Ab initio Prediction of RNA Nucleotide Interactions with Backbone k-Tree Model

Liang Ding[1][*], Xingran Xue[1], Sal LaMarca[1], Mohammad Mohebbi[1],
Abdul Samad[4], Russell L. Malmberg[2,3], and Liming Cai[1,2][*]

[1]Department of Computer Science, [2]Institute of Bioinformatics
and [3]Department of Plant Biology, University of Georgia, GA 30602, USA,
{lding, xrxue, slamarca, mohebbi}@uga.edu, russell@plantbio.uga.edu,
cai@cs.uga.edu
[4] Department of Computer Science, BUITEMS, Pakistan.
abdul.samad1@buitms.edu

Abstract. Given the importance of non-coding RNAs to cellular regulatory functions and rapid growth of RNA transcripts, computational prediction of RNA tertiary structure remains highly demanded yet significantly challenging. Even for a short RNA sequence, the space of tertiary conformations is immense; existing methods to identify native-like conformations mostly resort to random sampling of conformations to gain computational feasibility. However native conformations may not be examined and prediction accuracy may be compromised due to sampling. In particular, the state-of-the-art methods have yet to deliver the desired prediction performance for RNAs of length beyond 50.

This paper presents the work to tackle a key step in the RNA tertiary structure prediction problem, the prediction of the nucleotide interactions that constitute the desired tertiary structure. The research is established upon a novel graph model, called *backbone k-tree*, to markably constrain nucleotide interaction relationships in RNA tertiary structure. It is shown that the new model makes it possible to efficiently predict the optimal set of nucleotide interactions from the query sequence, including the interactions in all recently revealed families. Evident by the preliminary results, the new method can predict with a high accuracy the nucleotide interactions that constitute the tertiary structure of the query sequence, thus providing a viable solution towards *ab initio* prediction of RNA tertiary structure.

1 Introduction

In the past decade, there have been many revelations of the importance of non-coding RNAs to cellular regulatory functions and thus a growing interest in computational prediction of RNA tertiary structure [15], [17]. Nevertheless, RNA tertiary structure prediction from a single RNA sequence is a significant challenge. One major unresolved issue is in the immense space of tertiary conformations even for a short RNA sequence. Existing methods usually employ random sampling algorithms for computation feasibility, which assemble sampled tertiary motifs into native-like structures [6], [8], [12], [22], [25], [28]. To reduce the chance to miss native structures, the assembly

---

[*] To whom correspondence should be addressed.

Ding, Xue, LaMarca, Mohebbi, Samad, Malmberg and <u>Cai</u>

algorithms have mostly been guided with constraining structural models. For example, MC-Fold/MC-Sym [22] assumes the tertiary structure consists of 4-nt cyclic tertiary motifs constructible from the predicted secondary structure. Rosetta [6,7] *de novo* assembles tertiary structure from a database of 3-nt tertiary fragments. Other methods follow samplings that preserve the secondary structure [4], [25,26] or intervention from human experts [13], [20]. However, these constraining models do not necessarily ensure that native conformations are examined. In particular, the state-of-the-art methods have yet to deliver the desired prediction accuracy for RNA sequences of lengths beyond 50 [15].

In this work, we introduce a novel method to predict nucleotide interactions from sequences as a key step toward accurate *ab initio* prediction of tertiary structure. Accurate knowledge of the nucleotide interactions is crucial to predicting the tertiary structure of an RNA and subsequently predicting its functional roles. To predict nucleotide interactions, our method is guided by a novel graph model called a *backbone k-tree*, for small integer k, to globally constrain the nucleotide interaction relationships (NIRs) that constitute the tertiary structure. In such a k-tree graph, nucleotides are organized into groups of size $k+1$, such that NIRs are permitted only for nucleotides belonging to the same group and groups are connected to each other with a tree topology (see section 2). This model was inspired by our recent discovery of the small treewidth of the NIR graphs for more than 3,500 RNA chains extracted from 1,984 resolved RNAs (Figure 1). We have been able to develop dynamic programming algorithms with $O(n^{k+1})$ time and space complexities, efficient for small k, to compute the optimal backbone k-tree spanning over the nucleotides on the query sequence, given a scoring function [9,10]
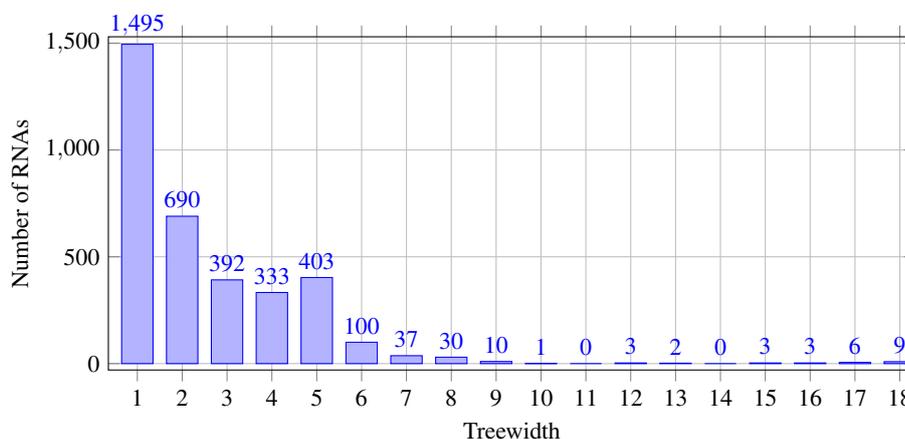

Fig. 1. Treewidth distribution of NIR graphs of more than 3,500 chains deriving from 1,984 resolved RNA tertiary structures in the RNA Structure Atlas [27]. The RNAs with treewidth larger than 18 are omitted due to their very small number. These treewidths are actually upper bounds computed by a program [5]; it is likely that the exact treewidths of the NIR graphs may actually be smaller.

Ab initio Prediction of RNA Nucleotide Interactions with Backbone *k*-Tree Model

To ensure that the computed optimal *k*-tree can actually yield the set of nucleotide interactions that constitutes the native tertiary structure, our method defines the scoring function over detailed patterns of nucleotide interactions within every group of $k+1$ nucleotides. We consider nucleotide interactions from the established geometric nomenclatures [16] and nucleotide interaction families [18], [29], [31], including base-base, base-phosphate, and base-ribose as well as base-stacking interactions. To test our method, we adopted an improved 3-tree model and pre-computed candidates of interaction patterns for every group of 4 nucleotides, by searching through RNA Structure Atlas [27]; this contains annotated atom-level nucleotide interactions for nearly 3,000 resolved tertiary structures. We trained artificial neural networks (ANNs) to compute the confidence of every given nucleotide interaction and the confidence of every admissible nucleotide interaction pattern for every group of 4 given nucleotides. We filtered out unlikely interaction patterns and kept only those with high confidences. With this 3-tree model, our algorithm efficiently predicts an optimal set of nucleotide interactions from the query sequence within computational time $O(c^5 M n^3)$, where M is a constant and $c \leq 20$ is the maximum number of candidate interaction patterns for one group of 4 nucleotides. We have implemented the algorithm into a program called BkTree, which may use known or predicted canonical (i.e., *cis* Watson-Crick) base pairs on the query sequence.

To evaluate our method for nucleotide interaction prediction, we tested BkTree on a benchmark set of 43 high resolution RNAs, which had been used to survey a number of state-of-the-art tertiary structure prediction methods [15]. The resolved, atom-level interactions were extracted with FR3D [27]. BkTree performed impressively well across the set of tested RNAs (Table 3), achieving the averaged sensitivity, PPV, and MCC values of 0.86, 0.78, and 0.82, respectively (discounting the input canonical base pairs). In comparison with previous programs MC [22], Rosetta [6], and NAST [12] that all assumed the secondary structure as a part of the input [15], it is clear that BkTree outperformed the other three programs in the MCC measure on this set of benchhmark RNAs (Table 4, Figure 3). In particular, on the four representative RNAs that contain typical helices and junctions [15], BkTree gave the best performance on all but one RNA, for which BkTree acquired a higher sensitivity value but lower PPV than the MC program, resulting in a slightly lower MCC value (Table 5).

To evaluate the significance of our method to 3D conformation prediction, we used the program MC-Sym to model 3D conformations from the interactions predicted by BkTree and calculated RMSDs against the resolved structures. Since MC-Sym requires secondary structure for 3D conformation modeling, we identified 30 RNAs from the benchmark set for which their secondary structures are covered by the BkTree-predicted nucleotide interactions together with its input canonical base pairs. For the 4 representative RNAs listed in Table 5, BkTree outperforms MC and Rosetta on 3 of them.

2 Model and Methods

In this work, we consider all known types of nucleotide interactions of atomic-resolution [16], [18], [31]. In particular, with the base triangle model consisting of Watson-Crick (W), Hoogsteen (H), and sugar (S) edges, base-base interactions has been fully charac-

Ding, Xue, LaMarca, Mohebbi, Samad, Malmberg and Cai

terized into rich 12 geometric types and 18 interaction families [16], [18], according to involved edges, *cis* or *trans*, and parallel or anti-parallel, observed in crystal structures. For example the cWW family contains, in addition to the canonical (i.e., *cis* Watson-Crick) base pairs, many non-canonical base-base interactions through W edges. More recently, classifications of nucleotide interactions have been extended to base-backbone interactions. There are 10 families identified for base-phosphate interactions based on the position of the interacting hydrogen atom in the base [31]. Similarly, 9 additional families have been identified for base-ribose interactions [32]. A few base stacking interactions have also been classified. Table 1 summarizes these classes of nucleotide interactions, which also includes the backbone interaction between two neighboring nucleotides.

Table 1. Categories, types and families of RNA nucleotide interactions, mostly summarized from works [16], [18], [31,32]. It also includes the phosphodiester interaction between two neighboring nucleotides.

Categories	Types (Interaction Families)	Number
Base pairs	cWW, tWW, cWH, tWH, cHW, tHW, cWS, tWS, cSW, tSW, cHH, tHH, cHS, tHS, cSH, tSH, cSS, tSS	18
Base-phosphates	0BPh, 1BPh, 2BPh, 3BPh, 4BPh, 5BPh, 6BPh, 7BPh 8BPh, 9BPh	
Base-riboses	0BR, 1BR, 2BR, 3BR, 4BR, 5BR, 6BR, 7BR, 9BR	9
Bases stackings	s35, s53, s33, s55	4
Backbone-backbone	phosphodiester	1

2.1 Backbone k-Tree Model

Let the query RNA sequence be $S = S_1 S_2, \ldots S_n$, where $S_i \in \{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{U}\}$, for $1 \leq i \leq n$. We denote an interaction between the ith and jth nucleotides, where $i < j$, with triple $\langle S_i^{(i)}, S_j^{(j)}, t \rangle$, for some interaction type t shown in Table 1. Note that there are possibly two or more simultaneous interactions between the two nucleotides.

Given the native tertiary structure of the sequence S, we model the *nucleotide interaction relationship*s (NIRs) within the tertiary structure with a graph $G = (V, E)$, where $V = \{S_i^{(i)} : 1 \leq i \leq n\}$, such that $(S_i^{(i)}, S_j^{(j)})$ is an edge in E if and only if $i \neq j$ and $\langle S_i^{(i)}, S_j^{(j)}, t \rangle$ is an interaction for some t. We call G *the NIR graph* of the sequence with the given structure. Because every two consecutive nucleotides are connected with the phosphodiester bond, every NIR graph of n vertices contains all edges $(S_i^{(i)}, S_{i+1}^{(i+1)})$, for $1 \leq i \leq n-1$. These edges are called *backbone edges*.

In our recent investigation [9], we constructed NIR graphs for all RNAs whose tertiary structures were known from RNA Structure Atlas [26]. We discovered that an overwhelming majority of these RNAs are of small treewidths (Figure 1). Treewidth is a graph metric, which intuitively indicates how much a graph is tree-like. If a graph has treewidth bounded by k, any clique obtained by deleting vertices and edges and

Ab initio Prediction of RNA Nucleotide Interactions with Backbone *k*-Tree Model

contracting edges of the graph can contain at most $k+1$ vertices [2]. Thus the distribution of treewidths suggest that NIRs in the RNA tertiary structures are in general not arbitrarily complex.

The concept of treewidth originated from the algorithmic graph theory. It is closely related to, and may be better explained with the notion of *k-tree*, which is central to this work.

Definition 1. [24] Let integer $k \geq 1$. The class of *k-trees* are graphs defined by the following inductive steps:

1. A *k*-tree of $k+1$ vertices is a clique of $k+1$ vertices;
2. A *k*-tree of n vertices, for $n > k+1$, is a graph consisting of a *k*-tree G of $n-1$ vertices and a vertex v, which does not occur in G, such that v forms a $(k+1)$-clique with some k-clique already in G.

Figure 2 shows a 3-tree with seven vertices in (a) and illustrates it in (b) with a tree-topology that connects the four 4-cliques in the graph.
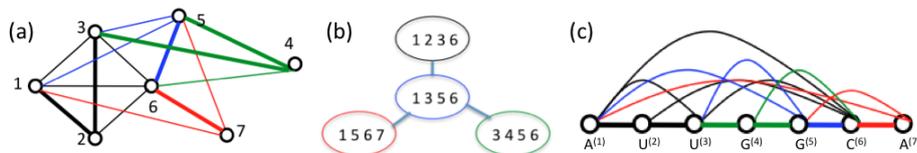

Fig. 2. (a) 3-tree of 7 vertices by Definition 1, with the order of forming the four 4-cliques: with initial clique $\{1, 2, 3, 6\}$ (black edges), vertex 5 and blue edges added, then vertex 7 and red edges added, and finally vertex 4 and green edges added. (b) Illustration of the graph of (a) with a tree-topology connecting the four 4-cliques. (c) A backbone 3-tree for sequence AUUGGCA, of the same topology as shown in (a); backbone edges are in bold.

By [30], for any $k \geq 1$, a graph is of treewidth $\leq k$ if and only if it is a subgraph of a *k*-tree. Therefore, NIR graphs for an overwhelming majority of known RNA tertiary structures are constrained in topology by *k*-trees, for small values of *k*. Because technically, every graph of treewidth bounded by *k* can be augmented with additional edges into a *k*-tree, we adopt such *k*-trees as the model for NIRs of the RNA tertiary structure.

Definition 2. Let $k \geq 1$ be an integer. The *backbone k-tree* for an RNA sequence is an augmented NIR graph of the sequence, which is a *k*-tree.

Figure 1(c) shows a backbone 3-tree for sequence AUUGGCA. Note that backbone *k*-trees differ from general *k*-trees in that a backbone *k*-tree has to the designated Hamiltonian path (consisting of all the backbone edges).

With the backbone *k*-tree model, in order to predict the set I of nucleotide interactions from the query sequence, we propose to identify a backbone *k*-tree $G = (V, E)$ such that

$$(S_i^{(i)}, S_j^{(j)}) \in E \text{ if and only if } \exists t \ \langle S_i^{(i)}, S_j^{(j)}, t \rangle \in I$$

Ding, Xue, LaMarca, Mohebbi, Samad, Malmberg and <u>Cai</u>

To ensure the identified G actually corresponds to the set of interactions that constitute the native structure of the query sequence, we need to quantify nucleotide interactions for combinatorial optimization of such a backbone k-tree G, as explained in the subsequent sections.

2.2 Quantification of Nucleotide Interactions

Definition 3. Let q be a $(k+1)$-clique in a backbone k-tree of query sequence S. An *interaction pattern* (ip) for clique q is a set P_q of interactions for the nucleotides in q such that for every interaction $\langle S_i^{(i)}, S_j^{(j)}, t \rangle$ in P_q, both nucleotides $S_i^{(i)}$ and $S_j^{(j)}$ are in clique q.

Given an ip P_q for clique q, we define the *induced subgraph by P_p*, denoted with $B_{P_q} = (q, E_{B_{P_q}})$ to be a subgraph of q such that edge $(S_i^{(i)}, S_j^{(j)}) \in E_{B_{P_q}}$ only if interaction $\langle S_i^{(i)}, S_j^{(j)}, t \rangle \in P_q$ for some t.

Definition 4. Let q be a $(k+1)$-clique in the in a backbone k-tree of query sequence S. The *confidence* of a given ip P_q for clique q is defined as

$$f(q, P_q, S) = \sum_{\langle S_i^{(i)}, S_j^{(j)}, t \rangle \in P_q} c_{q, B_{P_q}, t}^{(i,j)} \tag{1}$$

where $c_{q, B_{P_q}, t}^{(i,j)}$ is the *confidence* of interaction $\langle S_i^{(i)}, S_j^{(j)}, t \rangle$ given q and subgraph B_{P_q} induced by P_q.

In the Section 3, we will introduce artificial neural networks (ANNs) to compute confidence $c_{q, B_{P_q}, t}^{(i,j)}$.

For every clique q, with $\mathscr{Q}(q)$, we denote the finite set of all ips for q. In the practical application, we may only include those ips in $\mathscr{Q}(q)$ which have "high" confidences (e.g., above certain threshold). Let I be a set of interactions. By notation $I|_q$, we mean the maximal size subset of I that is an ip for q.

Definition 5. Let k be any fixed integer ≥ 2. The *nucleotide interaction prediction* problem NIP(k) is, given an input query sequence S, to identify a backbone k-tree $G^* = (V, E^*)$ as well as a set I^* of nucleotide interactions that constitutes the tertiary structure of S, such that every interaction $\langle S_i^{(i)}, S_j^{(j)}, t \rangle \in I^*$ implies edge $(S_i^{(i)}, S_j^{(j)}) \in E^*$ and

$$(I^*, G^*) = \arg \max_{(I,G)} \left\{ \sum_{q \text{ in } G, I|_q \in \mathscr{Q}(q)} f(q, I|_q, S) \right\} \tag{2}$$

2.3 Overview of the Method

Our method consists of three major components to solve the NIP(k) problem, for any fixed $k \geq 2$. The first component is data repositories including NIPDB and NIPCCTable. NIPDB is a database of all possible interaction patterns (ips) for every $(k+1)$-clique,

Ab initio Prediction of RNA Nucleotide Interactions with Backbone *k*-Tree Model

which was established by searching through the RNA Structure Atlas [27]. For every such clique, its ips in NIPDB are extracted and ranked when the query sequence is pre-processed. NIPCCTable is a matrix for compatibility between every pair of ips for two cliques that share all but one nucleotide. The compatibility is checked by the dynamic programming algorithm computing the NIP(k) problem.

The second component is a set of artificial neural networks (ANNs) to compute confidence for any given interaction type t between any two given nucleotides $S_i^{(i)}$ and $S_j^{(j)}$ on the query sequence. The computed confidences for interactions are then used to compute confidence of an ip for every $(k+1)$-clique, as formulated in equation (1). For every such clique q, all ips of q obtained from database NIPDB are ranked according to their confidence values. Often the number of ips with significant confidence values is small, e.g., ≤ 20; ips of significant scores are included as ip candidates into the set $\mathcal{Q}(q)$ for q. The detailed construction of the ANNs will be described in the next section.

The third component is a dynamic programming algorithm solving the NIP(k) problem, using the prepared data and preprocessing results from the first two components. From the input query sequence, the algorithm produces a backbone k-tree G^* as well as a set I^* of nucleotide interactions, maximizing the aggregate confidence value across all $(k+1)$-cliques in G^* (see equations (2) and (1)). The relationship between G^* and I^* is that, for every $(k+1)$-clique q in the k-tree G^*, there is a maximal subset of nucleotide interactions $I|_q \subseteq I^*$ being an ip for q, such that $I^* = \bigcup_{q \text{ in } G^*} I^*|_q$. The next section describes the details of the dynamic programming algorithm.

3 Algorithms

3.1 ANNs for Computing Interaction Confidence

Let the query sequence $S = S_1 S_2 \ldots S_n$ of n nucleotides, where $S_i \in \{\texttt{A}, \texttt{C}, \texttt{G}, \texttt{U}\}$, for $1 \leq i \leq n$. Technically we considered all $(k+1)$-cliques formed by $k+1$ vertices $\{S_{h_0}^{(h_0)}, S_{h_1}^{(h_1)}, \ldots, S_{h_k}^{(h_k)}\}$, where $1 \leq h_0 < h_1 < \cdots < h_k \leq n$. Let $q = (V, E)$ be such a clique and $B_q = (V, E_{B_q})$, where $E_{B_q} \subseteq E$, be any subgraph of q. For every edge $(S_i^{(i)}, S_j^{(j)}) \in E_{B_q}$ and every possible interaction $\langle S_i^{(i)}, S_j^{(j)}, t \rangle$ of type t, we constructed an ANN $\mathcal{N}_{q,B_q,t}^{(i,j)}$ to calculate *confidence* $c_{q,B_q,t}^{(i,j)}$ that interaction $\langle S_i^{(i)}, S_j^{(j)}, t \rangle$ occurs in the subgraph B_q of clique q.

Each ANN $\mathcal{N}_{q,B_q,t}^{(i,j)}$ consists of an input layer, a hidden layer, and an output layer. The output layer is a single unit depicting a confidence value for interaction $\langle S_i^{(i)}, S_j^{(j)}, t \rangle$. The input layer consists of input units representing the selected global and local features shown in Table 2. The features included the sequence length and the distance between the involved nucleotides as well as neighboring nucleotide types. In addition, we included the information of *assumed* canonical base pairs[1] within the query sequence. The complete list of features selected for the trainings are given in the Table 2.

---

[1] These are known or predicted Watson-Crick and wobble base pairs. Note that they do not necessarily constitute all information about the secondary structure.

Ding, Xue, LaMarca, Mohebbi, Samad, Malmberg and Cai

Table 2. Features selected from a given $(k+1)$-clique q and given subgraph B_q of q for training ANN $\mathcal{N}_{q,B_{P_q},t}^{(i,j)}$. CBP is an abbreviation for canonical base pair. A *Component* (Cp) is defined as the maximal subsequence consisting of two or more nucleotides each involved in a CBP.

Feature	Value	Comments
Seq. length	An integer	Length of a training sequence containing q.
Distances	k integers	Distances between every two nucleotides in the sequential order in q.
Number of Cps	k integers	Number (one of $\{0,1,2,3,-1\}$) of Cps on the subsequence between every two nucleotides in the sequential order. 3 means there are at least 3 Cps; -1 means the two nucleotides are neighboring nucleotides on the sequence.
Neighbor nts.	$k+1$ 4-mers	One 4-mer (of letter A, C, G, U) for every nucleotide in q, where the first two letters and the last two letter of the 4-mer indicate the two nts to the left and to the right of the nucleotide, respectively, and letter N is used when there is no neighbor.
Neighbor CBPs	$k+1$ 4-mers	One 4-mer (of binary bits) for every nucleotide in q, where the first two bits and the last two bits of the 4-mer indicate the two nts to the left and to the right of the nucleotide are involved in CBPs, respectively, and letter N is used when there is no neighbor.
Edge properties	up to $\frac{k(k+1)}{2}$ integers	For every edge in the subgraph B_q of q, value 0 indicates both nts are involved in a CBP; -1 (resp. +1) indicates exclusively left (resp. right) nt is involved in a CBP; 2 indicates either is near a CBP; and -2 indicates both are far away (distant beyond 3 nts) from a CBP.

We adopted conventional methods to construct and train the ANNs [21], typically the technique of back-propagation with gradient descent, using a fixed-size network. This is based on the calculation of the error by taking the first derivatives of half the Euclidean distance between the output and target and back-propagating it towards the input layer, over the whole training set. Each weight is then updated according to the error contribution of each unit, the error of each output unit and a learning rate. The logistic sigmoid was used as the activation functions for each unit. The updating is repeated until the training error converges to a minimum or the cross-validation error starts to rise, due to over-fitting. The learning rate 0.03 was the value that yielded the best results for a subset of 895 RNAs from RNA Structure Atlas.

The trained ANNs can be applied to compute confidence for interaction patterns. In particular, given a $(k+1)$-clique $q = \{S_{h_1}^{(h_1)}, S_{h_2}^{(h_2)}, \ldots, S_{h_{k+1}}^{(h_{k+1})}\}$, $1 \leq h_1 < \cdots < h_{k+1} \leq n$, let P_q be an ip for q and let B_{P_q} be the underlying graph for P_q, which is a subgraph of clique q. Then the trained ANN $\mathcal{N}_{q,B_{P_q},t}^{(i,j)}$ can be applied on each edge $(S_i^{(i)}, S_j^{(j)}) \in E_{B_{P_q}}$ and each type t to compute the confidence score $c_{q,B_{P_q},t}^{(i,j)}$ for interaction $\langle S_i^{(i)}, S_j^{(j)}, t \rangle$. The confidence $f(q, P_q, S)$ of P_q for q is computed with the equation (1).

Ab initio Prediction of RNA Nucleotide Interactions with Backbone *k*-Tree Model

Then for q, all the ips P_q's are ranked according to their confidences $f(q, P_q, S)$, and only significant top m ips are included in the candidate set $\mathscr{Q}(q)$. We have chosen $m \leq 20$ in the performance evaluations as our experiments results had showed that a larger m could not help to improve the results.

3.2 Algorithm for NIP(k) problem

Roughly speaking, the algorithm for NIP(k) problem considers every $(k + 1)$-clique, from which recursive creations of more cliques are all examined. For every newly created clique q, all ips from $\mathscr{Q}(q)$ are considered but eventually exactly one of them is chosen for q. The algorithm follows the basic process of creating k-tree given in Definition 1. However, because the identified k-tree is a backbone k-tree that contains all backbone edges, the process is not straightforward. We need the following notations for an introduction to the algorithmic idea. By *interval* $[i..j]$, for $i \leq j$, we mean the set of consecutive integers between i and j, inclusive. Two intervals $[i..j]$ and $[h..l]$ are *non-overlapping* if either $j \leq h$ or $l \leq i$. Formally, let the query sequence be $S = S_1 S_2 \ldots S_n$ and q be a clique formed by $k + 1$ vertices $\{S_{h_1}^{(h_1)}, S_{h_2}^{(h_2)}, \ldots, S_{h_{k+1}}^{(h_{k+1})}\}$, where $1 = h_0 \leq h_1 < h_2 < \cdots < h_{k+1} \leq n = h_{k+2}$. Let A be a set of non-overlapping intervals and $P_q \in \mathscr{Q}(q)$ be an ip for clique q.

We define function $M(q, A, P_q, S)$ to be the maximum confidence of a k-tree constructed beginning from clique q, which includes all backbone edge $(S_i^{(i)}, S_{i+1}^{(i+1)})$ for integers i and $i + 1$ both contained in the same interval in A. Then we obtain the following recurrence:

$$M(q, A, P_q, S) = \max_{\substack{S_x^{(x)} \in q, \\ S_y^{(y)} \notin q, \\ y \in [i..j] \in A, \\ p = q|_y^x}} \max_{\substack{P_p \in \mathscr{Q}(p), \\ \mathscr{R}(B,C), \\ \mathscr{P}(P_q, P_p)}} \{M(p, B, P_p, S) + M(q, C, P_q, S) + f(q, P_q, S)\} \quad (3)$$

where abbreviations $q|_y^x = q \cup \{S_y^{(y)}\} \setminus \{S_x^{(x)}\}$, $\mathscr{P}(P_p, P_q)$ asserts that the chosen ip P_p be compatible with P_q, and $\mathscr{R}(B,C)$ represents the choices of two sets of intervals, B and C, which satisfy constraints

(a) $\{[i..y], [y..j]\} \subseteq B$, $\{[w..x], [x..z]\} \subseteq C$, for applicable w and z; and
(b) $B \cup C = A \cup \{[i..y], [y..j]\} \setminus \{[i..j]\}$, and $B \cap C = \emptyset$.

Recurrence (3) gives an iterative process to produce a backbone k-tree. The intuitive idea is to create a new clique p from q by introducing a new nucleotide vertex $S_y^{(y)}$ to the partially constructed k-tree. This results in possibly two or more sub-k-trees, one starting from p and the others from q (but not including $S_y^{(y)}$). Since the two or more sub-k-trees will never join together again, interval sets are used to ensure backbone edges will be properly created. Essentially, the constructed k-tree corresponding to the value of function $M(q, A, P_q, S)$ contains only those backbone edges that connect the nucleotides of indexes specified in the intervals in A. In particular, starting from clique

Ding, Xue, LaMarca, Mohebbi, Samad, Malmberg and <u>Cai</u>

q of $k+1$ vertices $\{S_{h_1}^{(h_1)}, S_{h_2}^{(h_2)}, \ldots, S_{h_{k+1}}^{(h_{k+1})}\}$, to compute an backbone k-tree that contains all the backbone edges, we need to set $A = \{[h_i..h_{i+1}] : 0 \le i \le k+1\}$, where $h_0 = 1$ and $h_{k+2} = n$.

The confidence score of the produced k-tree is computed as the sum of confidence scores of ips chosen for all involved $(k+1)$-cliques. The chosen ips need to be compatible across the cliques when they share nucleotide interactions or even just nucleotides. This is ensured by the assertion $\mathscr{P}(P_q, P_p)$, which checks (1) P_q and P_p have the same set of interactions on the edges shared by cliques q and p by looking up table NIPC-CTable; and (2) any pattern of interactions between a single nucleotide and multiple others has to exist in the structure database.

To complete the recurrence, we need the following base case:

$$M(q, A, P_q, S) = 0 \text{ if } A = \emptyset$$

To identify the desired backbone k-tree G^*, we maximize $M(q, A, P_q, S)$ over all starting clique q and all ip $P_q \in \mathscr{Q}(q)$. The associated set I^* of nucleotides is just the union of the chosen ips for all $(k+1)$-cliques in G^*.

Recurrence (3) naturally offers a dynamic programming solution. Function $M(q, P_q, A, S)$ can be computed by establishing a table with dimensions for q, P_q, and A. With the base cases, the table is computed bottom-up, from $A = \emptyset$, using the recurrence (3).

3.3 Improved Algorithms

Simply implementing the above outlined algorithm would require $O(n^{k+1})$ memory space and $O(n^{k+2})$ computation time for every fixed value of k. Following the same idea but creating $(k+1)$-cliques from k-cliques instead leads to an improved dynamic programming algorithm to solve the NIP(k) problem, with a little more sophisticated steps to navigate through k-cliques. The improved algorithm uses $O(n^k)$ amount of memory space and $O(n^{k+1})$ amount of time for every fixed value of k [9,10].

The efficiency can be further improved by demanding that every $(k+1)$-clique in backbone k-trees contains two consecutive nucleotides $S_i^{(i)}$ and $S_{i+1}^{(i+1)}$ for some i. That is, every interaction pattern for a $(k+1)$-clique always contains at least one backbone edge. This allows a further reduction of computation time to $O(n^k)$. Testing on the case $k = 3$ has shown that the constrained backbone 3-tree model maintains the similar capability to account for sophisticated nucleotide interactions as the "standard" backbone 3-tree model. In addition the constraint may enforce the construction of the 3-tree to follow backbone edges, providing more controls on the 3-tree construction. Finally, the constraint also significantly reduced the number of cases that the ANNs need to consider in their construction.

3.4 Implementation

The NIPDB database construction was implemented by Python, where Prody package [3] was adopted to search RNA Structure Atlas. Afterward, NIPCCTable, the matrix for ip consistence and compatibility was developed using Python. Training and building of ANNs were realized with WEKA package [19]. Finally, confidences of ips admissible

Ab initio Prediction of RNA Nucleotide Interactions with Backbone *k*-Tree Model

for every clique $(k + 1)$-clique in the query sequence was computed by programs in Python.

We implemented in C++ the dynamic programming algorithm into a program called BkTree. We ran the evaluation tests on a Red Hat 4.8.2-7 server with 4 Intel Quad core X5550 Xeon Processors, 2.66GHz 8M Cache and 70GB Memory.

4 Performance Evaluation

4.1 Test Data

We implemented our method in the program BkTree. We evaluated our method through testing BkTree on a list of 43 RNAs of high resolution structure data, which had been used as a benchmark set to evaluate a number of state-of-the-art tertiary structure prediction methods in the survey [15]. 18 of the RNA sequences are of length ≥ 50. In developing the ANNs for computing interaction confidences, 7 of these RNAs were not included in the training data.

Given the recent progress made in RNA secondary structure prediction [15], [26], we believe that canonical base pairs may be routinely predicted with a fair accuracy. Therefore, we have allowed the program BkTree to accept known or predicted canonical base pairs along with the query sequence as input. Note that the knowledge of canonical base pairs does not necessarily imply the whole secondary structure, which is often a part of input to most of the existing RNA 3D prediction methods. In our test, we extracted canonical base pairs of a RNA from FR3D analyzed interactions [27].

4.2 Overall Performance

We evaluated the quality of the predicted nucleotide interactions by the sensitivity (STY) and positive predictive value (PPV) against the FR3D-analyzed interactions [27]. In order to take into account the effects of both true positive and false positive rates in one measure, the *Matthews correlation coefficient* (MCC), defined in [15] as

$$\mathrm{MCC} := \sqrt{\mathrm{PPV} \times \mathrm{STY}},$$

was also calculated.

Table 3 summarizes the overall performance of BkTree on the benchmark set. On a large majority of RNAs, the sensitivity is decently high. Note that the STY and PPV calculations excluded the canonical base pairs. The sensitivity result indicates that our method has a high accuracy in identifying non-canonical interactions that may be crucial to tertiary structures. This is true even for those longer RNAs. We further note that for the 7 RNAs that were not included in the training data, BkTree also performed extremely well.

4.3 Performance Comparison with Other Methods

We compared our program BkTree with the programs MC, Rosetta, and NAST on the capability to predict nucleotide interactions. These other methods had been surveyed

Ding, Xue, LaMarca, Mohebbi, Samad, Malmberg and <u>Cai</u>

and evaluated in [15] based on their ability to identify both base pairing and base stacking interactions. We removed base-phosphate and base-ribose interactions from our prediction results. We incorporated the canonical base pairs into our results because these other methods include all interactions from the input secondary structure.

Figure 3 shows the MCC curves for MC, Rosetta, NAST, and BkTree on the benchmark set of RNAs. Data of RNAs failed by a program were not included in the calculation. We note that for every RNA, these other programs produced more than one conformation so the results were averaged for these comparisons. The figure demonstrates that BkTree overall outperformed the other three programs in predicting non-canonical base pairing and base stacking interactions.
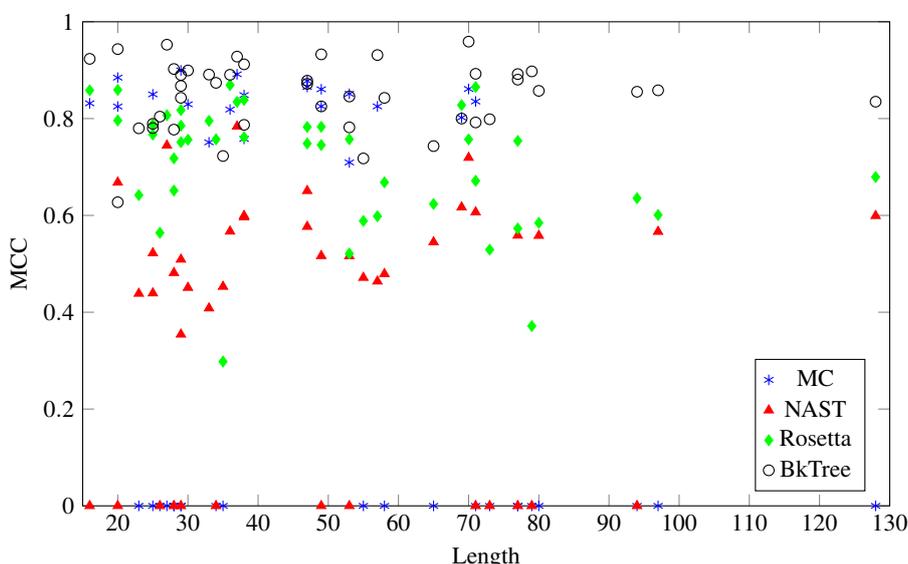

Fig. 3. Comparison of the MCC generated by MC, NAST, Rosetta and BkTree. The MCC of the 43 RNAs are calculated by including canonical base pairs in the results and sorted by their lengths. The plot was derived by merging the results obtained by BkTree and the data computed in the survey [14,15]. In that survey, the tertiary structure predictions with the other 3 methods were based on resolved secondary structures and the secondary structures were included in the calculations. Therefore, the canonical base pairs were also been added to the perdiction results by BkTree.

Table 4 gives comparisons on average performance between the four methods. In general, Bktree produced much better average results than Rosetta and NAST, and comparable average results with MC, for which BkTree shows better average STY value than MC, whereas MC gives better average PPV. On MCC values, BkTree had an edge over MC. On RNAs of length ≥ 50, BkTree maintained almost the same average MCC as it did on the whole set.

Ab initio Prediction of RNA Nucleotide Interactions with Backbone *k*-Tree Model

4.4 Significance to 3D conformation prediction

To evaluate the significance of our method to 3D conformation prediction, we used MC-Sym [22] to model 3D conformations from the interactions predicted by BkTree and calculated RMSDs against the resolved structures. We note that MC-Sym does not accept interactions of categories other than base-pair and base stacking; the correctly predicted base-phosphate and base-ribose interactions by our methods were discarded by MC-Sym to produce 3D folds. The *deviation index* (DI) [23], a measure that accounts for both RMSD and MCC, defined as the quotient of them, was also calculated. Table 5 presents the performance values on the 4 representative RNAs chosen in [15] which typically contain two hairpins and two junctions. Since both MC and Rosetta allow prediction of multiple optimal or suboptimal folds, we chose the averaged values of their solutions. We note that to model 3D conformations with MC using our predicted interaction data, we needed the secondary structure of the tested RNA to be covered by the input canonical base pairs together with the interactions predicted by BkTree. RNA 2QUS failed on this requirement. The averaged RMSDs achieved by BkTree for the rest 3 RNAs are significantly smaller than those achieved by MC and Rosetta.

5 Discussion and Conclusion

Our method is the first to *ab initio* predict RNA non-canonical interactions of all types. Evaluation of the results have highlighted its potential as an important step toward accurate *ab initio* 3D structure prediction. We attribute the encouraging preliminary results to the recent growth of knowledge in high-resolution nucleotide interaction data as well as to the novel backbone *k*-tree modeling of nucleotide interaction relationships. The latter makes it possible to markedly reduce the space of solutions for the nucleotide interaction prediction problem to one that can be feasibly searched in polynomial time.

Our method differs from others also in its direct prediction of nucleotide interactions whereas the others mostly attempt 3D conformation construction before producing nucleotide interactions. The difference makes it difficult to compare their performances, especially when a 3D structure is not the direct output of a software, e.g., RNA-MoIP [26]. Therefore, the MCC comparison with MC was probably more appropriate than the comparison with RNA-MoIP, since the results of MC were based on interactions from the RNA Structure Atlas and so did BkTree, while RNA-MoIP used Interaction Network Fidelity [11] in calculating the MCC values. The contrast is more evident when using MC-Sym to model 3D conformations from interaction data predicted by BkTree. Even though the predicted base-phosphate and base-ribose interactions have to be discarded, the resulted RMSDs seem to correlate with the MCC values (Table 5).

The evaluation tests have also revealed some issues with BkTree. First, the complexity of structures has an impact on our prediction results. Typically, BkTree underperformed on some of the RNAs with pseudoknots or 4-way junctions. Table 5 shows that BkTree loses to MC on MCC value for only one representative RNA 2QUS, which contains a pseudoknot. The underperformance is likely due to the 3-tree model that is a little too weak for complex structures. For example, the best 3-tree can include at most 83 interactions out of total 95 interactions of tRNA 2DU3, indicating a higher treewidth is needed for the NIR graph of this RNA. To improve prediction performance for such

Ding, Xue, LaMarca, Mohebbi, Samad, Malmberg and <u>Cai</u>

RNAs, an algorithm may need to be based on the backbone 4-tree model. Our method is not ineffective for handling multi-way junctions or pseudoknots, e.g., RNA 2GIS in Table 3. Fixing a specific k-tree model, it is the NIR graph treewidth of an RNA that determines the performance on the RNA.

Second, the NIR graph treewidth is also related to scalability of our method. The current algorithm for the nucleotide prediction problem has the complexity $O(n^3)$ for both time and memory requirements. With a large hidden constant in the polynomial, the implemented program BkTree typically runs in 2 to 3 hours on an RNA of length 100 and uses several Gigabytes of memory. This is because the current prototype has aimed at accuracy without optimization in computational efficiency. However, the problem (based on the k-tree model) has an inherent complexity of $O(n^k)$; our method is scalable to suit longer and more complex RNAs, e.g., which require the 4-tree model.

Third, due to the lack of tools to model 3D conformations from nucleotide interactions of all types, it is an immediate future task of ours is to develop such a tool that can be pipelined with a program like BkTree for *ab initio* 3D structure prediction. We perceive such a task to be feasible. This is because the output of program BkTree contains not only the predicted nucleotide interactions but also a backbone 3-tree that decomposes nucleotides according to their interconnectivity. The given 3-tree can be the basis for very efficient algorithms for computing a desirable optimization function on 3D conformations [1].

Acknowledgments

References

1. Arnborg, S. and Proskurowski, A. (1989) Linear time algorithms for NP-hard problems restricted to partial k-trees. *Discrete Applied Mathematics*, 23(1):11-24.
2. Arnborg, S., Proskurowski, A., and Corneil, DG. (1990) Forbidden minors characterization of partial 3-trees. *Discrete Mathematics*, 80(1):1-19.
3. Bakan, S., Meireles, L., and Bahar, I. (2011) ProDy: Protein Dynamics Inferred from Theory and Experiments. *Bioinformatics*, 27(11):1575-1577.
4. Bida, J.P., and Maher, L.J. III (2012) Improved prediction of RNA tertiary structure with insights into native state dynamics. *RNA*, 18:385-393.
5. Bodlaender, H.L. and Koster, A.M.C.A. (2010) Treewidth computations I. Upper bounds. *Information and Computation*, 208(3):259-275.
6. Das, R. and Baker, D. (2007) Automated *de novo* prediction of native-like RNA tertiary structures. *Proc. Natl Acad Sci*, 104:14664-14669.
7. Das, R. Karanicolas, J., and Baker, D. (2010) Atomic accuracy in predicting and designing noncanonical RNA structure. *Nature Methods*, 7:291-294.
8. Ding, F., Sharma, S., Chalasani, P., Demidov, V.V., Broude, N.E., and Dokholyan, N.V. (2008) *Ab initio* RNA folding by discrete molecular dynamics: From structure prediction to folding mechanisms. *RNA*, 14:1164-1173.

Ab initio Prediction of RNA Nucleotide Interactions with Backbone *k*-Tree Model

9. Ding, L., Xue, X., LaMarca, S., Mohebbi, M., Samad, A., Malmberg, R., and Cai, L. (2014) Stochastic *k*-tree grammar and its application in biomolecular structure modeling. *Lecture Notes in Computer Science*, 8370:308-322.

10. Ding, L., Samad, A., Li, G., Robinson, R.W., Xue, X., Malmberg, R., and Cai, L. (2014) Finding maximum spanning *k*-trees on backbone graphs in polynomial time. *Submitted*.

11. Gendron P., Lemieux S., and Major F. (2001) Quantitative analysis of nucleic acid three-dimensional structures. *J Mol Biol*, 308:919-936.

12. Jonikas, M.A., Radmer, R.J., Laederach, A., Das, R., Pearlman S., Herschlag, D. and Altman R.B. (2009) Coarse-grained modeling of large RNA molecules with knowledge-based potentials and structure filters. *RNA*, 15:189-199.

13. Jossinet, F., Ludwig, T.E., and Westhof, E. (2010) Assemble: An interactive graphical tool to analyze and build RNA architectures at the 2D and tertiary levels. *Bioinformatics*, 26:2057-2059.

14. Laing, C. (2014) Personal communication.

15. Laing, C. and Schlick, T. (2010) Computational approaches to tertiary modeling of RNA. *Journal of Physics: Condensed Matter*, 22:283101.

16. Leontis, N.B. and Westhof, E. (2001) Geometric nomenclature and classification of RNA base pairs. *RNA*, 7(4):499-512.

17. Leontis, N.B. and Westhof, E.(Eds.) (2012) RNA 3D Structure Analysis and Prediction. *Springer*.

18. Leontis, N.B., Stombaugh, J., and Westhof, E. (2002) The non-Watson-Crick base pairs and their associated isostericity matrices. *Nucleic Acids Res.*, 30(16):3497-531.

19. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., and Witten, I.H. (2009) The WEKA Data Mining Software: An Update. *ACM SIGKDD Explorations Newsletter*, 11(1):10-18.

20. Martinez, H.M., Maizel, J.V., Jr., and Shapiro, B.A. (2008) RNA2Dtertiary: A program for generating, viewing, and comparing 3-dimensional models of RNA. *Journal of Biomolecular Structure Dynamics*, 25:669-683.

21. Mitchell, T. Machine Learning. *McGraw Hill*, New York, NY, USA, (1997).

22. Parisien, M. and Major, F. (2008) The MC-Fold and MC-Sym pipeline infers RNA structure from sequence data. *Nature*, 452:51-55.

23. Parisien M., Cruz J. A., Westhof E., and Major F. (2009) New metrics for comparing and assessing discrepancies between RNA 3D structures and models. *RNA* 15:1875-1885.

24. Patil, H. P. (1986) On the structure of *k*-tree. *Journal of Combinatorics, Information and System Sciences*, 11(2-4):57-64.

25. Popenda, M., Szachniuk, M., Antczak, M., Purzycka, K.J., Lukasiak, P., Bartol, N., Blazewicz, J., and Adamiak, R.W. (2012) Automated tertiary structure composition for large RNAs. *Nucleic Acids Research* 40(14):e112.

26. Reinharz, V., Major, F., and Waldispĥl, J. (2013) Towards tertiary structure prediction of large RNA molecules: an integer programming framework to insert local tertiary motifs in RNA secondary structure. *Bioinformatics*, 28:i207-i214.

27. Sarver, M., Zirbel, C.L., Stombaugh, J., Mokdad, A., and Leontis, N.B. (2008) FR3D: Finding Local and Composite Recurrent Structural Motifs in RNA 3D Structures. *Journal of Mathematical Biology*, 56:215-252.

28. Sharma, S., Ding, F., and Dokholyan, N.V. (2008) iFoldRNA: Three-dimensional RNA structure prediction and folding. *Bioinformatics*, 24:1951-1952.

29. Stombaugh, j., Zirbel, C.L., Westhof, E., and Leontis, N.B. (2009) Frequency and isostericity of RNA base pairs. *Nucleic Acids Research* 37(7):2294-2312.

30. van Leeuwen, J. (1990) Graph algorithms, *Handbook of Theoretical Computer Science, A: Algorithms and Complexity theory*, North Halland.

Ding, Xue, LaMarca, Mohebbi, Samad, Malmberg and <u>Cai</u>

31. Zirbel, C.L. Judit E. Sponer, J.E., Sponer, J., Stombaugh, J., and Leontis, N.B. (2009) Classification and energetics of the base-phosphate interactions in RNA. *Nucleic Acids Research*, 37(15):4898-4918.

32. Zirbel, CL. et. al. (2011) FR3D list of base-phosphate and base-ribose interactions in 1EHZ, `http:// rna.bgsu.edu/FR3D/AnalyzedStructures/1EHZ/1EHZ_base_phosphate.html`.

Ab initio Prediction of RNA Nucleotide Interactions with Backbone *k*-Tree Model

Table 3. Nucleotide interaction prediction results by BkTree on the benchmark set used in the survey [15]. The number of canonical base pairs (CPBs) and number of non-canonical interactions (NCIs) are listed. The sensitivity (STY), PPV and MCC were calculated, excluding the canonical bases pairs used as a part of the input. The data of the 7 RNAs not used for training ANNs are displayed with the bold font.

PDB ID	Length	# CBPs	# NCIs	STY	PPV	MCC	Structure complexity
2F8K	**16**	**6**	**14**	**85**	**85**	**0.8571**	**Hairpin**
2AB4	**20**	**6**	**20**	**100**	**90**	**0.9534**	**Hairpin**
361D	**20**	**5**	**17**	**70**	**57**	**0.6351**	**Hairpin**
2ANN	23	3	24	75	66	0.7071	Hairpin
1RLG	25	5	22	95	63	0.7793	Hairpin, internal loop
2QUX	25	9	22	90	71	0.8058	Hairpin
387D	26	4	23	86	68	0.7744	Hairpin
1MSY	27	6	39	97	92	0.9502	Hairpin
1L2X	28	8	34	88	88	0.8823	Pseudoknot
2AP5	28	8	29	82	66	0.7427	Pseudoknot
1JID	29	8	31	93	72	0.8235	Hairpin, internal loop
1OOA	29	8	29	93	72	0.8242	Hairpin, internal loop
430D	29	6	37	94	77	0.8577	Hairpin, internal loop
3SNP	30	12	31	93	85	0.8932	Hairpin, internal loop
2OZB	33	10	33	93	79	0.8641	Hairpin, internal loop
1MJI	34	10	44	84	84	0.8409	Hairpin, internal loop
1ET4	35	8	40	67	84	0.7546	Pseudoknot
2HW8	36	12	44	93	80	0.8655	Hairpin, internal loop
1I6U	37	15	47	91	89	0.9053	Hairpin, internal loop
1F1T	38	10	38	81	63	0.7184	Hairpin, internal loop
1ZHO	**38**	**13**	**46**	**95**	**83**	**0.8911**	**Hairpin, internal loop**
1S03	47	18	53	88	79	0.8404	Hairpin, internal loop
1XJR	47	15	55	83	80	0.8215	Hairpin, internal loop
1U63	49	17	50	94	65	0.7833	Hairpin, internal loop
2PXB	49	16	66	98	94	0.9632	Hairpin, internal loop
2FK6	53	20	58	77	70	0.7385	Pseudoknot, 3-way junction
3E5C	53	21	65	84	73	0.7877	3-way junction (riboswitch)
1MZP	55	17	73	64	73	0.6876	Hairpin internal
1DK1	57	24	65	100	89	0.9436	3-way junction
1MMS	58	20	86	74	82	0.7814	3-way junction
3EGZ	65	23	72	70	66	0.6849	3-way junction (riboswitch)
2QUS	69	26	81	75	76	0.7577	Pseudoknot, 3-way junction
1KXK	70	28	87	96	92	0.9440	Hairpin, internal loop
2DU3	71	27	75	78	70	0.7433	4-way junction (tRNA)
2OIU	**71**	**29**	**84**	**90**	**83**	**0.8692**	**3-way junction (riboswitch)**
1SJ4	73	19	83	78	81	0.7976	Pseudoknot, 4-way junction
1P5O	77	29	86	97	77	0.8716	Hairpin, internal loop
3D2G	77	28	103	80	88	0.8435	3-way junction (riboswitch)
2HOJ	79	27	100	87	84	0.8572	3-way junction (riboswitch)
2GDI	**80**	**32**	**100**	**84**	**80**	**0.8197**	**3-way junction (riboswitch)**
2GIS	94	36	125	87	82	0.8485	Pseudoknot, 4-way junction (riboswitch)
1LNG	97	38	124	85	79	0.8254	3-way junction (SRP)
1MFQ	**128**	**49**	**164**	**81**	**76**	**0.7895**	**3-way junction (SRP)**

Ding, Xue, LaMarca, Mohebbi, Samad, Malmberg and Cai

Table 4. Average performances of MC, Rosetta, NAST and BkTree, with results in two categories: average over all successfully resolved RNAs and average over all successfully resolved RNAs of length > 50. The best performance data are displayed in bold.

	All RNAs				RNAs of length >50			
	Success/Total	STY	PPV	MCC	Success/Total	STY	PPV	MCC
MC	21/43	80.7	**86.2**	0.8344	6/18	77.1	**86.0**	0.8145
Rosetta	43/43	62.8	80.3	0.7101	18/18	53.4	78.5	0.6474
NAST	30/43	44.5	68.2	0.5508	12/18	44.0	71.4	0.5604
BkTree	43/43	**88.6**	81.3	**0.8482**	18/18	**86.0**	82.7	**0.8433**

Table 5. List of performance values predicted using MC, Rosetta and BkTree on 4 representative RNAs chosen by [15]. The results generated MC and Rosetta are obtained from the survey paper [14,15]. For every RNA, the best results are displayed in bold.

		MC					Rosetta					BkTree				
PDB	Length	STY	PPV	MCC	RMSD	DI	STY	PPV	MCC	RMSD	DI	STY	PPV	MCC	RMSD	DI
1KXK	70	81	89	0.849	9.49	11.16	74	85	0.793	17.23	21.69	97	94	**0.9589**	**8.33**	**8.68**
1XJR	47	76	87	0.8131	8.74	10.74	71	83	0.7676	11.63	15.21	91	84	**0.8782**	**6.00**	**6.83**
2OIU	71	76	92	0.8361	16.85	20.14	63	87	0.7403	18.10	24.72	92	86	**0.8925**	**13.21**	**14.8**
2QUS	69	78	86	**0.819**	18.41	22.44	58	86	0.7062	15.73	22.80	80	80	0.8	-	-