RNA Secondary Structures as Tree Data Structures with RnaSec

Jon-Michael Deldin
Intro to Bioinformatics
Graduate Project Report

2010-12-17

Contents

0.1 Introduction 2
0.1.1 Background 2
0.1.2 Trees 2
0.1.3 Data 2
0.2 Methods 3
0.3 Results 4
0.4 Discussion 4

0.1 Introduction

For this project, I developed RnaSec, a Ruby library that represents RNA secondary structures as tree data structures[1]. Tree representation is a step towards determining secondary structure similarity using tree edit distance.

0.1.1 Background

With primary structures, we can align sequences rather easily, but alignment programs, such as BLAST[Altschul et al., 1990], do not always produce correct results for certain cases. For example, there may be a secondary structure that is identical to another structure, but it is inverted. A primary sequence alignment would likely output a low similarity score in this case. However, a human would conclude that these two structures are similar. What we need is an automated tool to align secondary structures and determine their similarity. The outcome of this project is part of the solution to comparing structures.

0.1.2 Trees

Tree data structures are a good candidate for representing RNA secondary structures because they have been extensively researched in computer science, which has resulted in a number of algorithms that can be adapted to finding the similarity between two RNA structures. The most relevant of which is tree edit distance, which determines the cost to turn one tree into another.

For this project, trees are loosely based on the representation proposed by Höchsmann [2005, 19]. One of the difficulties in representing RNA as a tree is deciding what the root node should be. In this project, a *root node* is defined as a node that contains two elements that represent the $5'$ and $3'$ ends of the (sub)sequence. For example, the root node may contain the start and ending nucleotides of a hairpin loop.

[1]The original proposal was "Using tree edit distance as an RNA secondary structure similarity metric", but after consulting with Dr. Raiford and researching the subject, I realized I needed to narrow the scope of the project for the semester time frame.

```
procedure RNA-Sequence-To-Tree(data)
  seqs ← FASTA-File-To-Array(data)
  trees ← Array
  for all seqs as seq do
    vienna ← Get-Vienna-Notation(seq)
    tree ← Vienna-To-Tree(vienna)
    trees.append(tree)
  end for
  return trees
```

Figure 1: High level algorithm for converting primary structure into trees.

0.1.3 Data

The data used in this project was a sample of 100 DNA sequences generated randomly and by Systematic Evolution of Ligands by Exponential Enrichment (SELEX) by M. Ellenbecker, J.M. Lanchy, and J.S. Lodmell. The sequences were provided in FASTA format.

0.2 Methods

RnaSec is implemented in Ruby 1.9.2 because it combines Perl's text-processing capabilities with an expressive object-oriented interface.

The tree data structure stores one required and one option element as its value and stores subtrees as an array. I stored the child nodes as an array because a tree can have zero or more subtrees.

The high level algorithm for converting RNA sequences into tree structures is presented in Figure 1. The input data is in FASTA format, which contains a number of label and sequences as shown in Figure 2.

In Figure 1, the tree is constructed from Vienna notation. I selected this format because it is a common, ASCII format produced by many secondary structure predictors. Predictors like `UNAFold`[Markham and Zuker, 2008] and `RNAFold`[Hofacker, 2009] can generate Vienna and graphical representations of secondary structure based on the primary sequence. The downside to Vienna is it does not explicitly represent structures because it's alphabet consists of three characters, periods, opening parenthesis, and closing parenthesis. One can infer the other structures with more complicated parsers, but for this project, only hairpins and bases are detected.

My main script integrates with `RNAFold`[2], which produces output as shown in Figures 3 and 4. The output is fed into a parser that scans the Vienna string and constructs a series of tree objects.

Parsing the root node from Vienna notation is described in Figure 5. The tree building itself is similar, except that it works by moving through the string character by character adding subtrees to the root node.

0.3 Results

RnaSec's driver script was able to read 100 SELEX sequences, generate the Vienna structure with `RNAFold`, and convert the generated trees back to Vienna format to verify correctness. The truncated output is shown in Figure 6.

0.4 Discussion

Overall, the project succeeded in building trees from primary structures. The computed trees were shown to be equivalent to their Vienna structures. However, there were a number of weaknesses in the library. First, the tree definition should be more precisely defined pending further research. While it is acceptable in converting Vienna structures into tree objects, but it may be cumbersome in tree edit distance.

Second, the created trees only supported hairpins and bases; it does not handle bulge loops, interior loops, or junctions. The current string parser

[2] `RNAFold` and `UNAFold` produce similar Vienna structures, so I integrated with the former because it has fewer dependencies.

```
>LABEL-1
GGCATTACGGCCGGG TGTCGCTGATTTCTGAATTTTGTGTGGGG GCGTCTTCTAG
:
:
>LABEL-n
GGCATTACGGCCGGG TGTCGTACTGATTGATGATTTACTCTGGGG GCGTCTTCTAG
```

Figure 2: FASTA format. A line beginning with ">" is a label, and the following line is the corresponding sequence.

```
GGCAUUACGGCCGGGUGUCGUACUGAUUGAUGAUUUACUCUGGGGGCGUCUUCUAG
..((.((((((.....)))))).)).............(((((((...)))))))
```

Figure 3: Vienna representation of aptamer MBE2A generated by `RNAFold`.

should be replaced with a parsing expression grammar (PEG). PEGs are similar to Chomsky's context-free grammars, but choices are prioritized. This will aid in determining terminating bases of hairpin loops and making the parser more robust. Additionally, there is a Ruby library, Treetop, that provides a powerful syntax, allowing one to map methods to tokens[Sobo].

Finally, once a true parser is implemented, work will begin on calculating the tree edit distance to determine similarity.

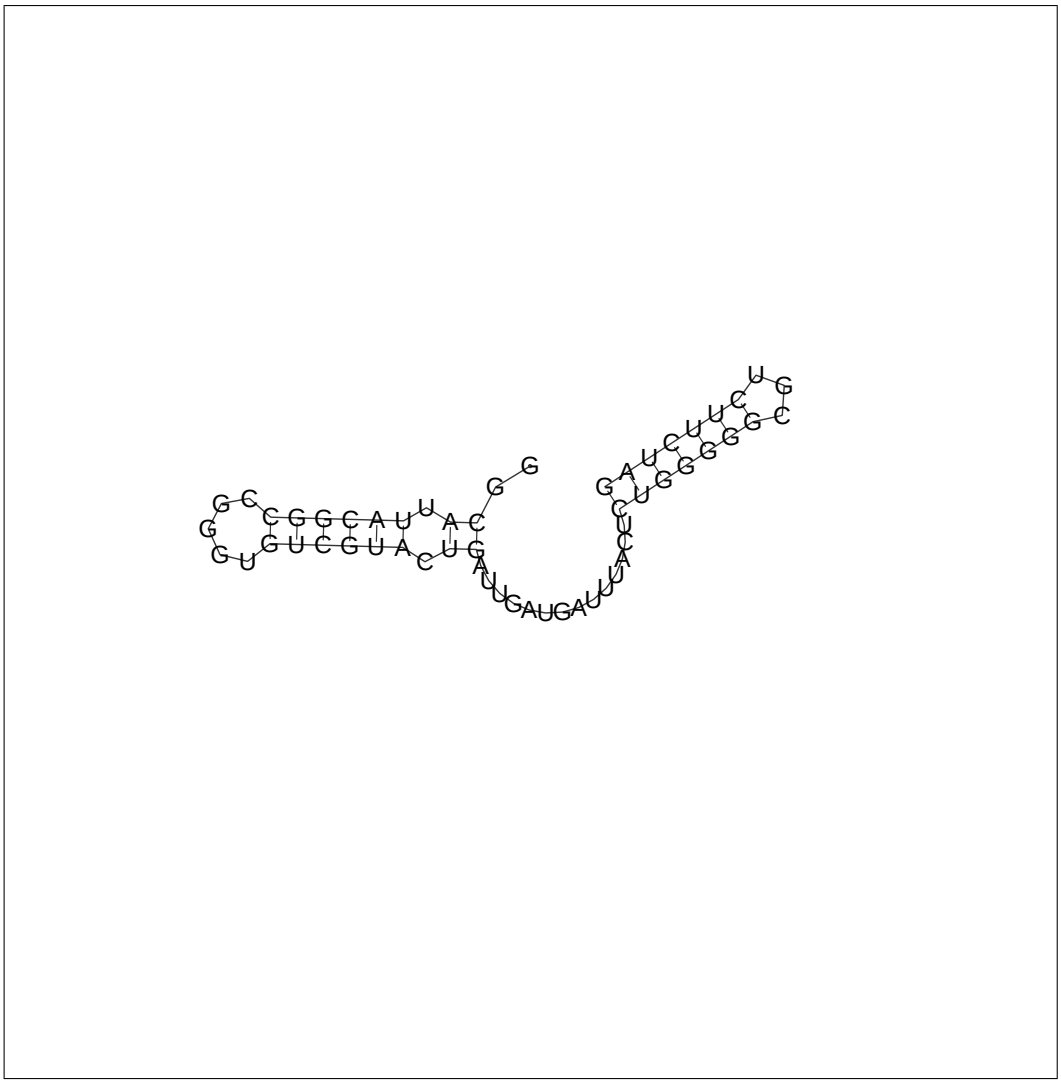

Figure 4: Postscript representation of aptamer MBE2A generated by `RNAFold`.

```
procedure Parse-Root(seq)
  first ← seq[0]
  last ← seq[n]
  if first = '(' AND last = ')' then
    tree ← Tree.new(Hairpin(first), Hairpin.new(last))
  else if first = '.' AND last = '.' then
    tree ← Tree.new(Base.new(first), Base.new(last))
    {Handle asymmetric 5′ and 3′ ends:}
  else if first = '.' AND last = ')' then
    tree ← Tree.new(Base.new(first), Hairpin.new(last))
  else if first = '(' AND last = '.' then
    tree ← Tree.new(Hairpin.new(first), Base.new(last))
  else
    print  Cannot find root
    exit
  end if
  return  tree
```

Figure 5: Algorithm for finding the root node in Vienna notation. The `last` variable represents the last character in the `seq` string. The `Hairpin` and `Base` objects are simplified – they accept a nucleotide and whether it is a start or end pair in the case of hairpins.

```
$ bin/rnasec data/2010-11-15.aptamers.txt
...read 100 entries
...folded 100 sequences
...built 100 trees

...compare RNAFold structure to tree
ggcauuacggccgggugucgcugauuucugaauuuuguguggggggcgucuucuag
        (((......))).(..(((.(((....(........)..))).))))..)......
        (((......))).(..(((.(((....(........)..))).))))..)......
ggcauuacggccgggugucguacugauugaugauuuacucuggggggcgucuucuag
        ..((.((((((.....)))))).)).............(((((((...)))))))
        ..((.((((((.....)))))).)).............(((((((...)))))))
```

Figure 6: Truncated output of `rnasec` on a 100-entry FASTA file. The Vienna structure immediately following the RNA sequence is generated by RNAFold. The following line is a result of converting the tree back to Vienna format to verify correctness.

Bibliography

S F Altschul, W Gish, W Miller, E W Myers, and D J Lipman. Basic local alignment search tool. *J Mol Biol*, 215(3):403–410, Oct 1990. ISSN 0022-2836 (Print); 0022-2836 (Linking). doi: 10.1006/jmbi.1990.9999.

Matthias Höchsmann. *The Tree Alignment Model: Algorithms, Implementations and Applications for the Analysis of RNA Secondary Structures*. PhD thesis, Technischen Fakult at der Universita t Bielefeld, 2005.

Ivo L Hofacker. Rna secondary structure analysis using the vienna rna package. *Curr Protoc Bioinformatics*, Chapter 12:Unit12.2, Jun 2009. doi: 10.1002/0471250953.bi1202s26.

Nicholas R Markham and Michael Zuker. Unafold: software for nucleic acid folding and hybridization. *Methods Mol Biol*, 453:3–31, 2008. doi: 10.1007/978-1-60327-429-6_1.

Nathan Sobo. Treetop: Bringing the simplicity of ruby to syntactic analysis. URL `http://treetop.rubyforge.org/`.