

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be “used for any purpose other than private study, scholarship, or research.” If a user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of “fair use” that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select “Pages from: first page # to: last page #” on the print dialog screen

The Van Houten library has removed some of the personal information and all signatures from the approval page and biographical sketches of theses and dissertations in order to protect the identity of NJIT graduates and faculty.

ABSTRACT

DATA MINING OF TETRALOOP-TETRALOOP RECEPTORS IN RNA XML FILES

by

Sinan Ramazanoglu

RNA (Ribonucleic acid) Motifs are tertiary structures that play an important role in the folding mechanism of the RNA molecule. The overall function of a RNA Motif depends on its specific bp (base pairs) sequence that constitutes the secondary structure. Data mining is a novel method in both discovering potential tertiary structures within DNA (Deoxyribonucleic acid), RNA, and protein molecules and storing the information in databases. The RNA Motif of interest is the tetraloop-tetraloop receptor, which is composed of a highly conserved 11 nt (nucleotide) sequence and a tetraloop with the generic form of GNRA (where N = any base and R = A or U). There consists of eight variations of the tetraloop in the tertiary structure of the RNA motif.

The initial procedure in data mining the tetraloop-tetraloop receptor is to search for all potential PDB (Protein Data Bank) files that contain the particular RNA Motif. Once all the PDB files have been allocated, each individual file will need to be processed by RNAView to convert them to Xml format. Xml files are excellent in that they give a great in-depth summary of the residue bp and position within the molecule. Parsing each Xml file by a custom written, advanced python script will give a text file output pertaining to in which PDB files may or may not contain the full tertiary structure of the

tetraloop-tetraloop receptor. The results lead to the GAAA tetraloop as well as the highly conserved 11 nt sequence which together form a tertiary structure, residing in nine PDB files.

**DATA MINING OF TETRALOOP-TETRALOOP RECEPTORS
IN RNA XML FILES**

by
Sinan Ramazanoglu

**A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Masters of Science in Bioinformatics**

Department of Computer Science

May 2012

Blank Page

APPROVAL PAGE

**DATA MINING OF TETRALOOP-TETRALOOP RECEPTORS
IN RNA XML FILES**

Sinan Ramazanoglu

Dr. Jason T.L. Wang, Thesis Advisor
Professor of Bioinformatics and Computer Science, NJIT

Date

Dr. Usman W. Roshan, Committee Member
Associate Professor of Bioinformatics and Computer Science, NJIT

Date

Dr. Zhi Wei, Committee Member
Assistant Professor of Bioinformatics and Computer Science, NJIT

Date

BIOGRAPHICAL SKETCH

Author: Sinan Ramazanoglu

Degree: Masters of Science

Date: May 2012

Undergraduate and Graduate Education:

- Masters of Science in Bioinformatics
New Jersey Institute of Technology, Newark, NJ 07102
- Bachelor of Science in Bioinformatics
New Jersey Institute of Technology, Newark, NJ 07102

Major: Bioinformatics

Dedicated to
Professor Jason TL Wang
And myself for all the hard work I put into it

ACKNOWLEDGEMENTS

I would like to thank Professor Jason Wang, my thesis advisor who has been a guiding me throughout and without whose support this thesis would have never been possible. I would also like to thank Dr. Usman W. Roshan and Dr. Zhi Wei for being part of my thesis committee. My sincere appreciation goes to my fellow student Andrew Roberts for coding this algorithm in Python and without it I would have been able to complete my thesis.

TABLE OF CONTENTS

Chapter	Page
1 INTRODUCTION.....	1
1.1 What is a RNA Motif.....	1
1.2 Types of RNA Motifs and Tertiary Structure.....	2
1.3 What is a Tetraloop-Tetraloop Receptor.....	3
1.4 Base Pair Interaction of Tetraloop-Tetraloop Receptor.....	4
2 MATERIALS AND METHODS.....	6
2.1 Data Mining Process.....	6
2.2 Data Extraction.....	7
2.3 PDB Files Conversion to XML Files.....	9
2.4 Tetraloop-Tetraloop Receptor Search Methodology in XML Files.....	12
2.5 Sequence in Context.....	16
2.5.1 SIC Patterns Part I.....	16
2.5.2 SIC Patterns Part II.....	17
2.6 Data Processing for Motif Identification.....	18
2.7 Program Output Details.....	19
3 EXPERIMENTS.....	21
3.1 Finding RNA 3D (FR3D).....	21
3.2 Visual Molecular Dynamics (VMD).....	25
4 DISCUSSION.....	27
4.1 Discussion and Further Studies.....	27

TABLE OF CONTENTS

(Continued)

Chapter	Page
APPENDIX A PYTHON SCRIPT.....	29
APPENDIX B PROGRAM OUTPUT.....	42
REFERENCES.....	44

LIST OF TABLES

Table	Page
1.1 Geometric Base-Pairing Families.....	2
3.1 PDB ID and Starting Position of Tertiary Structure Location.....	22

LIST OF FIGURES

Figure	Page
1.1 VMD image of the RNA structure from PDB ID: 1GID.....	1
1.2 Pairs of interacting edges.....	3
1.3 Tetraloop-Tetraloop Receptor.....	4
1.4 Base pair interaction between tetraloop and tetraloop-receptor.....	5
2.1 Tetraloop-Tetraloop Receptor search on pdb.org.....	8
2.2 Tetraloop-Tetraloop Receptor at the Primary structure level.....	10
2.3 Tetraloop hairpin loop at the Secondary structure level.....	10
2.4 Receptor stem/internal loop at the Secondary structure level.....	10
2.5 Tetraloop-Tetraloop Receptor Tertiary structure with hydrogen bond interaction.....	11
2.6 Grouping count of SIC occurrences.....	16
2.7 Statistics of occurrences at the primary, secondary and tertiary level.....	18
3.1 FR3D web interface.....	23
3.2 3D image of PDB ID: 1GID.....	23
3.3 3D structure of PDB ID: 1GID with the tertiary structure of the Tetraloop- Tetraloop Receptor bolded.....	25
3.4 3D structure of the Tetraloop-Tetraloop Receptor.....	26
3.5 3D structure of the Tetraloop-Tetraloop Receptor with a variation of style/format.....	26

CHAPTER 1

INTRODUCTION

1.1 What is a RNA Motif

RNA motifs play an important role in RNA folding and other significant biochemical functions. They are tertiary structures within the RNA molecule; their small motif segments interact strongly with its environment. These motifs control the release of information in the cell from the RNA molecule itself. Functions of the motifs depend on a conserved bp (base pairing) sequence of the secondary structure. Specific functions include protein binding, base-pairing to other RNA molecules, and modifying nucleic acid bonds.

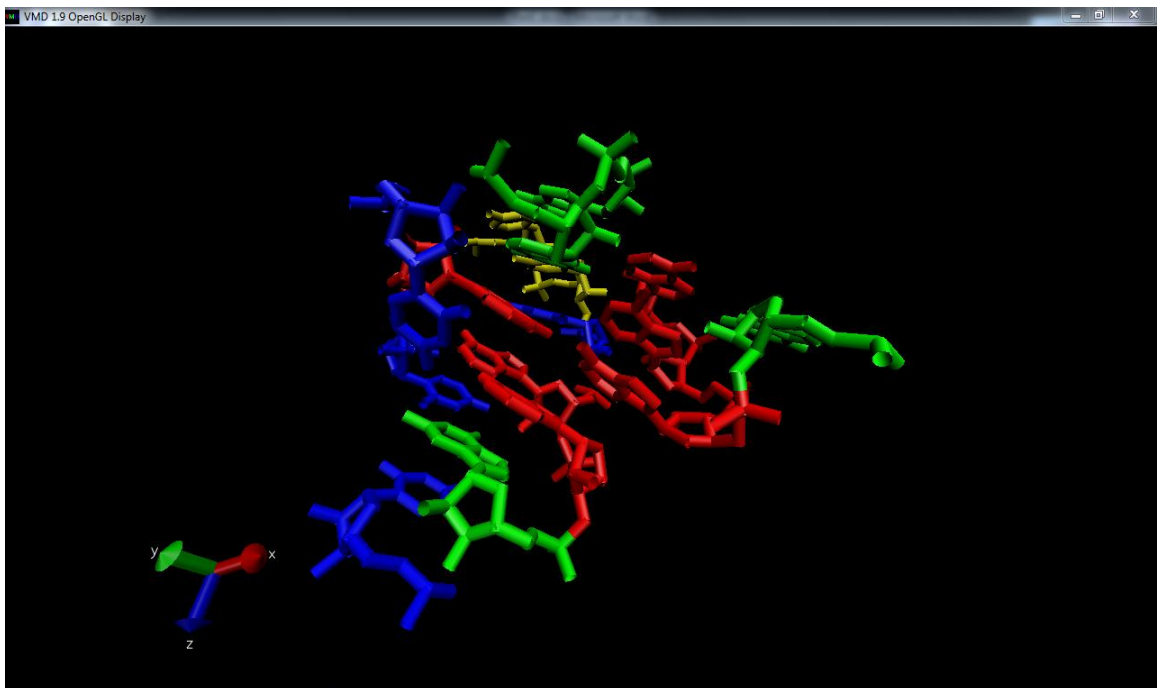


Figure 1.1 VMD image of the RNA motif structure from PDB ID: 1GID [13].

1.2 Types of RNA Motifs and Tertiary Structure

In total there are seven distinct RNA motifs: A-minor, Coaxial helix, Ribose Zipper, Pseudoknot, Kissing hairpin, tRNA D-loop/T-loop, and the Tetraloop-Tetraloop receptor. Each motif consists of its own unique function within the RNA molecule.

Tertiary RNA motifs are generally conserved structural sequences. In terms of pairs of interacting edges, there is the Watson-Crick (W), Hoogsteen (H), Sugar (S), and Glycosidic bond orientation cis (c) and trans (t). Tertiary structures can also be formed by pairwise interactions between nucleotides such as base-pairing, base stacking and base-phosphate.

Table 1.1 12 Geometric base-pairing families [3].

No.	GLYCOSIDIC BOND ORIENTATION	INTERACTING EDGES	SYMBOL	DEFAULT LOCAL STRAND ORIENTATION
1	<i>Cis</i>	Watson-Crick / Watson-Crick		Anti-parallel
2	<i>Trans</i>	Watson-Crick / Watson-Crick		Parallel
3	<i>Cis</i>	Watson-Crick / Hoogsteen		Parallel
4	<i>Trans</i>	Watson-Crick / Hoogsteen		Anti-parallel
5	<i>Cis</i>	Watson-Crick / Sugar Edge		Anti-parallel
6	<i>Trans</i>	Watson-Crick / Sugar Edge		Parallel
7	<i>Cis</i>	Hoogsteen / Hoogsteen		Anti-parallel
8	<i>Trans</i>	Hoogsteen / Hoogsteen		Parallel
9	<i>Cis</i>	Hoogsteen / Sugar Edge		Parallel
10	<i>Trans</i>	Hoogsteen / Sugar Edge		Anti-parallel
11	<i>Cis</i>	Sugar Edge / Sugar Edge		Anti-parallel
12	<i>Trans</i>	Sugar Edge / Sugar Edge		Parallel

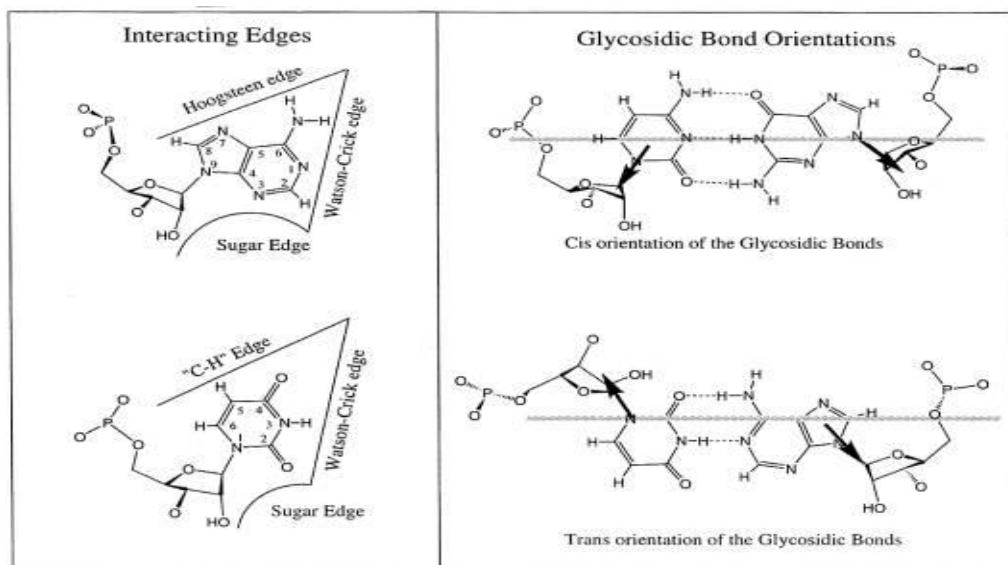


Figure 1.2 Pairs of interacting edges [3].

1.3 What is a Tetraloop-Tetraloop Receptor

The tetraloop is a hairpin loop consisting of 4 nt (nucleotide), GNRA, where N = any base and R = A or U. Giving a total of 8 possible variations of the tetraloop: GAAA, GAUA, GCAA, GCUA, GGAA, GGUA, GUAA, and GUUA. The tetraloop is also part of a tetraloop-receptor which is a 11 nt highly conserved sequence made up of CCUAAG....UAUAGG, the structure forms a stem/internal loop. To get a better understanding of the tetraloop-tetraloop receptor and its function the *Tetrahymena* thermophile is good reference that contains the tertiary structure. In this particular species the motif is located in the P4-P6 domain of the group I self-splicing intron. It has been shown that the GAAA is often found together with the 11 nt motif, interactions between the two cause self-splicing intron [7]. It is safe to assume that the other variations of the tetraloop interact in the same manner with the tetraloop-receptor. In the thesis GAAA is the reference tetraloop as well as PDB ID 1GID [9].

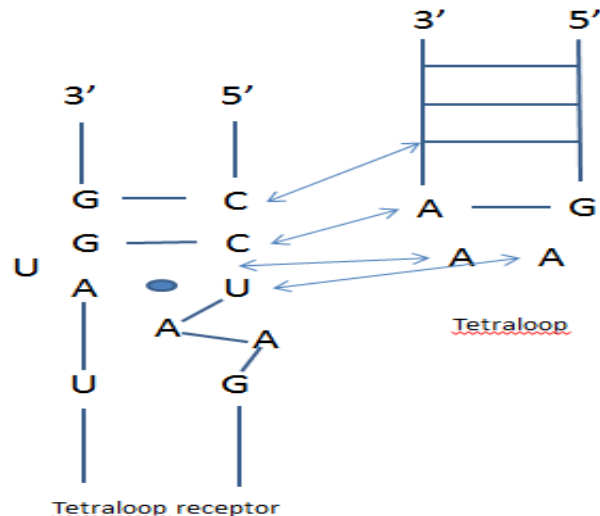


Figure 1.3 Tetraloop-Tetraloop Receptor.

1.4 Base Pair Interaction of Tetraloop-Tetraloop Receptor

The 11 nt motif consists of a C-G base pair (bp), 5 nt internal loop and a G-U bp. Studies suggest that the C-G bp interacts with the last A in GAAA. The structure of GAAA (tetraloop) is a rigid unit, all loops are in a anti conformation and all of the three A's in GAAA are stacked on one another from the 5` side of the 11nt motif (tetraloop-receptor). This type of side by side interaction will cause a kink in the backbone of the ribose allowing for an opening in the minor groove of the tetraloop-receptor. Each A in GAAA forms hydrogen bonds to the tetraloop receptor. The first A in GAAA makes a bond with A-U-A. The second A stacks on the A-U-A bp and the third A makes a bonds to C-G forming a triple pair. The GAAA interaction mentioned above also with the A-rich bulge (this part is not covered) help bring the two helical parts of the P4-P6 domain closer together, causing a close packing with the ribose phosphate backbone. Hence, in

essence the tetraloop-tetraloop receptor not only helps with self-splicing of the intron but also the folding of the RNA molecule [7].

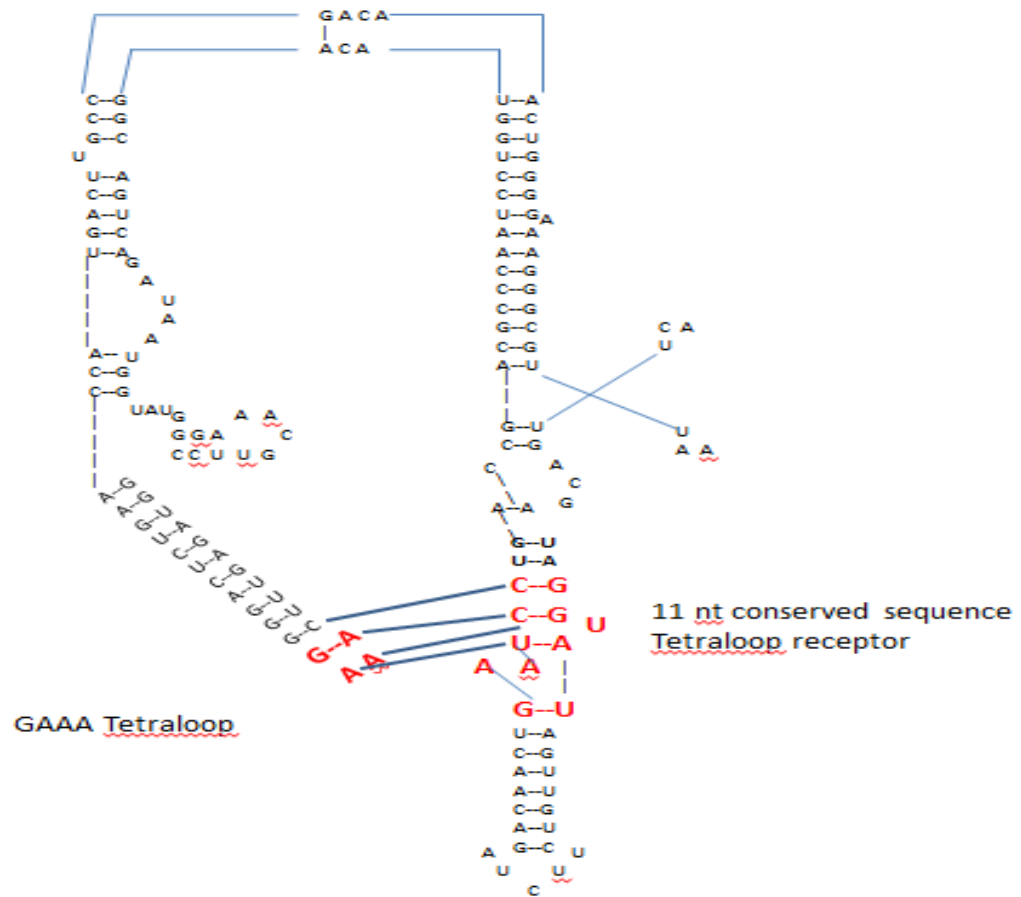


Figure 1.4 Base pair interaction between tetraloop and tetraloop-receptor.

CHAPTER 2

MATERIALS AND METHODS

2.1 Data Mining Process

Searching, or at least filtering, for a full tertiary motif can be attempted at 3 different levels:

- Primary, trying to find the primary motif of required nucleotides via sequence search
- Secondary, trying to find the secondary motif, the tetraloop and receptor secondary structures, each of course composed of the necessary nucleotide sequences in the primary motif.
- Tertiary, finally by searching full 3-D RNA structures for the fully defined motif -
- the correct primary structure sequences of nucleotides, arranged in the correct secondary structure elements, all located in the correct geometry in space and thus forming the correct hydrogen bonds in the tertiary structure

It is of course possible for the primary motif to occur in a molecule without forming the secondary motif, or for the primary/secondary motif to occur without forming the full tertiary motif. Therefore, each successive level of search method gets closer to identifying the true motif among false positives; however each search level also requires more computational resources (e.g., 3D search is much slower per molecule than sequence search) as well as much rarer experimental data (there are many sequences to search for the primary motif, but only a few crystal/NMR RNA structures to search for the full tertiary motif).

The motif is the tertiary interaction tetraloop-tetraloop receptor. In this motif, a tetraloop hairpin loop made up of 4 nucleotides (GNRA, where N=any base and R = A or U) forms a particular hydrogen bonding pattern to nucleotides in a tetraloop receptor formed by a stem/internal loop structure (CCUAAG...UAUGG) elsewhere in the RNA molecule. [1,2]. Note that the full tertiary motif comprises three motifs, one for each level of RNA structure:

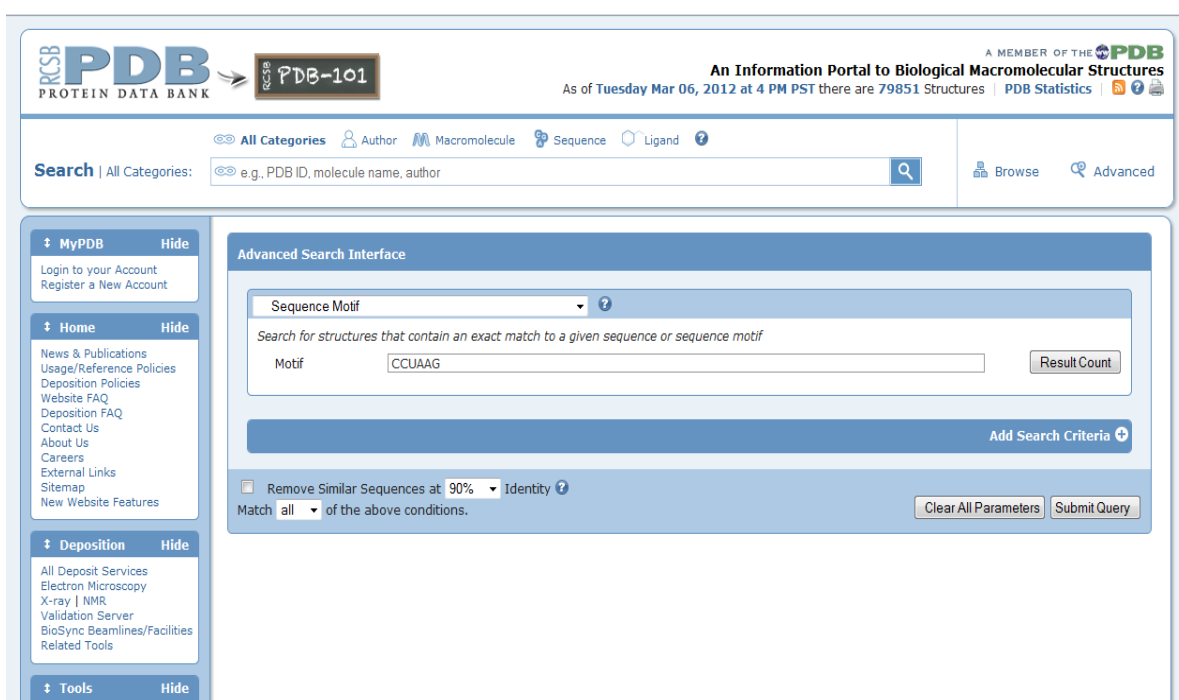
- The primary motif (the required sequence of nucleotides within the motif elements).
- The secondary motif (the tetraloop hairpin loop and receptor stem/internal loop structure).
- The tertiary motif itself (the distinctive hydrogen bonding between the tetraloop and the receptor).

Create the data set of RNA molecules. Each molecule must have full 3D data (crystal/NMR), and this should ensure that any and all examples of the motif documented in the literature are included. This data set will be taken from PDB [9].

2.2 Data Extraction

Knowing that the tetraloop GNRA and the tetraloop receptor CCUAAG...UAUAGG are both highly conserved sequence, this is how the PDB files with the motif were found: In Protein Data Bank (PDB) [9] under the Search tab click Advanced Search. From there a drop down box Choose a Query Type click on it and scroll down till Sequence motif is visible, select that and input any string desired. For the input, CCUAAG is entered and retrieved a list of 411 potential PDB files that may or may not contain the tertiary structure of the tetraloop-tetraloop receptors within them. Since CCUAAG is part of the

11 nt highly conserved sequence, its counterpart UAUAGG should be in the PDB files as well as with the tetraloop GNRA. The main article that is used as a reference to for this thesis [7], the PDB ID's: 2J00 and 1VQO is mentioned in the article as having tetraloop-tetraloop receptors. From the search hits obtained, these two PDB ID's were checked to see if they are in the results, which indeed they were. Based on this, the assumption is made that the search was done correctly and to extract the PDB files [9].



The screenshot displays the PDB website's search interface. At the top, the RCSB PDB logo is visible, along with a 'PDB-101' badge and a status bar indicating the date and time (Tuesday Mar 06, 2012 at 4 PM PST) and the number of structures (79851). The main search bar contains the text 'e.g., PDB ID, molecule name, author'. Below the search bar, there are navigation links for 'All Categories', 'Author', 'Macromolecule', 'Sequence', and 'Ligand'. The 'Advanced Search Interface' section is active, showing a 'Sequence Motif' dropdown menu and a search input field containing 'CCUAAG'. A 'Result Count' button is present next to the search input. Below the search input, there is an 'Add Search Criteria' button. At the bottom of the search interface, there is a checkbox for 'Remove Similar Sequences at 90% Identity' and a 'Match all' dropdown menu. 'Clear All Parameters' and 'Submit Query' buttons are also visible.

Figure 2.1 Tetraloop-tetraloop Receptor search on pdb.org [9].

List of PDB ID's:

1C2W:1, 1FFK:1, 1FJG:1, 1FKA:1, 1GID:1, 1GIX:1, 1GIY:1, 1GRZ:1, 1HNW:1, 1HNX:1, 1HNZ:1, 1HR0:1, 1HR2:1, 1I94:1, 1I95:1, 1I96:1, 1I97:1, 1IBK:1, 1IBL:1, 1IBM:1, 1J5A:1, 1J5E:1, 1JGO:1, 1JGP:1, 1JGQ:1, 1JJ2:1, 1JZX:1, 1JZY:1, 1JZZ:1, 1K01:1, 1K73:1, 1K8A:1, 1K9M:1, 1KC8:1, 1KD1:1, 1KQS:1, 1L8V:1, 1L9A:1, 1M1K:1, 1M90:1, 1MFQ:1, 1MJ1:3, 1ML5:1, 1ML5:4, 1N32:1, 1N33:1, 1N34:1, 1N36:1, 1N8R:1, 1NJI:1, 1NJM:1, 1NJJ:1, 1NJO:1, 1NJP:1, 1NKW:1, 1NWX:1, 1NWX:1, 1NWX:1, 1OND:1, 1P85:1, 1P86:1, 1P9X:1, 1PNS:1, 1PNU:1, 1PNX:1, 1PNY:1, 1Q7Y:1, 1Q81:1, 1Q82:1, 1Q86:1, 1QVF:1, 1QVG:1, 1RY1:2, 1S1H:1, 1S1I:1, 1S72:1, 1SM1:1, 1TLR:1, 1U6B:1, 1VOQ:1, 1VOR:1, 1VOS:1, 1VOU:1, 1VOV:1, 1VOW:1, 1VOX:1, 1VOY:1, 1VOZ:1, 1VP0:1, 1VQ4:1, 1VQ5:1, 1VQ6:1, 1VQ7:1, 1VQ8:1, 1VQ9:1, 1VQK:1, 1VQL:1, 1VQM:1, 1VQN:1, 1VQO:1, 1VQP:1, 1VS6:2, 1VS8:2, 1VSA:1, 1VSP:1, 1VT2:1, 1W2B:1, 1X8W:1, 1XBP:1, 1XMO:1, 1XMQ:1, 1XNQ:1, 1XNR:1, 1Y0Q:1, 1Y69:1, 1YHQ:1, 1YI2:1, 1YIJ:1, 1YIT:1, 1YJ9:1, 1YJN:1, 1YJW:1, 1YL3:2, 1YL4:1, 1Z58:1, 1ZN1:1, 1ZZN:1, 2AAR:1, 2ADT:1, 2AW4:2, 2AWB:2, 2B64:1, 2B66:2, 2B9M:1, 2B9N:2, 2B9O:1, 2B9P:2, 2D3O:1, 2E5L:1, 2F4V:1, 2GO5:1, 2GYA:1, 2GYC:1, 2HGI:1, 2HGJ:1, 2HGP:1, 2HGQ:1, 2HGR:1, 2HGU:1, 2HHH:1, 2I2T:2, 2I2V:2, 2I7E:1, 2I7Z:1, 2J00:1, 2J01:10, 2J02:1, 2J03:10, 2J28:10, 2J37:4, 2J37:8, 2JYF:1, 2JYH:1, 2JYJ:1, 2NOQ:1, 2O43:1, 2O44:1, 2O45:1, 2OGM:1, 2OGN:1, 2OGO:1, 2OTJ:1, 2OTL:1, 2OW8:1, 2QA4:1, 2QAM:2, 2QAO:2, 2QBA:2, 2QBC:2, 2QBE:2, 2QBG:2, 2QBI:2, 2QBK:2, 2QEX:1, 2QNH:1, 2QOV:2, 2QOX:2, 2QOZ:2, 2QP1:2, 2R8S:1, 2RDO:2, 2UU9:1, 2UUA:1, 2UUB:1, 2UUC:1, 2UXB:1, 2UXC:1, 2UXD:1, 2V46:1, 2V47:10, 2V48:1, 2V49:10, 2VHM:8, 2VHN:7, 2VQE:1, 2VQF:1, 2WDG:1, 2WDH:1, 2WDI:11, 2WDJ:11, 2WDK:1, 2WDL:11, 2WDM:1, 2WDN:11, 2WH1:1, 2WH2:11, 2WH3:1, 2WH4:11, 2WRI:1, 2WRJ:11, 2WRK:1, 2WRL:11, 2WRN:1, 2WRO:11, 2WRQ:1, 2WRR:11, 2WWQ:11, 2X9R:1, 2X9S:11, 2X9T:1, 2X9U:11, 2XFZ:1, 2XG0:11, 2XG1:1, 2XG2:11, 2XQD:1, 2XQE:11, 2XSY:1, 2XTG:11, 2XUX:11, 2XUY:1, 2Y0U:1, 2Y0V:11, 2Y0W:1, 2Y0X:11, 2Y0Y:1, 2Y0Z:11, 2Y10:1, 2Y11:11, 2Y12:1, 2Y13:11, 2Y14:1, 2Y15:11, 2Y16:1, 2Y17:11, 2Y18:1, 2Y19:11, 2Z4L:2, 2Z4N:2, 2ZJP:1, 2ZJQ:1, 2ZJR:1, 2ZKR:2, 2ZM6:1, 2ZUE:2, 2ZUF:2, 3BBO:1, 3BBX:2, 3BO2:1, 3BO3:1, 3BO4:1, 3BWP:1, 3CC2:30, 3CC4:30, 3CC7:30, 3CCE:30, 3CCJ:30, 3CCL:30, 3CCM:30, 3CCQ:30, 3CCR:30, 3CCS:30, 3CCU:30, 3CCV:30, 3CD6:30, 3CF5:1, 3CMA:30, 3CME:30, 3CPW:1, 3CXC:1, 3D5A:1, 3D5B:1, 3D5C:1, 3D5D:30, 3DF2:2, 3DF4:2, 3DG0:2, 3DG2:2, 3DG4:2, 3DG5:2, 3DLL:1, 3E1B:2, 3E1D:2, 3EOG:1, 3EOH:1, 3F1E:1, 3F1F:1, 3F1G:1, 3F1H:1, 3F1C:1, 3FIK:31, 3FIN:11, 3FWO:1, 3G4S:1, 3G6E:1, 3G71:1, 3G78:1, 3HUW:1, 3HUX:1, 3HUY:1, 3HUZ:1, 3I1N:1, 3I1P:1, 3I1R:1, 3I1T:30, 3I20:1, 3I22:1, 3I55:1, 3I56:30, 3I8F:1, 3I8G:1, 3I8H:1, 3I8I:1, 3I9B:1, 3I9C:1, 3I9D:1, 3I9E:1, 3IGI:1, 3IIN:2, 3IZ9:1, 3IZF:1, 3IZT:2, 3IZU:2, 3JQ4:1, 3JYX:3, 3KCR:2, 3KIQ:22, 3KIR:32, 3KIS:22, 3KIT:32, 3KIU:22, 3KIW:32, 3KIX:22, 3KIY:32, 3KNH:1, 3KNI:1, 3KNJ:1, 3KNK:1, 3KNL:1, 3KNM:1, 3KNN:1, 3KNO:1, 3KTV:1, 3KTV:3, 3MR8:1, 3MRZ:1, 3MS0:1, 3MS1:1, 3O58:1, 3O5H:1, 3OAS:1, 3OAT:1, 3OFC:1, 3OFD:1, 3OFQ:1, 3OFR:1, 3OFZ:1, 3OG0:1, 3OGE:1, 3OGY:1, 3OH5:10, 3OH7:10, 3OHC:1, 3OHD:1, 3OHJ:10, 3OHK:10, 3OHY:1, 3OHZ:10, 3OI0:1, 3OI1:10, 3OI2:1, 3OI3:10, 3OI4:1, 3OI5:10, 3ORB:1, 3OTO:1, 3PIO:1, 3PIP:1, 3PYN:1, 3PYO:1, 3PYQ:1, 3PYR:1, 3PYS:1, 3PYS:22, 3PYT:1, 3PYU:1, 3PYU:22, 3PYV:1, 3Q1Q:3, 3Q1R:3

2.3 PDB Files Conversion to XML Files

These were then put through the RNAView software [11], which produced 1) a post-script file with an illustration of all pair bonding in the molecule – similar to a secondary structure diagram, but enhanced with non-Watson-Crick hydrogen bonds that are NOT part of secondary structure, but are crucial to tertiary structures; and 2) an XML file in RNAML format, a standard encapsulating all of the base nucleotides, base pairs of all

types, and other structural information about an RNA molecule.

To investigate the full tertiary motif, the use of the fact that it actually comprises three motifs, one for each level of RNA structure:

- The primary motif (the required sequence of nucleotides within the motif elements)

gggaUAUGGaagaaccgggGAAAcuugguucuuCCUAAGuc
CU

Figure 2.2 Tetraloop-Tetraloop Receptor at the Primary structure level.

- The secondary motif (the tetraloop hairpin loop and receptor stem/internal loop structure)

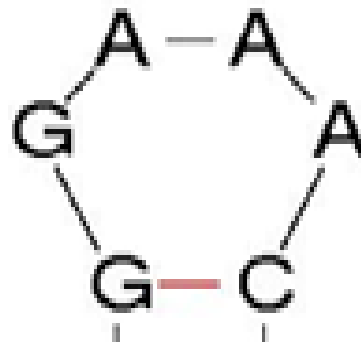


Figure 2.3 Tetraloop hairpin loop at the Secondary structure level [10].

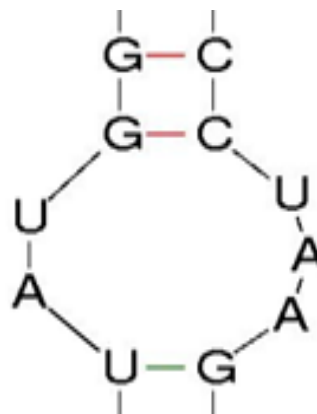


Figure 2.4 Receptor stem/internal loop at the Secondary structure level [10].

- The tertiary motif itself (the distinctive hydrogen bonding between the tetraloop and the receptor).

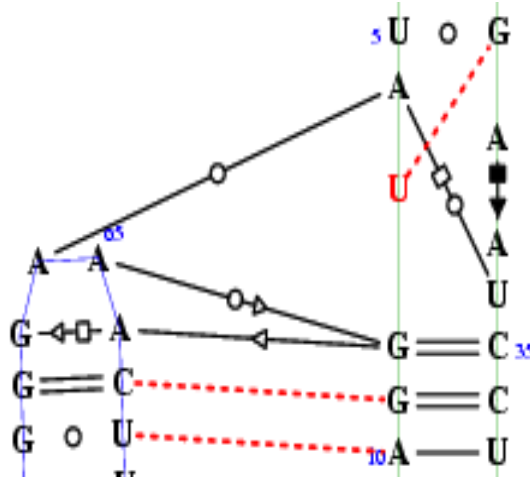


Figure 2.5 Tetraloop-Tetraloop Receptor tertiary structure with hydrogen bond interaction [10].

There are three primary motifs: the sequence forming a tetraloop (GNRA) forms the loop primary motif, and the two different sequences that join together to form a tetraloop receptor (CCUAAG and UAUGG), which will be known as receptor A and receptor B primary motifs.

There are two secondary motifs: a loop formed by the tetraloop primary motif, and an internal loop and stem structure, the receptor, formed by a pattern of base pairing between the two different receptor primary motifs.

There is only one tertiary motif, formed when the two different secondary motifs hydrogen bond together.

2.4 Tetraloop-Tetraloop search Methodology in XML files

Use the RNAView software [3] to obtain a graph of tertiary structure information in RNAXML format for all the molecules in the data set. Extract all examples of the motif from this data, to form the gold standard list of tetraloop-tetraloop receptor occurrences. This qualifies as a gold standard as this method is returning only those structures that satisfy all the requirements of the motif definition in 3D data. Also, very few examples of the tetraloop-tetraloop receptor motif are explicitly identified in the literature, but it should be confirmed that these literature instances are included in the gold standard.

Primary structure search: Search the sequence data for the data set to identify all molecules containing the primary structure motif; using the search page on the NDB web site and/or a small script we create to directly search the sequences. Evaluate the hypothesis "all instances of primary motif form the tertiary motif": collect statistics, determine true vs. false positives. See if it can be figured out, other patterns in the sequence that correlate with true positives (e.g., certain bases always occurring near the motif itself for true positives).

Secondary structure search: Search the RNAXML data to identify all molecules containing the secondary structure motif (keeping in mind that this includes the primary structure motif by definition). Evaluate the hypothesis "All instances of secondary motif form the tertiary motif": collect statistics, determine true vs. false positives. See if there are patterns in the overall secondary structure that correlate with true positives (e.g. length/type of stems adjacent to tetraloop hairpin loop and receptor stem/internal loop; overall pattern of molecule-wide secondary structure elements).

Secondary structure prediction: Predict secondary structure from sequence for the data set using RNAfold [4] or mfold [5]. Compare predicted structure to actual secondary structure from the RNAXML data, specifically for the elements in the motif; note that even if the prediction for the whole molecular structure is wrong, the particular structures forming the motif could be correct. Based on this and the results from the previous step, evaluate how well secondary structure prediction tools can successfully predict the full tertiary motif.

Tertiary structure prediction: Seeing as the motif is fully defined by a 3D tertiary structure pattern, predicting the motif from 3D data isn't really an issue. However the gold standard will be put into the molecules using the VMD tool [13] and the FR3D tool [6] to confirm that it finds the same motifs that RNAView tagged as golden. Also input any molecules that do contain the primary/secondary motif, but do NOT contain the full tertiary motif (according to RNAView), to confirm that VMD and FR3D also does not find the tertiary motif. This will check the two 3D motif identification tools (RNAView, VMD, and FR3D) against each other.

An XML file description:

```

<rnaml>
  <molecule>
    <sequence>
      <seq-data>
    <structure>
      <model>
        <base>
          <position>
          <base-type>
        <str-annotation>
          <base-pair>
            <base-id-5p>
              <base-id>
                <position>
            <base-id-3p>
              <base-id>
                <position>
          <edge-5p>
          <edge-3p>
          <bond-orientation>
        <single-strand>
          <segment>
            <base-id-5p>
              <base-id>
                <position>
            <base-id-3p>
              <base-id>
                <position>

```

The tags are shown at their proper levels, and were used to quickly build up the data structure described above. Below is a description of each tag. If two tags follow each other then that means one means little without the other [10].

<rnaml> This is the root tag that begins an RNAmL XML document. This document can be verified with rnaml.dtd.

<molecule> This tag begins the description of a single RNA molecule. A single XML file can have more than one molecule, each beginning with the <molecule> tag.

<sequence> **<seq-data>** This contains a quick preview of the entire RNA sequence.

<structure> <model> This begins the description of the physical structure of the RNA sequence. It starts with each base one by one, then follows all of the connection data, followed by special single strand data.

<base> This begins the description of a single base element.

<position> This is the index used inside the XML document to reference this base.

<base-type> this is the actual base type A, C, G, or U. If it is modified it will be listed in lower case.

<str-annotation> This begins the description of the connections between the bases.

<base-pair> This begins a description of a connection between two base elements.

<base-id-5p> This is the 5' base element in the pair **20**.

<base-id> <position> This is the index of the base being referenced. This is the same as the position shown above and is used to match up two bases to create a pair.

<base-id-3p> This is the 3' base element in the pair.

<base-id> <position> This is the index of the base being referenced. This is the same as the position shown above and is used to match up two bases to create a pair.

<edge-5p> This is the edge type for the 5' side. This can be W, H, S, +, or –

<edge-3p> This is the edge type for the 3' side. This can be W, H, S, +, or –

<bond-orientation> This is the Glycosidic Bond Orientation, it can be c, or t.

<single-strand> <segment> This begins the description of a range of bases indexes that make up a single strand segment.

2.5 Sequence in Context

To examine variation in each primary motif in context, the 3 bases 5' of, and the 3 bases 3' of, the motif itself is shown. Examples: loop – cucGAAAugg receptor, part A – augCCUAAGgca receptor, part B – cccUAUGGcua Will call this larger sequence pattern SIC for Sequence in Context. In addition to counting all instances of the primary motif, by the SIC occurrence it can be grouped and get counts for each.

```

66 cccUAUGGagc
53 gaaUAUGGggg
53 gggUAUGGcug
44 uguUAUGGcgu
... 21 rows omitted ...
1 augUAUGGggg
1 gcuUAUGGacc
501 Total Hits
27 Distinct SICs

```

Figure 2.6 Grouping count of SIC occurrences.

2.5.1 SIC patterns Part I 2.5.2

Examination of counts for each distinct SIC revealed an interesting pattern. Out of 300 total receptor A motif hits, 198 of them (66%) have the pattern: gg[au]CCUAAG. Here [au] means either an A or U base. If base choice was random, the expectation would be $1/4 \times 1/4 \times 2/4 = 1/32 = 3.125\%$ with this pattern.

Secondary motif processes, for each primary motif, examine all base pairs which include the motif bases. Loop the primary motif examined alone, receptor A and receptor B primary motifs examined in combination with each other. Lastly, determine whether

the primary motif(s) form the secondary motif.

The tetraloop at the secondary structure level is defined as: all 4 bases in GAAA motif are NOT in Watson-Crick (WC) base pairs -- but can be involved in other kinds of interaction the base just 5' of G and the base just 3' of the last A form a WC pair.

The secondary structure level of the tetraloop receptor is defined as: CC in receptor A and GG in receptor B sequence are WC paired – form a helix G in receptor A and first U in receptor B form a wobble pair (non-standard WC-edge pairing), the other bases not in WC pair form.

2.5.2 SIC patterns Part II

Of 4937 primary loop motifs (GAAA), 1013 have SIC pattern [cg][cg]GAAA[cg][cg], so 20.5% -- random chance would yield 6.25%. Of the 735 secondary loop motifs (GAAA is indeed a tetraloop), 558 have the SIC pattern – so 78.9%. Over 3/4 of GAAA tetraloops match this pattern. So the pattern [cg][cg]GAAA[cg][cg] has an elevated chance of occurring, and an elevated chance of forming a tetraloop when it occurs – thus accounting for the majority of GAAA tetraloops.

The tertiary motif at the tertiary structure level is defined as: Any hydrogen bonding between bases in secondary loop motif and secondary receptor motif.

Input: 192 XML files; 420 RNA molecules		
Pri Loop	Pri Receptor A	Pri Receptor B
4937 / 163	300 / 17	501 / 27
Sec Loop	Sec Receptor	
735 / 22 14.9%	29 / 5 A: 9.67%; B: 5.79%	
Tert Motif		
14 / 2 loop: 1.9%; receptor: 48.3%		

Figure 2.7 Statistics of occurrences at the primary, secondary, and tertiary level.

2.6 Data Processing for Motif Identification

A software script in the Python language to extract motifs from the RNAML files. This script, "rmaxml_parser.py", parses an RNA XML file, creates data objects for the base nucleotides and all the base pairs connecting them, and then finds and outputs all the examples of the tetraloop tetraloop-receptor primary, secondary, and tertiary motif components, along with various counts and formatted versions of this information.

The RNAML file does not extract important some important metadata from the input PDB file, namely the title and citation information concerning the journal article associated with the PDB database entry. Also a small perl script is written to extract this data directly from the PDB file and save it in a text file. When the rmaxml_parser.py program runs, it fetches this information from the text file and combines it with the RNA structural information from the RNAML file to provide complete, annotated data output.

The `rmaxml_parser.py` program outputs a variety of data, much of it for debugging purposes, but the "official" output consists of 4 different kinds of records, each record being a single line of text with tab-delimited data fields. Please refer to appendix A to view the program.

2.7 Program Output Details

An `RNA_MOLECULE` record is output for a single chain of RNA within the XML file. The parser can only work on a single chain at a time, but if you pass a chain number along with the PDB file as an input argument, it will operate on only that chain. This allows an RNAML file with several chains to be completely examined by running the parser over it several times in sequence. The `RNA_MOLECULE` record contains general title and citation information, the full sequence of the chain molecule, and set of counts describing the number of each of the three primary motifs, two secondary motifs, and single tertiary motif found in the molecule.

A `PRIMARY_MOTIF` record is output for each instance of any of the three primary motifs in the RNA sequence. For each motif it provides the type (loop, receptor A, or receptor B) and the starting position of the motif in the sequence. It also displays the SIC, the "sequence in context:" this is just the motif sequence with 3 bases added before and another 3 bases added after, to produce a slightly long sequence that shows the primary motif in the "neighborhood" of bases in which it occurs. This allows one to look for patterns.

A `SECONDARY_MOTIF` record is output for each instance of the two secondary motifs. For each motif it provides the type (loop or receptor) and the numeric IDs for the

primary motif records (one ID for loop, two IDs for receptor as it has two primary motif components) which form the secondary motif. This ties each secondary motif to the primary motifs from which it is composed in a normalized data scheme. Finally, a string representing all base pairs between the bases in the motif is shown, in a string format derived from the symbols used in RNAML to represent the different kinds of base pairing described by Leontis and Westhof [3].

A TERTIARY_MOTIF record is output for each instance of the full tertiary motif. The two secondary motifs included in the tertiary motif are identified by their numeric IDS. Also, the (quite long) base pair representation for all base pairs including ANY of the bases in the motif is output. Please refer to appendix B to view a sample program output where the tertiary structure of the tetraloop-tetraloop receptor exists.

CHAPTER 3

EXPERIMENTS

3.1 Find RNA 3D (FR3D)

Find RNA 3D (FR3D) is use online tool which gives the user the ability to select a PDB ID and input nucleotide base position to view a 3D image of the interaction occurring. FR3D also gives the capability to either do a geometric or symbolic search, for the purpose of this thesis geometric is applied, constraints can implemented based on the search being made but again for the thesis constraints were not used. Based on the data received from using the program developed to search for the tertiary structure of tetraloop-tetraloop receptors, FR3D is implemented to get a visualization of the tertiary structure of the tetraloop-tetraloop receptor. The numbers that are bolded is the given base pair location of the tetraloop-tetraloop receptor in the given PBD file. From the table 3.1 or please refer to appendix B.

Table 3.1 PDB ID and Starting Position of Tertiary Structure Location.

<p>TERTIARY_LOCATION: 1GID 150-222-247 IG1~SH~IA4 IA2~WW~bA2 IA3~!!~bG4 IA3~!!~aU3 IA4~SS~bG4 IA4~!!~aC2 sIC1~!!~bG5 aC1~+++~bG5 aC2~+++~bG4 aU3~WH~bA2 aA4~SH~aA5 aA5~WW~bU3 aG6~WW~bU1 aG6~!!~bU3</p>
<p>TERTIARY_LOCATION: 1GRZ 150-222-247 IG1~SH~IA4 IG1~WH~sIC1 IA2~WW~bA2 aC1~+++~bG5 aC2~+++~bG4 aU3~WH~bA2 aA4~!!~aA5 aA5~---~bU3 aG6~WW~bU1</p>
<p>TERTIARY_LOCATION: 1HR2 150-222-247 IG1~SH~IA4 IA2~WW~bA2 IA3~!!~bG4 IA3~!!~aU3 IA4~SS~bG4 IA4~!!~aC2 sIC1~!!~bG5 aC1~+++~bG5 aC2~+++~bG4 aU3~WH~bA2 aA4~SH~aA5 aA5~WW~bU3 aG6~WW~bU1 aG6~!!~bU3</p>
<p>TERTIARY_LOCATION: 1L8V 150-222-247 IG1~SH~IA4 IA2~WW~bA2 IA3~!!~bG4 IA3~!!~aU3 IA4~SS~bG4 IA4~!!~aC2 sIC1~!!~bG5 aC1~+++~bG5 aC2~+++~bG4 aU3~WH~bA2 aA4~SH~aA5 aA5~WW~bU3 aG6~WW~bU1 aG6~!!~bU3</p>
<p>TERTIARY_LOCATION: 1U6B 24-146-160 IG1~SH~IA4 IA2~WW~bA2 IA3~!!~aU3 IA4~SS~bG4 IA4~!!~aC2 sIC1~!!~bG5 aC1~+++~bG5 aC2~+++~bG4 aU3~WH~bA2 aA4~SH~aA5 aA5~---~bU3 aG6~WW~bU1 aG6~WW~bU3</p>
<p>TERTIARY_LOCATION: 1ZZN 24-146-160 IG1~SH~IA4 IA3~!!~aU3 IA4~!!~aC2 sIC1~!!~bG5 aC1~+++~bG5 aC2~+++~bG4 aU3~WW~bG4 aA4~SH~aA5 aG6~WW~bU1</p>
<p>TERTIARY_LOCATION: 2R8S 150-222-247 IG1~SH~IA4 IA2~WW~bA2 IA3~!!~bG4 IA3~!!~aU3 IA4~SS~bG4 IA4~!!~aC2 sIC1~!!~bG5 aC1~+++~bG5 aC2~+++~bG4 aU3~WH~bA2 aA4~SH~aA5 aG6~WW~bU1</p>
<p>TERTIARY_LOCATION: 3B04 24-146-160 IG1~!!~sIC1 IG1~!!~IA4 IA2~WW~bA2 IA3~!!~bG4 IA3~!!~aU3 IA4~SS~bG4 IA4~!!~aC2 sIC1~!!~bG5 aC1~+++~bG5 aC2~+++~bG4 aU3~WH~bA2 aA4~!!~aA5 aG6~WW~bU1</p>
<p>TERTIARY_LOCATION: 3IIN 24-146-160 IG1~WW~sIC1 IG1~!!~IA3 IA2~WW~bA2 IA3~!!~bG4 IA3~!!~aU3 IA4~SS~bG4 IA4~!!~aC2 sIC1~!!~bG5 aC1~+++~bG5 aC2~+++~bG4 aU3~WH~bA2 aA4~!!~aA5 aG6~WW~bU1</p>

WebFR3D Geometric Search Switch to Select an example Help

Structures to search

- 1FYP
- 1G1X
- 1G2E
- 1G2J
- 1G3A
- 1G4Q
- 1G59
- 1G70
- 1GAX
- 1GID**
- 1GIX
- 1GIY

Options

Discrepancy: 0.4

Email address (optional):

Reset

Help

Enter up to 25 nucleotides separated by commas. You can specify ranges using colons. [More](#)

Query PDB: 1GID Query nucleotides: 150,153,222,228,247,251 Interaction Matrix View Query Search

	NT150	NT151	NT152	NT153	NT222	NT223	NT224	NT225	NT226	NT227	NT228	NT247	NT248	NT249	NT250	NT251
NT150	N															
NT151		N														
NT152			N													
NT153				N												
NT222					N											
NT223						N										
NT224							N									
NT225								N								
NT226									N							
NT227										N						
NT228											N					
NT247												N				
NT248													N			
NT249														N		
NT250															N	
NT251																N

A A A A A A A A A A A A A A A A

Figure 3.1 FR3D web interface [12].

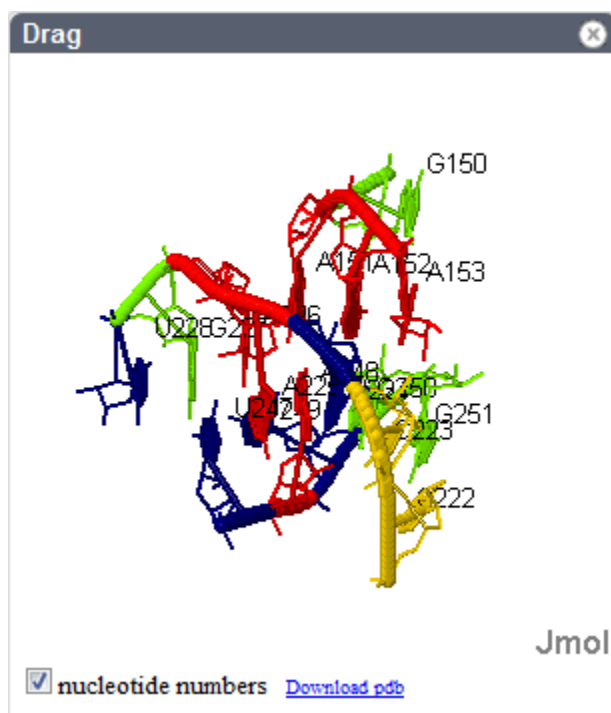


Figure 3.2 3D image of PDB ID: 1GID [12].

The 3D structure of the tetraloop-tetraloop receptor in 1GID represents the interactions occurring between the bases A, U, C, and G in the motif. The hexagons are the bases which are color coded red being Adenosine, blue being Uracil, green being Guanine, and yellow being Cytosine. FR3D has an interesting feature where it allows the user to display the base number position with the corresponding base listed beside it. The small branched out chains symbolize the carbons chains. As for the thicker and bulkier chains those are the backbones made up of phosphate and sugar. The image right above the 3D structure is a screenshot of FR3D and our input query. The blue and yellow rectangles are used for constraints and modify the output of the structure. The constraint function is not implemented since all that is needed is to the view all interactions being made in the tetraloop-tetraloop receptor motif.

3.2 Visual Molecular Dynamics (VMD)

Visual Molecular Dynamics (VMD) is similar to FR3D but much more detailed software. The input is a PDB [9] file in its pdb format, uploading the file into the tool gives a full 3D structure image whereas FR3D only gives an image based on the input of the base positions. VMD gives the user the capability to manipulate with the structure, allowing the selection of specific base position, gives the ability to change the style and format of the image, and other great features as well.

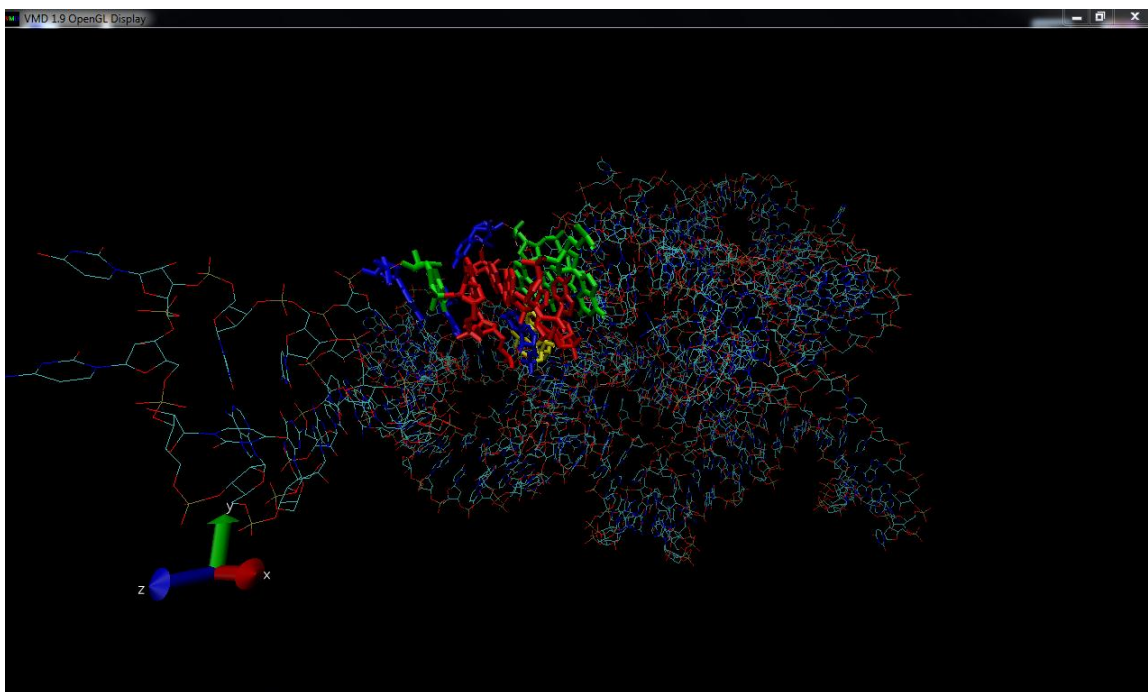


Figure 3.3 3D structure of PDB ID: 1GID with the tertiary structure of the Tetraloop-Tetraloop Receptor bolded.

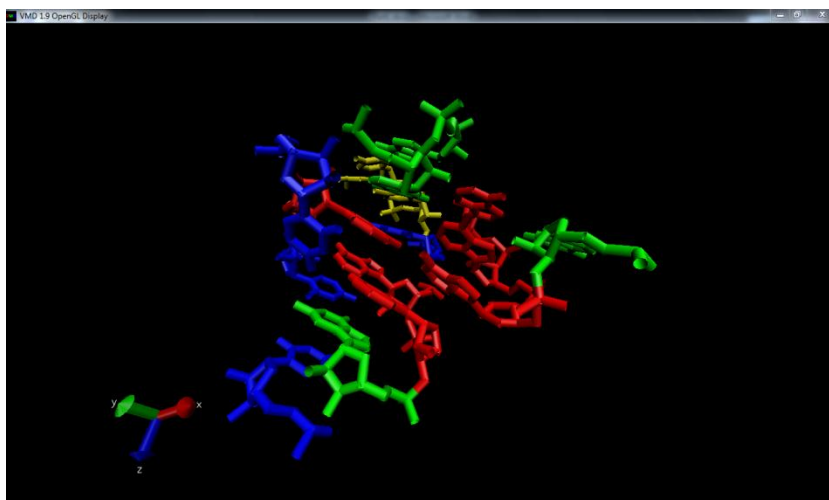


Figure 3.4 3D structure of the Tetraloop-Tetraloop Receptor.

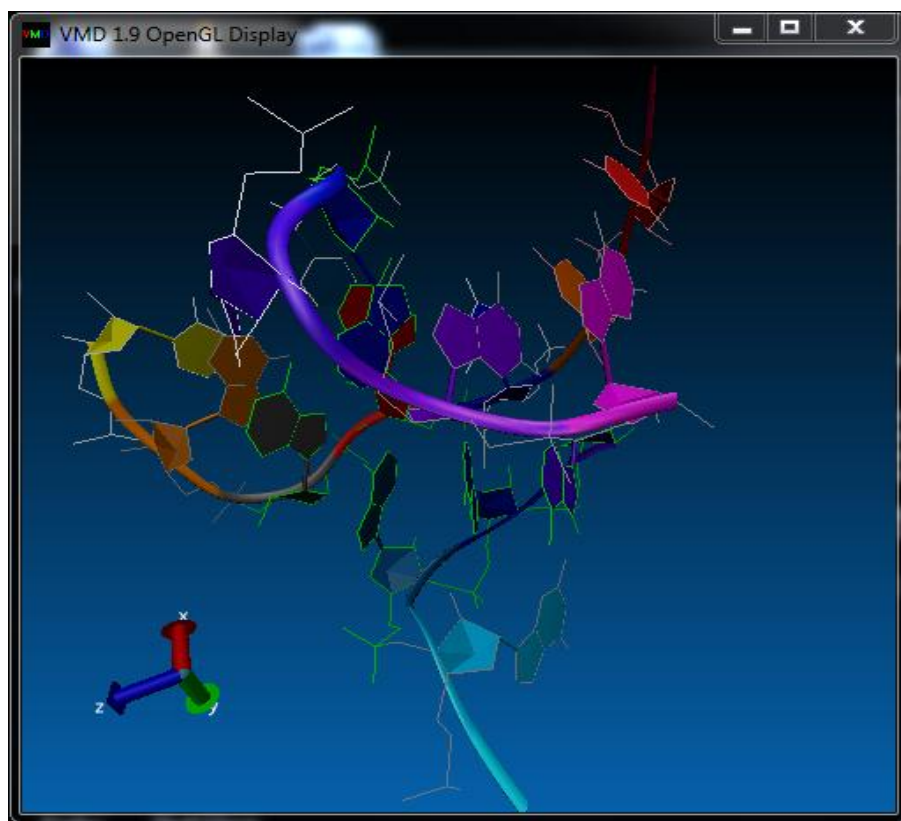


Figure 3.5 3D structure of Tetraloop-Tetraloop Receptor with a variation of style/format.

CHAPTER 4

DISCUSSION

4.1 Discussion and Further Studies

Examining the papers associated with the 14 full motif hits reveals that: SIC I has 10 instances, all of which are studies of the group I intron in *Tetrahymena thermophila* protozoa. SIC II has 4 instances, all of which are studies of the group I intron in *Azoarcus* bacteria. In conclusion to these findings there are actually only two distinct tetraloop-tetraloop receptors with GAAA.

Relatively few distinct RNA molecules have been crystallized. These are studied by different groups under slightly different conditions, so each RNA has multiple PDB entries with basically the same molecule. The specific tetraloop-tetraloop receptor motif, widely discussed as significant and important, appears to only have two known examples in PDB. Google searches of the topic appear to confirm this -- everyone discusses about the same two examples.

As it is mentioned earlier in the text the tetraloop-receptor is GNRA (generic form) where N = any base and R = A or U and all the possible variations are GAAA, GAUA, GCAA, GCUA, GGAA, GGUA, GUAA, and GUUA. Out of all of these possible combinations only the tetraloop GAAA was found from the 400+ list of PDB's. These results are expected from the text literature [7], the tetraloop GAAA is the most occurring receptor out of all the other varieties. Although from the PDB id's found with the potential of including the tertiary structure of the tetraloop-tetraloop receptor, the search made could have filtered out possible pdb files which may or may not contain the

other variations of the tetraloop.

APPENDIX A

Python Script

This is the Python program written to locate the tertiary structure of the Tetraloop-Tetraloop Receptors from the extracted PBD files.

```
#!/usr/bin/python
import sys, os
import xml.dom.minidom as minidom

#-----
def getText(node):
    # print node.nodeType, ":", node
    xxx = []
    for n in node.childNodes:
        # print " ", n.nodeType, ":", n
        if n.nodeType == n.TEXT_NODE:
            xxx.append( n.data.strip() )
    return ".join(xxx)
#-----
def get1Element(node, tag):
    return node.getElementsByTagName(tag)[0]
#-----
def getElements(node, tag):
    return node.getElementsByTagName(tag)

#####
#####
class Base(object):

    def __init__(self, rna_obj, pos):

        self.pos = pos
        try:
            self.nucleotide = rna_obj.sequence[pos]
        except KeyError:
            print >>sys.stderr, "PDB ID", rna_obj.pdb_id
            raise
        self.pdb_pos = rna_obj.pdb_numbering[pos]
        self.motif = "?"
        self.motif_sort = 4
        self.canon_pos = self.pdb_pos
        rna_obj.pos_2_base[pos] = self
#-----
```

```

def display(self, style="pdb"):
    if style == "pdb":
        disp_base = "%s%d" % (self.nucleotide, self.pdb_pos)
    elif style == "canon":
        disp_base = "%s%s%d" % (self.motif, self.nucleotide, self.canon_pos)
    else:
        disp_base = "%s%d" % (self.nucleotide, self.pos)
    return disp_base
#####
#####
class Basepair(object):

    def __init__(self, rna_obj, bp_element):

        # get position and base for 5p base in pair
        tmp = bp_element.getElementsByTagName("base-id-5p")[0]
        pos = int(getText(tmp.getElementsByTagName("position")[0]))
        base = rna_obj.pos_2_base[pos]
        self.base_5p = base

        try:
            rna_obj.base_pairs_by_5p[pos].append(self)
        except KeyError:
            print >>sys.stderr, "PDB ID", rna_obj.pdb_id
            raise

        # get position and base for 3p base in pair
        tmp = bp_element.getElementsByTagName("base-id-3p")[0]
        pos = int(getText(tmp.getElementsByTagName("position")[0]))
        base = rna_obj.pos_2_base[pos]
        self.base_3p = base

        try:
            rna_obj.base_pairs_by_3p[pos].append(self)
        except KeyError:
            print >>sys.stderr, "PDB ID", rna_obj.pdb_id
            raise

        self.type = "%s%s" % (
            getText(bp_element.getElementsByTagName("edge-5p")[0]),
            getText(bp_element.getElementsByTagName("edge-3p")[0])
        )
#-----
def display(self, style="pdb")

```

```

    if style == "canon" and (self.base_3p.motif_sort < self.base_5p.motif_sort):
        return "%s~%s~%s" % ( self.base_3p.display(style), self.type,
self.base_5p.display(style) )
    else:
        return "%s~%s~%s" % ( self.base_5p.display(style), self.type,
self.base_3p.display(style) )
#-----
def WC(self):
    return True if self.type == "++" or self.type == "--" else False

#####
#####
class RNA(object):

    def __init__(self, xml_fname, molecule_id):

        has_molecule_id = False
        id_text = 'id="%s"' % molecule_id
        for line in open(xml_fname):
            if "<molecule" in line:
                if id_text in line:
                    has_molecule_id = True
                    break

        #
        if not has_molecule_id:
            print >>sys.stderr, "Skipping file %s, does not have a molecule of id %s" %
(xml_fname, molecule_id)
            sys.exit()

        fname_base = os.path.basename(xml_fname)
        if fname_base.endswith(".pdb.xml"):
            self.pdb_id = fname_base[0 : fname_base.find(".") ]
        else:
            self.pdb_id = "???"
        full_dom = minidom.parse(xml_fname)
        for molecule_elem in getElements(full_dom, "molecule"):
            if molecule_elem.getAttribute("id") == molecule_id:
                rna_dom = molecule_elem

        try:
            rna_dom
        except UnboundLocalError:
            print >>sys.stderr, "No molecule with id=%s found in PDB ID %s", (molecule_id,
self.pdb_id)
            raise
        self.rna_dom = rna_dom

```

```

self.molecule_id = molecule_id

# get numbering start/end actually used in this file
tmp = get1Element(rna_dom, "numbering-range")
self.numbering_start = int(getText(get1Element(tmp, "start")))
self.numbering_end = int(getText(get1Element(tmp, "end")))

# get table converting to/from original PDB sequence numbering
tmp = getText( get1Element(rna_dom, "numbering-table") )
seq_nbr_list = [ int(i) for i in tmp.split() ]
if len(seq_nbr_list) != (self.numbering_end - self.numbering_start + 1):
    raise Exception("numbering-table has %d elements, does not match start:end of
%d:%d" % (
        len(seq_nbr_list), self.numbering_start, self.numbering_end)
    )
self.pdb_numbering = [0] + seq_nbr_list

# get sequence itself
tmp = getText( get1Element(rna_dom, "seq-data") )
sequence = ".join( tmp.split() ) # nukes all whitespace
if len(sequence) != len(seq_nbr_list):
    raise Exception("sequence has %d elements, but sequence number has %d
elements" % (
        len(sequence), len(seq_nbr_list) )
    )
self.sequence = (" " * self.numbering_start) + sequence

# initialize database structures for bases and base-pairs
self.pos_2_base = {} # map of pos# to nucleotide base objects
base_pairs_by_5p = {} # map of pos# to list of base pairs where pos# is 5p base
base_pairs_by_3p = {} # map of pos# to list of base pairs where pos# is 3p base
# iterate over sequence/pos#s, adding base to pos#
i = self.numbering_start
while i <= self.numbering_end:
    Base(self, i)
    base_pairs_by_5p[i] = [] # initialize list
    base_pairs_by_3p[i] = [] # initialize list
    i += 1
# save these data structures in RNA object
self.base_pairs_by_5p = base_pairs_by_5p
self.base_pairs_by_3p = base_pairs_by_3p

# get base-pairs
for bp_element in rna_dom.getElementsByTagName("base-pair"):
    # note that base pairs link themselves into
    # base_pairs_by_[53]p dictionaries in RNA object (self

```

```

    Basepair(self, bp_element)

#-----
def display_bps(self, base_pos_list, style="pdb", include_external=True):

    bps_seen = set()
    bps_list = []

    for pos in base_pos_list:
        base_bps_list = []
        for bp in rna.base_pairs_by_3p[pos]:
            if not include_external and bp.base_5p.pos not in base_pos_list:
                continue
            base_bps_list.append(bp)
            if bp not in bps_seen:
                bps_list.append(bp)
                bps_seen.add(bp)
        # end loop over 3p pos
        for bp in rna.base_pairs_by_5p[pos]:
            if not include_external and bp.base_3p.pos not in base_pos_list:
                continue
            base_bps_list.append(bp)
            if bp not in bps_seen:
                bps_list.append(bp)
                bps_seen.add(bp)
        # end loop over 5p pos
    # end loop over passed pos

    if style == "canon":
        bps_list.sort(key=lambda x: (x.base_5p.motif_sort, x.base_5p.pos,
x.base_3p.motif_sort, x.base_3p.pos) )
    else:
        bps_list.sort(key=lambda x: (x.base_5p.pos, x.base_3p.pos) )

    return " ".join( [ bp.display(style) for bp in bps_list ] )

#-----
def findPrimary(self, string):
    primary_starts = []
    i = self.sequence.find(string, 0)
    while i > -1:
        primary_starts.append(i)
        i = self.sequence.find(string, i+1)
    return primary_starts
#-----

```

```

def seqInContext(self, s, e):
    if s-3 < 0:
        prefix = "-" * -(s-3) + self.sequence[0:s].lower()
    else:
        prefix = self.sequence[s-3:s].lower()
    suffix = self.sequence[e+1:e+4].lower()
    if len(suffix) < 3:
        suffix = suffix + ("-" * (3-len(suffix)))
    return prefix + self.sequence[s:e+1] + suffix
#-----
def findBPs(self, mypos, other_pos=None, other_base=None, bp_type=None):

    matched_bps = []

    for bp in self.base_pairs_by_5p[mypos]:
        if other_pos and other_pos != bp.base_3p.pos: continue
        if other_base and other_base != bp.base_3p.nucleotide: continue
        if bp_type and bp_type != bp.type: continue
        matched_bps.append(bp)

    for bp in self.base_pairs_by_3p[mypos]:
        if other_pos and other_pos != bp.base_5p.pos: continue
        if other_base and other_base != bp.base_5p.nucleotide: continue
        if bp_type and bp_type != bp.type[:-1]: continue
        matched_bps.append(bp)

    return matched_bps
#-----
def mark_motif(self, motif, start_pos):
    if motif == "l":
        length = 4
        motif_sort = 1
    elif motif == "a":
        length = 6
        motif_sort = 2
    elif motif == "b":
        length = 5
        motif_sort = 3

    # do canonical labeling for all the bases in the motif proper
    j = 1
    for i in range(start_pos, start_pos+length):
        base = self.pos_2_base[i]
        base.motif = motif
        base.motif_sort = motif_so

```



```

    base.canon_pos = j
    j += 1
# and for the 3 bases 5' of the motif, the prefix
j = 1
for i in range(start_pos-1, max(start_pos-4, self.numbering_start), -1):
    base = self.pos_2_base[i]
    base.motif = "p" + motif
    base.motif_sort = motif_sort
    base.canon_pos = j
    j += 1
# and for the 3 bases 3' of the motif, the suffix
j = 1
for i in range(start_pos+length, min(start_pos+length+3, self.numbering_end)):
    base = self.pos_2_base[i]
    base.motif = "s" + motif
    base.motif_sort = motif_sort
    base.canon_pos = j
    j += 1

#-----
def test_tetraloop(self, start_pos):
    "Tests whether 4 bases starting at start_pos in sequence are a tetraloop (of any
    variety)"

    # are we on a stem? this is true only if there is a WC base pairing
    # between relative positions 0 and 5 (1-4 then being the tetraloop)
    on_stem = False
    for bp in self.base_pairs_by_5p[start_pos-1]:
        if bp.base_3p.pos == (start_pos+4) and bp.WC():
            on_stem = True
            break
    if not on_stem: return False
    # end on_stem test

    # is our loop made of unpaired bases (WC pairing only)?
    # true only if none of bases 1-4 is in ANY WC base pairing
    # note that this excludes pseudoknots, but we assume a
    # true tetraloop cannot be in a pseudoknot
    for i in range(start_pos, start_pos+4):
        for bp in self.base_pairs_by_5p[i]:
            if bp.WC(): return False
        for bp in self.base_pairs_by_3p[i]:
            if bp.WC(): return False
    # end no_WC_in_loop test

```

```

# if we made it this far, we pass all tests and we are in a tetraloop!

# mark out the motif for the canonical display of the bases involved
self.mark_motif("l", start_pos)

return True
#-----
def test_receptor(self, A_start_pos, B_start_pos):
    "Tests whether two sequences of bases form our tetraloop receptor"

    # 123456 12345
    # a=CCUAAG b=UAUGG

    # test aC1 ++ bG5 (Watson-Crick base pair)
    if not self.findBPs(A_start_pos, other_pos=B_start_pos+4, bp_type="++"):
        return False
    # test aC2 ++ bG4 (Watson-Crick base pair)
    if not self.findBPs(A_start_pos+1, other_pos=B_start_pos+3, bp_type="++"):
        return False
    # test aG6 WW bU1 ("Wobble" base pair)
    if not self.findBPs(A_start_pos+5, other_pos=B_start_pos, bp_type="WW"):
        return False
    # make sure aU3, aA4, aA5 are not in Watson-Crick base pairs
    for i in range(A_start_pos+2, A_start_pos+4):
        for bp in self.base_pairs_by_5p[i]:
            if bp.WC(): return False
        for bp in self.base_pairs_by_3p[i]:
            if bp.WC(): return False
    # make sure bA2, bU3 are not in Watson-Crick base pairs
    for i in range(B_start_pos+1, B_start_pos+2):
        for bp in self.base_pairs_by_5p[i]:
            if bp.WC(): return False
        for bp in self.base_pairs_by_3p[i]:
            if bp.WC(): return False
    # if we have made it here, we pass all tests and return True, a receptor!

    # mark out the motif for the canonical display of the bases involved
    self.mark_motif("a", A_start_pos)
    self.mark_motif("b", B_start_pos)

    return True
#-----
def test_full_motif(self, L_start_pos, A_start_pos, B_start_pos):

```

"Tests whether three sequences of bases form the full tetraloop - tetraloop receptor motif"

```

receptor_base_set = set()
for i in range(A_start_pos, A_start_pos+5):
    receptor_base_set.add(i)
for i in range(B_start_pos, B_start_pos+4):
    receptor_base_set.add(i)

for pos in range(L_start_pos, L_start_pos+4):
    for bp in self.findBPs(pos):
        if bp.base_3p.pos in receptor_base_set: return True
        if bp.base_5p.pos in receptor_base_set: return True
    return False
#####
#####
try:
    fname = sys.argv[1]
except IndexError:
    raise Exception("You must enter an RNAXML filename to parse")

try:
    molecule_id = sys.argv[2]
except IndexError:
    molecule_id = "1"

if not os.path.exists(fname):
    raise Exception("The entered file %s does not exist" % fname)

rna = RNA(fname, molecule_id)
pmotif_id_ctr = 0
smotif_id_ctr = 0
tmotif_id_ctr = 0

pdb_title = "Unknown"
pdb_jnl_title = "Unknown"
pdb_authors = "Unknown"
pdb_reference = "Unknown"
pdb_pubmed_ID = "Unknown"
pdb_ndb_ID = "none"

for line in
open("/Users/drew/home/school/bnfo644_data_mining/TermProject/Data/PDB_structure
s/pdb_file_metadata.txt"):
    if not line.startswith(rna.pdb_id): continue

```

```

line = line.strip()
pdb_redux, code, data = line.split("~")
if code == "TITLE":
    pdb_title = data
elif code == "JNL_TITLE":
    pdb_jnl_title = data
elif code == "AUTH":
    pdb_authors = data
elif code == "REF":
    pdb_reference = data
elif code == "PMID":
    pdb_pubmed_ID = data
elif code == "NDB_ID":
    pdb_ndb_ID = data

# get title
#title = "Unknown"
#basename = os.path.basename(fname)
#for line in open("RNAML/get_rnaml_from_ndb.LOG"):
#    if line.find(basename) > -1:
#        line = line.strip()
#        title = line[ line.find("Title") + 6:]
#        break
#print "TITLE", title

# find tetraloops
pri_tetraloops = rna.findPrimary("GAAA")
sec_tetraloops = []
pos_2_pmotif_id = {}
pos_2_smotif_id = {}
print "Tetraloop Pmotifs:"
for s in pri_tetraloops:
    pmotif_id_ctr += 1
    pos_2_pmotif_id[s] = pmotif_id_ctr
    e = s + 3
    print rna.pdb_numbering[s], "->", rna.pdb_numbering[e], rna.sequence[s:e+1]
    print "PRIMARY_MOTIF: %s\t%s\t%d\tloop\t%d\t%d\t%s" % (rna.pdb_id,
rna.molecule_id, pmotif_id_ctr, s, rna.pdb_numbering[s], rna.seqInContext(s,e))
    print " BP:", rna.display_bps(range(s,e+1))
    if rna.test_tetraloop(s):
        smotif_id_ctr += 1
        pos_2_smotif_id[s] = smotif_id_ctr
        sec_tetraloops.append(s)

```

```

canonical_bp_display = rna.display_bps(range(s,s+4), style="canon",
include_external=False)
    print "Tetraloop? True"
    print "SECONDARY_MOTIF: %s\t%s\t%d\tloop\t%d\t0\t%s\t%s" % (rna.pdb_id,
rna.molecule_id, smotif_id_ctr, pmotif_id_ctr, rna.seqInContext(s,e),
canonical_bp_display)
    else:
        print "Tetraloop? False"
        print "-----"

print "-----"
# CCUAAG...UAUGG
# find receptors part A
pri_receptor_A = rna.findPrimary("CCUAAG")
print "Receptor, Part A:"
for s in pri_receptor_A:
    pmotif_id_ctr += 1
    pos_2_pmotif_id[s] = pmotif_id_ctr
    e = s + 5
    print rna.pdb_numbering[s], "->", rna.pdb_numbering[e], rna.sequence[s:e+1]
    print "PRIMARY_MOTIF: %s\t%s\t%d\treceptor_A\t%d\t%d\t%s" % (rna.pdb_id,
rna.molecule_id, pmotif_id_ctr, s, rna.pdb_numbering[s], rna.seqInContext(s,e))
    print "  BP:", rna.display_bps(range(s,e+1))

print "-----"
# find receptors part B
pri_receptor_B = rna.findPrimary("UAUGG")
print "Receptor, Part B:"
for s in pri_receptor_B:
    pmotif_id_ctr += 1
    pos_2_pmotif_id[s] = pmotif_id_ctr
    e = s + 4
    print rna.pdb_numbering[s], "->", rna.pdb_numbering[e], rna.sequence[s:e+1]
    print "PRIMARY_MOTIF: %s\t%s\t%d\treceptor_B\t%d\t%d\t%s" % (rna.pdb_id,
rna.molecule_id, pmotif_id_ctr, s, rna.pdb_numbering[s], rna.seqInContext(s,e))
    print "  BP:", rna.display_bps(range(s,e+1))

print "-----"
sec_receptors = []
tert_motifs = []
print "Full Receptor Candidates"
for A in pri_receptor_A:
    for B in pri_receptor_B:
        print rna.pdb_numbering[A], "<>", rna.pdb_numbering[B]

```

```

if rna.test_receptor(A,B):
    smotif_id_ctr += 1
    sec_receptors.append((A,B))
    print "Receptor? True"
    canonical_bp_display = rna.display_bps(range(A,A+6) + range(B,B+5),
style="canon", include_external=False)
    print "SECONDARY_MOTIF: %s\t%s\t%d\treceptor\t%d\t%d\t%s\t%s\t%s" % (
        rna.pdb_id, rna.molecule_id, smotif_id_ctr, pos_2_pmotif_id[A],
pos_2_pmotif_id[B],
        rna.seqInContext(A,A+5), rna.seqInContext(B,B+4),canonical_bp_display
    )
    for L in sec_tetraloops:
        if rna.test_full_motif(L, A, B) :
            tmotif_id_ctr += 1
            tert_motifs.append((L,A,B))
            canonical_bp_display = rna.display_bps(range(L,L+4) + range(A,A+6) +
range(B,B+5),
                                style="canon", include_external=True
            )
            print "TERTIARY_MOTIF: %s\t%s\t%d\t%d\t%d\t%s\t%s\t%s\t%s" % (
                rna.pdb_id, rna.molecule_id, tmotif_id_ctr, pos_2_smotif_id[L],
smotif_id_ctr,
                rna.seqInContext(L,L+3), rna.seqInContext(A,A+5),
rna.seqInContext(B,B+4),
                canonical_bp_display
            )
            print "TERTIARY_LOCATION: %s\t%d-%d-%d\t%s" % (
                rna.pdb_id, rna.pdb_numbering[L], rna.pdb_numbering[A],
rna.pdb_numbering[B], canonical_bp_display
            )
        else:
            print "Receptor? False"

print "-----"
all_primotif_flag = (True if pri_tetraloops and pri_receptor_A and pri_receptor_B else
False)
all_secmotif_flag = (True if sec_tetraloops and sec_receptors else False)
all_tertmotif_flag = (True if tert_motifs else False)
print "SUMMARY: %12s MOL: %s PRI: loop %3d, recA %3d, recB %3d, %5s SEC:
loop %3d, rec%3d, %5s; TERT: %3d %5s" % (
    fname, rna.molecule_id,
    len(pri_tetraloops), len(pri_receptor_A), len(pri_receptor_B), all_primotif_flag,
    len(sec_tetraloops), len(sec_receptors), all_secmotif_flag,
    len(tert_motifs), all_tertmotif_flg
)

```

```
)  
print 'RNA_MOLECULE: %s\t%s\t%s\t%d\t%d\t%d\t%d\t%d\t%d\t%s\t%s; "%s"; %s;  
PubMedID %s\t%s\t%s\t%s' % (  
    rna.pdb_id, rna.molecule_id,  
    pdb_ndb_ID,  
    len(pri_tetraloops), len(pri_receptor_A), len(pri_receptor_B),  
    len(sec_tetraloops), len(sec_receptors),  
    len(tert_motifs),  
    pdb_title,  
    pdb_authors, pdb_jnl_title, pdb_reference, pdb_pubmed_ID,  
    fname,  
    "",  
    rna.sequence )
```

APPENDIX B

Program Output

This is a sample output which the program gives. As seen what is bolded in the text is what is being looked for to prove the work done.

Tetraloop Pmotifs:

112 -> 115 GAAA

PRIMARY_MOTIF: 1GID 1 1 loop 10 112 cggGAAAggg

BP: G112~+++~C208 A113~SH~A207 A114~HS~A206

Tetraloop? False

150 -> 153 GAAA

PRIMARY_MOTIF: 1GID 1 2 loop 48 150 gggGAAAcuu

BP: G150~SH~A153 A151~WW~A248 A152~!!~U224 A152~!!~G250
A153~!!~C223 A153~SS~G250

Tetraloop? True

SECONDARY_MOTIF: 1GID 1 1 loop 2 0 gggGAAAcuu

IG1~SH~IA4

Receptor, Part A:

222 -> 227 CUAAG

PRIMARY_MOTIF: 1GID 1 3 receptor_A 120 222

aguCUAAGuca

BP: A152~!!~U224 A153~!!~C223 C222~+++~G251 C223~+++~G250
U224~WH~A248 A225~SH~A226 A226~WW~U249 G227~WW~U247
G227~!!~U249

Receptor, Part B:

177 -> 181 UAUGG

PRIMARY_MOTIF: 1GID 1 4 receptor_B 75 177

gggUAUGGuaa

BP: C137~+++~G181 C138~+++~G180 G163~!!~U177 U177~!!~A178 U179~!!~G180
G181~SS~A186

247 -> 251 UAUGG

PRIMARY_MOTIF: 1GID 1 5 receptor_B 145 247

ugaUAUGGaug

BP: A151~WW~A248 A152~!!~G250 A153~SS~G250 C154~!!~G251
C222~+++~G251 C223~+++~G250 U224~WH~A248 A226~WW~U249
G227~WW~U247 G227~!!~U249

Full Receptor Candidates

222 <> 177

Receptor? False

222

<>

247Receptor?Tru

SECONDARY_MOTIF: 1GID 1 2 receptor 3 5
 aguCCUAAGuca ugaUAUGGaug aC1~+++~bG5 aC2~+++~bG4
 aU3~WH~bA2 aA4~SH~aA5 aA5~WW~bU3 aG6~WW~bU1 aG6~!!~bU3

TERTIARY_MOTIF: 1GID 1 1 1 2 gggGAAAcuu
 aguCCUAAGuca ugaUAUGGaug 1G1~SH~1A4 1A2~WW~bA2
 1A3~!!~aU3 1A3~!!~bG4 1A4~!!~aC2 1A4~SS~bG4 s1C1~!!~bG5 aC1~+++~bG5
 aC2~+++~bG4 aU3~WH~bA2 aA4~SH~aA5 aA5~WW~bU3 aG6~WW~bU1
 aG6~!!~bU3

TERTIARY_LOCATION: 1GID 150-222-247 1G1~SH~1A4 1A2~WW~bA2
 1A3~!!~aU3 1A3~!!~bG4 1A4~!!~aC2 1A4~SS~bG4 s1C1~!!~bG5 aC1~+++~bG5
 aC2~+++~bG4 aU3~WH~bA2 aA4~SH~aA5 aA5~WW~bU3 aG6~WW~bU1
 aG6~!!~bU3

 SUMMARY: 1GID.pdb.xml MOL: 1 PRI: loop 2, recA 1, recB 2, **True SEC: loop 1, rec 1, True; TERT: 1 True**

RNA_MOLECULE: 1GID 1 none 2 1 2 1 1 1
 Unknown Unknown; "Unknown"; Unknown; PubMedID Unknown
 1GID.pdb.xml

GAAUUGCGGGAAAGGGGUCAACAGCCGUUCAGUACCAAGUCUCAGGGGAA
 ACUUUGAGAUGGCCUUGCAAAGGGUAUGGUAAUAAGCUGACGGACAUGGU
 CCUAACCACGCAGCCAAGUCCUAAGUCAACAGAUCUUCUGUUGAUAUGGA
 UGCAGUUC

References

- 1: Costa and Michel, 1995. "Frequent use of the same tertiary motif by self-folding RNAs", *EMBO Journal* 14.6:1276-1285
- 2: Cate, et al., 1996. "Crystal Structure of a Group I Ribozyme Domain: Principles of RNA Packing," *Science* 273:1678
- 3: Yang, H., Jossinet, F., Leontis, N., Chen, L., Westbrook, J., Berman, H.M., Westhof, E., 2003. "Tools for the automatic identification and classification of RNA base pairs," *Nucleic Acids Research*. 31.13: 3450-3460.
- 4: I.L. Hofacker, W. Fontana, P.F. Stadler, S. Bonhoeffer, M. Tacker, P. Schuster, 1994. "Fast Folding and Comparison of RNA Secondary Structures," *Monatshefte f. Chemie* 125: 167-188
- 5: Zuker, 2003. "Mfold web server for nucleic acid folding and hybridization prediction," *Nucleic Acids Research*. 31.13: 3406-3415.
- 6: Michael Sarver; Craig L. Zirbel; Jesse Stombaugh; Ali Mokdad; Neocles B. Leontis, 2008. "FR3D: Finding Local and Composite Recurrent Structural Motifs in RNA 3D Structures," *Journal of Mathematical Biology*. 56:215–252
- 7: Jamie H. Cate, Anne R. Gooding, Elaine Podell, Kaihong Zhou, Barbara L. Golden, Craig E. Kundrot, Thomas R. Cech, *Jennifer A. Doudna*, September 2005. "Crystal Structure of a Group I Ribozyme Domain: Principles of RNA Packing". *Science*. 273: 1678-1685.
- 8: Zhihua Du, Jinghua Yu, Raul Andino, Thomas L. James, 2003. "Extending the Family of UNCG-like Tetraloop Motifs: NMR Structure of a CACG Tetraloop from Coxsackievirus B3". *Biochemistry*. 42: 4373-4383
- 9: PDB database
<http://www.rcsb.org/pdb/home/home.do>
- 10: NDB database
<http://ndbserver.rutgers.edu/>
- 11: RNAView software available at:
<http://ndbserver.rutgers.edu/services/download/index.html>
- 12: Find RNA 3D (FR3D) web program:
<http://rna.bgsu.edu/WebFR3D/geometric.php>

13: Visual Molecular Dynamics (VMD):

<http://www.ks.uiuc.edu/Research/vmd/>