Exploring the Space of Consensus RNA Secondary Structure Motifs Using Suffix Arrays

Truong Nguyen and Marcel Turcotte

School of Information Technology and Engineering

University of Ottawa, 800 King Edward, Ottawa, Ontario K1N 6N5, Canada

Abstract

In the last few years, we have seen a rapid increase of the number of known RNA families. For a significant fraction of them, the mechanisms of action remain unclear. Their signature combines structure and sequence information. In most cases, they are difficult to identify from sequence alone. Traditional approaches to identify RNA motifs seek to find a conserved structure with minimum free energy in an ensemble of aligned sequences.

We present a novel approach for discovering consensus secondary structure motifs in a set of unaligned RNA sequences. The secondary structure motifs combine sequence and structure information. State-of-the-art data structures, suffix arrays in particular, are used to enumerate exhaustively the space of possible motifs. Suffix arrays (SAs) are used for two purposes. First, to enumerate efficiently stem structures, including internal loops. Second, SAs are used to match secondary structure expressions. The algorithms have been implemented in a software system called Seed.

Applications of the approach on test cases shows that i) complex search spaces can be exhaustively explored and that ii) the search spaces contains biologically relevant candidates.

Keywords: nucleic acids, structure, motifs discovery.

1. Background

The history of molecular biology is punctuated by a series of discoveries demonstrating the surprising breath of biological roles of RNAs. The identification of novel non-protein coding RNAs (ncRNAs) requires extensive human examination. This paper presents a new software system that allows searching exhaustively the space of RNA sequence and structure motifs, therefore assisting the identification and characterisation of new motifs.

The repertoire of known ncRNAs has grown rapidly[1]. The housekeeping roles of RNAs, such as those of tRNA, rRNA, RNAseP, snRNA and snoRNA, were discovered early. While in the recent years, it became clear that RNAs also have important regulatory functions. Examples include microRNAs, which regulate the expression of protein genes by targeting a complementary region of their mRNAs. MicroRNAs constitute one of the most abundant classes of regulatory molecules, and are key to many developmental processes[2]. Several discoveries collectively demonstrate that untranslated messenger RNAs can sense the level of metabolites, and modulate the expression of certain genes accordingly. Those RNAs are referred to as RNA sensors and riboswitches, and have been reviewed[3, 4]. Post-transcriptional regulation of gene expression often involves secondary structure elements located in the untranslated regions of mRNAs[5]. Through all those discoveries, a new understanding of gene expression regulation is emerging.

Lately, several resources have been established to help understanding the RNA universe. Sequence and functional elements of the 5' and 3' untranslated regions of eukaryotic mRNAs are collected in the UTRdb and UTRsite databases[6]. Sequence information is available from the Noncoding RNAs database[7]. Finally, Rfam compiles a large collection of multiple sequence alignments and covariance models for many common non-coding RNA families[8].

Knowledge about RNA secondary structure motifs serves two important purposes. First, an RNA secondary structure motif presents the essential/conserved features of an RNA family. It directs the research efforts by restricting the set of hypotheses to be tested. Ultimately, a motif may help formulating a mechanism of action. Second, an RNA motif helps finding new members of an RNA family. Once identified, the candidate must be validated experimentally.

Regular languages have been successfully used to characterise DNA and protein sequence patterns. Algorithms have been developed to automatically infer sequence motifs, either from aligned or unaligned sequences. With RNAs, the sequence information alone is generally not enough. Members

of an RNA gene family cannot be found using only the sequence information of other known members.

Several database searching techniques have been developed that combine sequence and structure information. Approximate matching and stochastic models are two approaches to accommodate for the fact that biological data are often incomplete, and may contain errors. BIOSMatch is an example of a software system for the approximate matching of secondary structure expressions[9]. While covariance models are examples of the later[10, 11]. Approaches that are general, are also extremely time/memory consuming. Consequently, specialised programs are often developed to recognise members of a given family.

However, there are few algorithms that directly address the problem of finding those motifs. Much work has been done on predicting RNA structure through energy minimization. Simply, the free energy of an RNA molecule is modeled as the sum of the contributions of independent cycles/loops (so called nearest neighbour model[12]). Melting experiments are performed to determine the free energy parameters for small structures. Since the free energy can be decomposed into a sum of independent loop contributions, it can be solved exactly and efficiently when formulated as a dynamic programming problem[13]. Steady progress has been made, mainly through the determination of more complete and accurate sets of free energy parameters[14]. The computer programs mfold[13] and RNAfold[15] are two widely used implementations. The performance of mfold, versions 2.3 and 3.1, has been evaluated on a large dataset[16]. However, there are several reasons why free energy minimization can fail.

- The lowest free energy conformation may not coincide with the native conformation. This can be due to experimental errors in determining the free energy parameters, errors due to the extrapolation of the parameters, or simply because there are numerous lowest free energy conformations, and it can be difficult to distinguish the native conformation from the others;
- Certain classes of RNA have more than one active structure. This is the case for several RNA regulatory elements termed riboswitches[3, 4, 17];
- The nearest neighbour model does not take into account the contributions of the cellular environment: proteins, other RNAs, metabolites and solvent. Such contributions may be particularly important for modeling regulatory elements present the untranslated regions of mRNAs. Similarly, RNAs are often modified after their transcription, the modifications can play an important role while folding;
- Pseudo-knot structures are often not considered. For some RNAs, neglecting the contributions of pseudo-knots may entail that the native conformation and the lowest free energy conformation are quite different. However, taking into account pseudo-knots severely increases the time/space complexity of the algorithms. There is also a lack of experimental data that can be used to deduce the free energy parameters.

The accuracy of the predictions can be increased significantly if a multiple sequence alignment is used as input. These sequences are assumed to share a common secondary structure. Hofacker et al. incorporated a new term into the total energy function for taking into account covariations[18]. This approach has been implemented in the program RNAalifold. The number of required input sequences is less than that of traditional covariations analyses, yet the results are superior to the implementations based on a single input sequence. Tahi, Gouy and Régnier have taken a different approach deciding not to include thermodynamics constraints into their program, DCFold[19]. This software system handles large sequences, and was reported to effectively recover the common secondary structure of rRNAs. The relative performance of comparative RNA structure prediction approaches has been evaluated recently[20].

Stochastic context-free grammars are a powerful paradigm allowing for both the inference and the database searching of secondary structure motifs[10, 11, 21]. However, the secondary structure inference works best when the input consists of a set of aligned sequences, and the heavy time/space complexity limits their application to small sequences.

Often, an alignment is not readily available. It could be that the similarity of the sequences is too low to construct a multiple sequence alignment; consequently, knowledge about the secondary structure would be required to construct a reliable alignment. Or, alternatively, the common motif perhaps represents a small portion of each sequence; and it can be discontinuous.

David Sankoff has developed recurrence equations to simultaneously fold and align k sequences ([22], to be more precise, the work also proposes the reconstruction of the ancestral sequence on a phylogenetic tree, a real “tour de force”). Dynalign is an implementation for 2 sequences[23]. It differs from the original proposal in that there are no substitution costs present in the recurrence equation. Prohibitive time/space complexity limits its application to sequences that are a few hundreds nucleotides long,

and approximately the same size. Indeed, the maximum distance between the aligned nucleotides is restricted by a factor m, when $m \sim n$, where n is the size of the input sequences, the complexity of the algorithm is $O(n^6)$. In order to reduce the complexity of the problem, Gorodkin et al. are focusing on hairpin structures only[24, 25].

When a high quality alignment is available, comparative analysis has proven to be an effective approach[26]. Following the experimental determination of the structure of two ribosomal RNAs, 30S and 50S, [27] reported that over 97% of the base pairs predicted by comparative analysis were correct. It is accepted that homologous sequences, sequences that are related by common ancestry, adopt a similar structure. The similarity of the sequences can be low, yet the majority of the base pairs will be preserved. Consequently, in a multiple sequence alignment that reflects this structural homology, pairs of columns that correspond to nucleotides involved in base pairs show a high degree of covariations (simultaneous coordinated changes). Secondary structure elements (stems) are seen as pairs of segments *i:j*, *i+1:j-1* … with correlated changes. The degree of association is often quantified using the mutual information content[28, 29]. The analysis is powerful enough to detect reliably tertiary interactions as well. The application of this approach is limited by 1) the availability of related yet divergent sequences and 2) more importantly, by the difficulty to build a reliable alignment without prior information about the structure. Accordingly, comparative analyses are mostly done by hand, iteratively, starting with the most conserved sequences[16].

The methods presented thus far are designed to find complete secondary structures. In fact, most approaches predict the secondary structure for the individual sequences then seek to find common secondary structure elements. Sometimes, the application requires focusing a restricted subset of the secondary structure. We present a method that focuses on finding secondary structure motifs.

2. Algorithms

2.1. Overview and design issues

Seed is a data exploration tool specifically designed to search a space of conserved RNA secondary structure motifs. We list here the main issues that influenced its design. The space of valid RNA secondary structures is extremely large, even when restricted to a given input sequence; exponential w.r.t. the length of the input sequence[30]. In order to make this search space more tractable, we adopt a data-driven approach. A seed sequence serves to induce a search space that is exhaustively explored for finding motifs that also match a significant fraction of the k input sequences. The search space is traversed from the most general to the most specific motif. Whenever a motif is found that is not supported (does not match enough input sequences) the motif and its descendant are pruned from the search space.

The assumption that the input sequences share some common features may not be true. Accordingly, the motif discovery algorithm performs a "sequential covering" of the input sequences. This means the algorithm repeatedly selects a sequence, searches the space of motifs induced by that sequence, selects the "best" motif and removes all the input sequences that match the "best" motif. The process stops when all the input sequences have been processed. The top level organisation of Seed is as follows.

1. Select a seed sequence;
2. Construct the most specific motif;
3. General-to-specific search of the motif space;
4. Select the "best" motif;
5. Remove all the input sequences containing the selected motif;
6. If there are no more sequences then stop, otherwise goto 1.

No heuristics are used, the algorithm exhaustively explores the space of RNA secondary structure motifs for a given input and set of parameters. Execution times, although large, are small compared to the whole process of identify, proving and characterising RNA motifs experimentally.

2.2. Suffix trees and suffix-arrays

Suffix trees are a prominent data structure in computational biology, powering efficient sequence comparison and repeat finding algorithms that can be applied to genomic scale data[31, 32]. A related data structure, suffix arrays[33], offers some advantages over suffix trees, namely reducing the memory requirements and easier to implement algorithms[34]. Important and recent achievements now allow use of suffix arrays every where suffix trees were used[34]. Those achievements are: a direct approach for the linear-time construction of the suffix array[35-37], an algorithm for finding the longest-common-prefix in linear-time[38], and simulating the bottom-up[39] and top-down[40] traversal of suffix trees, also in linear time. We used suffix arrays for the implementation of Seed but we will use suffix trees herein for clarity. [34] shows the relationships between the two data structures.

A suffix tree for a text $T = t_1 \ldots t_n$ is a rooted labelled tree such that,

- the edges of the tree are labelled with substrings of the text;
- each internal node has at least two children, with the possible exception of the root of the tree;
- any two outgoing edges of the same internal node start with a different letter;
- every suffix of the text is spelled out on a path from the root to a leaf, and that leaf is labeled with the start position of that suffix.

Several algorithms and implementation techniques have been proposed for constructing the data structure in linear-time and space. Applications include pattern matching and repeat finding. A pattern P occurs in a text T iff the suffix tree of T contains a path (from the root of the tree) that spells P; this follows from the fact that P occurs in T iff P is the prefix of at least one suffix of T. A suffix tree exposes all the internal repeats of a text. By definition, every internal node has at least two descendants, corresponding to suffixes that share a common prefix, spelled out on the path from the root to that node.

A generalized suffix tree is a suffix tree that contains all the suffixes of two or more strings. A generalized suffix tree allows finding substrings that are common to an ensemble of strings.

Briefly, a suffix array for a text $T = t_1 \ldots . t_n$ is an array of integers that specifies the lexicographic order of the suffixes of T; each entry of this array is the start position a suffix of T. This simple data structure is enhanced by pre-calculating other indexing structures in order to perform the top-down and bottom-up traversal, as well as calculating the longest-common-prefix.

2.3. Most specific motif

The search space is induced from a seed sequence that has been selected in the first step of the algorithm. The basic algorithm is as follows.

1. Construct a suffix tree for T and T^R;
2. For every starting position i in T, $1 \ldots n$;
   a) For every starting position j in T^R, such that $j=|T|-i-L+1$;
   b) find the lowest common ancestor, l, of i and j
   c) if the length of the complementary region is larger than some user defined value then save this stem.

where T^R is the reverse complement of T, and L is a user defined constraint on the maximum distance between the 5' end and the 3' end of a stem. The basic algorithm is extended in two ways. First, up to e mismatches per stem are allowed. This involves adding an inner loop, executed e times. For each iteration, i and j are incremented by one. This increases the time complexity by a factor e. The second extension allows for interior loops; where the maximum length for interior loops is bound by a used defined constant m.

The location of each stem is recorded to be used in the later stages of the algorithms. Similar ideas have been proposed by Gusfield[41], for suffix trees only.

By using suffix arrays and range minimum query, we are enumerating stems more efficiently than GPRM[42, 43], $O(n + em^2Ln)$ instead of $O(L^3n)$.

2.4. General to specific search

The search algorithm consists of three distinct phases: initialisation, instantiation and composition. During the first phase, the algorithm initialises a list (queue) of open nodes to contain structural motifs (see below). The motifs have been derived from the selected seed sequence. Only the motifs that have a minimum support, i.e. that also match other sequences from the input set, are part of the list. Structural motifs have no base pair instantiated.

In the second phase, all the possible sequence instantiations for every motif of the open list are considered. Systematically and exhaustively, all the base pairs of every stem motif in the open list are replaced by the actual base pair that occurs in the seed sequence. This information is readily available since the location of every stem within the seed sequence has been saved. Each newly created instance is matched against the remaining sequences. Only the motifs that have a minimum level of support are added at the rear of the queue. Figure 1 illustrates this process for a single stem. In the actual implementation, the instantiations of all the motifs currently in the open list are interleaved. In other words, all the one base pair motifs for all the structural motifs are explored first, followed by the exploration of all the two base pair motifs, and so on. Progressively, all the possible instantiations are validated. This is done efficiently so that the same instantiation is never considered twice. At the end of the second phase, the open queue contains a mixture of structural, partially and fully instantiated motifs, all consisting of a single stem.

Finally, the third phase consists of creating multi-stems motifs by selecting and composing two motifs at a time from the open list. The composition of two motifs is dictated by their occurrence within the seed sequence. Given two motifs, there are two possible relationships. One motif follows the other or one motif is nested within the other. The seed sequence is used to determine which relationship to use and to calculate the distance parameters. Motifs that are structurally

invalid (because they overlap in the sequence space) or that don't have the required minimum support are discarded. During the execution of the third phase, the open list contains a mixture of single and multi-stems motifs, that are structural, partially or fully instantiated.

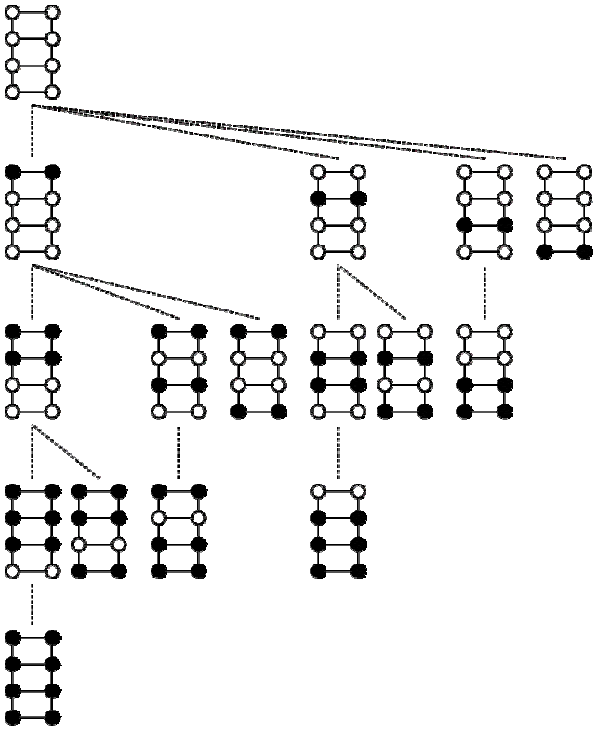

Fig. 1: Schematic illustration of the sequence instantiation process. Open circles correspond to generic base pairs, n:n', while the filled circles represent specific base pair, a:u, u:a, c:g, g:c.

2.5. Objective function

Our research has focused on developing a framework allowing to exhaustively search a space of possible secondary structure motifs. The objective function that has been used consists of calculating the information content of the motif. See the Results section for further details.

2.6. Matching algorithm

For all three phases of the search algorithm, the newly created motifs must be matched against the *k-1* remaining sequences in order to determine the level of support.

2.7. Suffix array-based matcher

We introduce an algorithm for matching secondary structure expressions. The basic idea is to "thread" a secondary structure expression onto the suffix tree of the input sequence. This means simultaneously traversing the expression, from its 5' end, and the suffix tree, starting from its root.

The main steps of this algorithm are as follows. First, build a suffix tree for the input string. Then, match the characters of the secondary structure expression along the unique path in the suffix tree until either 1) the end of the secondary structure expression has been reached, 2) the end of a branch has been reached, 3) a mismatch has been found, or 4) the secondary structure expression contains a joker (don't care symbol, any base type should be allowed).

In the former case, every leaf of the subtree below the last match represents the starting location of an occurrence. For cases 2 and 3, this is a failure and the algorithm must restart from the last branch point (see below), if there are no more branch points, this means the expression does not occur in the input sequence.

Finally, case 4, there are four issues to be considered: the joker occurs in a loop region, the joker occurs in the 5' end region of a stem, it occurs in the 3' end region of a stem, or it occurs at an internal node (fork) of the suffix tree.

For the first three issues, it is assumed that the last match was not the last letter of the label of an edge (it occurred at some intermediate position). The first issue is easy to deal with; the next character along this path is accepted. Second issue, a joker has been found in a 5' end region of a stem. The algorithm accepts the next symbol along the current path, and pushes that symbol onto a stack. Next and third issue, a joker is encountered in a 3' end region of a stem, the top of the stack contains the base that occurred at the 5' position of the pair, if the next character along the current path inside the tree is its complement then the top element is discarded and the algorithm continues, otherwise this is a failure and the algorithm restarts from an earlier branch point, or stop indicating a failure. Finally, whenever a joker is found, and the last match occurred at the end of a label, then a branch point must be created. Effectively, this means that the algorithm is applied recursively for all the edges out of the internal node where the last match has occurred. The system stack serves to memorise all the branch points. When the end of a secondary structure expression is reached (case 1) the stack must also be empty; otherwise, the expression is not valid.

The algorithm can answer two specific questions: 1) does this secondary structure expression occur in this input string? and 2) how many occurrences of this

expression are there? For the decision question, the algorithm stops whenever the end of the secondary structure expression is found. For the later question, all the leaves of the subtree below the node where the last character of the expression was matched must be visited in order to count the number of occurrences.

Figure 2 illustrates the search. When a joker is found in a 5' end region of a stem, the current path, in the suffix tree, is extended by one character, which is also pushed onto a stack. Later, when a joker is found in a 3' end region of a stem, the character onto the top of the stack must be the complement of a valid extension of the current path.

Determining the level of support is critical, as this allows pruning the search tree. The matching algorithm is called for each newly created node, therefore, most of the CPU time is spend matching sequences.

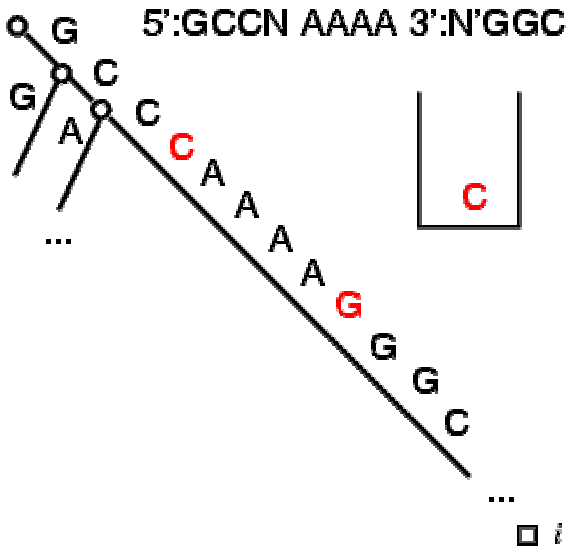

Fig. 2: Secondary structure matching.

3. Results and discussion

We validated the approach using the secondary structure model proposed by Le Quesne et al. for the c-myc IRES[44]. The eight sequences that served to derive the model were used as input; the sequences are approximately 400 nucleotides long. We also applied mfold, a widely used computer program for secondary structure prediction. Within the top 1000 motifs having the highest information content, 279 motifs overlapped with Le Quesne et al.'s model. The positive predictive value for those motifs varied from 23.5% to 100%, while the coverage varied from 7.8% to 21.1%. PPV is defined as the fraction of the predicted base pairs that are also occurring in the experimentally derived model. The coverage is the fraction of the base pairs from the model that are predicted. The low coverage is expected since Seed is aimed to produce small motifs. On the same dataset, the PPV score measured for mfold was 12.3%, while the coverage was 35%.

Our objective of building a software system capable of enumerating exhaustively all possible secondary structure motifs, w.r.t. a user defined set of parameters, has been attained. Furthermore, the set of motifs produced by the software system contains biologically relevant candidates.

Several improvements to the algorithms are considered. Such as replacing the suffix array-based matcher by an algorithm derived from Myers' generalisation the Cocke-Younger-Kasami[45]. However, the most urgent issue is to study alternative objective functions.

Information content alone cannot distinguish biologically relevant motifs from the rest. We are currently investigating alternative scoring functions. Recently, we extended the work of Mathews and Turner, and implemented a software system for the simultaneous alignment and structure prediction of three RNA sequences[46, 47]. We will compare a function based on a linear combination of the free energy of all the matches to a more complex information-based function that takes into account positive and negative examples, as well as the complexity of the motif.

Primary sequence information contributes to defining the identity of RNA motifs. Indeed, in the case of the T box system, for example, the acceptor end of an uncharged tRNA forms base pairs with the antiterminator element[48]. Accordingly, the identity of the bases in the loop regions will also be considered.

4. Conclusions

Determining RNA secondary structure motifs is important for understanding the structure-function relationship and post-transcriptional regulation, as well as identifying RNA targets. We presented a combinatorial algorithm for the detection of RNA secondary structure motifs that are common to a set of unaligned sequences. To our knowledge, this is the first algorithm that directly attempts to exhaustively explore the space of sequence and structure motifs using suffix arrays. We also introduced a new suffix arrays-based algorithm for matching RNA secondary expressions. Our next research efforts will be focusing on improving the objective function, and therefore the ability of Seed to discriminate biologically interesting motifs from the rest.

5. References

1. Storz, G., *An Expanding Universe of Noncoding RNAs.* Science, 2002. **296**: p. 1260-1263.
2. Bartel, D.P., *MicroRNAs: Genomics, Biogenesis, Mechanism, and Function.* Cell, 2004. **116**: p. 281-297.
3. Lai, E.C., *RNA Sensors and Riboswitches: Self-Regulating Messages.* Current Biology, 2003. **13**: p. R285-R291.
4. Nudler, E. and A.X. Mironov, *The riboswitch control of bacterial metabolism.* Trends Biol. Sci., 2004. **29**(1): p. 11-17.
5. Mignoe, F., et al., *Untranslated regions of mRNAs.* Genome Biology, 2003. **3**(3): p. 0004.1-0004.10.
6. Pesole, G., et al., *UTRdb and UTRsite: specialized databases of sequences and functional elements of 5' and 3' untranslated regions of eukaryotic mRNAs. Update 2002.* Nucl. Acids Res., 2002. **30**(1): p. 335-340.
7. Szymannski, M., V.A. Erdmann, and J. Barciszewsk, *Noncoding regulatory RNAs database.* Nucl. Acids Res., 2003. **31**(1): p. 429-431.
8. Griffiths-Jones, S., et al., *Rfam: an RNA family database.* Nucl. Acids Res., 2003. **31**(1): p. 439-441.
9. El-Mabrouk, N. and M. Raffinot, *Approximate matching of secondary structures*, in *Proceedings of the sixth annual international conference on computational molecular biology (RECOMB)*. 2002, ACM. p. 156-164.
10. Eddy, S.R. and R. Durbin, *RNA sequence analysis using covariance models.* Nucl. Acids Res., 1994. **22**(11): p. 2079-88.
11. Sakakibara, Y., et al., *Stochastic context-free grammars for tRNA modeling.* Nucl. Acids Res., 1994. **22**: p. 5112-5120.
12. Borer, P.N., et al., *Stability of Ribonucleic acid Double-stranded Helices.* J. Mol. Biol., 1974. **86**: p. 843-853.
13. Zuker, M. and P. Stiegler, *Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information.* Nucl. Acids Res., 1981. **9**: p. 133-148.
14. Mathews, D.H., et al., *Expanded Sequence Dependence of Thermodynamic Parameters Improves Prediction of RNA Secondary Structure.* J. Mol. Biol., 1999. **288**: p. 911-940.
15. Hofacker, I.L., et al., *Fast Folding and Comparison of RNA Secondary Structures.* Monatshefte für Chemie, 1994. **125**: p. 167-188.
16. Doshi, K.J., et al., *Evaluation of the suitability of free-energy minimization using mearest-neighbor energy parameters for RNA secondary structure prediction.* BMC Bioinformatics, 2004. **5**: p. 105.
17. Voss, B., C. Meyer, and R. Giegerich, *Evaluating the predictability of conformational switching in RNA.* Bioinformatics, 2004. **20**(10): p. 1573-1582.
18. Hofacker, I.L., M. Fekete, and P.F. Stadler, *Secondary Structure Prediction for Aligned RNA Sequences.* J. Mol. Biol., 2002. **319**: p. 1059-1066.
19. Tahi, F., M. Gouy, and M. Régnier, *Automatic RNA secondary structure prediction with a comparative approach.* Computers and Chemistry, 2002. **26**: p. 521-530.
20. Gardner, P.P. and G. Robert, *A comprehensive comparison of comparative RNA structure prediction approaches.* BMC Bioinformatics, 2004. **5**: p. 140.
21. Knudsen, B. and J. Hein, *Pfold: RNA secondary structure prediction using stochastic context-free grammars.* Nucleic Acids Res, 2003. **31**(13): p. 3423-8.
22. Sankoff, D., *Simultaneous solution of RNA folding, alignment and protosequence problems.* SIAM J. Appl. Math., 1985. **45**(5): p. 810-825.
23. Mathews, D.H. and D.H. Turner, *Dynalign: An Algorithm for Finding the Secondary Structure Common to Two RNA Sequences.* J. Mol. Biol., 2002. **317**: p. 191-203.
24. Gorodkin, J., L.J. Heyer, and G.D. Stormo, *Finding the most significant common sequence and structure motifs in a set of RNA sequences.* Nucl. Acids Res., 1997. **25**(18): p. 3724-3732.
25. Gorodkin, J., S.L. Stricklin, and G.D. Stormo, *Discovering common stem-loop motifs in unaligned RNA sequences.* Nucl. Acids Res., 2001. **29**(10): p. 2135-2144.
26. Woese, C.R. and N.R. Pace, *Probing RNA Structure, Function and History by Comparative Analysis*, in *The RNA World*, R.F. Gesteland and J.F. Atkins, Editors. 1993, Cold Sprint Harbor Laboratory Press. p. 91-117.
27. Gutell, R.R., J.C. Lee, and J.J. Cannone, *The accuracy of ribosomal RNA comparative*

structure models. Curr. Opin. Struct. Biol., 2002. **12**(3): p. 301-310.
28. Chiu, D.K. and T. Kolodziejczak, *Inferring consensus structure from nucleic acid sequences.* CABIOS, 1991. **7**: p. 347-352.
29. Gutell, R.R., et al., *Identifying constraints on the higher-order structure of RNA: continued development and application of comparative sequence analysis methods.* Nucl. Acids Res., 1992. **20**: p. 5785-5795.
30. Zuker, M. and D. Sankoff, *RNA Secondary Structure and Their Prediction.* Bulletin of Mathematical Biology, 1984. **46**(4): p. 591-621.
31. Kurtz, S., et al., *REPuter: the manifold applications of repeat analysis on a genomic scale.* Nucl. Acids Res., 2001. **29**(22): p. 4633-4642.
32. Kurtz, S., et al., *Versatile and open software for comparing large genomes.* Genome Biology, 2004. **5**(2): p. R12.
33. Manber, U. and G.E. Myers, *Suffix arrays: a new method for on-line string searches.* SIAM J. Comput., 1993. **22**(5): p. 935-948.
34. Abouelhoda, M.I., S. Kurtz, and E. Ohlebusch, *Replacing suffix trees with enhanced suffix arrays.* Journal of Discrete Algorithms, 2003. **2**: p. 53-86.
35. Käkkäinen, J. and P. Sanders, *Simple Linear Work Suffix Array Construction*, in *Annual Symposium on Combinatorial Pattern Matching*. 2003, Springer-Verlag: Berlin. p. 943-955.
36. Kho, P. and S. Aluru, *Space efficient linear time construction of suffix arrays*, in *Annual Symposium on Combinatorial Pattern Matching*. 2003, Springer-Verlag: Berlin. p. 200-210.
37. Kim, D.K., et al., *Linear-time construction of suffix arrays*, in *Annual Symposium on Combinatorial Pattern Matching*. 2003, Springer-Verlag: Berlin.
38. Kasai, T., et al., *Linear-Time Longest-Common-Prefix Computation in Suffix Arrays and Its Applications*, in *Annual Symposium on Combinatorial Pattern Matching*. 2001, Springer-Verlag: Berline. p. 181-192.
39. Abouelhoda, M.I., S. Kurtz, and E. Ohlebusch, *The Enhanced Suffix Array and its Applications to Genome Analysis*, in *2nd Workshop on Algorithms in Bioinformatics*. 2002, Springer-Verlag. p. 449-463.
40. Abouelhoda, M.I., S. Kurtz, and E. Ohlebusch, *Optimal Exact String Matching Based on Suffix Arrays*, in *9th International Symposium on String Processing and Information Retrieval*. 2002, Springer-Verlag: Berlin. p. 31-43.
41. Gusfield, D., *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. 1997: Cambridge University Press.
42. Hu, Y.J., *Prediction of consensus structural motifs in a family of coregulated RNA sequences.* Nucleic Acids Res, 2002. **30**(17): p. 3886-93.
43. Hu, Y.J., *GPRM: A genetic programming approach to finding common RNA secondary structure elements.* Nucleic Acids Res, 2003. **31**(13): p. 3446-9.
44. Le Quesne, J.P., et al., *Derivation of a structural model for the c-myc IRES.* J Mol Biol, 2001. **310**(1): p. 111-26.
45. Myers, G., *Approximately matching context-free languages.* Information Processing Letters, 1995. **54**: p. 85-92.
46. Masoumi, B. and M. Turcotte. *Simultaneous Alignment and Structure Prediction of RNAs: Are Three Input Sequences Better than Two?* in *International Conference on Computational Science (ICCS 2005), Lecture Notes in Computer Science 3515*. 2005. Atlanta, USA: Springer Verlag.
47. Masoumi, B. and M. Turcotte, *Simultaneous alignment and structure prediction of three RNA sequences.* International Journal of Bioinformatics Research and Applications, In Press - 2005.
48. Grundy, F.J. and T.M. Henkin, *Regulation of gene expression by effectors that bind to RNA.* Curr. Opin. Microbiology, 2004. **7**: p. 126-131.