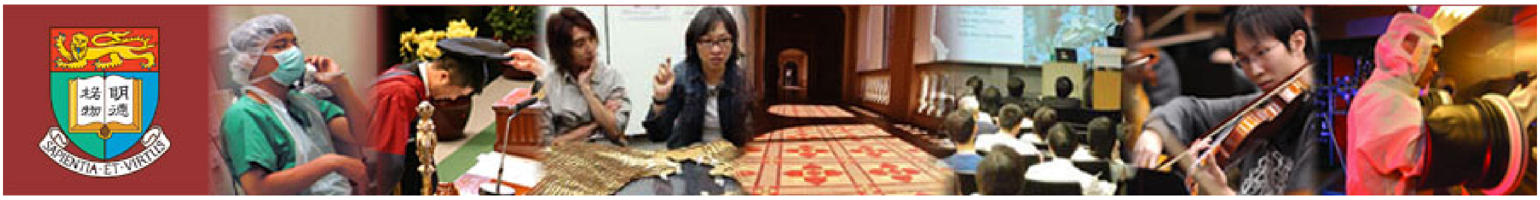


JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 18, Number 1, 2011
© Mary Ann Liebert, Inc.
Pp. 97–108
DOI: 10.1089/cmb.2009.0275

Structural Alignment of RNA with Complex Pseudoknot Structure

THOMAS K.F. WONG[1], T.W. LAM[1], WING-KIN SUNG[2],
BRENDA W.Y. CHEUNG[1], and S.M. YIU[1]

ABSTRACT

The secondary structure of an ncRNA molecule is known to play an important role in its biological functions. Aligning a known ncRNA to a target candidate to determine the sequence and structural similarity helps in identifying de novo ncRNA molecules that are in the same family of the known ncRNA. However, existing algorithms cannot handle complex pseudoknot structures which are found in nature. In this article, we propose algorithms to handle two types of complex pseudoknots: simple non-standard pseudoknots and recursive pseudoknots. Although our methods are not designed for general pseudoknots, it already covers all known ncRNAs in both Rfam and PseudoBase databases. An evaluation of our algorithms shows that it is useful to identify ncRNA molecules in other species which are in the same family of a known ncRNA.

Key words: dynamic programming, secondary structure, sequences.

1. INTRODUCTION

A non-coding RNA (ncRNA) is a RNA molecule that does not translate into a protein. It has been shown to be involved in many biological processes (Frank and Pace, 1998, Nguyen et al., 2001, Yang et al., 2001). The number of ncRNAs within the human genome was underestimated before, but recently some databases reveal over 212,000 ncRNAs (He et al., 2007) and more than 1,300 ncRNA families (Griffiths-Jones et al., 2003). Large discoveries of ncRNAs and their families show the possibilities that ncRNAs may be as diverse as protein molecules (Eddy, 2001). Identifying ncRNAs is an important problem in biological study.

It is known that the secondary structure of an ncRNA molecule usually plays an important role in its biological functions. Some researches attempted to identify ncRNAs by considering the stability of secondary structures formed by the substrings of a given genome (Le et al., 1990). This method is not effective because a random sequence with high GC composition also allows an energetically favorable secondary structure (Rivas and Eddy, 2000). A more promising direction is comparative approach which makes use of the idea that if a DNA region from which a RNA is transcribed has similar sequence and structure to a known ncRNA, then this region is likely to be an ncRNA gene whose corresponding ncRNA is in the same family of the known ncRNA. Thus, to locate ncRNAs in a genome, we can use a known

[1] Department of Computer Science, University of Hong Kong, Hong Kong.
[2] School of Computing, National University of Singapore, Singapore.

ncRNA as a query and search along the genome for substrings with similar sequence and structure to the query. The key of this approach is to compute the structural alignment between a query sequence with known structure and a target sequence with unknown structure. The alignment score represents their sequence and structural similarity. RSEARCH (Klein and Eddy, 2003) and FASTR (Zhang et al., 2005) belong to this category.

However, these tools do not support pseudoknots. Given two base pairs at positions (i, j) and (i', j'), where $i < j$ and $i' < j'$, pseudoknots are base pairs *crossing* each other, i.e., $i < i' < j < j'$ or $i' < i < j' < j$. In some studies, secondary structures including pseudoknots are found involved in some functions such as telomerase (Chen and Greider, 2005), catalytic functions (Dam et al., 1992), and self-splicing introns (Adams et al., 2004). The presence of pseudoknots makes the problem computationally harder. Usually, the large time complexity and considerable memory required for these algorithms make it impractical to search long pseudoknotted ncRNA along the genome.

Recently, Han et al. (2008) developed PAL to solve the problem that supports secondary structures with standard pseudoknot of degree k and their algorithm runs in $O(kmn^k)$ where m is the length of the query sequence and n is the length of the target sequence. Their algorithm cannot handle more complex pseudoknot structures such as one with 3 base pairs mutually crossing each other (i.e., any two of them are crossing) as in Figure 1a or the structure allowing recursive pseudoknots (i.e., pseudoknot/regular structures exist within another pseudoknot structure) as in Figure 1b. In Rfam 9.1 database (Griffiths-Jones et al., 2003), among 71 pseudoknotted families, 18 of them have complex pseudoknot structure. In the PseudoBase database (van Batenburg et al., 2000), among 304 pseudoknot RNAs, 8 of them have complex pseudoknot structures. It is possible that more and more ncRNAs with complex pseudoknots will be discovered later.

In this paper, we consider more complex pseudoknot structures which are found in nature. We define a class of pseudoknots called *simple non-standard pseudoknot* which allows some restricted cases with 3 base pairs mutually crossing each other. Our algorithm can apply to this complex structure using the same time complexity as the PAL algorithm (Han et al., 2008) for standard pseudoknot structure (i.e., $O(kmn^k)$ for degree k). Then, we propose an algorithm to handle a special type of 2-level recursive pseudoknot structure which runs in $O(kmn^{k+1})$ time. The algorithm can be extended to handle other recursive pseudoknot structures with the worst case time complexity of $O(kmn^{k+2})$.

Although our method is not designed for generic pseudoknots, we found that our method already covers all ncRNAs with complex pseudoknots in both Rfam 9.1 and PseudoBase databases. A preliminary experiment shows that our algorithms are useful in identifying ncRNAs from other species which are in the same family of a known ncRNA.

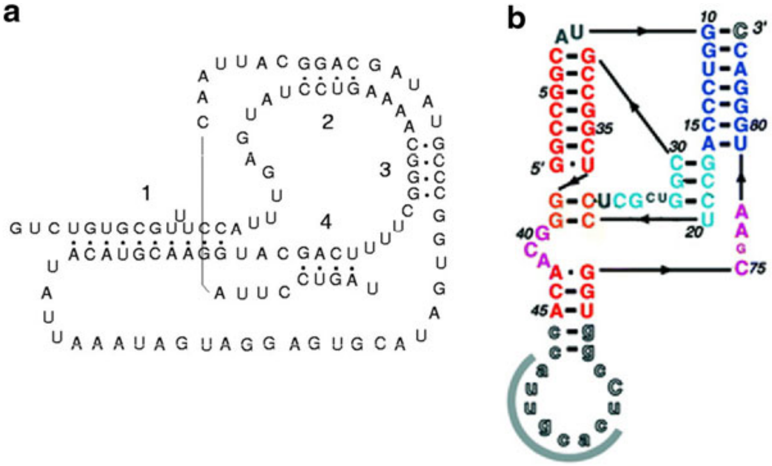


FIG. 1. (**a**) The secondary structure of RF00140 from Rfam 9.1 database (Griffiths-Jones et al., 2003). Consider three base pairs: one from region 1, one from region 2 and one from region 4, they are mutually crossing each other (i.e., any two of them are crossing). (**b**) The secondary structure of self-cleaving ribozymes of hepatitis delta virus from Ferré-D'Amaré et al. (1998) (i.e., RF00094 from Rfam 9.1 database).

2. PSEUDOKNOT DEFINITIONS

Let $A = a_1a_2 \ldots a_m$ be a length-m ncRNA sequence and M be the secondary structure of A. M is represented as a set of base pair positions. i.e., $M = \{(i, j) | 1 \leq i < j \leq m, (a_i, a_j) \text{ is a base pair}\}$. Let $M_{x,y} \subseteq M$ be the set of base pairs within the subsequence $a_xa_{x+1} \ldots a_y$, $1 \leq x < y \leq m$, i.e., $M_{x,y} = \{(i, j) \in M | x \leq i < j \leq y\}$, with $M = M_{1,m}$. We assume that there is no two base pairs sharing the same position, i.e., 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 $j_1 = j_2$.

Definition 1. *$M_{x,y}$ is a* regular structure *if there does not exist two base pairs $(i, j), (k, l) \in M_{x,y}$ such that $i < k < j < l$ or $k < i < l < j$. An empty set is also considered as a regular structure.*

A regular structure is one without pseudoknots. A structure is a standard pseudoknot of degree k if the RNA sequence can be divided into k consecutive regions (Fig. 2a) such that base pairs must have end points in adjacent regions and base pairs that are in the same adjacent regions do not cross each other. The formal definition is as follows.

Definition 2. *$M_{x,y}$ is a* standard pseudoknot of degree $k \geq 3$ *if there exists a set of* pivot points *$x_1, x_2, \ldots, x_{k-1} (x = x_0 < x_1 < x_2 < \ldots < x_{k-1} < x_k = y)$ that satisfy the following. Let $M_w(1 \leq w \leq k-1) = \{(i, j) \in M_{x,y} | x_{w-1} \leq i < x_w \leq j < x_{w+1}\}$. Note that we allow $j = x_k$ for M_{k-1} to resolve the boundary case.*

- *For each $(i, j) \in M_{x,y}, (i, j) \in M_w$ for some $1 \leq w \leq k-1$.*
- *$M_w(1 \leq w \leq k-1)$ is a regular structure.*

A standard pseudoknot of degree 3 is usually referred as a *simple pseudoknot*. Now, we define a simple *non-standard* pseudoknot to include some structures with three base pairs crossing each other. For a simple non-standard pseudoknot of degree k, similar to a standard pseudoknot, the RNA sequence can be divided into k regions with the region at one of the ends (say, the right end) designated as the special region. Base pairs with both end points in the first $k-1$ regions have the same requirements as in a standard pseudoknot. And there is an extra group of base pairs that can start in one of the first $k-2$ regions and end at the last special region and again these pairs do not cross each other (Fig. 2b). See the formal definition below.

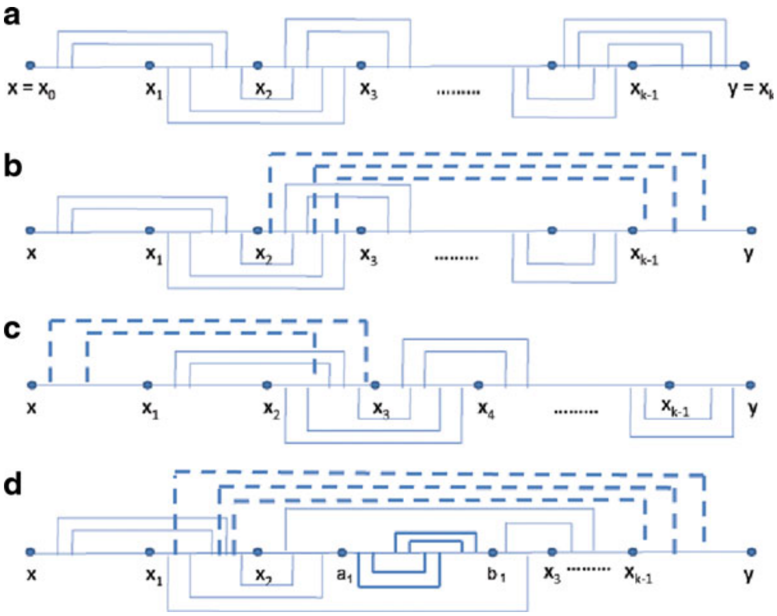


FIG. 2. (**a**) Standard pseudoknot of degree k. (**b**) Simple non-standard recursive pseudoknot of degree k (Type I). (**c**) Simple non-standard recursive pseudoknot of degree k (Type II). (**d**) Recursive pseudoknot (the region $[a_1, b_1]$ is a recursive region).

Definition 3. *$M_{x,y}$ is a* simple non-standard pseudoknot of degree $k \geq 4$ (Type I) *if there exist $x_1, \ldots, x_{k-1}$ and t where $x = x_0 < x_1 < \ldots < x_{k-1} < x_k = y$ and $1 \leq t \leq k-2$ that satisfy the following. Let $M_w(1 \leq w \leq k-2) = \{(i, j) \in M_{x,y} | x_{w-1} \leq i < x_w \leq j < x_{w+1}\}$. Let $X = \{(i, j) \in M_{x,y} | x_{t-1} \leq i < x_t, x_{k-1} \leq j \leq y\}$.*

- *For each $(i, j) \in M_{x,y}$, either $(i, j) \in M_w(1 \leq w \leq k-2)$ or $(i, j) \in X$.*
- *M_w and X are regular structures.*

Type II simple non-standard pseudoknots (Fig. 2c) are symmetric to Type I simple non-standard pseudoknots with the special region on the left end. In the rest of the paper, we only consider Type I simple non-standard pseudoknots and simply refer it as simple non-standard pseudoknots.

Lastly, we define what a recursive pseudoknot is (Fig. 2d).

Definition 4. *$M_{x,y}$ is a recursive pseudoknot of degree $k \geq 3$ if $M_{x,y}$ is either regular, standard pseudoknot of degree k or simple non-standard pseudoknot of degree k (if $k \geq 4$), or $\exists a_1, b_1, \ldots, a_s, b_s (x \leq a_1 < b_1 < \ldots < a_s < b_s \leq y)$ that satisfy the followings. Each M_{a_i, b_i} is called a* recursive region.

- *M_{a_i, b_i}, for $1 \leq i \leq s$, is a recursive pseudoknot of degree $\leq k$.*
- *For each M_{a_i, b_i}, $1 \leq i \leq s$, there does not exist $(i,j) \in M$ that $i \in [a_i, b_i]$ but $j \notin [a_i, b_i]$, or $i \notin [a_i, b_i]$ but $j \in [a_i, b_i]$.*
- *$(M_{x,y} - \bigcup_{1 \leq i \leq s} M_{a_i, b_i})$ is either regular structure, standard pseudoknot of degree $\leq k$ or simple non-standard-pseudoknot of degree $\leq k$.*

3. ALGORITHM FOR SIMPLE NON-STANDARD PSEUDOKNOTS

3.1. Structural alignment

Let $S[1 \ldots m]$ be a query sequence with known secondary structure M, and $T[1 \ldots n]$ be a target sequence with unknown secondary structure. S and T are both sequences of {A,C,G,U}. A structural alignment between S and T is a pair of sequences $S'[1 \ldots r]$ and $T'[1 \ldots r]$ where $r \geq m, n$, S' is obtained from S and T' is obtained from T with spaces inserted to make both of the same length. A space cannot appear in the same position of S' and T'. The score of the alignment, which determines the sequence and structural similarity between S' and T', is defined as follows (Zhang et al., 2005).

$$score = \sum_{i=1}^{r} \gamma(S'[i], T'[i]) + \sum_{\substack{i,j \text{ s.t. } \eta(i), \eta(j) \in M, \\ S'[i], S'[j], T'[i], T'[j] \neq `\_'}} \delta(S'[i], S'[j], T'[i], T'[j]) \qquad (1)$$

where $\eta(i)$ is the corresponding position in S according to the position i in S'; $\gamma(t_1, t_2)$ and $\delta(x_1, y_1, x_2, y_2)$ where $t_1, t_2 \in \{A, C, G, U, `\_'\}$ and $x_1, x_2, y_1, y_2 \in \{A, C, G, U\}$, are scores for character similarity and for base pair similarity, respectively. The problem is to find an alignment to maximize the score.

3.2. Substructure of simple non-standard pseudoknot

We solve the problem using dynamic programming. The key is to define a substructure to enable us to find the solution recursively. For ease understanding of what a substructure is, we draw the pseudoknot structure using another approach (Fig. 3).

We use simple non-standard pseudoknots with degree 4 for illustration. The result can be easily extended to general k. Figure 3b shows the same pseudoknot structure as in Figure 3a. By drawing the pseudoknot structure this way, the base pairs can be drawn without crossing and can be ordered from the top to bottom. According to this ordering, we can define a substructure based on four points on the sequence (Fig. 3c, substructure is highlighted in bold) such that all base pairs are either with both end points inside or outside the substructure. Note that in Figure 3c, $t = 1$ (t is odd), if $t = 2$ (t is even), we have to use a slightly different definition for substructures, otherwise base pairs cannot be ordered from top to bottom without crossing each other (Fig. 3d,e). Note that the two base pairs that cross in Figure 3d is due to the way we draw the pseudoknot, they do not actually cross each other.). These are the only cases we need to consider.

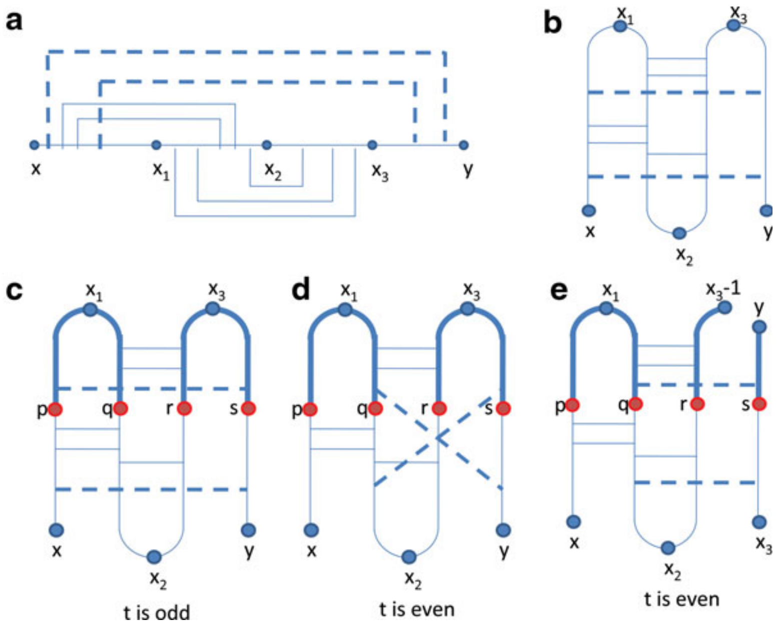


FIG. 3. (**a**) An example of simple non-standard pseudoknot when $t = 1$ (t is odd). (**b**) Another view of the same structure when t is odd. (**c**) Definition of substructure of a simple non-standard pseudoknot when t is odd. (**d**) An illustration that the same definition of substructure cannot be used when t is even due to the cross-looking of base pairs. (**e**) Definition of substructure of a simple non-standard pseudoknot when t is even.

Now, we formally define what a substructure is. Let $S[x..y]$ be an RNA sequence with known simple non-standard pseudoknot structure M of degree 4. Note that x_1, x_2, x_3 and t are known. Let $v = (p, q, r, s)$ be a quadruple with $x \le p < x_1 \le q < x_2 \le r \le x_3 < s \le y$. If t is odd, define the *subregion* $R_{odd}(S, v) = [p, q] \cup [r, s]$. Otherwise, define the subregion $R_{even}(S, x_3, v) = [p, q] \cup [r, x_3 - 1] \cup [s, y]$. Note that x_3 is not a parameter, but a fixed value for S. Let $Struct(R_x) = \{(i, j) \in M | i, j \in R_x\}$ where R_x is a subregion.

We say that a subregion R_x defines a valid substructure ($Struct(R_x)$) of M if there does not exist $(i, j) \in M$ such that one endpoint of (i, j) is in R_x and the other is outside the region. Obviously, $Struct(R_x)$ is also a simple non-standard pseudoknot structure.

3.3. Dynamic programming

Let $S[1, m]$ be the query sequence with known structure M and $T[1, n]$ be the target sequence with unknown structure. Note that the pivot points x_1, x_2, x_3 and t for S is known. We can apply the definitions of R_{odd} and R_{even} to T. If t is odd, for any $v' = (e, f, g, h)$ such that $1 \le e < f < g < h \le n$, we define the subregion $R_{odd}(T, v') = [e, f] \cup [g, h]$. If t is even, for any $v' = (e, f, g, h)$ and x'_3 such that $1 \le e < f < g < x'_3 \le h \le n$, we define the subregion $R_{even}(T, x'_3, v') = [e, f] \cup [g, x'_3 - 1] \cup [h, n]$. Note that since the structure of T is unknown, x'_3 is a parameter.

Define $C(R_x, R_y)$ be the score of the optimal alignment between a subregion R_x in S with substructure $Struct(R_x)$ and a subregion R_y in T. The score of the optimal alignment between S and T can be obtained as follows. If t is odd, setting $v* = (1, x_2 - 1, x_2, m)$ includes the whole query sequence S, the entry $\max_{x'_2}\{C(R_{odd}(S, v^*), R_{odd}(T, v' = (1, x'_2 - 1, x'_2, n)))\}$ provides the answer. On the other hand, if t is even, setting $v* = (1, x_2 - 1, x_2, x_3)$, the entry $\max_{x'_2} \max_{x'_3 > x'_2}\{C(R_{even}(S, x_3, v^*), R_{even}(T, x'_3, v' = (1, x'_2 - 1, x'_2, x'_3)))\}$ provides the optimal score.

The value of $C(R_x, R_y)$ can be computed recursively. Assume that t is odd. Let $R_x = R_{odd}(S, (p, q, r, s))$ and $R_y = R_{odd}(T, (e, f, g, h))$. If (p, q) is a base pair in $Struct(R_x)$, there are four cases to consider. Case 1: MATCH$_{both}$ - aligning the base pair (p, q) of S with (e, f) of T; Case 2: MATCH$_{single}$ - aligning only one of the bases in (p, q) with the corresponding base in (e, f); Case 3: INSERT - insert a space on S; Case 4: DELETE - delete the base-pair (p, q) from S. Lemma 1 summarizes these cases.

The other cases, (q, r) is a base pair or (p, s) is a base pair, are similar. Note that if more than one such base pair exists (e.g. both (q, r) and (p, s) are base pairs), we only need to follow the recursion on one of the pairs. However, you cannot pick any of them in an arbitrary manner, otherwise, when we fill the dynamic programming table, we need to fill all entries for all possible subregions of S. We will address this issue in the later part of this section.

Lemma 1. *Let $v = (p, q, r, s)$ and $v' = (e, f, g, h)$. Let t be odd. And $R_x = R_{odd}(S, v)$, $R_y = R_{odd}(T, v')$. If (p, q) is a base pair, then* $C(R_x, R_y) = \max$

$$\begin{cases} //MATCH_{both} \\ C(R_{odd}(S,(p+1,q-1,r,s)), R_{odd}(T,(e+1,f-1,g,h))) \\ \quad +\gamma(S[p],T[e]) + \gamma(S[q],T[f]) + \delta(S[p],S[q],T[e],T[f]); \\ //MATCH_{single} \\ C(R_{odd}(S,(p+1,q-1,r,s)), R_{odd}(T,(e+1,f,g,h))) + \gamma(S[p],T[e]) + \gamma(S[q],\text{‘\_’}), \\ C(R_{odd}(S,(p+1,q-1,r,s)), R_{odd}(T,(e,f-1,g,h))) + \gamma(S[p],\text{‘\_’}) + \gamma(S[q],T[f]); \\ //INSERT \\ C(R_{odd}(S,(p,q,r,s)), R_{odd}(T,(e+1,f,g,h))) + \gamma(\text{‘\_’},T[e]), \\ C(R_{odd}(S,(p,q,r,s)), R_{odd}(T,(e,f-1,g,h))) + \gamma(\text{‘\_’},T[f]), \\ C(R_{odd}(S,(p,q,r,s)), R_{odd}(T,(e,f,g+1,h))) + \gamma(\text{‘\_’},T[g]), \\ C(R_{odd}(S,(p,q,r,s)), R_{odd}(T,(e,f,g,h-1))) + \gamma(\text{‘\_’},T[h]), \\ //DELETE \\ C(R_{odd}(S,(p+1,q-1,r,s)), R_{odd}(T,(e,f,g,h))) + \gamma(S[p],\text{‘\_’}) + \gamma(S[q],\text{‘\_’}) \end{cases}$$

On the other hand, if none of these are base pairs, assume that $p+1 < x_1$ and $S[p]$ is a single base, then we can compute $C(R_x, R_y)$ recursively according to another three cases. Case 1: Match - aligning $S[p]$ with $T[e]$; Case 2: INSERT - insert a space on S; Case 3: Delete - delete $S[p]$.

Lemma 2. *Let $v = (p, q, r, s)$ and $v' = (e, f, g, h)$. Let t be odd. And $R_x = R_{odd}(S, v)$, $R_y = R_{odd}(T, v')$. If $p+1 < x_1$ and $S[p]$ is a single base, then* $C(R_x, R_y) = \max$

$$\begin{cases} C(R_{odd}(S,(p+1,q,r,s)), R_{odd}(T,(e+1,f,g,h))) + \gamma(S[p],T[e]) //MATCH \\ //INSERT\text{: same as the one defined in Lemma 1} \\ C(R_{odd}(S,(p+1,q,r,s)), R_{odd}(T,(e,f,g,h))) + \gamma(S[p],\text{‘\_’}) //DELETE \end{cases}$$

For t is even, we consider whether (p, q),(q, r), and (q, s) are base pairs in $Struct(R_x)$ and we need to consider all possible cases for x'_3 since the structure of T is unknown (i.e., the pivot points are unknown).

To fill the dynamic programming table, not all entries for all possible subranges of S needs to be filled. For any given subregion $v = (p, q, r, s)$ in S, we first define $pair_{min}(v)$ and $single_{min}(v)$ as follow. If there exists a set of base pairs, say $\{(i_1, j_1), \ldots, (i_d, j_d)\}$, such that all $i_k, j_k (1 \le k \le d)$ equals to p (if $x \le p < x_1$), q (if $x_1 \le q < x_2$), r (if $x_2 \le r < x_3$) or s (if $x_3 \le s \le y$), then $pair_{min}(v)$ is the pair with minimum value of i. Also, if there exists a set of single bases (i.e. the positions which do not belong to any base pair), say $\{u_1, \ldots, u_d\}$, such that all $u_k (1 \le k \le d)$ equals to p (if $x \le p < x_1$), q (if $x_1 \le q < x_2$), r (if $x_2 \le r < x_3$) or s (if $x_3 \le s \le y$), then $single_{min}(v)$ is the one with minimum value.

Now, we define a function $\zeta(v)$ to determine for which subregions in S, we need to fill the corresponding C entires.

Case 1. If $(i, j) = pair_{min}(v)$ exists, then

$$\zeta(v) = \begin{cases} (p+1, q-1, r, s), & \text{if } (i,j) = (p,q) \\ (p, q-1, r+1, s), & \text{if } (i,j) = (q,r) \\ (p+1, q, r, s-1), & \text{if } (i,j) = (p,s) \text{ i.e. } t \text{ is odd} \\ (p, q-1, r, s+1), & \text{if } (i,j) = (q,s) \text{ i.e. } t \text{ is even} \end{cases} \tag{2}$$

Case 2. If $pair_{min}(v)$ does not exist, then $u = single_{min}(v)$ should exist and

$$\zeta(v) = \begin{cases} (p+1, q, r, s), & \text{if } u = p \\ (p, q-1, r, s), & \text{if } u = q \\ (p, q, r+1, s), & \text{if } u = r \\ (p, q, r, s-1), & \text{if } u = s \text{ and } t \text{ is odd} \\ (p, q, r, s+1), & \text{if } u = s \text{ and } t \text{ is even} \end{cases} \tag{3}$$

It is obvious that if v defines a subregion with a valid substructure, $\zeta(v)$ also defines a valid substructure. For t is odd, let $v^* = (1, x_2 - 1, x_2, m)$. We only need to fill in the entries for C provided v can be obtained from v^* by applying ζ function repeatedly. If t is even, let $v^* = (1, x_2 - 1, x_2, x_3)$. Intuitively, ζ guides which recursion formula to use. And there are only $O(m)$ such v values. The following lemma summarizes the time complexity for this algorithm.

Lemma 3. *For any sequence $S[1..m]$ with simple non-standard pseudoknot of degree 4 and any sequence $T[1..n]$, let c be the max length of $[x_3', n]$, the optimal alignment score between $S[1..m]$ and $T[1..n]$ can be computed in $O(cmn^4)$.*

Note that the factor c is only needed when t is even due to the extra parameter x_3'. We examined all sequences in Rfam and PseudoBase, we found that usually the length of the final segment c is short (<15) and the average length is only 5.4 with most of the cases having lengths from 5 to 7. So, we can assume that c is a constant. The algorithm can be extended to simple non-standard pseudoknot of degree k easily.

Theorem 1. *For any sequence $S[1..m]$ with simple non-standard pseudoknot of degree k and any sequence $T[1...n]$, the optimal alignment score between $S[1..m]$ and $T[1..n]$ can be computed in $O(kmn^k)$.*

4. ALGORITHM FOR 2-LEVEL RECURSIVE PSEUDOKNOT

In this section, we describe the algorithm for handling a special type of recursive pseudoknots in which each recursive region is a regular structure and after excluding all recursive regions, the remaining base pairs form a simple pseudoknot or a simple non-standard pseudoknot. We refer this recursive pseudoknot as *2-level pseudoknot with regular recursive regions*. We use a recursive pseudoknot of degree 4 to illustrate the algorithm. We show the case for simple non-standard pseudoknot. The algoirthm for simple pseudoknot is simpler and the approach can be easily extended to general k.

Let $S[1..m]$ be the query sequence with recursive pseudoknot structure M. Recall the definition of a recursive pseudoknot. There can be disjoint recursive regions, namely $M_{a_1,b_1}, \ldots, M_{a_s,b_s}$, in M. By removing all these recursive regions, the remaining structure $M - (M_{a_1,b_1} \cup \ldots \cup M_{a_s,b_s})$ together with the remaining sequence $S[1..a_1 - 1]S[b_1 + 1..a_2 - 1] \ldots S[b_s + 1..m]$ are referred as level-0. For each removed recursive region M_{a_i,b_i}, we can apply the same procedure to define level-1, level-2, ..., level-ℓ structures. In our case, we only have level-1 structures (Fig. 4).

Let $T[1..n]$ be the target sequence. Define $H[a_i, b_i, x', y']$ be the score of the optimal alignment between the recursive region $S[a_i, b_i]$ with structure M_{a_i,b_i} and $T[x'..y']$, where $1 \le x' < y' \le n$. We now show how to compute the score of the optimal alignment between S and T recursively for Type I simple non-standard pseudoknot (i.e., t is odd). The case for even value of t is similar. Let $v = (p, q, r, s)$ be a quadruple that

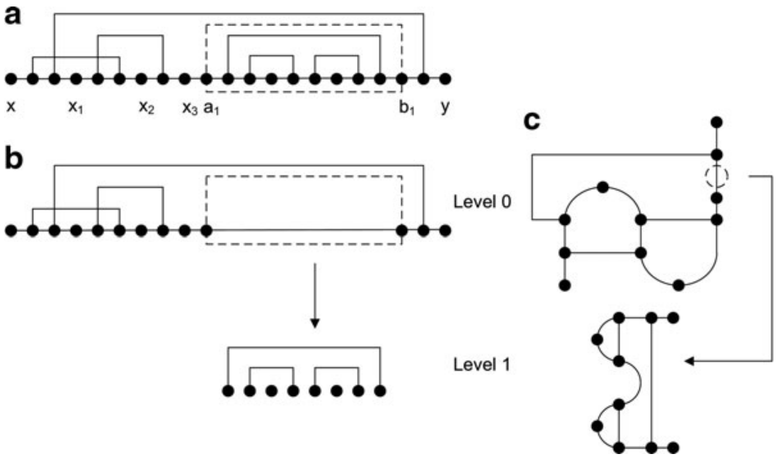


FIG. 4. (**a**) An example of a level-2 simple non-standard pseudoknot with regular recursive regions. (**b**) Level-0 and level-1 structures. (**c**) Another view for the same example.

defines a substructure of S. Let $S[p..y_p]$ be a recursive region. The following lemma shows how to compute $C(R_x, R_y)$, the score of the optimal alignment between R_x and R_y where $R_x = R_{odd}(S, v)$ and $R_y = R_{odd}(T, v')$.

Lemma 4. *Let $v = (p, q, r, s)$ and $v' = (e, f, g, h)$. Assume that t is odd. $R_x = R_{odd}(S, v)$ and $R_y = R_{odd}(T, v')$. If $S[p..y_p]$ is a recursive region, then*

$$C(R_x, R_y) = \max \begin{cases} //MATCH \\ \max_{e \le w \le f}\{C(R_{odd}(S, (y_p + 1, q, r, s)), R_{odd}(T, (w + 1, f, g, h))) + H(p, y_p, e, w)\} \\ //INSERT \\ \textit{same as INSERT defined in Lemma 1} \\ //DELETE \\ C(R_{odd}(S, (y_p + 1, q, r, s)), R_{odd}(T, (e, f, g, h))) + \sum_{p \le w \le y_p} \gamma(S[w], \text{'\_'}) \end{cases}$$

Other cases, where $S[x_q..q]$ or $S[r..y_r]$ or $S[x_s..s]$ is a recursive region, can be handled in a similar way. Again, we do not need to compute C for all possible values of (p, q, r, s). We need to determine for which subregions in S, we need to fill in the corresponding C entries. So, we enhance ζ function as follows.

Consider a quadruple $v = (p, q, r, s)$ in a region $S[x..y]$ where the structure is a simple non-standard pseudoknot of degree 4 if all the next-level subregions inside are excluded. Let us define subregion$_{min}(v)$ as follows: if there exists a set of next-level subregions, say $\{[i_1, j_1], \ldots, [i_d, j_d]\}$ where $x \le i_k < j_k \le y$ for all $1 \le k \le d$ such that either i_k or j_k equals to p (if $x \le p < x_1$), q (if $x_1 \le q < x_2$), r (if $x_2 \le r < x_3$) or s (if $x_3 \le s \le y$), then let subregion$_{min}(v)$ be the region with minimum value of i. We add the following case to ζ function. Note that the t value refers to the structure for $S[x..y]$ excluding all next-level subregions.

Case 0 of ζ(v): If $[i, j] =$ subregion$_{min}(v)$ exists, then

$$\zeta(v) = \begin{cases} (j + 1, q, r, s), & \text{if } i = p \\ (p, i - 1, r, s), & \text{if } j = q \\ (p, q, j + 1, s), & \text{if } i = r \\ (p, q, r, i - 1), & \text{if } j = s \text{ //i.e. } t \text{ is odd} \\ (p, q, r, j + 1), & \text{if } i = s \text{ //i.e. } t \text{ is even} \end{cases}$$

There are at most $O(m)$ v values we need to consider. So, assuming all $H()$ values have been computed, it takes $O(mn^5)$ time to fill all C entries. For $H()$, since the recursive region is a regular structure, we can make use of the algorithm in (Zhang et al., 2005) algorithm to compute all $H()$ values for all possible subregions of T in $O(mn^3)$ time. The following theorem summarizes the result of this section. The algorithm presented in this section can be extended to general recursive pseudoknots with more than 2 levels and with recursive regions having other structures as defined in Definition 4 with an increase of $O(n)$ factor in the time complexity.

Theorem 2. *To compute the optimal alignment score between a query sequence $S[1..m]$ with a 2-level pseudoknot of degree $k \ge 4$) with regular recursive regions and a target sequence $T[1..n]$, it can be done in $O(kmn^{k+1})$ time.*

5. EXPERIMENTAL RESULTS

We implemented both algorithms for simple non-standard pseudoknot and the 2-level pseudoknot with regular recursive regions in C++. By inputting a query ncRNA sequence (Q) and its secondary structure, the program can scan a long DNA sequence (T) and output the score for every region in T. Higher score indicates that the sequence and the structure of the region is more similar to those of Q. To evaluate the effectiveness of our algorithm, we selected a set of families in Rfam 9.1 database for which the structures of the ncRNAs in these families are either simple non-standard pseudoknot or recursive pseudoknot. For each family, we selected one of the seed members (in Rfam 9.1 database, for each family, there is a set of reliable members which are regarded as seed members) as the query sequence Q. To demonstrate the power of structural alignment, the query sequence selected has the lowest *sequence* similarity with the other seed members. The details of the families including the sequence selected as the query, the length of this sequence, and the number of members in each family are given in Table 1.

We constucted a long random genome sequence of length about 300 times the length of the query sequence. Then, we embed all the ncRNA sequences (seed members or non-seed members) of the family, except the query sequence, into this long random sequence in arbitrary positions. The resulting sequence is our T. For

Table 1. Details of the ncRNA Families Used in the Experiments

Family	*Pseudoknot Type*	*Query Sequence ID*	*Length of Query Sequence*	*Number of members*
RF00140	Degree-4 simple non-standard	AM286690/459188-459298	111	164
RF00094	Degree-4 recursive (non-standard)	AB037947/685-775	91	432
RF00622	Degree-4 recursive (non-standard)	AAGV01475186/596-519	78	47
RF01084	Degree-3 recursive (simple)	AF325738/2039-2167	129	263
RF01075	Degree-3 recursive (simple)	AY787207/2721-2816	96	23
RF01085	Degree-3 recursive (simple)	K01776/83-200	118	8
RF00176	Degree-3 recursive (simple)	D00719/409-499	91	80

The first family is of simple non-standard pseudoknot. All the other families are 2-level pseudoknot with regular recursive regions. Among these six families, the first two have a simple non-standard pseudoknot in level-0 while the rest have a simple pseudokont in level-0.

every region in *T* with length similar to that of the query sequence,[1] we compute the structural alignment score of the region and the query sequence. We use the same scoring scheme as in Klein and Eddy (2003).

We assume that regions other than the real members of the family are false hits as they are likely not to be members of the family. Figure 5 shows the distribution of the alignment scores of the true hits (real members) and false hits. It is quite clear that based on the structural alignment scores, the real members can be easily distinguished from the false hits except for the family RF00176. We have investigated why the approach does not work well for RF00176. We found that the length of the query sequence is much longer than those of the other member sequences. Since our current tool is designed for global alignment, the big differences in length lead to a big penalty score in our method and thus the resulting structural alignment scores becomes very low. To verify this observation, we identified a conserved region inside the multiple sequence alignment of the family. Then we only use the corresponding conserved region (the length is 37 while the total length of the sequence is 91) of the selected query sequence as the new query sequence. The result has been improved substantially (Fig. 6). From this observation, we believe that developing a tool for local structural alignment is desirable and would be our next step.

Since there is no existing software which available freely for performing structural alignment for complex pseudoknot structures, we follow the evaluation method of Han et al. (2008) to compare the performance of our method with BLAST. That is, we want to compare the effectiveness of considering only sequence similarity (based on BLAST) and our method which also consider the structural similarity. We use default parameters for BLAST except that the wordsize is set to 7 to increase its sensitivity. For each family, we use the same query sequence and the random sequence *T* as in the above experiment. Again, for each region in *T*, we compute a BLAST score between the region and the query sequence. To compute the effectiveness of our method and BLAST, we set the threshold as the maximum score that can be achieved by the false hits and a hit is considered to be true positive if the score of the region is larger than this threshold. By setting the threshold this way, none of the false hits will be considered to be answer, but a real hit will be missed if the computed score is smaller than or equal to this threshold. In other words, we will compare how many real hits will be missed by each method. We also try different thresholds and the results are similar.

Table 2 shows the comparison result. Note that we omit the family RF00176 in this comparison. For most of the families, our algorithm does not miss as many as BLAST does. For example, in Family RF01084, our algorithm misses only 22 sequences, while BLAST misses 202 sequences. It is clear that our method is more effective than BLAST demonstrating that structural similarity is important for aligning ncRNA sequences.

Figure 7 shows the detailed scores for our method and BLAST for family RF00094. For ease of illustration, we only show the scores for the seed members. Among 32 seed members (except the one selected as query sequence), BLAST missed 13 of them. However, all the regions of these 32 members got the highest scores if using our algorithm and thus none of them is missed. To take a closer look at the missed cases for BLAST, we found that the missed sequence is usually not similar

[1] We set the length of each region equal to the length of the query plus 20.

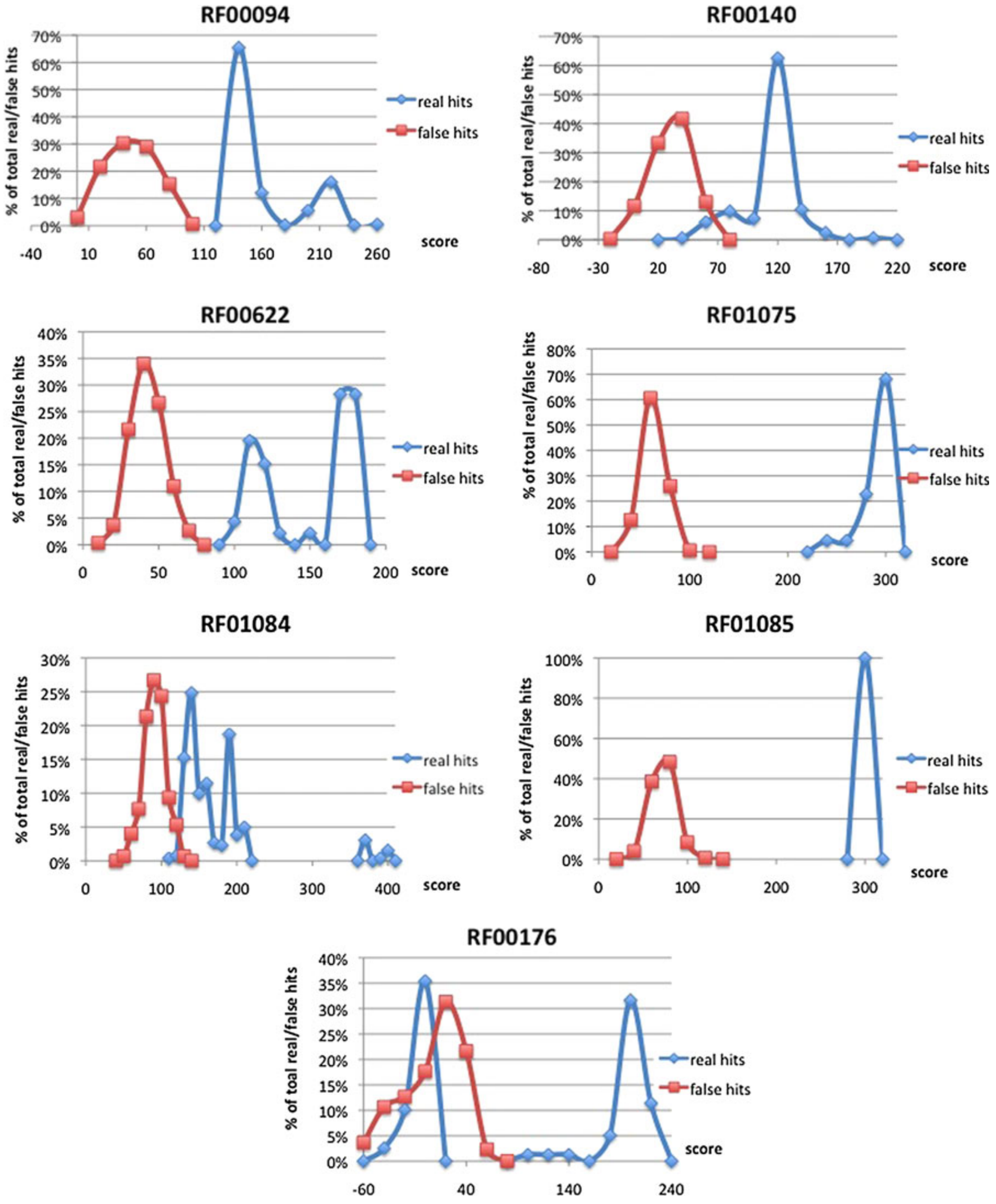


FIG. 5. The distribution of alignment scores of true hits and false hits.

to the query sequence in terms of *sequence* similarity while the corresponding secondary structure is similar to that of the query sequence. Figure 8 shows some examples of these cases. The top one circled by red line is the query sequence, and the others are those missed by BLAST but can still be identified by our algorithm. We can see that although the sequence similarity between the query sequence and the other sequences may not be very high, all of their secondary structures are highly conserved.

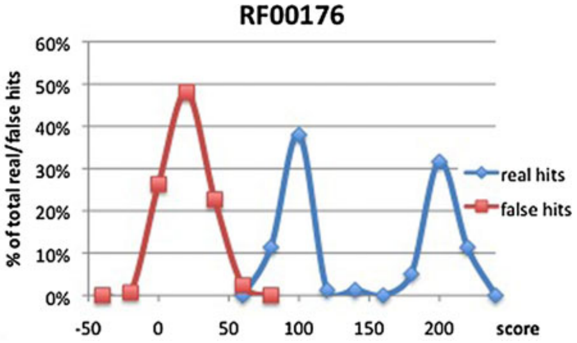


FIG. 6. The distribution of alignment scores of true hits and false hits for the family RF00176 after considering the conserved region.

Table 2. Summary of Comparison on Results between BLAST and Our Method

Family	*No. of real hits*	*BLAST misses, no.*	*Our method misses, no.*
RF00094	431	334	0
RF00622	46	15	0
RF00140	163	8	9
RF01084	262	202	22
RF01075	22	0	0
RF01085	7	0	0

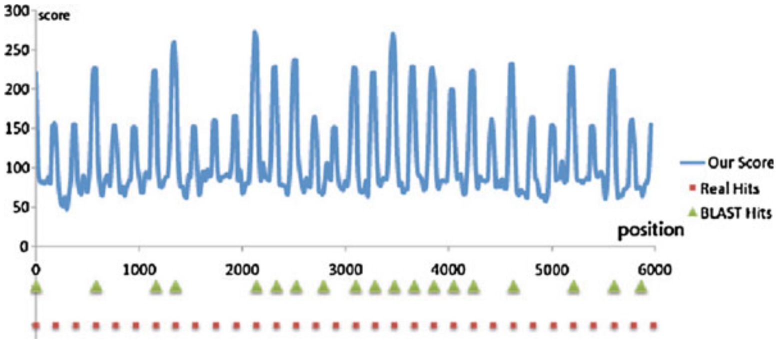
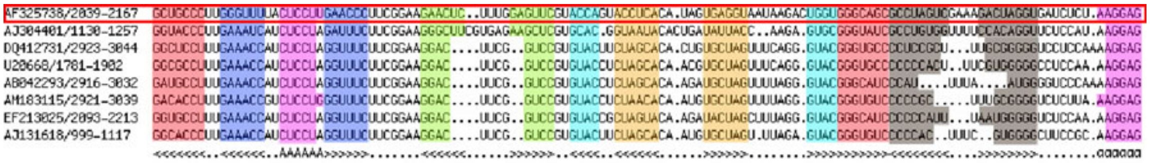


FIG. 7. Comparison of resulting scores from our program and BLAST of family RF00094. The squares represent the positions of real hits, the triangles represent the positions of BLAST hits, and the line represents our scores along different positions. Among 32 real hits, BLAST missed 13 of them. However, they got the highest scores by our method, and so our method did not miss any of them.



```
AF325738/2039-2167  GCUGCCCUUGGGUUUUACUCCUUGAACCCUUCGGAAGAACUC..UUUGGAGUUCGUACCAGUACCUCACA.UAGUGAGGUAAUAAGACUGGUGGGCAGCGCCUAGUCGAAAGACUAGGUGAUCUCU.AAGGAG
AJ304401/1130-1257  GGUACCCUUGAAAUCAUCUCCUAGAUUUCUUCGGAAGGGCUUCGUGAGAAGCUCGUGCAC.GGUAAUACACUGAUAUUACC..AAGA.GUGCGGGUAUCGCCUGUGGUUUUCCACAGGUUCUCCAU.AAGGAG
DQ412731/2923-3044  GGCUCCCUUGAAACCAUCUCCUAGGUUUCUUCGGAAGGAC....UUCG..GUCCGUGUACUUCUAGCACA.CUGUGCUAGUUUCAGG.GUACGGGUGCCCCUCCGCU...UUGCGGGGGUCCUCCAAAAGGAG
U20668/1781-1902    GGCGCCCUUGAAACCAUCUCCUAGGUUUCUUCGGAAGGAC....UUCG..GUCCGUGUACCUCUAGCACA.ACGUGCUAGUUUCAGG.GUACGGGUGCCCCCCACU..UUCGUGGGGGCCUCCAA.AAGGAG
AB042293/2916-3032  GAUGCCCUUGAAACCAUCUCCUAGGUUUCUUCGGAAGGAC....UUCG..GUCCGUGUACCUCUAGCACA.AGAUGCUAGUUUUAGG.GUACGGGCAUCCCAU....UUUA....AUGGGGUCCCAAAAGGAG
AM183115/2921-3039  GACACCCUUGAAACCGUCUCCUGGGUUUCUUCGGAAGGAC....UUCG..GUCCGUGUACCUCUAACACA.AUGUGCUAGUUUUAGG.GUACGGGUGUCCCCCGC.....UUUGCGGGGUCUCUUA.AAGGAG
EF213025/2093-2213  GGUGCCCUUGAAACCAUCUCCUAGGUUUCUUCGGAAGGAC....UUCG..GUCCGUGUACCGCUAGUACA.AGAUACUAGCUUUAGG.GUACGGGCAUCCCCCCAUU...UAAUGGGGGUCUCCAA.AAGGAG
AJ131618/999-1117   GGCACCCUUGAAACCAUCUCCUAGGUUUCUUCGGAAGGAC....UUCG..GUCCGUGUACUUCUAGCACA.AUGUGCUAGU.UUAGA.GUACGGGUGUCCCCCAC..UUUC..GUGGGGCUUCCGC.AAGGAG
                    <<<<<<<..<<<<<<..AAAAAA>>>>>>......<<<<<<.....>>>>>>..<<<<..<<<<<<......>>>>>>.........>>>>>>>>>>>><<<<<<<<....>>>>>>>>.......aaaaaa
```

FIG. 8. A multiple sequence alignment of some members in the family RF01084 with recursive pseudoknot of degree 3. By using the selected query sequence (which is circled in red), BLAST cannot locate the other member sequences. However, since both the structure of the query sequence and that of the member sequences are highly conserved, our method can locate all of them.

6. CONCLUSION

In the article, we provided the first set of algorithms to handle structural alignment of RNA with complex pseudoknot structures: recursive pseudoknots and simple non-standard pseudoknots. Further directions include speeding up these algorithms, developing a local structural alignment algorithm, and considering other more complicated pseudoknot structures.

DISCLOSURE STATEMENT

No competing financial interests exist.

REFERENCES

Adams, P.L., Stahley, M.R., Kosek, A.B., et al. 2004. Crystal structure of a self-splicing group I intron with both exons. *Nature* 430, 45–50.

Chen, J.-L., and Greider, C.W. 2005. Functional analysis of the pseudoknot structure in human telomerase RNA. *Proc. Natl. Acad. Sci. USA* 102, 8080–8085.

Dam, E., Pleij, K., and Draper, D. 1992. Structural and functional aspects of RNA pseudoknots. *Biochemistry* 31, 11665–11676.

Eddy, S.R. 2001. Non-coding RNA genes and the modern RNA world. *Nat. Rev. Genet.* 2, 919–929.

Ferré-D'Amaré, A.R., Zhou, K., and Doudna, J.A. 1998. Crystal structure of a hepatitis delta virus ribozyme. *Nature* 395, 567–574.

Frank, D.N., and Pace, N.R. 1998. Ribonuclease P: unity and diversity in a tRNA processing ribozyme. *Annu. Rev. Biochem.* 67, 153–180.

Griffiths-Jones, S., Bateman, A., Marshall, M., et al. 2003. Rfam: an RNA family database. *Nucleic Acids Res.* 31, 439–441.

Han, B., Dost, B., Bafna, V., et al. 2008. Structural alignment of pseudoknotted RNA. *J. Comput. Biol.* 15, 489–504.

He, S., Liu, C., Skogerbo, G., et al. 2007. Noncode v2.0: decoding the non-coding. *Nucleic Acids Res.* 36, D170–D172.

Klein, R.J., and Eddy, S.R. 2003. Rsearch: finding homologs of single structured RNA sequences. *BMC Bioinform.* 4, 44.

Le, S.Y., Chen, J.H., and Maizel, J.V. 1990. *Efficient Searches for Unusual Folding Regions in RNA Sequences.* Volume 1. Adenine Press, New York.

Nguyen, V.T., Kiss, T., Michels, A.A., et al. 2001. 7sk small nuclear RNA binds to and inhibits the activity of cdk9/cyclin t complexes. *Nature* 414, 322–325.

Rivas, E., and Eddy, S.R. 2000. Secondary structure alone is generally not statistically significant for the detection of noncoding RNAs. *Bioinformatics* 16, 583–605.

van Batenburg, F.H., Gultyaev, A.P., Pleij, C.W., et al. 2000. Pseudobase: a database with RNA pseudoknots. *Nucleic Acids Res.* 28, 201–204.

Yang, Z., Zhu, Q., Luo, K., et al. 2001. The 7sk small nuclear RNA inhibits the cdk9/cyclin t1 kinase to control transcription. *Nature* 414, 317–322.

Zhang, S., Haas, B., Eskin, E., et al. 2005. Searching genomes for noncoding RNA using fastr. *IEEE/ACM Trans. Comput. Biol. Bioinform.* 2, 366–379.

Address correspondence to:
Dr. S.M. Yiu
Department of Computer Science
University of Hong Kong
Hong Kong

E-mail: smyiu@cs.hku.hk