Automated Discovery of Active Motifs in Multiple RNA Secondary Structures

Jason T. L. Wang
Computer & Information Sci. Dept.
New Jersey Institute of Technology
Newark, New Jersey 07102, USA
jason@village.njit.edu

Bruce A. Shapiro
Division of Basic Sciences
National Institutes of Health
Frederick, Maryland 21702, USA
bshapiro@ncifcrf.gov

Dennis Shasha
Courant Institute of Math. Sciences
New York University
New York, New York 10012, USA
shasha@cs.nyu.edu

Kaizhong Zhang
Computer Science Dept.
The University of Western Ontario
London, Ontario, Canada N6A 5B7
kzhang@csd.uwo.ca

Chia-Yo Chang
Computer & Information Sci. Dept.
New Jersey Institute of Technology
Newark, New Jersey 07102, USA
changc@homer.njit.edu

Abstract

In this paper we present a method for discovering approximately common motifs (also known as active motifs) in multiple RNA secondary structures. The secondary structures can be represented as ordered trees (i.e., the order among siblings matters). Motifs in these trees are connected subgraphs that can differ in both substitutions and deletions/insertions. The proposed method consists of two steps: (1) find candidate motifs in a small sample of the secondary structures; (2) search all of the secondary structures to determine how frequently these motifs occur (within the allowed approximation) in the secondary structures. To reduce the running time, we develop two optimization heuristics based on sampling and pattern matching techniques. Experimental results obtained by running these algorithms on both generated data and RNA secondary structures show the good performance of the algorithms. To demonstrate the utility of our algorithms, we discuss their applications to conducting the phylogenetic study of RNA sequences obtained from GenBank.

Introduction

Data mining is fun and useful (Agrawal 1994). Most of the research has been concentrating on record-oriented applications (Piatetsky-Shapiro & Frawley 1991; Silberschatz, Stonebraker, & Ullman 1991; Han, Cai, & Cercone 1993). Here we study a different type of data mining, namely, discovering structural patterns in scientific data. We focus on finding approximately common motifs (also known as *active motifs*) in multiple RNA secondary structures. This problem is important in computational biology (Le *et al.* 1989). For example, in predicting secondary structures for a given mRNA, one may first find a set of 'optimal' and 'suboptimal' structures using existing algorithms (Zuker 1989). Then to determine which one among these structures is closest to the one occurring naturally, one may search for active motifs in the structures (Le *et al.* 1989). The motifs appearing in many predicted structures are more likely to be present in the real structure. Finding active motifs in secondary structures of different RNA molecules is useful as well. Often, the information obtained from such motifs, in conjunction with results obtained from sequence alignments, helps to conduct the phylogenetic study of the structure for a class of sequences (Shapiro & Zhang 1990).

To find the motifs in RNA secondary structures by a computer, we need a suitable representation for the structures. This paper adopts the tree representation previously proposed in (Shapiro & Zhang 1990). We define both the helical stems and loops to be nodes in a tree. Figure 1 illustrates a RNA secondary structure and its tree representation. The structure is decomposed into five terms: stem, hairpin, bulge, internal loop and multi-branch loop. In the tree, H represents hairpin nodes, I represents internal loops, B represents bulge loops, M represents multi-branch loops, R represents helical stem regions (shown as connecting arcs) and N is a special node used to make sure the tree is connected. The tree is considered to be an ordered one where the ordering is imposed based upon the 5' to 3' nature of the molecule. This representation allows one to encode detailed information of RNA by associating each node with a property list. Common properties may include sizes of loop components, sequence information and energy.

We consider a motif in a tree T to be a connected subgraph of T, viz., a subtree U of T with certain nodes being cut at no cost. (Cutting at a node n in U means

though our techniques should generalize to any scientific domains where data are naturally represented as trees.

Our Approach

Terminology

We say a tree T contains a motif M within distance d (or M approximately occurs in T within distance d) if there exists a subtree U of T such that the *minimum* distance between M and U is less than or equal to d, allowing zero or more cuttings at nodes from U. Let $\mathcal{S}$ be a set of trees. The occurrence number of a motif M is the number of trees in $\mathcal{S}$ that contain M within the allowed distance. Formally, the occurrence number of a motif M with respect to distance d and set $\mathcal{S}$, denoted $occurrence\_no^d_{\mathcal{S}}(M)$, is k if there are k trees in $\mathcal{S}$ that contain M within distance d. For example, consider Fig. 2 again. Let $\mathcal{S}$ contain the three trees in Fig. 2(a). Then $occurrence\_no^0_{\mathcal{S}}(M_1) = occurrence\_no^0_{\mathcal{S}}(M_2) = 3$; $occurrence\_no^1_{\mathcal{S}}(M_3) = occurrence\_no^1_{\mathcal{S}}(M_4) = 3$. Given a set $\mathcal{S}$ of trees, our algorithm finds all the motifs M where M is within the allowed distance *Dist* of at least *Occur* trees in $\mathcal{S}$ and $|M| \geq Size$, where $|M|$ represents the size, i.e., the number of nodes, of the motif M. (*Dist*, *Occur* and *Size* are user-specified parameters.)

Discovery Algorithm

Our algorithm consists of two phases: (1) find promising motifs among a randomly chosen sample $\mathcal{A}$ of the trees in $\mathcal{S}$; and (2) calculate the occurrence numbers of the promising motifs in all of $\mathcal{S}$ to determine which promising motifs satisfy the specified requirements.

Phase (1) consists of two sub-phases. In sub-phase A, we consider all

$$\binom{|\mathcal{A}|}{2} = \frac{|\mathcal{A}|(|\mathcal{A}|-1)}{2}$$

tree pairs in the sample. Then we find candidate motifs from the tree pairs as follows. Suppose the nodes in a tree T are numbered according to some order (e.g., a preorder numbering). Let $t[i]$ represent the node of T whose position in the left-to-right preorder traversal of T is i; $T[i]$ represents the subtree rooted at $t[i]$. We find, for each pair of sample trees T_1 and T_2, the largest approximately common motifs, within distance *Dist*, between $T_1[i]$ and $T_2[j]$ for all $1 \leq i \leq |T_1|$, $1 \leq j \leq |T_2|$. (The asymptotic time complexity of the algorithm is $O(Dist^2 n_1 n_2 (\min\{h_1, l_1\})(\min\{h_2, l_2\}))$, where n_i, $i = 1, 2$, is the number of nodes in tree T_i, h_i is the height of T_i and l_i is the number of leaves in T_i.) Let $\mathcal{C}$ contain the found motifs M with $|M| \geq Size$; these constitute candidate motifs. For each candidate motif M, the tree pairs from which M is discovered are recorded.

In sub-phase B, we store the candidate motifs into a prefix tree PRET (similar to Kosaraju's (1989) suffix tree for trees). Each candidate motif M is decomposed into a collection of paths, called *p-strings*. Each *p*-string contains a sequence of nodes starting at the root of M and ending at a leaf of M. For example, Fig. 3(a) shows four candidate motifs; Fig. 3(b) shows the *p*-strings of one of the motifs.

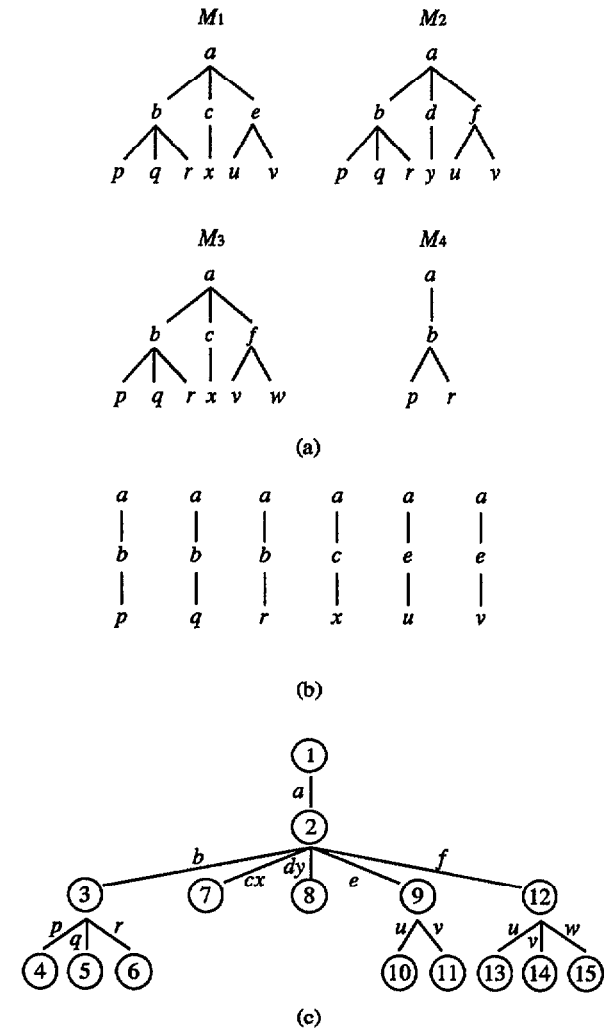

Figure 3: (a) Four candidate motifs M_1, M_2, M_3 and M_4. (b) M_1's *p*-strings. (c) Illustration of the PRET, constructed by inserting the *p*-strings of the motifs in (a) into the PRET; each node in the PRET is labeled by its preorder number.

The edges of the PRET are labeled with characters such that the concatenation of the edge labels on the path from the root to a leaf of the PRET is a *p*-string of some candidate motif. We insert the *p*-strings of candidate motifs into the PRET as into a trie except that if a series of nodes have only one child, we collapse these nodes to a single node whose parent edge is associated with a string instead of a single character. Figure 3(c) shows an example PRET (the nodes with the labels 7

and 8 show the result of a collapsing). For each node v in the PRET, let $string(v)$ be the string on the edge labels from the root to v. We associate v with two fields: $motif(v)$ and $count(v)$. The field $motif(v)$ tells which candidate motifs contain $string(v)$ ($string(v)$ may be a p-string or a prefix of some p-string in the candidate motifs). The field $count(v)$ shows the number of sample trees that contain $string(v)$ within the allowed distance. This field is calculated by traversing the PRET in a bottom-up fashion (e.g., by a postorder traversal) and counting the values in the field $motif(v)$ during the traversal.[1] The PRET can be constructed asymptotically in $O(N)$ time and space where N is the total length of all p-strings contained in the candidate motifs in $\mathcal{C}$ (Kosaraju 1989). After constructing the PRET, we traverse the PRET in a top-down fashion (e.g., by a preorder traversal), pruning unlikely candidate motifs using the optimization heuristics described in the next subsection. The result is a set of "promising" motifs.

In phase (2), we calculate the occurrence numbers of the promising motifs with respect to the entire set $\mathcal{S}$. Determining which are the most likely is a statistical exercise. We use *simple random sampling without replacement* (Cochran 1977) to select sample trees from the set $\mathcal{S}$. Consider a candidate motif M. Let N (n, respectively) denote the number of trees in the set $\mathcal{S}$ (the sample $\mathcal{A}$, respectively) that contain M within the allowed distance. $|\mathcal{S}|$ denotes the set size and $|\mathcal{A}|$ denotes the sample size; $F = N/|\mathcal{S}|$ and $f = n/|\mathcal{A}|$. If it is assumed that f is normally distributed, then we obtain the following.

Fact: *With probability = 99%, F is in the interval $(\hat{F}_L, \hat{F}_U)$ where*

$$\hat{F}_L = f - (t\sqrt{\frac{|\mathcal{S}| - |\mathcal{A}|}{|\mathcal{S}| - 1}}\sqrt{\frac{f(1-f)}{|\mathcal{A}|}} + \frac{1}{2|\mathcal{A}|}),$$

$$\hat{F}_U = f + (t\sqrt{\frac{|\mathcal{S}| - |\mathcal{A}|}{|\mathcal{S}| - 1}}\sqrt{\frac{f(1-f)}{|\mathcal{A}|}} + \frac{1}{2|\mathcal{A}|}).$$

The symbol t is the value of the normal deviate corresponding to the desired confidence probability. When the probability = 99%, $t = 2.58$ (Cochran 1977). The values of $|\mathcal{S}|$, $|\mathcal{A}|$ are given; f, n can be obtained from phase (1) of the discovery algorithm. Thus, if the estimator $(\hat{F}_U \times N) < Occur$ for the candidate motif M, then with probability $\geq 99\%$, M won't satisfy the specified requirements. We therefore eliminate it from consideration.

When checking whether a promising motif M occurs in a tree $T \in \mathcal{S}$ within the allowed distance $Dist$, we add variable length don't cares (VLDCs) to M as the new root and leaves to form a VLDC pattern V and then compare V with T using the pattern matching algorithm developed in (Zhang, Shasha, & Wang 1994). (A VLDC can be matched, at no cost, with a path or portion of a path in T. The algorithm calculates the minimum distance between V and T after implicitly computing an optimal substitution for the VLDCs in V, allowing zero or more cuttings at nodes from T.)

Besides statistical filtering, we incorporate a second optimization heuristic to eliminate the redundant calculation of occurrence numbers (an expensive dynamic programming calculation that must be repeated for every tree in $\mathcal{S}$). We say M_1 is a *sub-pattern* of M_2 if for every p-string of M_1, represented as $string(u)$ in the PRET, there exists a p-string of M_2, represented as $string(v)$ in the PRET, such that v is a descendant of u in the PRET. Observe that $occurrence\_no^k_{\mathcal{S}}(M_1) \geq occurrence\_no^k_{\mathcal{S}}(M_2)$ for all $0 \leq k \leq Dist$. Thus if M_2 is in the final output set, then we need not bother matching M_1 against trees in $\mathcal{S}$, since it will be too. If M_1 is not in the final output set, M_2 won't be either, since its occurrence number will be even lower.

By traversing the PRET in a top-down preorder traversal, we implicitly incorporate the preceding optimization heuristics. Let u, v be two nodes in the PRET where v is an descendant of u. Observe that $occurrence\_no^k_{\mathcal{S}}(string(u)) \geq occurrence\_no^k_{\mathcal{S}}(string(v))$ for all $0 \leq k \leq Dist$. Furthermore, for any p-string P of a motif M, $occurrence\_no^k_{\mathcal{S}}(P) \geq occurrence\_no^k_{\mathcal{S}}(M)$ for all $0 \leq k \leq Dist$. Thus in visiting a node u, we check the field $count(u)$ and use our sampling formula described above to estimate $string(u)$'s occurrence number. Suppose it is estimated that the occurrence number of $string(u)$ is below $Occur$, we eliminate all the motifs containing $string(u)$ from further consideration. Furthermore, we prune all motifs containing $string(v)$ where v is a descendant of u in the PRET. After traversing the PRET, we only calculate the occurrence numbers of the unpruned motifs with respect to the entire set $\mathcal{S}$.

Experimental Results

Data and Parameters

We carried out a series of experiments to evaluate the effectiveness and speed (measured by elapsed CPU time) of our approach. The programs were written in C and run on a SUN SPARC workstation under the SUN operating system version 4.1.2. The data was a set of randomly generated 80 trees. To make the experiments manageable, the size of the trees was fixed at 15. Each node label of the generated trees was drawn randomly from the range A to Z. To gain a better understanding of the performance of our algorithms, we also tested the algorithms on real RNA secondary structures. Eighty secondary structures (trees) were selected randomly from the database in the National

[1] The tree pairs from which a candidate motif is discovered are recorded. For each $string(v)$, we add up the numbers of distinct trees from which the candidate motifs in the field $motif(v)$ are discovered and assign the sum to the field $count(v)$.

Cancer Institute. The sizes of the secondary structures ranged from 10 to 15.

The experimental parameters and their base values were as follows: $SetSize = 80$, $SizeRatio = 80\%$, $Size = 5$ (i.e., the minimum size of an interesting motif is 5), $Dist = 1$ and $Occur = 70$. The sample size was obtained by multiplying $SetSize$ by $SizeRatio$. Twenty samples were chosen randomly; each time one sample was used in running the set and the average was plotted.[2] The metric used to evaluate the effectiveness of our algorithms was $HitRatio = (NumDiscovered/TotalNum) \times 100\%$ where $NumDiscovered$ is the number of interesting motifs discovered by our techniques. $HitRatio$ stands for the percentage of the interesting motifs obtained from the exhaustive search method. By exhaustive search, we mean selecting as candidates all motifs in the set that satisfy the size constraint. One would like this percentage to be as high as possible.

Performance Analysis

Figure 4 compares the effectiveness of our optimized approach (the discovery algorithm with the optimizations) with a non-optimized approach for varying sample sizes. Figure 5 compares their running times with that of the exhaustive search method. It can be seen that very few qualifying motifs were missed by the two proposed optimization heuristics (Fig. 4). Both heuristics sped up the discovery algorithm by a factor of 10. Moreover, our optimized approach was 10,000 times faster than the brute force method (Fig. 5).

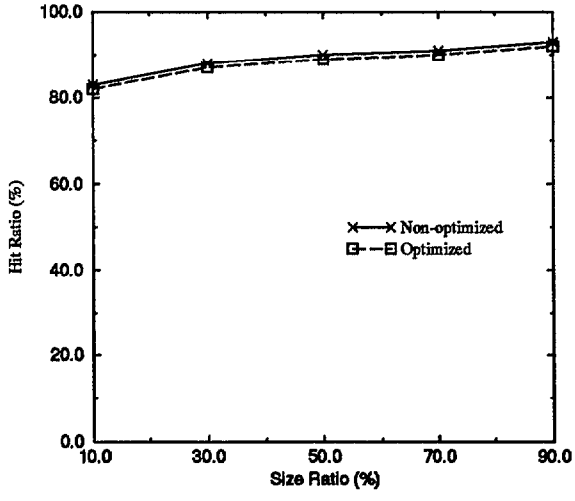

Figure 4: Performance of the pruning techniques for varying sample sizes.

[2]The results obtained from the generated and real data, for both the base values and other parameter values, were rather consistent. Hence, we only present here the results for the RNA molecules with the base values.

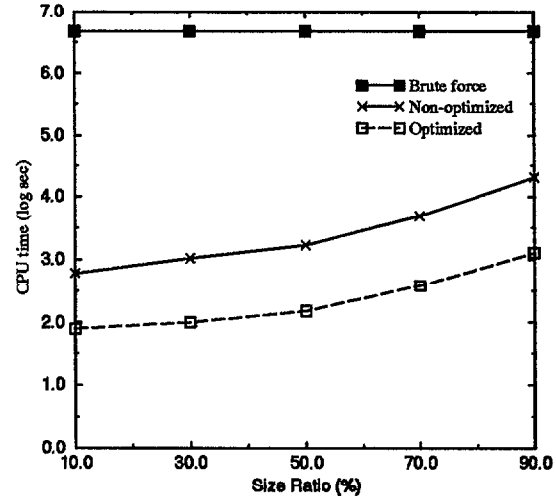

Figure 5: Comparison of the running times between the brute force method, our optimized approach and the approach without optimizations.

Phylogenetic Study

In this experiment, we were interested in seeing whether active motifs help with the phylogenetic study. We selected three families of mRNA sequences from GenBank (Burks *et al.* 1991) pertaining to the poliovirus, human rhinovirus and coxsackievirus. Poliovirus contained two sequences: polio3 sabin strain and pol3mut; human rhinovirus contained two sequences: rhino 2 and rhino 14; and coxsackievirus also contained two sequences: cox5 and cvb305pr. We folded the 5' non-coding region of these mRNA sequences and transformed the resulting suboptimal structures into trees using the algorithm developed in (Shapiro & Zhang 1990). This resulted in 6 files, where each file contained 3,000 trees and the trees had between 70 and 180 nodes.

Using the proposed method, we found 100 most active tree-structured motifs (i.e., those with the largest occurrence numbers) for each sequence. It was observed that rhino 2 had very few active motifs appearing in its family uniquely, whereas the other 5 sequences had many such motifs. To avoid biased results, we took away rhino 2. Then we found the intersections of every two sequences' motifs. Figure 6 summarizes the results. The figure shows that one can get more intersections from sequences of the same family. This result indicates that closeness in motif corresponds to closeness in family. Consequently, one may use the motifs as a way to predict ancestry.

Acknowledgments

We would like to thank the KDD referees whose comments helped to improve the paper. This work was supported by NSF grants IRI-9224601 and IRI-9224602, by ONR grant N00014-92-J-1719 and by the Natural Sciences and Engineering Research Council of

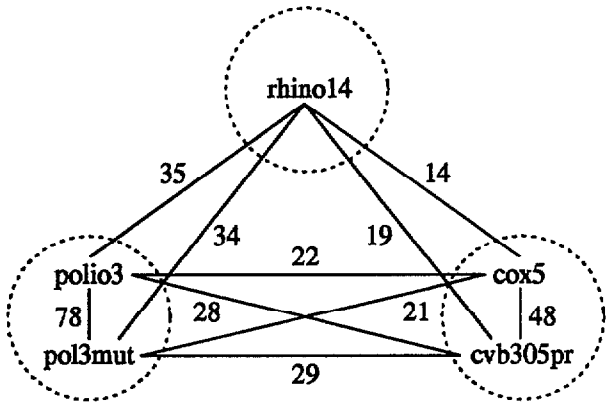

Figure 6: Sequences belonging to the same family are placed in the same circle. Each line connects two sequences and is associated with the number of the intersections of the two sequences' motifs.

Canada under Grant No. OGP0046373.

References

Agrawal, R. 1994. Tutorial: Database Mining. In Proceedings of the Thirteenth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, 75–76.

Burks, C.; Cassidy, M.; Cinkosky, M. J.; Cumella, K. E.; Gilna, P.; Hayden, J. E.-D.; Keen, G. M.; Kelley, T. A.; Kelly, M.; Kristofferson, D.; and Ryals, J. 1991. GenBank. *Nucleic Acids Research* 19:2221–2225.

Chevalet, C., and Michot, B. 1992. An Algorithm for Comparing RNA Secondary Structures and Searching for Similar Substructures. *Comput. Applic. Biosci.* 8:215–225.

Cochran, W. G. 1977. *Sampling Techniques.* New York: Wiley.

Djoko, S.; Cook, D. J.; and Holder, L. B. 1995. Analyzing the Benefits of Domain Knowledge in Substructure Discovery. In Proceedings of the First International Conference on Knowledge Discovery & Data Mining, 75–80.

Han, J.; Cai, Y.; and Cercone, N. 1993. Data-Driven Discovery of Quantitative Rules in Relational Databases. *IEEE Transactions on Knowledge and Data Engineering* 5(1):29–40.

Konings, D. A. M., and Hogeweg, P. 1989. Pattern Analysis of RNA Secondary Structure – Similarity and Consensus of Minimal Energy Folding. *J. Mol. Biol.* 207:597–614.

Kosaraju, S. R. 1989. Efficient Tree Pattern Matching. In Proceedings of the Thirtieth Annual Symposium on Foundations of Computer Science, 178–183.

Le, S.-Y.; Owens, J.; Nussinov, R.; Chen, J.-H.; Shapiro, B. A.; and Maizel, J. V. 1989. RNA Secondary Structures: Comparison and Determination of Frequently Recurring Substructures by Consensus. *Comput. Applic. Biosci.* 5(3):205–210.

Mannila, H.; Toivonen, H.; and Verkamo, A. I. 1995. Discovering Frequent Episodes in Sequences. In Proceedings of the First International Conference on Knowledge Discovery & Data Mining, 210–215.

Piatetsky-Shapiro, G., and Frawley, W. J. eds. 1991. *Knowledge Discovery in Databases.* Menlo Park, CA: AAAI Press/The MIT Press.

Shapiro, B. A., and Zhang, K. 1990. Comparing Multiple RNA Secondary Structures Using Tree Comparisons. *Comput. Applic. Biosci.* 6(4):309–318.

Silberschatz, A.; Stonebraker, M.; and Ullman, J. D. 1991. Database Systems: Achievements and Opportunities. *Communications of the ACM* 34(10):94–109.

Tsumoto, S., and Tanaka, H. 1995. Automated Discovery of Functional Components of Proteins from Amino-Acid Sequences Based on Rough Sets and Change of Representation. In Proceedings of the First International Conference on Knowledge Discovery & Data Mining, 318–324.

Wang, J. T. L.; Chirn, G.-W.; Marr, T. G.; Shapiro, B. A.; Shasha, D.; and Zhang, K. 1994a. Combinatorial Pattern Discovery for Scientific Data: Some Preliminary Results. In Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data, 115–125.

Wang, J. T. L.; Marr, T. G.; Shasha, D.; Shapiro, B. A.; and Chirn, G.-W. 1994b. Discovering Active Motifs in Sets of Related Protein Sequences and Using Them for Classification. *Nucleic Acids Research* 22(14):2769–2775.

Wang, J. T. L.; Zhang, K.; Jeong, K.; and Shasha, D. 1994c. A System for Approximate Tree Matching. *IEEE Transactions on Knowledge and Data Engineering* 6(4):559–571.

Zhang, K.; Shasha, D.; and Wang, J. T. L. 1994. Approximate Tree Matching in the Presence of Variable Length Don't Cares. *Journal of Algorithms* 16(1):33–66.

Zuker, M. 1989. On Finding All Suboptimal Foldings of an RNA Molecule. *Science* 244:48–52.