Pattern Matching in RNA: A Third Required Element of Standard Dot-Bracket Notation

October 2010
James F. Lynn www.RNAParse.com
Jlynn@acsalaska.net

Introduction

For the purpose of discussion in this paper, ribonucleic acid (RNA) can generally be thought of as a set of linear string of the characters {AGCU}. By ease of convention, U will be substituted for its DNA counterpart T. Also by ease of convention, the "rules" by which secondary structures are formed will follow standard Watson-Crick (WC) paring, such that A pairs with T, T pairs with A, G pairs with C, and C pairs wit G. It is, however, important to remember that exceptions exist to these rules and in fact many non-WC pairs are formed in many RNA structures as a result of close proximity of nucleotides in a tertiary structure RNA chain.

RNA as a linear Chain

Extensive biochemical process left aside, RNA forms long chains of individual nucleotides that are read from the 5' (Five prime) position to the 3' (Three prime.) as exampled below in figure 1.

```
5' - GGGGCTCAAGGGAGGCCCCAGAAACAAACTTTCCC  - 3'
     ((((((:::[[[[))))))::::::::::::]]]]
```

fig. 1 Short RNA from the Equine Infectious Anemic Virus shown with representation of its secondary structure.

In actuality, figure one represents a complex structure known as a pseudoknot in which several nucleotides in the string loop back on themselves to form a *secondary* structure that greatly adds complexity and imparts functionality to the string. The dot-bracket diagram shown represents how and where nucleotides interact: "( )" and "[ ]" are where pairs are formed and ":" represents unpaired nucleotides. Of note, the structure above cannot be unambiguously represented using one set of brackets , thus "[ ]" is added to represent the second stem of the structure. Another meaningful representation (figure 2) of the RNA above shows another nature of how the RNA is formed in space.

```
5' - GGGGCT CAA GGGA GGCCCC AGAAACAAACTT TCCC  - 3'
        |         |     |                  |
        |_________|_____|                  |
                  |                        |
                  |________________________|
```

fig. 2 Lines clearly show that the two stems formed have "crossing structure."

A successful computational pattern match of the secondary structure above must thus take into account four things, the left side of a paired stem (e.g. GGGGCT), the right side of a paired stem (e.g. GGCCCC), loops or unpaired nucleotides (e.g CAA.) and must also account that two stem-loop structures are present *and* either cross or do not cross – the latter being the great rub in computational pattern matching as stem-loop structures are much easier to match than pseudoknoted structures.

Simple stem structures are "context-free" while crossing systems of stems are "context-sensitive." Loops, the unpaired part of our string can be described using simple regular expressions such as "find -CAA-" in some given string.

To reiterate a bit, There are only two combinations possible that form RNA secondary structure – the stem and the loop by which all other patterns that are formed are variations.

```
-GCCC AAT GGGC-
 |_________|
```

fig. 3 Stem and loop where GCCC and GGGC are complementary pairs and ATT is unpaired.

A new structure that requires added notation: Tandem compliment repeats.

In the course of our work we realized there is a possible third combination of RNA strings, the tandem compliment repeat (TCR) which cannot be unambiguously represented using standard bracket notation. This occurs in a string when compliment pairs form a stem-loop both in the same direction (stems are bi-directional.) The best way to describe it is with a figure as shown below.

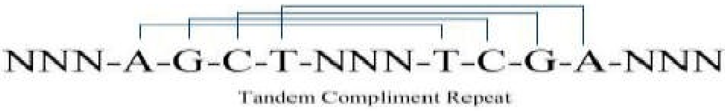

fig. 4 Tandem compliment repeat. (N = loops of any nucleotide) A context-sensitive structure not unlike a pseudoknot.

If we try to diagram a structure that contains a TCR we must introduce a third notation, curly brackets "{ }" to prohibt ambiguity in its representation: Consider the following hypothetical RNA as shown in below.

```
-AAAA  loop AGCT loop  GGGG loop TTTT loop TCGA loop CCCC-
 ((((  :::: :::: ::::  [[[[ :::: )))) :::: :::: :::: ]]]]
```

Standard notation ignoring TCR (underlined.)

```
-AAAA  loop AGCT loop  GGGG loop TTTT loop TCGA loop CCCC-
 ((((  :::: (((( ::::  [[[[ :::: )))) :::: )))) :::: ]]]]
 ((((  :::: [[[[ ::::  [[[[ :::: )))) :::: ]]]] :::: ]]]]
```

Attempts at using standard notation of TCR (underlined.) Which "( )[ ]" gos with which stem? You may also note that the TCR is not accurately depicted as both "( )" and "[ ]" represent bi-directionality. Thus, I've introduced a third set of characters, "{}" to resolve these conflicts:

```
-AAAA  loop AGCT loop  GGGG loop TTTT loop TCGA loop CCCC-
 ((((  :::: (((( ::::  [[[[ :::: )))) :::: )))) :::: ]]]]
 ((((  :::: [[[[ ::::  [[[[ :::: )))) :::: ]]]] :::: ]]]]
 ((((  :::: {{{{ ::::  [[[[ :::: )))) :::: }}}} :::: ]]]]
```

Thus the TCR "AGCT...TCGA" is unambiguously represented as "{{{{}}}}"

Discussion

All possible secondary structures are represented using dot bracket notation “( )” for stems, “[ ]” for pseudoknots, “{ }” for TCRs, and “:” or “.” for loops. Stems and pseudoknots have been empirically shown to exist in RNAs but TCRs have not. For purposes of this introduction the question of existence is somewhat moot – we introduced the new notation as a means of describing a third kind of *possible* configuration that seems to be overlooked in the scientific literature up until now. Through research we have shown that TCRs are common in RNA but make no claims as to their importance. Given very large strings of a small alphabet such as RNA, nearly any configuration is possible per random chance. None the less, we have developed grammar-only methods to parse TCRs from RNA strands. An example of such an application may be downloaded from the “downloads” section of www.rnaparse.com