A Local Structural Prediction Algorithm for RNA Triple Helix Structure

Bay-Yuan Hsu[1], Thomas K.F. Wong[2,⋆], Wing-Kai Hon[1], Xinyi Liu[3], Tak-Wah Lam[3], and Siu-Ming Yiu[3,⋆⋆]

[1] Department of Computer Science, National Tsing Hua University, Taiwan
[2] Ecosystem Sciences, CSIRO, Canberra, Australian Capital Territory, Australia
[3] Department of Computer Science, The University of Hong Kong, Hong Kong

Abstract. Secondary structure prediction (with or without pseudoknots) of an RNA molecule is a well-known problem in computational biology. Most of the existing algorithms have an assumption that each nucleotide can interact with at most one other nucleotide. This assumption is not valid for triple helix structure (a pseudoknotted structure with tertiary interactions). As these structures are found to be important in many biological processes, it is desirable to develop a prediction tool for these structures. We provide the first structural prediction algorithm to handle triple helix structures. Our algorithm runs in $O(n^3)$ time where n is the length of input RNA sequence. The accuracy of the prediction is reasonably high, with average sensitivity and specificity over 80% for base pairs, and over 70% for tertiary interactions.

1 Introduction

Prediction of a pseudoknotted secondary structure (base pairs crossing each other) of an RNA molecule is NP-hard in general [1]. In practice, the project focus on restricted classes of pseudoknots that are found in nature. Examples of these prediction algorithms include [1–7]. All these existing methods have an assumption that each nucleotide can interact with at most one nucleotide in the RNA. However, if tertiary interaction (where some single stranded nucleotides also form hydrogen bonds with nucleotides in base pairs) is considered, this assumption may not hold. Triple helix structure in ncRNA is a pseudoknotted structure with tertiary interaction. Figure 1 shows an example of a triple helix structure. Triple helix structures exist in yeast and human telomerase, and are found to be essential in quite a few biological processes (e.g. chromosome stability in stem cells, germline cells and cancer cells [8–10]; ribosomal frameshifting [11, 12]).

There are only two recent results [13, 14] that consider tertiary interactions. Siederdissen *et al.* provided a folding algorithm for RNA secondary structures which consider tertiary interactions inside only a regular structure (one without

⋆ B.Y. Hsu and T. Wong contributed equally.
⋆⋆ Corresponding author.

A. Ngom et al. (Eds.): PRIB 2013, LNBI 7986, pp. 102–113, 2013.

pseudoknots) while Wong *et al.* considered a structural alignment problem for RNA secondary structures with *standard triple helix structure* (tertiary interactions inside a simple pseudoknot). In this paper, we provide the first RNA secondary structure prediction algorithm for tertiary interactions over pseudoknots and focus on the standard triple helix structure defined in [14].

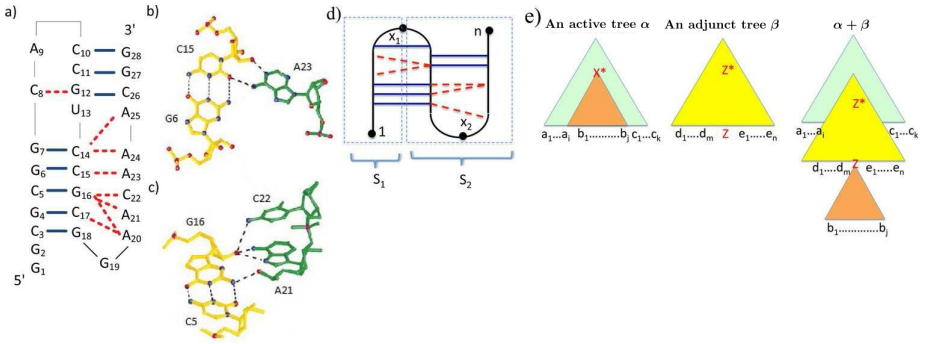

Fig. 1. (a) Triple helix in beet western yellows virus pseudoknot [11]. Blue lines represent the secondary structure. Red lines represent the tertiary interactions between single stranded nucleotides (according to the secondary structure) and base pairs. (b) and (c) Detailed view of some tertiary interactions in the structure [11]. (d) A standard triple helix structure. (e) Adjoining interaction between one active tree and one adjunct tree in simple tree adjoining grammar (STAG). The * indicates an active node. The active node X is replaced by the whole tree β.

We employ a machine learning approach (similar to the approach used by [15]) as follows to solve the problem. We define a grammar, which for any given RNA sequence, generate different possible secondary structures of the sequence. Based on some training datasets (the RNA sequences with known secondary structures), we assign probability to each grammar rule. Then, for each RNA sequence with unknown secondary structure, we can derive the optimal secondary structure (the one with the highest probability). Our contributions include the following. Existing grammars cannot handle standard triple helix structures. Based on the *simple tree adjoining grammar* (STAG) defined by [7] that can handle pseudoknots, we provide an extended version to cover the standard triple helix structures. Since STAG is an ambiguous grammar (i.e. there can be more than one derviation forming the same structure), we remove the ambiguity by introducing some restrictions on the grammar. Finally, we develop a dynamic programming algorithm that runs in $O(n^3)$ time, where n is the length of the input RNA sequence, to report the most probable structure[1] based on the probability measures. According to our experiments, the performance of our tool is

[1] The tool can be modified to report the top x possible structures, but for simplicity, we only consider the most probable structure in all our experiments.

reasonably good (with average sensitivity and specificity higher than 80% for base pairs and over 70% for tertiary interactions) when it is used for prediction of triple helix structures.

2 Standard Triple Helix

Based on [14], the formal definition of a standard triple helix is listed as follows. Let $A = a_1 a_2 \dots a_n$ be a length-n RNA sequence. Let M be the set of base pairs denoted as $M = \{(i,j) \mid 1 \leq i < j \leq n, (a_i, a_j) \text{ is a base pair}\}$. The tertiary interactions P of A are defined as follows. The interaction of the base pair (i,j) and the single stranded nucleotide k is denoted as $(i,j) * k$. That is, $P = \{(i,j) * k \mid (i,j) \in M, a_k \text{ is a single stranded nucleotide and interacts with } (a_i, a_j)\}$. Then, $H = (M, P)$ is referred as the triple helix structure of A.

The secondary structure still obeys the rule that no two base pairs share the same position. That is, for any $(i_1, j_1), (i_2, j_2) \in M$, $i_1 \neq j_2$, $i_2 \neq j_1$, and $i_1 = i_2$ if and only if $j_1 = j_2$. However, the tertiary interactions do not follow this rule, so that for any $(i_1, j_1) * k_1, (i_2, j_2) * k_2 \in P$, if $i_1 = i_2$ and $j_1 = j_2$, it does not imply $k_1 = k_2$; also, if $k_1 = k_2$, it does not imply $i_1 = i_2$ and $j_1 = j_2$. $H = (M, P)$ is a *standard triple helix structure*, as illustrated in Figure 1(d), if $\exists x_1, x_2 (1 \leq x_1 < x_2 \leq n)$, so that base pairs in M can be partitioned into two groups $R_1 = \{(i,j) \in M \mid 1 \leq i < x_1 \leq j < x_2\}$ and $R_2 = \{(i,j) \in M \mid x_1 \leq i < x_2 \leq j \leq n\}$, and H satisfies the following.

(1) For any two base pairs $(i_1, j_1), (i_2, j_2) \in R_k$, $k = 1$ or 2, either $i_1 < i_2 < j_2 < j_1$ or $i_2 < i_1 < j_1 < j_2$. That is, the base pairs in the same group do not cross. We say M forms a *simple pseudoknot* structure.
(2) For any $(i,j) * k \in P$, if $(i,j) \in R_1$, then $x_2 \leq k \leq n$ and $\nexists (i', j') \in R_2$ such that $j \leq i' \leq k \leq j'$ or $i' \leq j \leq j' \leq k$. This is to ensure that k is from a region outside that of R_1, and there does not exist base pairs in R_2 crossing with the tertiary interaction. Similarly, if $(i,j) \in R_2$, then $1 \leq k < x_1$ and $\nexists (i', j') \in R_1$ such that $k \leq i' \leq i \leq j'$ or $i' \leq k \leq j' \leq i$.
(3) For any $(i_1, j_1) * k_1, (i_2, j_2) * k_2 \in P$, if $(i_1, j_1), (i_2, j_2) \in R_1$, then $i_1 \leq i_2 \Leftrightarrow k_1 \leq k_2$, $i_2 \leq i_1 \Leftrightarrow k_2 \leq k_1$. This is to ensure that if the same single stranded nucleotide interacts with two base pairs, the interactions do not cross. Similarly, if $(i_1, j_1), (i_2, j_2) \in R_2$, then $j_1 \leq j_2 \Leftrightarrow k_1 \leq k_2$, $j_2 \leq j_1 \Leftrightarrow k_2 \leq k_1$.

3 Method

3.1 Simple Tree Adjoining Grammar

Simple Tree Adjoining Grammar (STAG) is a tree-based grammar for the generation of strings. The basic idea is to start with an initial tree, and then by repeatedly replacing some internal node of the current tree with another tree, bases or base pairs can simultaneously be added to the string that the tree represents. STAG can be used to predict pseudoknotted structures [7].

Let V be a finite set of alphabets and Σ be the terminal alphabet where $\Sigma \subset V$. Let γ be a tree over V such that (1) each internal node must be labeled with a nonterminal; (2) each leaf node can be labeled with a terminal or a nonterminal symbol; (3) each internal node can have any number of children; and (4) each node has a state, either active or inactive. A tree is *simple* and *active* if there is only one internal node active.

Let $Y(\gamma)$ (i.e. *yield* of a tree rooted at γ) be the string of labels of the leaf nodes of γ from top to bottom and from left to right. Precisely, it is defined as follows (let γ^1,γ^2,...,γ^n be the children of γ):

$$Y(\gamma) = \begin{cases} \text{label of } \gamma \text{ // if } \gamma \text{ is a leaf node} \\ Y(\gamma^1)Y(\gamma^2)...Y(\gamma^n) \text{ // otherwise} \end{cases}$$

In STAG, a tree β is an *adjunct tree* if: (1) there are only leaves labeled with nonterminal symbols; (2) there is only one internal node active; (3) the active internal node is along the backbone. The *backbone* is the path from the root to the leaf with nonterminal symbol.

A simple active tree α can be *adjoined* by an adjunct tree β and form a new tree denoted by $\alpha + \beta$. The adjoining interaction consists of the following operations (as shown in Figure 1e): (1) the active node in α is replaced by the tree β; and (2) the children of the active node in α become the children of the leaf with nonterminal symbol in β.

Definition 1. *$G(C, A)$ is defined as **Simple Tree Adjoining Grammar**, where C is a set of trees, all trees inside are simple and active, their yields are empty strings,and A is a set of adjunct trees.*

*A tree γ is a **derived** tree of G (where the set of the derived trees of G is denoted as $D(G)$) if either of the following constraints is satisfied (which is a recursive definition): (1) $\gamma = \alpha + \beta$ for $\alpha \in C$, $\beta \in A$. (2) $\gamma = d + \beta$ for $d \in D(G)$, $\beta \in B$.*

The language of G (denoted as $L(G)$) is defined as follows: $L(G) = \{w|w = Y(d)$ where $d \in D(G)\}$.

3.2 Structural Prediction for Triple Helix

To model the generation of an RNA with triple helix structure, we set the center tree and the adjunct trees as shown in Figure 2a. There is one center tree and nine adjunct trees. Every adjunct tree will contribute at least one base to the RNA sequence. More precisely, the trees TS1, TS2 and TS3 will produce a single base, while TP1 and TP2 will produce a base pair. Similarly, the trees TSH1 and TSH2 are for producing a single base which has tertiary interaction with an existing base pair, while the trees TPH1 and TPH2 are for producing a base pair which has tertiary interaction with a single base. In the following, we will describe these adjunct trees and how a triple helix structure is generated.

As shown in Figure 1e, the yield of an active tree can be viewed as the concatenation of three sequences: sequence S_1 (i.e. $a_1a_2...a_i$) which is from the

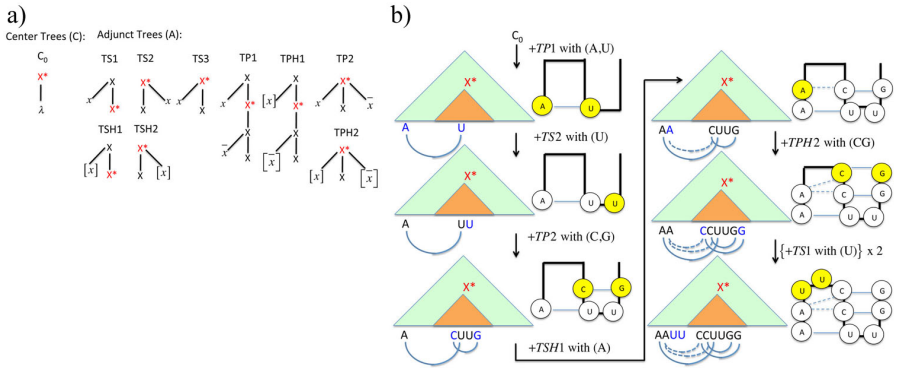

Fig. 2. a) The center tree and the adjunct trees of STAG for modeling RNA triple helix structure. The adjunct trees TS1, TS2 and TS3 generate single base. TP1 and TP2 generate base pairs. TSH1 and TSH2 generate single bases which interact with existing base pairs. TPH1 and TPH2 generate base pairs which interact with existing single bases. b) An example of generation of an RNA with triple helix structure by using STAG.

left part of the tree excluding the subtree of the active node; sequence S_2 (i.e. $b_1b_2...b_j$) which is from the subtree rooted at the active node; and sequence S_3 (i.e. $c_1c_2...c_k$) which is from the right part of the tree excluding the subtree of the active node. And by using the set of adjunct trees in Figure 2a, sequence S_3 is always an empty string, because none of the adjunct trees contribute any base to the sequence S_3. An RNA sequence can be viewed as the concatenation of S_1 and S_2 (as in Figure 1(d)), where S_1 represents the region $[1, x_1 - 1]$ while S_2 represents the regions $[x_1, n]$. The following lists out how the sequence S_1 and S_2 be modified when the tree is adjoined by a different adjunct tree. There are nine different operations (i.e one for each adjunct tree):

1. Adjoined by TS1: add a single base to the end of S_1.
2. Adjoined by TS2: add a single base to the end of S_2.
3. Adjoined by TS3: add a single base to the beginning of S_2.
4. Adjoined by TP1: add a base pair with bases at the end of S_1 and the beginning of S_2.
5. Adjoined by TP2: add a base pair with bases at the beginning and the end of S_2.
6. Adjoined by TSH1: add a single base at the end of S_1, which interacts with an existing base pair whose bases are at the beginning and the end of S_2, provided that the beginning and the end of S_2 are base pair.
7. Adjoined by TSH2: add a single base at the end of S_2, which interacts with an existing base pair whose bases are at the end of S_1 and the beginning of S_2, provided that the end of S_1 and the beginning of S_2 are base pair.
8. Adjoined by TPH1: add a base pair whose bases are at the end of S_1 and at the beginning of S_2, which interacts with the single base existing at the end of S_2, provided that the end of S_2 is a single base.

9. Adjoined by TPH2: add a base pair whose bases are at the beginning and the end of S_2, which interacts with the single base existing at the end of S_1, provided that the end of S_1 is a single base.

By using the above nine operations, it can build up any RNA with standard triple helix structure and any structure it comes up is a standard triple helix. An example of the generation of a standard triple helix is shown in Figure 2b. Under this model, different derivations may generate the same RNA sequence, but the corresponding secondary structures may be different. We associate probabilities for each tree operation (trained using real data); consequently, on given any input RNA sequence $A[1...n]$, we can report the derivation (and thus the corresponding secondary structure) that is the most probable.

To simplify the model, we assume that the probability of applying a particular tree p is independent of the current sequence, but depends on the previously applied tree p' and the bases involved in p. The probability of obtaining an input RNA sequence $A[1...n]$ with a particular secondary structure ζ is defined to be the product of the probabilities of the applied trees for operation in the corresponding derivation. To find out the most probable secondary structure ζ^* is equivalent to finding a ζ^* with the maximum summation of the log values of the corresponding derivation probabilities. Now, we define the following notations and present the recurrences.

- $M(i,j,k,p)$: the maximum score of the substructure $A[1...i] \cup A[j...k]$ of the sequence A if the last operation applied is p.
- $M_L(i,j,k,p)$: the maximum score of the substructure $A[1...i] \cup A[j...k]$ of the sequence A if the last operation applied is p and (i,j) is a base pair.
- $M_R(i,j,k,p)$: the maximum score of the substructure $A[1...i] \cup A[j...k]$ of the sequence A if the last operation applied is p and (j,k) is a base pair.
- $M_F(i,j,k,p)$: the maximum score of the substructure $A[1...i] \cup A[j...k]$ of the sequence A if the last operation applied is p and i is a single base.
- $M_G(i,j,k,p)$: the maximum score of the substructure $A[1...i] \cup A[j...k]$ of the sequence A if the last operation applied is p and k is a single base.
- $\texttt{score}(p,p',X)$: the score from previous operation p' to the current operation p with character set X. The scores are fixed in the parameter-tuning step of the method.
- $\texttt{charset}(i,j,k,p)$: the base(s) involved when the current operation p is applied.

$$M(i,j,k,p) = \max \begin{cases} M_L(i,j,k,p), M_R(i,j,k,p), M_F(i,j,k,p), M_G(i,j,k,p) \\ \text{// if } p \text{ is operation 3, also check the following score} \\ \max_{p'}\{M(i,j+1,k,p') + \texttt{score}(p,p',\texttt{charset}(i,j,k,p))\} \end{cases}$$

$$M_L(i,j,k,p) = \begin{cases} \text{// if } p \text{ is operation 1, 3, 5, 6 or 9} \\ -\infty \\ \text{// else if } p \text{ is operation 2 or 7} \\ \max_{p'}\{M_L(i,j,k-1,p') + \texttt{score}(p,p',\texttt{charset}(i,j,k,p))\} \\ \text{// else if } p \text{ is operation 4} \\ \max_{p'}\{M(i-1,j+1,k,p') + \texttt{score}(p,p',\texttt{charset}(i,j,k,p))\} \\ \text{// else if } p \text{ is operation 8} \\ \max_{p'}\{M_G(i-1,j+1,k,p') + \texttt{score}(p,p',\texttt{charset}(i,j,k,p))\} \end{cases}$$

The recurrence of $M_R(i,j,k,p)$ is analogous to that of $M_L(i,j,k,p)$. And the recurrence of $M_F(i,j,k,p)$ and $M_G(i,j,k,p)$ are similar too. The desired derivation corresponds to the entry among $M(i,i+1,n,p)$, for all possible i and p, that contains the maximum value. Once this entry is known, it is straightforward to obtain the corresponding secondary structure ζ^* by backtracking. By performing dynamic programming, each entry of $M(i,j,k,p)$, $M_L(i,j,k,p)$, $M_R(i,j,k,p)$, $M_F(i,j,k,p)$, and $M_G(i,j,k,p)$ can be computed in $O(1)$ time based on the previously computed entries. As there are altogether $O(n^3)$ entries to be filled, the time complexity of our prediction algorithm is $O(n^3)$.

A structural prediction grammar is ambiguous if there exists more than one derivation forming the same secondary structure, and [16] showed that an ambiguous grammar may not always report the optimal secondary structure correctly. The details of how the ambiguity of the grammar is removed is described in Appendix I. The accuracy of the prediction algorithm largely depends on how accurate the parameters $\texttt{score}(p,p',X)$ are. We only consider AU, UA, CG, GC, GU and UG as the possible base pairs and also regard the score for the operation with base pair AU (or CG or GU) is the same as that with base pair UA (or GC or UG). After considering all these together with the restrictions for preventing ambiguity, there are around 360 parameters $\texttt{score}(p,p',X)$ required to compute. We follow the maximum-likelihood approach mentioned by [17] to tune the grammar by a set of RNA sequences with known triple helix structures. $score(p,p',X)$ can be divided into two part: transition $a_{p'\to p}$ is score from previous operation p' to the current operation p, and emission $e_p(X)$ is score for X is involved in operation p. Since ambiguity is removed, operations series for each training sequence are known. We count the number of times each transition and emission, let these be $A_{p'\to p}$ and $E_p(X)$. Then the maximum likelihood estimators for $a_{p'\to p}$ and $e_p(X)$ are given by $a_{p'\to p} = \frac{A_{p'\to p}}{\sum_{l'} A_{p'\to l'}}$ and $e_p(X) = \frac{E_p(X)}{\sum_{X'} E_p(X')}$

With a set of training data, it takes O(n) time to calculate operation series for each sequence, and O(1) time to calculate all maximum likelihood estimators. For details, one may refer to [17].

4 Experimental Results

We implemented both the tuning and the prediction algorithms using C. There are three RNA families from Rfam 9.1 database with triple helix structures: RF00024, RF01050 and RF01074 (as listed in Table 1). The corresponding triple helix structure of each family can be deduced from [8, 9, 11, 18]. In the first experiment, we extracted the sequences of the triple helix regions of all the seed members (in Rfam 9.1 database, for each family, there is a set of reliable members that are regarded as seed members). It is found that the same model can hardly work well for the RNAs with large length difference. Since the lengths of the triple helix regions of the families RF00024 and RF01050 are similar, we put all the sequences from these two families together as set D_1, and the other sequences as set D_2.

Table 1. The families with triple helix structures

Family ID	# of seed members	Ave. length of triple helix region
RF00024	37	118
RF01050	13	99
RF01074	4	28

Table 2. Performance of triple helix prediction on the RNA sequences in set D_1 when using 10-fold cross-validation approach

Group	Base pairs		Tertiary interactions		Group	Base pairs		Tertiary interactions	
	Sen (%)	Spec (%)	Sen (%)	Spec (%)		Sen (%)	Spec (%)	Sen (%)	Spec (%)
G_1	90.5	89.6	76.2	88.9	G_6	95.1	90.1	78.3	78.3
G_2	81.3	77.1	72.2	76.5	G_7	56.6	65.8	44.4	38.1
G_3	97.1	90.3	88.9	88.9	G_8	90.8	90.1	95.7	88.0
G_4	79.5	76.5	38.5	71.4	G_9	92.5	87.5	71.4	74.1
G_5	86.8	87.5	73.9	77.3	G_{10}	74.5	82.0	64.5	66.7
On average: for base pairs, sensitivity **84.5** and specificity **83.7** for tertiary interactions, sensitivity **70.4** and specificity **74.8**									

We use the 10-fold cross-validation approach to evaluate the accuracy of our prediction tool. We evenly distributed all the sequences in the set D_1 into ten groups $G_1, G_2, \ldots, G_{10}$ such that the ratios of the sequences from each family are similar in each group. Next, we repeat the following procedure for each i from 1 to 10: We keep the group G_i aside, so that all the sequences as well as their corresponding triple helix structures from the other groups (i.e., $D_1 \backslash G_i$) were used for tuning our model; after that, the tuned model was used to predict the triple helix structure of each of the sequences in group G_i, and the predicted structure of each sequence was then compared with the corresponding real structure.

Our tool will report a set of base pairs as well as the tertiary interactions in the given region. Table 2 shows the summary of the performance of our prediction algorithm. Our method can predict the base pairs well with average sensitivity 84.5% and specificity 83.7%, and has a reasonable performance on the tertiary interaction prediction with over 70% in both sensitivity and specificity. Figure 3a shows an example of the predicted structure of the triple helix region of a sequence in family RF00024. The predicted structure is very similar to the real structure. Only one base pair (15,49) and one tertiary interaction (22,42)*74 are not predicted. Only one base pair (7,51) which should not exist is added.

For D_2, all RNA sequences are from the same family RF01074. The triple helix structures of the sequences are quite complex. There exist two or more single bases having tertiary interactions with the same base pair, and also two or more base pairs having tertiary interactions with the same single base. The tuned model may be over-fitted due to the small number of sequences in this set, but we still present the results here in order to show that our model is flexible enough to handle such a complex triple helix structure. We used 4-fold cross-validation technique in this set. For base pair prediction, the average sensitivity

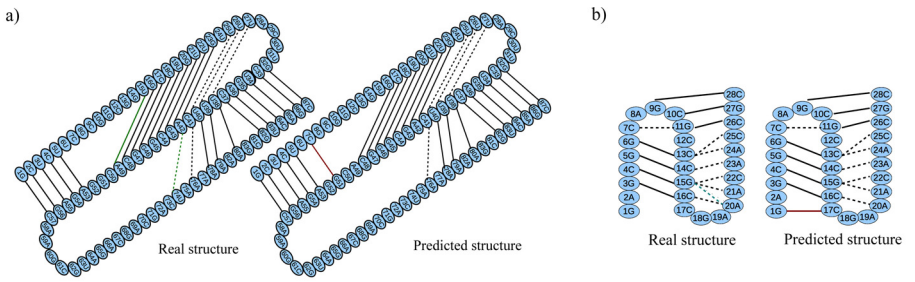

Fig. 3. a) Predicted triple helix region of sequence AF221906 of the family RF00024. b) Predicted triple helix region of sequence AF473561 of the family RF01074.

and specificity is 97.5% and 87.8%, respectively. For tertiary interaction prediction, the average sensitivity and specificity is 72.5% and 92.7%, respectively. Figure 3b shows an example of the predicted structure of the triple helix region for a sequence in the family RF01074. The predicted structure is very similar to the real structure, despite that the triple helix structure is quite complex.

Table 3. Experiment on the whole pipeline for the family RF00024

Seq ID	Annotated pseudoknot region	Reported pseudoknot region by vsfold5	Tertiary interaction predicted		Seq ID	Annotated pseudoknot region	Reported pseudoknot region by vsfold5	Tertiary interaction predicted	
			Sensitivity	Specificity				Sensitivity	Specificity
AF221911	55-143	7-173	80%	80%	AF221924	28-157	42-148	60%	60%
AF221913	63-148	52-148	100%	100%	AF221932	63-183	50-184	80%	80%
AF221916	19-139	2-165	50%	50%	AF221923	64-184	45-181	80%	80%
AF221926	55-138	51-172	80%	80%	AF221929	50-169	70-181	80%	80%
AF221940	56-135	45-169	100%	20%	AF221937	65-184	38-189	80%	80%
AF221934	60-155	48-184	100%	83%	AF221909	33-151	53-161	60%	60%
AF221927	60-156	31-152	80%	80%	AY058901	22-144	16-156	75%	75%
AF221910	62-151	74-197	100%	100%	AC121792	22-144	20-160	75%	50%
On average: Sensitivity **80%** Specificity **72%**									

In the second experiment, we try the whole pipeline for triple helix prediction on RNA sequences. Given an RNA sequence, the pseudoknotted structure will first be predicted by vsfold5 [19]. Then for those pseudoknotted regions reported by the tools, our tool predicts the triple helix structure. The maximum length of sequence vsfold5 supports is 450. The sequences in RF01050 are too long. Thus we selected those not-too-long sequences in RF00024 for the experiment. The pipeline is found to be feasible and quite effective (as shown in Table 3). On average, the sensitivity is 80%, while the specificity is 72%.

5 Discussion and Conclusions

To further evaluate our algorithm on the distinguishing power between regions containing a triple helix structure and those not containing one, we have selected

the families with simple pseudoknot structures (and no reported triple helix structures) as negative cases and it is found that our method can distinguish between regions with or without triple helix structure reasonably well. Since there are not much real data with known tertiary structures, further studies include collecting more real data, conducting a more comprehensive evaluation on the algorithm, and refining the grammar and the prediction algorithm to cater for more types of triple helix structures.

Acknowledgement. This project is partially supported by the General Research Fund (GRF) of the Hong Kong Government (HKU 719611E).

References

1. Lyngso, R., Pedersen, C.: A dynamic programming algorithm for RNA structure prediction including pseudoknots. In: Proc. of the Fourth Annual International Conferences on Compututational Molecular Biology (RECOMB 2000). ACM Press (2000)
2. Akutsu, T.: Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots. Discrete Applied Mathematics 104, 45–62 (2000)
3. Chen, H., Condon, A., Jabbari, H.: An $O(n^5)$ algorithm for MFE prediction of kissing hairpins and 4- chains in nucleic acids. Journal of Computational Biology 16(6), 803–815 (2009)
4. Dirks, R., Pierce, N.: A partition function algorithm for nucleic acid secondary structure including pseudoknots. Journal of Comput. Chem. 24(13), 1664–1677 (2003)
5. Reeder, J., Giegerich, R.: Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. BMC Bioinformatics 5, 104 (2004)
6. Rivas, E., Eddy, S.: A dynamic programming algorithm for RNA structure prediction including pseudoknots. Journal of Molecular Biology 285(5), 2053–2068 (1999)
7. Uemura, Y., Hasegawa, A., Kobayashi, S., Yokomori, T.: Tree adjoining grammars for RNA structure prediction. Theoretical Computer Science 210, 277–303 (1999)
8. Qiao, F., Cech, T.R.: Triple-helix structure in telomerase RNA contributes to catalysis. Nature Structural and Molecular Biology 15(6), 634–640 (2008)
9. Chen, J.L., Greider, C.W.: Functional analysis of the pseudoknot structure in human telomerase RNA. Proc. Natl. Acad. Sci. USA 102, 8080–8085 (2005)
10. Theimer, C.A., Blois, C.A., Feigon, J.: Structure of the human telomerase RNA pseudoknot reveals conserved tertiary interactions essential for function. Molecular Cell 17, 671–682 (2005)
11. Su, L., Chen, L., Egli, M., Berger, J.M., Rich, A.: Minor groove RNA triplex in the crystal structure of a ribosomal frameshifting viral pseudoknot. Nature Structural Biology 6(3), 285–292 (1999)
12. Chen, X., Chamorro, M., Lee, S.I., Shen, L.X., Hines, J.V., Tinoco Jr., I., Varmus, H.E.: Structural and functional studies of retroviral RNA pseudoknots involved in ribosomal frameshifting: nucleotides at the junction of the two stems are important for efficient ribosomal frameshifting. EMBO 14(4), 842–852 (1995)

13. Siederdissen, C., Bernhart, S., Stadler, P., Hofacker, I.: A folding algorithm for extended RNA secondary structures. Bioinformatics 27, i29–i36 (2011) (ISMB 2011)
14. Wong, T.K., Lam, T., Yiu, S.: Structural alignment of RNA with triple helix structure. Journal of Computational Biology 19(4), 365–378 (2012)
15. Matsui, H., Sato, K., Sakakibara, Y.: Pair stochastic tree adjoining grammars for aligning and predicting pseudoknot RNA structures. Bioinformatics 21, 2611–2617 (2005)
16. Dowell, R.D., Eddy, S.R.: Evaluation of several lightweight stochastic context-free grammars for rna secondary structure prediction. BMC Bioinformatics 5, 71 (2004)
17. Durbin, R., Eddy, S.R., Krogh, A., Mitchison, G.: Covariance models: SCFG-based RNA profiles. In: Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids. Cambridge University Press (1998)
18. Chastain, M., Tinoco, I.J.: A base-triple structural domain in RNA. Biochemistry 31, 12733–12741 (1992)
19. Dawson, W.K., Fujiwara, K., Kawai, G.: Prediction of RNA pseudoknots using heuristic modeling with mapping and sequential folding. PLoS One 2(9), e905 (2007)

Appendix I : Removing the Grammar Ambiguity

A structural prediction grammar is ambiguous if there exists more than one derivation forming the same secondary structure, and [16] showed that an ambiguous grammar often could not report the optimal secondary structure correctly. Therefore, we have to remove the ambiguity of the grammar such that each derivation can report a unique secondary structure.

First, there exist different operation series that come up with the same structure. For example, an operation 1 followed by an operation 2 would come up the same structure as an operation 2 followed by an operation 1. As shown in Figure 4a, we do not allow the operation series in right which produce the same structure as the operation series in left. Also, some operation sequences are not possible. For example, to perform operation 6, the beginning and the ending bases of B have to be a base pair. Therefore, it is not possible for an operation 3 followed by an operation 6 (because after operation 3, a single base will be added to the beginning of B). Figure 4b lists all of the cases.

Second, as one may notice, the positions of X_1 and X_2 may not be unique according to the definition in Section 2. In order to avoid the ambiguity, as shown in Figure 4c, we set the values of X_1 and X_2 as follows:

$$X_1 = \min\{\min j \mid (i,j) \in R_1, \min i \mid (i,j) \in R_2\}$$

$$X_2 = \max\{\max j \mid (i,j) \in R_1, \max i \mid (i,j) \in R_2\} + 1$$

where R_1 and R_2 are sets of base pairs defined in Section 2.

Since $S_1 = [0...X_1 - 1]$ and $S_2 = [X_1...n]$. Therefore, we have the following restrictions:

a)

The operation series which come up the same structure. (To avoid ambiguity, the operations in the right side of each equation are not allowed.)

$1 \to 2 \equiv 2 \to 1$	$2 \to 3 \equiv 3 \to 2$
$1 \to 3 \equiv 3 \to 1$	$4 \to 2 \equiv 2 \to 4$
$5 \to 1 \equiv 1 \to 5$	$4 \to 7 \equiv 2 \to 8$
$5 \to 6 \equiv 1 \to 9$	

b)

The followings are the impossible operation series.

$2 \to 6$	$8 \to 6$	$6 \to 7$	$9 \to 8$
$3 \to 6$	$1 \to 7$	$9 \to 7$	$4 \to 9$
$4 \to 6$	$3 \to 7$	$5 \to 8$	$7 \to 9$
$7 \to 6$	$5 \to 7$	$6 \to 8$	$8 \to 9$

c)

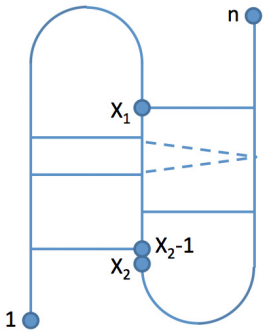

Fig. 4. a) The redundant operation series (in the right side of each equation). b) The impossible operation series. c) To remove the ambiguity, we define the exact values of X_1 and X_2.

1. S_2 cannot start with any single base. Since $3 \to 1$ and $3 \to 2$ are not allowed (see Figure 4a), we only need to restrict the operation 3 not being the last operation.

2. $X_2 - 1$ can be regarded as a center position of S_2 (which means all bases with positions $\leq X_2 - 1$ have to be added from the beginning of S_2, and all bases with positions $> X_2 - 1$ are added from the end of S_2) and the position $X_2 - 1$ cannot be a single base. There are two cases: the base $X_2 - 1$ belongs to a base pair $\in R_1$; or it belongs to a base pair $\in R_2$. In case 1, the operation 4 should be the first operation to add a base into S_2 and that position would be $X_2 - 1$. In case 2, there should be no operation 3 until the operation 5 or 9 exists. The left position of the base pair added would be $X_2 - 1$. According to the Figure 4a, since $2 \to 4$ and $2 \to 8$ are not allowed, therefore: we only need to restrict the operation 3 until the operation 4, 5 or 9 exists.

3. When R_1 and R_2 are empty, only operation 1 is allowed. i.e. When operations 4, 5, 8, 9 do not exist, only operation 1 can be the last operation.

4. If R_2 is not empty, R_1 has to be not empty too. i.e. If operation 5 or 9 exist, operation 4 or 8 has to exist before ends.

The above restrictions together with the restrictions listed in Figure 4 can make the grammar become unambiguous. Different derivation reports a unique secondary structure.