framework is composed of a serial of steps. The purpose and function of each step is explained as follows:

Data collection

Data collection is the first step in this work. To date, the PDB (Berman et al., 2000) and NDB (Berman et al., 1993) serve as a repository for raw structural data of molecules, including RNA. However, neither of these databases contains a complete representation of all available RNA structures. What is more, the data in them is in essentially raw form with few annotations and structures in PDB files often have exceptions like missing data

in sequence or missing data in the structure. Therefore manual work is necessary to select the appropriate data for study.

Structure segments using sliding window

After collecting the RNA three dimensional structures, we begin considering how to extract hidden structural patterns from the currently available RNA three dimensional structures. We have to transform the data into an appropriate representation. A structure is split into a set of structure segments by sliding window method. These structure segments are the targets that our algorithm will work on.

Distance matrix description

The structure segments are in three dimensional. To facilitate the calculation of similarity between two structure segments, we use a distance matrix to describe a structure segment. The distance matrix contains the Euclidean distances between any two nucleic acids in a window.

Improved K-means clustering

The improved K-means clustering algorithm is introduced in detail in Chapter 3. It is responsible for grouping structure segments into a suitable number of clusters. HFSS are extracted from the clusters.

Database setup

With the HFSS and their similarities available, we can set up a RNA structure database. It contains the RNA sequences, three dimensional structures, and strings of HFSS of each RNA three dimensional structure. The database will be used for substructure search and structure similarity evaluation, analysis of RNA structures, and mining new RNA tertiary motifs.

Rapid substructure search and structure similarity evaluation

Given a structure, we first convert it into a string of HFSS. By calculating the similarity between the given string and the strings of HFSS in the database, we can quickly filter out the structures similar to the given structure. RNA sequences are also determent to its structure and function. So the similarity of RNA sequences and related heuristic knowledge can be integrated into the final similarity evaluation for better performance.

Bibliography

Berman, H.M., Westbrook, J., Feng, Z., Gilliland, G., Bhat, T.N., Weissig, H., Shindyalov, I.N. and Bourne, P.E., "The Protein Data Bank". Nucleic Acids Research, 2000. 28: p. 235–242.

Berman, H.M., Olson, W.K., Beveridge, D.L., Westbrook, J., Gelbin, A., Demeny, T., Shieh, S.H., Srinivasan, A.R. and Schneider, B., The nucleic acid database. "A comprehensive relational database of three-dimensional structures of nucleic acids". Biophysical Journal, 1993. 63 p. 751–759.

Baader, F. and Nipkow, T., "Term Rewriting and All That". Cambridge, England: Cambridge University Press, 1999.

Clavel, M., F.D., Eker, S., Lincoln, P., Martí-Oliet, N., Meseguer,J. and Talcott,C., "The Maude 2.0 System". In Proc. Rewriting Techniques and Applications. Springer-Verlag LNCS 2003. 2706: p. 76-87.

Jain, A.K., Murty, M.N., Flynn, P.J., "Data clustering: a review". ACM Computing Surveys, 1999. 31(3): p. 264-323.

Batey, R.T., Rambo, R.P., Doudna, J.A., "Tertiary Motifs in RNA Structure and Folding". Angewandte Chemie International Edition English, 1999. 38(16): p. 2326-2343.

Creighton, T. E., "Proteins: Structures and Molecular Properties (Second Edition)". New York: Freeman, 1993.

Branden, C. and Tooze, J., "Introduction to protein structure". New York: Garland Publishing Inc., 1998.

Hand, D.J., Mannila, H., Smyth, P., "Principles of Data Mining". The MIT Press, 2001.

Batenburg, F., Gultyaev, A.P., Pleij,C.W.A., Ng,J. and Oliehoek,J., "Pseudobase: a database with RNA pseudoknots". Nucleic Acids Research, 2000. 28(1): p. 201-204.

Saenger, W., "Principles of Nucleic Acid Structure". C.R. Cantor, Ed. Springer Advanced Texts in Chemistry. Springer-Verlag, New York, 1984.

Staple, D.W., and Butcher, S.E., "Pseudoknots: RNA Structures with Diverse Functions". PLoS Biology, 2005. 3(6).

Chastain, M. and Tinoco Jr., I., "Structural elements in RNA". Progress in Nucleic Acid Research and Molecular Biology, 1991, 41:131-177.

Lyngsø, R.B. and Pedersen, C.N.S., "RNA pseudoknot prediction in energy-based models". Journal of Computational Biology, 2000. 7: p. 409–427.

MARTI-OLIET, N., and Meseguer, J., "Rewriting Logic: Roadmap and Bibliography". Theoretical Computer Science, Elsevier 2002. 285(2): p. 121-154.

Robinson, J.A. and Voronkov, A., "Handbook of Automated Reasoning (in 2 volumes)". Elsevier and MIT Press, 2001.

Dershowitz., N., "Functional Programming, Concurrency, Simulation and Automated Reasoning". Lecture Notes in Computer Science, Springer. , 1993. 693: p. 199-228.

Eker,S., Knapp, M., Laderoute,K., Lincoln,P., Talcott,C., "Pathway Logic: Executable Models of Biological Networks". Electronic Notes in Theoretical Computer Science, 2002. 71.

Talcott, C.L., Eker, S., Knapp, M., Lincoln, P., Laderoute,K., "Pathway Logic Modeling of Protein Functional Domains in Signal Transduction". Pacific Symposium on Biocomputing, 2004: 568-580.

Eker, S., "Fast matching in combination of regular equational theories". In José Meseguer, editor, Proceedings First International Workshop on Rewriting Logic and its Applications, volume 4 of Electronic Notes in Theoretical Computer Science, Elsevier. , 1996: p. 90-109.

Zuker. M., "Mfold web server for nucleic acid folding and hybridization prediction". Nucleic Acids Research, 2003. 1;31(13): p. 3406-15.

Hofacker, I.L., Fontana,W., Stadler,P.F., Bonhoeffer,S., Tacker,M. and Schuster,P., "Fast folding and comparison of RNA secondary structures". Monatsh. Chem. 1994. 125: p. 167-188.

Hofacker, I.L., "Vienna RNA secondary structure server". Nucleic Acids Research, 2003 31(13): p. 3429-31.

Ruan, J., Stormo, G.D., Zhang, W., "An Iterated loop matching approach to the prediction of RNA secondary structures with pseudoknots". Bioinformatics, 2004. 20(1): p. 58-66.

Fu, X.Z., Wang, H., Harrison, R., and Harrison, W., "RNA Pseudoknot Prediction using Term Rewriting". The Fifth IEEE Symposium on Bioinformatics & Bioengineering, 2005.

MacQueen, J.B., "Some Methods for classification and Analysis of Multivariate Observations". Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and

Probability, Berkeley, University of California Press, 1967. 1: p. 281-297.

Ben-Hur, A., Elisseeff, A., and Guyon, I., "A stability based method for discovering structure in clustered data". Pacific Symposium on Biocomputing, 2002.

Bryan, J., "Problems in gene clustering based on gene expression data". Journal of Multivariate Analyis, 2004. 90: p. 67-89.

Dudoit, S. and Fridlyand, J., "A prediction-based resampling method to estimate the number of clusters in a dataset". Genome Biology, 2002. 3: p. 0036.1-0036.21.

Chen, B., Tai, P.C., Harrison, R. and Pan, Y., "FIK model: A Novel Efficient Granular Computing Model for Protein Sequence Motifs and Structure Information Discovery". IEEE BIBE 2006 proceeding, 2006: p. 20-26

DiCiccio T.J. and Efron, B., "Bootstrap confidence intervals". Statistical Science, 1996. 11: p. 189-228.

DiCiccio, T.J. and Romano, J., "A review of bootstrap confidence intervals". Journal of the Royal Statistical Society, Series B 50, 1988: p. 338–370.

Young, G.A., "Bootstrap: more than a stab in the dark?". Statistical Science. 1994. 9: p. 382–415.

Xu, R. and Wunsch, D., "Survey of clustering algorithms". IEEE Transactions on Neural Networks, 2005. 16:33, 645-678.

Davison, A.C. and Hinkley, D.V., "Bootstrap Methods and their Application". Cambridge University Press, 1997.

Jain, A.K., and Dubes,R.C., "Algorithms for Clustering Data". Prentice Hall, 1988.

Maulik, U. and Bandyopadhyay, S., "Performance Evaluation of Some Clustering Algorithms and Validity Indices". IEEE Transanction on Pattern Analysis and Machine Intelligence, 2002. 24(12): p. 1650-1654.

BelMufti, G., P.B., and ElMoubarki, L., "Determining the Number of Groups from Measures of Cluster Validity". Proceedings of International Symposium on Applied Stochastic Models and Data Analysis, 2005: p. 404-414.

Tibshirani, R., Walther, G., and Hastie,T., "Estimating the Number of Clusters in a Data Set via Gap Statistic". Journal of the Royal Statistical Society. Series B (Statistical Methodology), 2001. 63(2): p. 411-423.

Lange, T., Roth, V., Braun, M.L. and Buhmann, J.M., "Stability-Based Validation of Clustering Solutions". Neural Computation, 2004. 16: p. 1299-1323.

Fowlkes, E.B. and Mallows, C.L., “A method for comparing two hierarchical clusterings”. Journal of the American Statistical Association, 1983. 78: p. 553–569.

Law, M.H. and Jain, A.K., “Cluster Validity by Boostrapping Partitions”. Technical Report MSU-CSE-03-5, Michigan State University, 2003.

Levine, E. and Domany, E., “Resampling Method for Unsupervised Estimation of Cluster Validity”. Neural Computation, 2001. 13: p. 2573-2593.

Monti, S., Tamayo, P., Mesirov, J., and Golub, T., “A Resampling Based Method for Class Discovery and Visualization of Gene Expression Microarray Data”. Machine Learning, 2003. 52: p. 91-118.

Matsumoto, M. and Nishimura, T., “Mersenne twister: a 623-dimensionally equidistributed uniform pseudorandom number generator”. ACM Transactions on Modeling and Computer Simulation, 1998. 8(3): p. 3-30.

Matsumoto, M. and Kurita, Y., “Twisted GFSR generators”. ACM Transactions on Modeling and Computer Simulation, 1992. 2: p. 179-194.

Knuth, D.E., “The Art of Computer Programming:Seminumerical Algorithms”.Vol. 2. Third Edition, Addison-Wesley, 1992.

Eddelbuettel, D., “Soft versus Hard: A comparison of random number generators between R, GSL and a non-deterministic generator”. 5th International Workshop on Directions in Statistical Computing, 2007.

Pena J.M., Lozano J.A., Larranaga P., “An Empirical comparison of four initialization methods for the k-means algorithm”, Pattern Recognition Letters, 1999, 20: p. 1027-1040.

Matlab, http://www.mathworks.com/products/matlab/ The MathWorks, Inc.1994.

Pyle, A.M., “Ribozymes—a distinct class of metalloenzymes”. Science, 1993 261: p. 709-714.

Lilley, D.M., “Folding and catalysis by the hairpin ribozyme”. FEBS Letters, 1999. 452: p. 26–30.

Hanna, R. and Doudna, J.A., “Metal ions in ribozyme folding and catalysis”. Current Opinion in Chemical Biology, 2000. 4: p. 166–170.

DeRose, V.J., “Metal ion binding to catalytic RNA molecules”. Current Opinion in Chemical Biology, 2003. 13: p. 317–324.

Nagaswamy, U., Larios-Sanz, M., Hury, J., Collins, S., Zhang, Z., Zhao, Q., Fox, G., "NCIR: a database of non-canonical interactions in known RNA structures". Nucleic Acids Research, 2002. 30 p. 395-397.

Duarte, C.M., Wadley, L.M., and Pyle, A.M., "RNA structure comparison, motif search and discovery using a reduced representation of RNA conformational space". Nucleic Acids Research. 2003 15;31(16): p. 4755-61.

Tabaska, J., Cary,R., Gabow, H., Stormo,G., "An RNA folding method capable of identifying pseudoknots and base triples". Bioinformatics, 1998, 14(8): p. 691-699.

Leontis, N.B., Westhof, E., "Geometric nomenclature and classification of RNA base pairs". RNA, 2001. 7(4): p. 499-512.

Zhang, Y. and Skolnick, J., "The protein structure prediction problem could be solved using the current PDB library". Proceedings of the National Academy of Sciences, USA, 2005. 102(4): p. 1029-34.

Rost, B., "Review: protein secondary structure prediction continues to rise". Journal of Structural Biology.Review, 2001. 134(2-3): p. 204-218.

Petrey, D. and Honig, B., "Protein structure prediction: inroads to biology". Molecular Cell, 2005. 20(6): p. 811-9.

Zhong,W., Altun, G., Harrison, R., Tai, P.C., Pan, Y., "Improved K-Means Clustering algorithm for Exploring Local Protein Sequence motifs Representing Common Structural Property". IEEE transactions on Nanobioscience, 2005. 4(3): p. 255-265.

Han, K.F. and Baker, D., "Recurring Local Sequence Motifs in Proteins". Journal of Molecular Biology, 1995. 251: p. 176-187.

G. Wang and R. L. Dunbrack, J., "PISCES: a protein sequence-culling server". Bioinformatics, 2003. 19(12): p. 1589-1591.

Sander, C. and Schneider, R., "Database of similarity derived protein structures and the structure meaning of sequence alignment". Proteins: Structure, Function, and Genetics, 1991. 9(1): p. 56-68.

Kabsch,W. and Sander, C., "Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features". Biopolymers, 1983. 22: p. 2577–2637.

Lesk, A.M., "Introduction to protein science: architecture, function and genomics". Oxford ; New York : Oxford University Press, 2004.

Fu, X.Z, Chen,B., Pan, Y., and Harrison, R., "Statistical Estimate for the Size of the

Protein Structural Vocabulary". 2007 International Symposium on Bioinformatics Research and Applications. May 7-10, 2007.