IranianJournal of Bi technology

PreRkTAG: Prediction of RNA Knotted Structures Using Tree Adjoining Grammars

Hesamedin Torabi Dashti [1], Fatemeh Zare-Mirakabad [2], Nima Aghaeepour [3], Hayedeh Ahrabian [4], Abbas Nowzari-Dalini [4,*]

[1] Department of Mathematics, University of Wisconsin, Madison, USA
[2] Department of Computer Science, Faculty of Mathematics and Computer Science, Amirkabir University of Technology, Tehran, IR Iran
[3] Department of Computer Science, University of British Columbia, Vancouver, Canada
[4] Department of Computer Science , School of Mathematics, Statistics, and Computer Science, University of Tehran, Tehran, IR Iran

ARTICLE INFO

Article type:
Research Article

Article history:
Received:09 Jun 2011
Revised: 11 Aug 2011
Accepted: 15 Oct 2012

Keywords:
Genetic Algorithms
RNA Secondary Structure
Pseudoknot
Tree Adjoining Grammars

ABSTRACT

Background: RNA molecules play many important regulatory, catalytic and structural roles in the cell, and RNA secondary structure prediction with pseudoknots is one the most important problems in biology. An RNA pseudoknot is an element of the RNA secondary structure in which bases of a single-stranded loop pair with complementary bases outside the loop. Modeling these nested structures (pseudoknots) causes numerous computational difficulties and so it has been generally neglected in RNA structure prediction algorithms.
Objectives: In this study, we present a new heuristic algorithm for the Prediction of RNA Knotted structures using Tree Adjoining Grammars (named PreRKTAG).
Materials and Methods: For a given RNA sequence, PreRKTAG uses a genetic algorithm on tree adjoining grammars to propose a structure with minimum thermodynamic energy. The genetic algorithm employs a subclass of tree adjoining grammars as individuals by which the secondary structure of RNAs are modeled. Upon the tree adjoining grammars, new crossover and mutation operations were designed.The fitness function is defined according to the RNA thermodynamic energy function, which causes the algorithm convergence to be a stable structure.
Results: The applicability of our algorithm is demonstrated by comparing its iresults with three well-known RNA secondary structure prediction algorithms that support crossed structures.
Conclusions: We performed our comparison on a set of RNA sequences from the RNAseP database, where the outcomes show efficiency and practicality of the proposed algorithm.

Implication for health policy/practice/research/medical education:
This article is recommended for those who researches in RNA area.

Please cite this paper as:
Torabi Dashti H, Zare-Mirakabad F, Aghaeepour N, Ahrabian H, Nowzari-Dalini A. PreRkTAG: Prediction of RNA Knotted Structures Using Tree Adjoining Grammars. *Iran J Biotech*. 2013: **11**(1): 3-13. DOI: 10.5812/ijb.9213

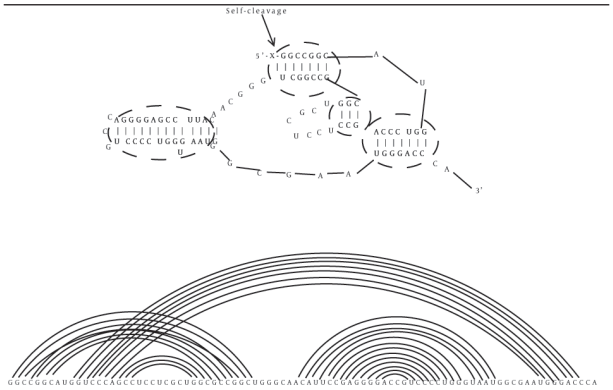
* *Corresponding author:* Abbas Nowzari-Dalini, Department of Computer Science , School of Mathematics, Statistics, and Computer Science, University of Tehran, Tehran, IR Iran.Tel: + 98-2166412478, Fax: + 98-2166412478, *E-mail:* nowzari@ut.ac.ir

DOI: 10.5812/ijb.9213

1. Background

RNA is an active component in various cellular processes. They act as carriers of genetic information, catalysts in cellular processes and mediators in determining the expression level of genes. Whereas the folding of RNA or its secondary structure is very important for functionality of RNA, thus it is of great interest to propose computational techniques for predicting RNA secondary structural features. Several algorithms are currently designed for the prediction of RNA secondary structure (1-8). The most successful approach for RNA structure prediction is comparative sequence analysis in which covering residues are identical to a multiple sequence alignment of different RNAs sequences with similar structures (9, 10). The accepted secondary structures of most structural and catalytic RNAs were generated by comparative sequence analysis. Comparative methods (7, 11-13) use a given set of RNA sequences and their corresponding structures to predict a structure for a new RNA sequence. In the comparative methods, the sequences in the given set must be in the same family with the new RNA, structurally or as a nucleotides arrangement. Hence, these methods fail to predict a structure for a single RNA sequence or a small family of RNAs with little sequence diversity. For these cases, prediction of RNA structure based on free energy minimization is the most widely used technique (5, 6, 14-17). One of the well-known applied approaches in these methods is the dynamic programming. Zuker introduced the dynamic programming approach for predicting RNA structures (17). Furthermore, Walter et al. (18) improved the algorithm by using thermodynamics rules and Zuker (8) presented a web-based predictor application. Nevertheless, due to undiscovered biochemical reasons, results of the comparative methods are more similar to RNA natural structure than results of latter methods. However, one of the difficulties on the problem of RNA structure prediction is the existence of a certain type of RNA secondary structures called pseudoknots. This type of structure is composed of both nested and crossed base pairs. *Figure 1* represents the structure of Hepatitis Delta Virus (HDV) in two graphical shapes. Rivas and Eddy (6) introduced an approach to predict pseudoknot structures by using a subset of context sensitive grammar. This approach is also considered by Uemura et al. (16), Lyngsø and Pedersen (3), and Cai et al. (19). In addition, heuristic approaches were also applied to capture nested structures. Shapiro and Navetta (20) introduced a parallel genetic algorithm to predict nested structures that was one of the famous predictors. Thereafter van Batenburg et al. (1) demonstrated that GA is a promising method in RNA structure prediction. Batenburg's GA method is part of Structure Analysis of RNA (STAR) package. Ren et al. (21) presented a heuristic algorithm called Hotknots for prediction of RNA structure including pseudoknots. They have improved the quality of the former prediction algorithms has become and their algorithm often performs better than the genetic algorithm from the STAR package and PKnotsRG-mfe algorithm by Reeder and Giegerich (15). Uemura et al. (16), Akutsu (14), and Kato et al. (22) demonstrated, Tree Adjoining Grammars (TAG) are suitable data structure for a capturing an RNA secondary structure.

Figure 1. Two Representations for RNA Secondary Structure



The upper graph represents planar structure and the bottom graph shows crossed representation of the structure (21).

A TAG is a kind of formal rewriting system to generate a set of trees associated to strings (23). More specially, the structured information of the derived tree in TAGs is of great significance in both syntactical and semantical concepts. Uemura et al. (16) and Kato et al. (22) illustrated wide ranges of crossed structures which could be captured by a subclass of TAG, called Extended Simple Linear Tree Adjoining Grammars (ESL-TAG), where their corresponding parsing algorithm uses $O(n^5)$ time and $O(n^3)$ space. There may be numerous adjunct trees in the grammar which represent secondary structures of an RNA. This means that for an RNA sequence, many structures might be possibly found. Thus, the problem is like as a search problem for an optimal structure with respect to thermodynamics energy parameters.

2. Objectives

In this manuscript, we present a new genetic algorithm, PreRkTAG, to predict RNA secondary structures including pseudoknots that is developed based on a sub-class of the adjoining grammars. PreRkTAG uses a practical data structure for capturing RNA crossed structures. Our designed genetic algorithm can search a broad area of feasible solutions of the given RNA structure. This genetic algorithm is composed of two evolutionary functions which allow free pairing of nucleotides in structures. This increases the probability of discovering new structures that cannot be represented by the regular ESL-TAG. This ability yields to more coverage area of the feasible solutions rather than ESL-TAG covers. On the other hand, a common problem of heuristic algorithms

is converging to local minimums, which is highly possible especially with a wide range of feasible solutions. In PreRkTAG we attempt to tackle this problem by using novel evolutionary functions. The evaluation of PreRkTAG against celebrated energy-based prediction algorithms, namely, the STAR, Hotknots, and Pknots, shows high sensitivity and high specificity of PreRkTAG in the most cases. The evaluation was performed on 24 RNA sequences chosen from the RNAseP database. Our algorithm correctly identifies more than 80% of base pairs.

3. Materials and Methods

The employed data structure for capturing RNA crossed structures was introduced in preceding research (14) and in order to synchronize the notifications the first part of this section is focused on ESL-TAG. Consequently, the details of PreRkTAG algorithm are described including the employed inner functions.

3.1. RNA Secondary Structure Modeling

Generally, an ESL-TAG is a 5-tuple grammar $G = (N, T, S, I, A)$ where N and T are finite sets of non-terminal and terminal symbols, respectively. S is the start symbol and I is finite set of initial trees (center trees). Finally, the finite set A is a set of adjunct trees (auxiliary trees). The members of the set N are divided into two subsets; *Active* and *Inactive*. During transition process of the grammar G, the active nodes can be replaced by a member of the set A, where inactive nodes do not. Based on the division, when a tree does not have any active node, it would be a *matured tree*. By adjoining two trees, s and t, a tree t' would be obtained, and $t \rightarrow_s t'$ represents the adjuncted transition. The transitive closure of $\rightarrow$ is $\rightarrow^*$ and a tree t' is derived from t, where $t \rightarrow^* t'$ for some $t \in (A \cup I)$. A set of trees of an ESL-TAG G is defined as $T(G) = \{t \mid s \rightarrow^* t, s \in$ I where t is matured tree$\}$. For modeling the secondary structure of an RNA with pseudoknot, the tree adjoining grammar $G = (N, T, S, I, A)$ is defined as follows. The set N is a finite set of non-terminals {X, Y, Z}, where active non-terminals are marked with asterisk {X*, Y*, Z*}. The set T is finite set of terminals {a, u, c, g} as abbreviation of nucleotide acids. Moreover, S is a start symbol and the set of initial trees I is composed of just one tree as depicted in *Figure 2* (this tree is tree of type 1 and denoted by center tree). The last parameter A, is a finite set of adjunct trees, such as given in *Figure 2*. This set is composed of four types. As depicted in the *Figure 2* , type 2 and type 3 represent base pairs. Moreover via preorder traversal of trees of these types, the type 2 represents crossed base pairs denoted by T2u and T2d, and the type 3 represents nested base pairs denoted by T3L and T3R. In addition, type 4 is introduced to generate a single base without any pairing. Trees of type 5 are used to combine substructures by using trees of type T5Ld and T5Rd, and inserting a tree into the current one by using trees of type T5Lu and T5Ru. As mentioned, the key idea of this modeling is from Uemura et al. (16). *Figure 3* depicts the process of constructing the crossed base pairs "agacuu" which is not generated by any context free grammar. Uemura et al. (16) described a computational obstacle of ESL-TAG trees via an example that is shown in *Figure 4* . In this example, it is shown that two dissimilar sequences can be captured by one ESL-TAG structure.

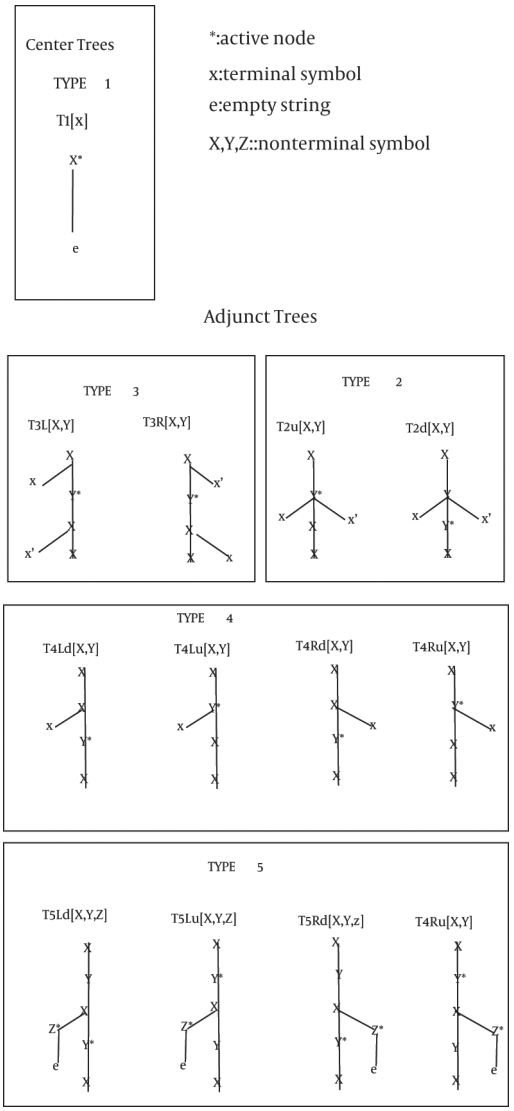


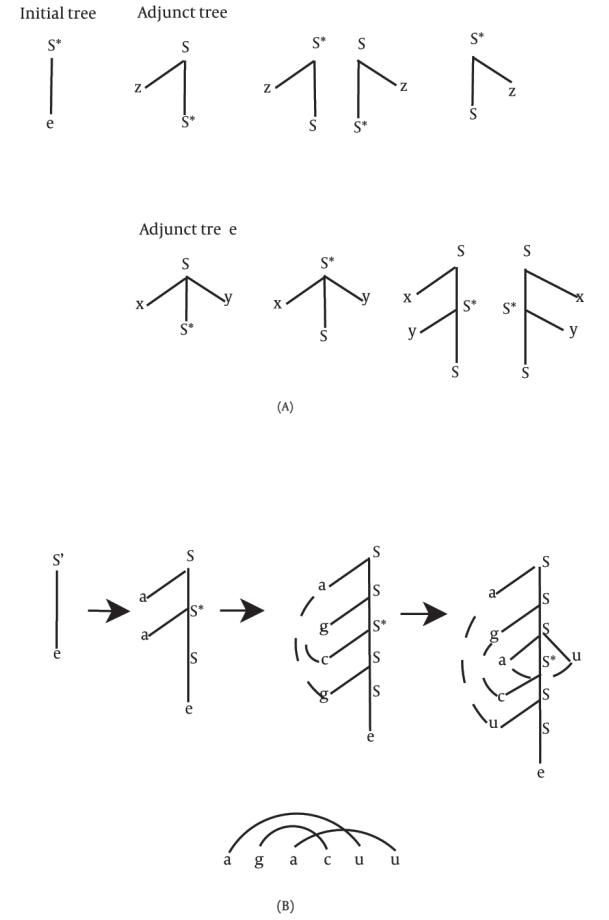
Figure 2. Five Types of Adjunct Trees for Modeling the Secondary Structure of RNA by ESL-TAG (16)

3.2. Genetic Algorithm

Genetic Algorithm (GA) is kind of Evolutionary Algorithms (EA) which via broad wide searching inside of probable solution space, analyzes a population of solutions instead of a single candidate. Goldberg (24) proved the GA convergence theory. Thereafter applicability of

the algorithm was extended in wide range of optimization problems.

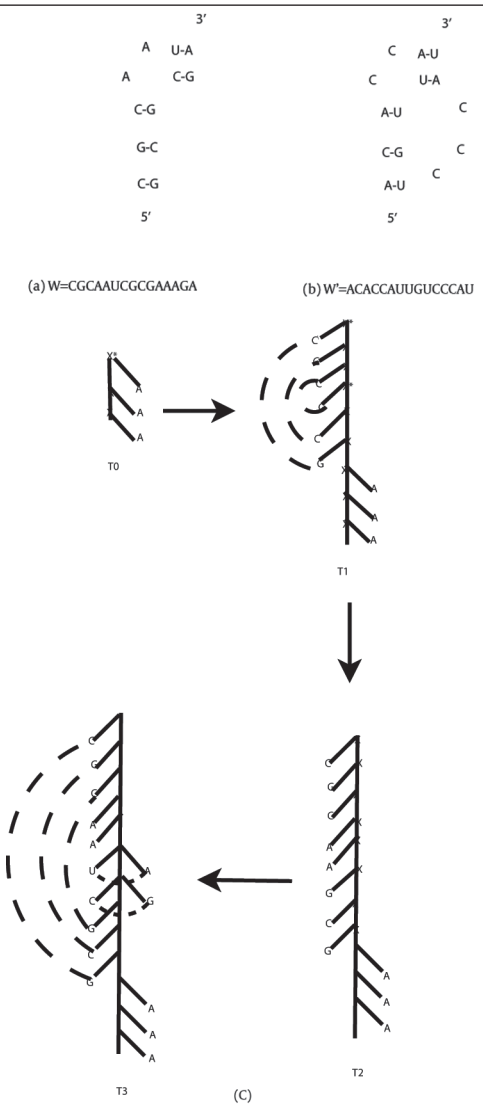
Figure 3. The Process of Constructing the Crossed Base Pairs "Agacuu"



a) Initialization of a sample ESL-TAG grammar; b) Capturing a sequence "agacuu" using ESL-TAG trees in part (a) (22).

The principle of GA is competition of the solutions where a scoring function (fitness function) measures their optimality. GA methodology is composed of two evolutionary processes; the first one is applied on to a solution and by randomly modifying the solution's properties creates a new solution (Mutation). Another one works on two different randomly selected solutions and generates two new solutions by combining predetermined features of the randomly selected solutions (Crossover). These two evolutionary functions are applied during the progress of the algorithm, where both these functions are derived from biological concepts (24). In this optimization method, the evolutionary functions are supposed to inhibit solutions to converge to local minimums. Therefore, designing efficient and practical algorithms for these functions play significant role as they converge to desirable results. Since these functions apply on the domain of feasible solutions, consideration of the data structure which represents the solutions is a critical step in designing the evolutionary functions. Novel evolutionary functions are designed according to tree adjoining grammars and equip the PreRkTAG algorithm towards global convergence. In advance to illustrating the evolutionary functions, a short description of feasible population and data structure is presented as follows.

Figure 4. Derivation RNA Structures Using ESL-TAG



a) An RNA sequence and its secondary structures. b) ESL-TAG representation of RNA structure given in (a). c) Another sequence with the same ESL-TAG representation (16)

3.3. Generating Population

Population of a GA is a set of feasible solutions and a solution is feasible if the length of its corresponding structure is the same as the length of the given sequence. Individuals of the population in PreRkTAG are generated

randomly by using ESL-TAG. As described in the previous section, an ESA-TAG starts with an initial state and could be extended easily to cover as many nodes as required. Hence, using ESL-TAG we are able to generate a tree with any given numbers of residues. To construct each element of population, we randomly select a transitive closure operator and apply it on initial trees. In this way a matured tree can be constructed in length of a given RNA sequence. Hence, a tree shows a probable structure for RNA sequences with the same length. Generating the population only depends on the length of the given RNA sequence and the rest of the algorithm adjusts these structures to nucleotide arrangement of the given RNA sequence. It is noted that outcome of the algorithm is an optimal structure for the given RNA sequence, which is a feasible structure for all RNA sequences with the same length but is not the optimal structure for them. Generating the population in this way leads to a wide range of feasible solutions where the fitness function will narrow them down to the optimal solution of the given RNA sequence. *Figure 5* shows an example of a constructed structure for a sequence of length 5.

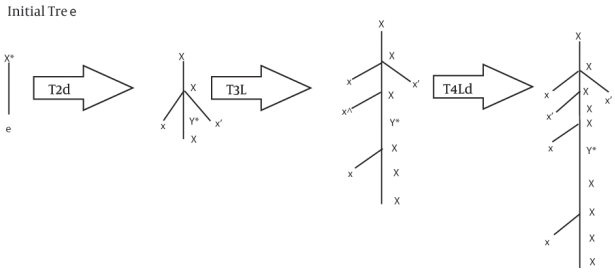


Figure 5. The Generated Sample Tree for a Sequence of Length 5

3.4. Evolutionary Functions

We present a detailed description of the evolutionary functions. These functions perform blindly in terms of considering the context of the given RNA. The mutation function takes a structure from the population and based on its length calculates the number of terminal nodes that might be mutated. Subsequently the mutated terminal nodes are selected by uniform distribution. Terminal nodes inside of adjunct trees have three different types, thus this function acts in three different ways. These treatments are discussed as follows:

1) If the selected node is a single node, then the mutation function selects another single node uniformly at random and pairs them up. The novel structure has a new pair which its ESL-TAG transitive closure is unknown. Hence, the new structure must be parsed by ESL-TAG parser to obtain the corresponding transitive closure.

2) If the selected node is a member of nested pair, then the tree reconstruction is not needed and just their adjunct tree is replaced with two adjunct trees with single terminals.

3) If the selected node is a member of crossed pair, then breaking up this pair is equal to replacing the corresponding adjunct tree by two adjunct trees; each of them has a terminal node in opposite side of the tree.

Algorithm 1 shows the mutation function. This algorithm includes two sub-functions; the "Random (L)" returns a random value L' ($0 < L' < L$), and the "Length (tree)" returns the number of terminal nodes of the given tree. In addition, as illustrated (*Algorithm 1*), in some cases new obtained structures must be parsed again to reconstruct the ESL-TAG tree, so in this algorithm the "Reconstruct" function is employed to find ESL-TAG tree of new structures. The next evolutionary function is Crossover that crosses two selected solutions and merges complimentary parts. The crossover function selects two solutions via normal distribution from the population. Thereafter this function extracts two sub-trees with the same length (number of terminal nodes) and random starting points from each of the selected trees and the crossover function swaps the sub-trees. Removing a subtree from a tree might come with breaking pairs whenever a pair contains a node inside and another node outside of the sub-tree (this pair is called an exteriorly pair). These nodes will remain single when the swapping process is done. Constructing the trees blindly, with respect to context of the given RNA sequence, makes the crossover function much easier and helps to avoid local minimums. The crossover algorithm is presented in *Algorithm 2*. Two introduced inner functions in the Mutation function (*Algorithm 1*) are used in this algorithm as well. In addition, two other functions are applied; "Move Down (node, length)" is a function for traversing the given tree from its node to attain length number of terminal nodes, and "Swap (Sub Tree0, Sub Tree1)" is a function for swapping sub-trees which results in two new trees. *Figure 6* demonstrates a sample of the crossover process.

3.5. Fitness Function

To measure the optimality of a predicted RNA structure, as it has been recently proposed by most methods, thermodynamics rules (17, 18, 25) are applied as the fitness function. In this case, the fitness function estimates the discrepancy of simulated structures by measuring energy of structures, where better structure has lower energy. Upon thermodynamics rules, constructing a base pair needs a specific amount of energy, where in some cases the base pair releases energy (negative value) and in other cases it gains energy (positive energy). In the thermodynamics rules, many parameters effect on the amount of energy and in a simplified model, this amount depends on nucleotide acids that compose a pair and also their neighbor base pairs. For example when '*AA*' in *5'-3'* and '*UU*' in *3'-5'*, the orientation $\binom{AA}{UU}$

makes two consecutive pairs, and the energy of corresponding structure will be reduced by -0.9. One example of assigning energy to two consecutive pairs is borrowed from (5) that is shown in *Table 1*. In this table, every row and every column shows a pair and the entry *(i, j)* shows amount of energy of appearing *j*'th pair after *i*'th pair. Since these are thermodynamics energies and also were assessed experimentally, there is no need to have a symmetric table. We also used the same energy values in our fitness function. For a given sequence *S*, with no base pair, the energy value is equal to zero. Assume *e (i, j)* denotes the energy value of base pair between *i* th and *j* th nucleotides, thus $\sum_{i,j\in S} e(i,j)$ shows the energy value of sequence S. The energy of RNA sequences obtained by the thermodynamic rules implies that number of single bases (without pair), coaxial, and stacked base pairs have impressive effects on the energy value of the corresponding sequence (17, 18). Our *fitness function* uses the current version of Turner group thermodynamic rules (5).

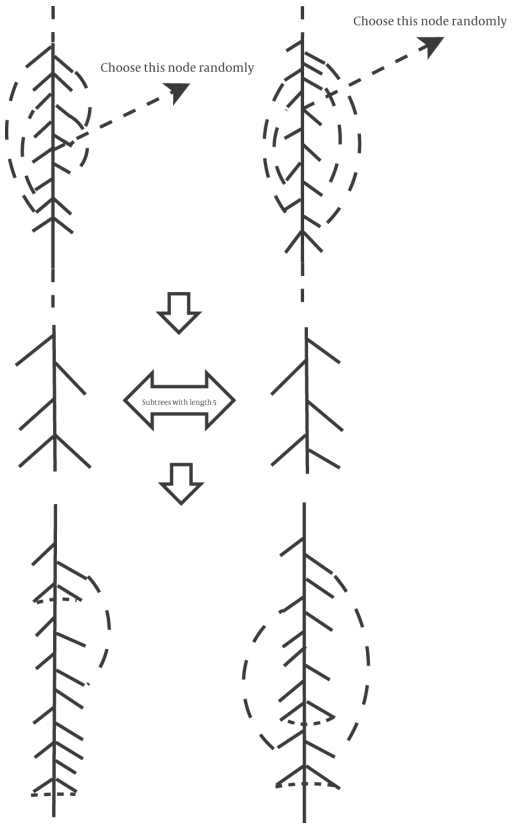


Figure 6. Graphical Representation of the Crossover Process

Within the *i*th iteration of the genetic algorithm, ten percent of highest score of the current population (that is called P_i) are directly copied into the next population (P_{i+1}). Then remaining 90 percent of P_{i+1} are produced by applying the roulette wheel method on the union of P_i and evaluated individuals (Emembers) of P_i. The evaluated individuals (Emembers) are new members generated by performing the evolution functions on the members of P_i.

```
Function Mutation (t: Tree)
Begin
    MutationNumber = Random (Length(t))
    For i=1 to MutationNumber Do Begin
        node0 = A random node from t;
        If node0 is a bases of a pair Then Begin
          node1 = pair of node0;
            If pair (node0, node1) is a crossed pair Then Begin
                Break up pair (node0, node1);
                Reconstruct t;
            End
            Else
                Mark bases node0 and node1 as single bases;
        End
        Else Begin
            node1 = A random single node from t;
            Mark node0 and node1 as a base pair;
            Reconstruct t;
        End;
    End;
    Return (t);
End.
```

Algorithm 1. Mutation Function

```
Function Crossover (t0, t1: Tree)
Begin
    CrossLength = Random (Length (t0));
    node0 = A random node from t0;
    node1 = A random node from t1;
    SubTree0 = MoveDown (node0, CrossLength);
    SubTree1 = MoveDown (node1, CrossLength);
    If sub-trees have exteriorly pair Then
    Break them up;
    Swap (SubTree0, SubTree1);
    Return (t0, t1);
End.
```

Algorithm 2. Crossover Function

This function includes six inner functions; "Random (L)" returns a random value L' (0 < L' < L), the "Length (Sequence)" function that counts the number of terminal nodes of Sequence, "PopulationGeneration (Pop, Number, S)" that generates Number trees of length of S using ESL-TAG, "GetRand ()" that returns a random value between [0:1], our fitness function which is called "ComputScore (Pop)" function by using thermodynamic roles computes the Pop's members energy value, and finally the "RouletteWheel (N, P_i, P_j)" via the roulette wheel method copies ((N*PopulationNumber)/100) members of P_i into P_j.

Table 1. The Two Dimensional Table Represents Energy of Two Consecutive Pairs

	A:U	C:G	G:C	U:A	G:U	U:G
A:U	-0.9	-2.1	-1.7	-0.9	-0.5	-1.0
C:G	-1.8	-2.9	-2.0	-1.7	-1.2	-1.9
G:C	-2.3	-3.4	-2.9	-2.1	-1.4	-2.1
U:A	-1.1	-2.3	-1.8	-0.9	-0.8	-1.1
G:U	-1.1	-2.1	-1.9	-1.0	-0.4	-1.5
U:G	-0.8	-1.4	-1.2	-0.5	-0.2	-0.4

4. Results

To assess our algorithm its accuracy has been evaluated and compared with three common thermodynamic based algorithms: Hotknot (21), Pknots (9), and the STAR (1). To compare PreRkTAG with these algorithms, we measured the accuracy of these algorithms on a set of RNAs from the RNAseP data base. Since genetic algorithms have random behavior, every execution of our algorithm on a given sequence may result in different structure. Hence, the PreRkTAG algorithm was applied three times on every sequence and the best structure of each run was stored. Then one of these three with the minimum energy was reported as the predicted structure. Because of the proposed evolution algorithms, in many cases, the mean of the three energy values were close to the picked minimum energy. The RNAseP data base includes natural structures of this set, therefore in this study the accuracy is measured by considering two different parameters; sensitivity and specificity of folded sequences. Sensitivity is defined by the number of base pairs that are predicted correctly per number of base pairs in natural structure. The specificity is equal to the number of true predicted base pairs per number of predicted base pairs. Predictors were employed on 24 sequences from RNAseP data base, which are tabulated in *Table 2* . The comparison of sensitivity between our algorithm and the other algorithms are given in *Table 3* . Also *Table 4* depicts results of comparing specificity of these algorithms. Unfortunately because of the length of the test sequences, which are fairly long, the Pknots algorithm could not obtain results within a reasonable timeframe on our hardware. Therefore, in *Table 3* and *Table 4* the Pknots results are not shown. We also provide running time complexity of the examined algorithms in the *Table 5*. As mentioned, for every RNA sequence, the PreRkTAG has been applied three times and we indicated outcome of the one with minimum energy. For calculating the running time of the algorithm, we followed the same idea and considered the running time of the one with minimum energy. It is well known that choosing suitable parameter values for genetic algorithms is very difficult (24). There is an inverse relationship between the number of generations and the size of population, so that large population size converges fast and requires the consideration of a few number of generations. On the other hand, large population size requires more evaluations per generation so as to avoid the decline of the rate of convergence. Generally, the optimal size of population depends on the domain complexity of the feasible solutions. In addition, the probabilities of performing evolutionary functions play a critical role to avoid trapping in local minimums. Small values for the crossover and mutation rates yield to copying most of the individuals to the next population directly which is beneficial in terms of decreasing the running time complexity as it is unnecessary to compute fitness values for most of the individuals. However, the small rates may

```
Function PreRkTAG (S: RNA Sequences)
Begin
  PNumber = Random (Length(S));
  PopulationGeneration (P0, PNumber, S);
  i= 0; stop-condition = False ;
  While (stop-condition = False) Do Begin
    Emembers = {∅} ;
    ComputScore(Pi) ;
    Copy ten percent of high scores Pi into Pi1+;
    For each t in Pi Do Begin
     If GetRand() > Mutation Probability Then Begin
        t′ = Mutation (t) ;
        Emembers = Emembers ∪ t′;
      End;
    End;
    For each t0 in Pi Do Begin
     If GetRand() > Crossover Probability Then Begin
        t1 = A random member of Pi ;
        (t′0, t′1) = Crossover(t0, t1);
        Emembers = Emembers ∪ (t′0, t′1);
      End;
    End ;
    ComputScore(Emembers) ;
    RouletteWheel (90, Pi ∪ Emembers, Pi1+);
    If Three previous iterations had equal best score Then
     stop-condition = True ;
    If i exceeds flip flops limitation Then
       stop-condition=True ;
     i++;
  End;
End.
```

Algorithm 3. PreRkTAG Algorithm

Table 2. List of Tested RNA Sequences

RNA's Name	Length	Seq ID
A. Ambivalens	262	AF192349
A. Fulgidus	229	AE000782
A. Truei	320	AF004373
B. Bufo	397	AF044737
C. Paradoxa	351	X89853
D. Janneae	406	AF004372
E. 1B	252	AF192351
E. RH	301	AF192353
G. Gorilla	320	L08685
H. Morrhuae	475	U42981
L. Muta	331	AF004376
M. Formicicum	301	AF121774
M. Musculus	288	L08802
M. Sedula	304	AF121773
M. Thermoautotrophicum	293	U42986
P. Purpurea	383	U38804
S. Carlsbergensis	359	L12746
S. Pastorianus	365	AF186225
S. Uvarum	369	L12749
S. Versatilis	286	NA
U. N2 fixer TP1	294	NA
U. N2 fixer WC1	294	NA
V. Cholerae	399	AE004310
Y. Pestis	377	NA

Table 3. Comparison of Sensitivity Various Algorithms

RNA's Name	PreRkTAG	HotKnots	STAR
A. Ambivalens	0.86	0.43	0.33
A. Fulgidus	0.83	0.70	0.59
A. Truei	0.95	0.62	0.54
B. Bufo	0.31	0.51	0.65
C. Paradoxa	0.92	0.41	0.51
D. Jeanneae	0.67	0.19	0.44
E. 1B	0.88	0.37	0.17
E. RH	0.76	0.13	0.20
G. Gorilla	0.51	0.07	0.38
H. Morrhuae	0.98	0.59	0.06
L. Muta	0.71	0.54	0.31
M. Formicicum	0.54	0.62	0.34
M. Musculus	0.87	0.68	0.53
M. Sedula	0.91	0.95	0.79
M. Thermoautotrophicum	0.94	0.22	0.43
P. Purpurea	0.96	0.68	0.37

S. Carlsbergensis	0.81	0.89	0.75
S. Pastorianus	0.85	0.46	0.88
S. Uvarum	0.87	0.89	0.81
S. Versatilis	0.71	0.41	0.31
U. N2 fixer TP1	0.91	0.56	0.77
U. N2 fixer WC1	0.96	0.88	0.50
V. Cholerae	0.71	0.91	0.87
Y. Pestis	0.87	0.66	0.84

Table 4. The Obtained Specificity Value by the Algorithms

RNA's Name	PreRkTAG	HotKnots	STAR
A. Ambivalens	0.71	0.34	0.31
A. Fulgidus	0.87	0.79	0.51
A. Truei	0.87	0.65	0.67
B. Bufo	0.11	0.17	0.40
C. Paradoxa	0.89	0.29	0.44
D. Jeanneae	0.45	0.04	0.49
E. 1B	0.65	0.23	0.21
E. RH	0.71	0.11	0.07
G. Gorilla	0.45	0.08	0.41
H. Morrhuae	0.93	0.65	0.03
L. Muta	0.68	0.52	0.44
M. Formicicum	0.43	0.78	0.21
M. Musculus	0.84	0.76	0.41
M. Sedula	0.81	0.73	0.64
M. Thermoautotrophicum	0.85	0.16	0.39
P. Purpurea	0.89	0.61	0.31
S. Carlsbergensis	0.70	0.72	0.65
S. Pastorianus	0.71	0.38	0.45
S. Uvarum	0.61	0.73	0.65
S. Versatilis	0.55	0.29	0.17
U. N2 fixer TP1	0.79	0.56	0.23
U. N2 fixer WC1	0.91	0.91	0.38
V. Cholerae	0.63	0.80	0.82
Y. Pestis	0.87	0.66	0.84

cause to converge to local minimums. Considering the behavior of genetic algorithms according to these rates shows that high probabilities for performing the mutation function allow for a greater diversity of solutions at very little expense. However, these high probabilities change the concept of genetic algorithms and convert them into random search procedures. The probability of performing the crossover function also has its own difficulties. The high value of the crossover rate helps the evolution progress and causes to it to jump out of local minimums while increasing the chance of converging to the global minimum. On the other hand, it increases the running time complexity dramatically. A theoretical investigation of the optimal size of population, crossover and mutation rates has been carried out by Goldberge (24) who has found that the best values for the population size is a value base on the length of the solutions. Goldberge showed that a practical crossover rate is in the range of [0.6, 0.95] and for the mutation rate should be in [0.001, 0.2]. In order to evaluate PreRkTAG algorithm, we used Goldberg's investigation for determining our GA's control parameter values. We tested different values for the size of population, crossover and mutation rates on a training set of RNAs. We empirically

Table 5. Comparison of Running Time of the Examined Algorithms in Minutes

RNA's Name	PreRkTAG	HotKnots	STAR
A. Ambivalens	4.575	5.445	6.562
A. Fulgidus	5.123	4.790	5.505
A. Truei	5.622	5.720	6.784
B. Bufo	7.130	7.816	8.011
C. Paradoxa	6.503	5.963	7.976
D. Jeanneae	7.544	6.933	8.715
E. 1B	4.222	4.636	6.637
E. RH	5.297	6.296	5.100
G. Gorilla	5.412	5.738	5.486
H. Morrhuae	9.035	8.435	10.033
L. Muta	5.525	6.773	5.631
M. Formicicum	5.473	5.312	6.336
M. Musculus	5.006	5.979	6.580
M. Sedula	6.633	6.499	5.101
M. Thermoautotrophicum	5.148	5.439	6.857
P. Purpurea	7.598	7.796	8.279
S. Carlsbergensis	7.335	6.399	6.955
S. Pastorianus	7.743	7.265	6.569
S. Uvarum	6.519	6.836	7.334
S. Versatilis	5.251	6.482	5.359
U. N2 fixer TP1	5.177	5.773	5.489
U. N2 fixer WC1	5.901	6.789	6.111
V. Cholerae	7.561	7.853	6.763
Y. Pestis	7.104	6.790	7.811

determined the best value for "population size" as a function of length of RNA sequences. In addition, we found that the best "mutation rate" is 0.2 and "crossover rate" is 0.8.

5. Discussion

In this manuscript, we proposed a new genetic algorithm for RNA secondary structure prediction based on a subclass of Tree Adjoining Grammars. This algorithm gives accurate results in a reasonable amount of time and has specificity and sensitivity comparable with existing tools such as Hotknot, Pknots, and the STAR algorithms. Because of the capability of regular ESL-TAG to cover a large class of RNAs' knotted structures, the time complexity of the parsing algorithm damages its integrity, where in our study, we modified implementation of the parsing algorithm to reduce running complexity.

Since using thermodynamics rules has been demonstrated as the most successful approach for prediction of single RNA sequence, we used these rules in fitness function. We designed novel ideas for the evolutionary functions which yielded to accurate predictions close to the natural structure. The eligibility of tree adjoining grammar for capturing RNAs with knotted structure and also its efficient data structure led us to employ this representation of RNA in the genetic algorithm. It is strongly believed this data structure could be improved and extended toward covering bigger class of knotted structures and a heuristic algorithm like genetic algorithm can use it to predict complicated knotted structures. The algorithm, PreRkTAG, was implemented and tested on a set of 24 different RNA sequences and compared with three well-known algorithms.

The results of the algorithm show that sensitivity of seventeen out of twenty-four samples and specificity of eighteen out of the twenty-four samples are higher than three well-known algorithms. These results show the practicality of the algorithm. This algorithm handles certain class of pseudoknots captured by ESL-TAG and in future work, we intend to extend the algorithm for larger class of pseudoknots.

Acknowledgements

We would like to express our sincere thanks to Professor C.W. Pleij for his kindly helps and guidance on dis-

cussions that led to an improved understanding of the problem characteristics. The authors are also very grateful to Professor D.H. Turner for his guidance on solving our biophysical problems. Finally we thank Professors F.H.D. van Batenburg, M. Zuker, R. Markham, and E. Rivas for their helps for comparing our method vs. theirs. The authors would like to thank the anonymous referees for their helpful suggestions.

Authors' Contribution

Initial idea of the research was from Torabi Dashti, Ahrabian, Nowzari-Dalini, and Zare-Mirakabad. All authors participated in algorithm design, and the structure and organization of the manuscript. Torabi Dashti and Aghaeepour implemented the algorithms and tested on different data sets. All authors contributed to read and approved the final manuscript.

Financial Disclosure

None declared.

Funding/Support

The authors would like to acknowledge the financial support of University of Tehran for this research under grant number 6103010/1/02.

References

1. van Batenburg FH, Gultyaev AP, Pleij CW. An APL-programmed genetic algorithm for the prediction of RNA secondary structure. *J Theor Biol*. 1995;**174**(3):269-80.
2. Hu YJ. GPRM: A genetic programming approach to finding common RNA secondary structure elements. *Nucleic Acids Res*. 2003;**31**(13):3446-9.
3. Lyngso RB, Pedersen CN. RNA pseudoknot prediction in energy-based models. *J Comput Biol*. 2000;**7**(3-4):409-27.
4. Markham NR, Zuker M. DINAMelt web server for nucleic acid melting prediction. *Nucleic Acids Res*. 2005;**33**(Web Server issue):W577-81.
5. Mathews DH, Turner DH. Prediction of RNA secondary structure by free energy minimization. *Curr Opin Struct Biol*. 2006;**16**(3):270-8.
6. Rivas E, Eddy SR. A dynamic programming algorithm for RNA structure prediction including pseudoknots. *J Mol Biol*. 1999;**285**(5):2053-68.
7. Ruan J, Stormo GD, Zhang W. An iterated loop matching approach to the prediction of RNA secondary structures with pseudoknots. *Bioinformatics*. 2004;**20**(1):58-66.
8. Zuker M. Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res*. 2003;**31**(13):3406-15.
9. Eddy SR, Durbin R. RNA sequence analysis using covariance models. *Nucleic Acids Res*. 1994;**22**(11):2079-88.
10. Zare-Mirakabad F, Sadeghi M, Ahrabian H, Nowzari-Dalini A. RNA-Comp: a new method for RNA secondary structure alignment. MATCH Commun. *Math Comput Chem*. 2009;**61**(3):789-816.
11. Gorodkin J, Stricklin SL, Stormo GD. Discovering common stem-loop motifs in unaligned RNA sequences. *Nucleic Acids Res*. 2001;**29**(10):2135-44.
12. Gutell RR, Power A, Hertz GZ, Putz EJ, Stormo GD. Identifying constraints on the higher-order structure of RNA: continued development and application of comparative sequence analysis methods. *Nucleic Acids Res*. 1992;**20**(21):5785-95.
13. Tahi F, Gouy M, Regnier M. Automatic RNA secondary structure prediction with a comparative approach. *Comput Chem*. 2002;**26**(5):521-30.
14. Akutsu T. Dynamic programming algorithms for RNA secondary structure prediction with pseudoknots. *Discrete App Math*. 2000;**104**(1–3):45-62.
15. Reeder J, Giegerich R. Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. *BMC Bioinformatics*. 2004;**5**:104.
16. Uemura Y, Hasegawa A, Kobayashi S, Yokomori T. Tree adjoining grammars for RNA structure prediction. *Theoret Comp Sci*. 1999;**210**(2):277-303.
17. Zuker M, Jaeger JA, Turner DH. A comparison of optimal and suboptimal RNA secondary structures predicted by free energy minimization with structures determined by phylogenetic comparison. *Nucleic Acids Res*. 1991;**19**(10):2707-14.
18. Walter AE, Turner DH, Kim J, Lyttle MH, Muller P, Mathews DH, et al. Coaxial stacking of helixes enhances binding of oligoribonucleotides and improves predictions of RNA folding. *Proc Natl Acad Sci U S A*. 1994;**91**(20):9218-22.
19. Cai L, Malmberg RL, Wu Y. Stochastic modeling of RNA pseudoknotted structures: a grammatical approach. *Bioinformatics*. 2003;**19**(Suppl 1):i66-73.
20. Shapiro BA, Navetta J. A massively parallel genetic algorithm for RNA secondary structure prediction. *J Supercomp*. 1994;**8**(3):195-207.
21. Ren J, Rastegari B, Condon A, Hoos HH. HotKnots: heuristic prediction of RNA secondary structures including pseudoknots. *RNA*. 2005;**11**(10):1494-504.
22. Kato Y, Seki H, Kasami T. On the generative power of grammars for RNA secondary structure. *IEICE Trans Inf Syst*. 2005;**88**(1):53-64.
23. Joshi AK, Levy LS, Takahashi M. Tree adjunct grammars. *J Comp Sys Sci*. 1975;**10**(1):136-63.
24. Goldberg DE. *Genetic algorithms in search, optimization, and machine learning. Addison-Wesley*. 1989
25. Freier SM, Kierzek R, Jaeger JA, Sugimoto N, Caruthers MH, Neilson T, et al. Improved free-energy parameters for predictions of RNA duplex stability. *Proc Natl Acad Sci U S A*. 1986;**83**(24):9373-7.