Bioinformatics, 36, 2020, i227–i235
doi: 10.1093/bioinformatics/btaa469
ISMB 2020

Finding the direct optimal RNA barrier energy and improving pathways with an arbitrary energy model

Hiroki Takizawa[1], Junichi Iwakiri[1], Goro Terai[1] and Kiyoshi Asai[1,2,*]

[1]Department of Computational Biology and Medical Sciences, Graduate School of Frontier Sciences, University of Tokyo, Tokyo, Chiba 277-8561 Japan and [2]Artificial Intelligence Research Center (AIRC), National Institute of Advanced Science and Technology (AIST), Tokyo135-0064, Japan

*To whom correspondence should be addressed.

Abstract

Motivation: RNA folding kinetics plays an important role in the biological functions of RNA molecules. An important goal in the investigation of the kinetic behavior of RNAs is to find the folding pathway with the lowest energy barrier. For this purpose, most of the existing methods use heuristics because the number of possible pathways is huge even if only the shortest (direct) folding pathways are considered.

Results: In this study, we propose a new method using a best-first search strategy to efficiently compute the exact solution of the minimum barrier energy of direct pathways. Using our method, we can find the exact direct pathways within a Hamming distance of 20, whereas the previous methods even miss the exact short pathways. Moreover, our method can be used to improve the pathways found by existing methods for exploring indirect pathways.

Availability and implementation: The source code and datasets created and used in this research are available at https://github.com/eukaryo/czno.

Contact: asai@k.u-tokyo.ac.jp

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

RNA folding kinetics is important for the biological function of many RNA molecules (Gerdes *et al.*, 1997; Espah Borujeni and Salis, 2016). For example Espah Borujeni and Salis (2016) showed that the accessibility of Shine-Dalgarno sequences affects translation efficiency in prokaryotes; the authors also used kinetic features to design nucleotide sequences of prokaryotic mRNAs. In the *hok*/*sok* *E.coli* system, the folding kinetics of *hok* mRNA may have pivotal roles in R1 plasmid maintenance (Nagel *et al.*, 1999).

RNA folding pathways can be classified based on two aspects. We briefly introduce some related terms here, with more detailed descriptions in Section 2. The first aspect consists of *direct* versus *indirect* folding pathways. For two given RNA structures, a direct folding pathway is the shortest pathway. In other words, there is no bypassing or going backwards in inferring a direct folding pathway. Otherwise, such a pathway is called an indirect folding pathway. The second aspect consists of *optimal* versus *approximate* pathways. Among all possible (direct or indirect) pathways, the optimal folding pathway is the one with the lowest barrier energy. A pathway's barrier energy is the energy of the most thermally unstable structure in the folding pathway. All other pathways are called approximate (or simply heuristic or not optimal) pathways.

Finding better RNA folding pathways (i.e. those with lower barrier energy) is important for various reasons. In molecular kinetics, some quantities such as transition rate, minimum barrier energy and folding pathway are related to each other. The searches among minimum barrier energies and corresponding folding pathways are inseparable. For a given minimum barrier energy value, one can derive the transition rate with the Arrhenius equation (Wolfinger *et al.*, 2004). Determining the optimal barrier energy or establishing its tighter upper bound enables us to more accurately estimate transition rates. Determining the optimal direct pathway is also helpful for understanding structure landscapes. With multiple meta-stable structures given, one can determine the 'radius' of each basin that is bounded at a distance from the structure. The direct barrier energy value gives a rational upper-bound of that distance.

The timescale of RNA folding kinetics can reach several tens of minutes (Gerdes *et al.*, 1997). Furthermore, the transition states for these processes are often complex and diffusive (Chen and Dill, 2000). Hence, a direct analysis by molecular dynamics simulation is generally difficult. Therefore, we will consider the minimum energy barrier of RNA secondary structures (Morgan and Higgs, 1998).

In RNA secondary structure folding pathways, the minimum energy barrier problem is known to be NP-hard for an arbitrary pathway and energy model (Maňuch *et al.*, 2011). For that reason, most of the existing research has focused on approximation algorithms. There has been no method for applying Dijkstra's shortest path algorithm for the exact solution of direct pathways with an arbitrary energy model.

Because previous studies aimed at determining optimal RNA folding pathways are extensive, we briefly review the literature below. More detailed reviews can be found in recent articles (Dotu *et al.*, 2010; Li and Zhang, 2012). Morgan and Higgs (1998) and Flamm *et al.* (2001) reported pioneering research on direct approximate pathways, using a greedy method and beam-search method respectively; Voss *et al.* (2004) and Dotu *et al.* (2010) also reported semi-greedy methods. The work by Morgan and Higgs (1998) also included important research on indirect approximate pathways using a greedy method. Recently, RNA2DPATH by Lorenz *et al.* (2009), RNATABUPATH (Tabu-Search) by Dotu *et al.* (2010) and RNAEAPath (Evolutionary Algorithms) by Li and Zhang (2012) reported alternative methods for indirect approximate pathways. Finally, the following studies are notable, but not directly related to our proposed algorithm. RNAsubopt in ViennaRNA Package (Lorenz *et al.*, 2011) provides a sampling method (Ding and Lawrence, 2003). RNAsubopt also enumerates structures with energy below the user-specified threshold. BARRIERS (Flamm *et al.*, 2002) utilizes RNAsubopt and calculates the optimal (in)direct minimum barrier energy considering all (in)direct pathways. Note that BARRIERS works only when the barrier value is sufficiently close to the minimum-free-energy (MFE) of the sequence; otherwise, RNAsubopt must enumerate an intractably large number of structures. By limiting their method to a simple energy model, Thachuk *et al.* (2010) successfully obtained a long direct optimal pathway, although its worst-case complexity was still exponential.

As Morgan and Higgs noted, the total number of direct pathways is $O(|H|!)$ for Hamming distance $|H|$ (Morgan and Higgs, 1998). However, the total number of possible intermediate structures is $O(2^{|H|})$, because it can be regarded as a family of subsets of immediate transitions. $O(2^{|H|})$ is much smaller than $O(|H|!)$. The number of immediate transitions from each intermediate structure is $O(|H|)$. Thus, Dijkstra's algorithm can be transformed and applied (Dijkstra, 1959; Mohri, 2002). While the resulting algorithm is still exponential, it is much faster than the process of enumerating all pathways (Morgan and Higgs, 1998). Based on this method, we hypothesized that we could obtain exact solutions for longer direct pathways and improve solutions for indirect pathways.

2 Materials and methods

2.1 RNA definition

In our method, only Watson–Crick base pairs (A–U, C–G) or Wobble base pairs (G–U) are considered; no pseudo-knots are allowed. We consider production or destruction of one base pair to be an immediate transition, which means that the Hamming distance changes by one with an immediate transition.

2.2 Notation

We use the following notation scheme according to a previous study (Dotu *et al.*, 2010). Let X be an RNA sequence and Σ be a set of all secondary structures of X. Let us define a structure pair $\{\sigma_i, \sigma_j\} \in \Sigma$ as an 'immediate transition' when their difference is exactly one base pair (i.e. their Hamming distance is one). A sequence of structures ($A = \sigma_0, \sigma_1, \ldots, \sigma_n = B$) is called a 'folding pathway from A to B' if $\sigma_i \in \Sigma$ for $0 \leq i \leq n$ and each pair $\{\sigma_i, \sigma_{i+1}\}$ is an immediate transition. If A and B differ at exactly n base pairs, or equivalently if the Hamming distance between A and B is n, it is called a 'direct' folding pathway; otherwise, it is an 'indirect' folding pathway. Each $\sigma \in \Sigma$ has its free energy $E(\sigma|X)$ or simply $E(\sigma)$. The barrier energy of a folding pathway $(\sigma_0, \sigma_1, \ldots, \sigma_n)$ is defined as $\max(E(\sigma_0), E(\sigma_1), \ldots, E(\sigma_n))$.

2.3 Preliminary algorithms

Algorithm 1 accepts two base pairs, P and Q. It returns true if and only if the two base pairs cannot exist in one structure, that is, (i) they require the same base position or (ii) they are in a 'pseudo-knot' position.Considering all direct pathways from A to B, the number of possible intermediate structures is $(2^{|H|} - 2)$, where $|H|$ is the Hamming distance between structures A and B. Algorithm 2 accepts an index number $N \in [0, 2^{|H|})$ and two structures, A and B, returning an intermediate structure S. An example of the relation between an integer and intermediate structure is shown in Figure 1. In this example, there are three different base pairs between the initial and end structures, and hence, there are 2^3—2 intermediate structures between them.

Algorithm 1 IsExclusive

```
Input: (BasePair $P$ and $Q$)
Output: ($true$ or $false$)
1: '.l' and '.r' represent the left and right positions of a base
   pair, respectively.
2: if $P.l \leq Q.l \leq P.r \leq Q.r$ then
3:     return $true$
4: end if
5: if $Q.l \leq P.l \leq Q.r \leq P.r$ then
6:     return $true$
7: end if
8: return $false$
```

Algorithm 2 IndexToStructure

```
Input: (Integer $N$, Structure $A$ and $B$)
Output: (Structure $S$)
1: Ensure $A$ and $B$ are a set of base-pairs.
2: $H \Leftarrow$ an array of base-pairs different between $A$ and $B$
3: $S \Leftarrow A \cap B$
4: for ($i \in [0, |H|)$) do
5:     if ($H[i] \in A$) $xor$ ($i$-th bit of $N$ is 1) then
6:         $S \Leftarrow S \cup H[i]$
7:     end if
8: end for
9: return $S$
```

Algorithm 2 resolves the bijective relationship between an integer (index) and its corresponding structure with respect to the initial structure and the end structure. For binary integers (N_1, N_2) and the corresponding structures (S_1, S_2), the binary Hamming distance between N_1 and N_2 is equal to the Hamming distance between S_1 and S_2. For example, in Figure 1b, the binary Hamming distance between $N = 4$ and 6 is 1, meaning that there is exactly one base pair differing between the two structures. Accordingly, Algorithm 2 enables us to treat a one base-pair manipulation as a single bit manipulation. Each binary integer N retains the information regarding which base pairs have been added to or deleted from the initial structure A. For example in Figure 1b, the first bit of $N = 1$ is 1, which indicates a base pair $H[0] = bp1$ has been deleted from the initial structure A in the intermediate structure corresponding to $N = 1$.

2.4 Previous algorithms for obtaining approximate direct pathways

To clarify the difference between the proposed and the existing methods, the existing methods are specifically described here. Note that detailed explanations for the same methods have been previously described (Dotu *et al.*, 2010).

2.4.1 Morgan and Higgs greedy method

Morgan and Higgs (1998) proposed a greedy method for obtaining a direct pathway. They considered a simple energy model in which energy is determined by the number of base pairs. According to their

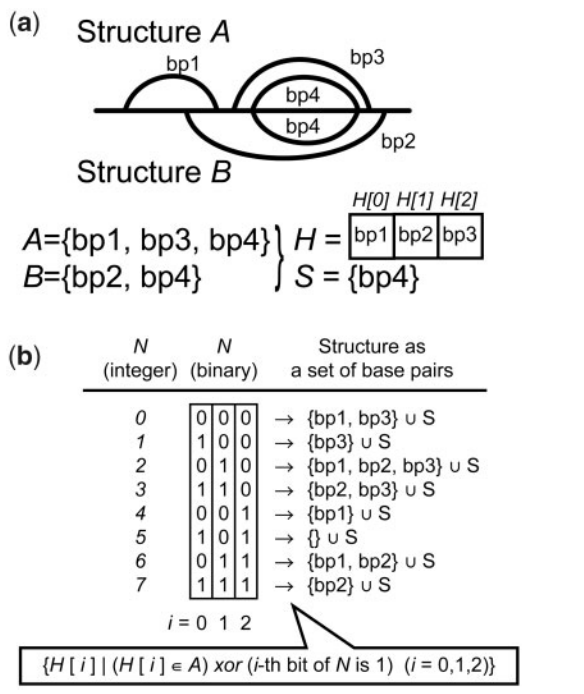

Fig. 1. An example of the relation between an integer and intermediate structures. (**a**) The initial structure *A* and end structure *B* have three and two base pairs, respectively. The structures *A* and *B* are represented by sets of base pairs (lower left). *H* is an array of base pairs that differ between *A* and *B*, and *S* is the set of the common base pairs. (**b**) All intermediate structures are represented by integers between 1 and $2^{|H|}$-2, where $|H| = 3$ in this example. *A* and *B* correspond to 0 and $2^{|H|}$-1, respectively. Each integer is converted to an intermediate structure represented by a set of base pairs as follows. For $i = 0, 1, 2$, a base pair $H[i]$ is included in the intermediate structure indicated by a particular integer *N* if $H[i] \in A$ *xor* the *i*th bit of *N* is 1

method, if the direct pathway from A to B consists only of removing base pairs, then this can be performed in an arbitrary order. In contrast, if the pathway involves transitions of base pairs, for each transition along the pathway, the number of base pairs that must be removed for the transition to occur is counted. Then, the transition with the lowest count is selected, and the selected transition is performed after conducting the necessary base-pair-removing transitions.

2.4.2 Findpath and the Voss *et al.* method

Flamm *et al.* (2001) proposed a semi-greedy method (Findpath) utilizing beam search, which is a well-known versatile pruning method that has been recently utilized in sequence generation tasks with neural networks (Graves, 2012). Figure 2a shows how beam-search works. Voss *et al.* (2004) later proposed a greedy method equivalent to Findpath's method with beam-breadth = 1. Note that the method by Voss *et al.* (2004) is a greedy method for arbitrary energy models, while the Morgan and Higgs (1998) method is a 'greedy' method only with respect to the simple energy model.

2.4.3 Modification of the Voss *et al.* method by Dotu *et al.*

Dotu *et al.* (2010) modified the Voss *et al.* (2004) method. The method by Dotu *et al.* (2010) is similar to that shown in Figure 2a. The difference is that at each depth, one of the best *k* candidates is chosen uniformly at random and the others are discarded. The methods by Dotu *et al.* (2010) and Voss *et al.* (2004) are equivalent when $k = 1$.

2.5 Proposed algorithms for obtaining the optimal direct pathway

In our method, all intermediate structures are connected to integers (indices) in a bijective way and treated as nodes of a directed graph from structure *A* to structure *B*. Some of the intermediate structures,

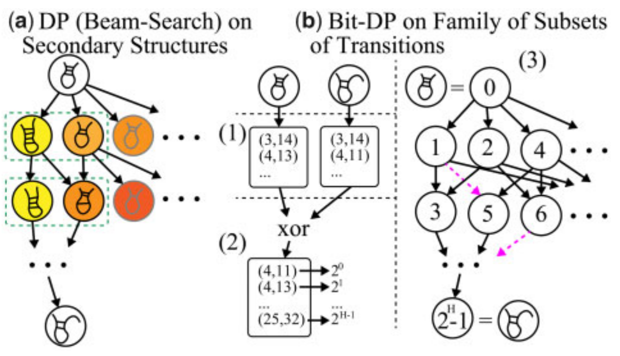
(a) DP (Beam-Search) on Secondary Structures
(b) Bit-DP on Family of Subsets of Transitions
(1)
(3,14)
(4,13)
(3,14)
(4,11)
(2)
xor
(4,11)
(4,13)
(25,32)
2^0
2^1
2^{H-1}
(3)
0
1
2
4
3
5
6
2^H-1

Fig. 2. Schematic illustrations of (**a**) a previous method and (**b**) the proposed method for direct pathway search. (a) The uppermost structure is the initial structure (i.e. depth = 0). First, all possible transitions (arrows) from each considered structure are enumerated. Second, all destination structures are sorted by barrier energy of the structure. [In (a), structures are sorted by barrier energy, which is represented by color.] Next, only the best *k* structures are stored (green dashed box; here, $k = 2$), and the others are discarded (gray structures). Finally, the next depth is searched. (b) (1) First, the initial and goal structures are converted to base-pair set representations. (2) Then, the exclusive OR set of the two sets is taken, and a set of base pairs is obtained. Note that, each direct pathway is in a bijective relationship with the reordering of the elements of this set. Next, each element of this set (i.e. an immediate transition) is made to correspond to the number 2^i. (3) In the aforementioned procedure, the initial structure is identified as 0. The goal structure is identified as $2^H - 1$. Each intermediate structure is identified with a distinct intermediate number. Each immediate transition is identified as a 1-bit change. Any search algorithms, including Dijkstra's method, can be efficiently performed with this representation. Note that, some transitions may be invalid (purple dashed arrows); they can be efficiently removed using an auxiliary table, as described

Algorithm 3 ConstructAuxilaryData

```
Input: (Structure $A$ and $B$)
Output: (Array $H$ and $b$)
1: Ensure that $A$ and $B$ are base pair sets.
2: $H \Leftarrow$ a vector of base pairs different between $A$ and $B$
3: $b \Leftarrow$ an array of length $|H|$ filled with zeros.
4: for $i \in [0, |H|)$ do
5:   for $j \in [0, |H|)$ do
6:     if $H[i] \in A$ and $H[j] \in B$ then
7:       if $IsExclusive(H[i], H[j])$ then
8:         /* Set the $i$-th bit of $b[j]$ to 1. */
9:         $b[j] \Leftarrow b[j] + 2^i$
10:      end if
11:    end if
12:  end for
13: end for
14: return $(H, b)$
```

however, may have base pairs that cannot coexist. For example, in Figure 1b, the structures for $N = 2$ and 6 have base pairs that cannot coexist. Such invalid structures should be efficiently avoided during the exploration of the optimal direct pathway. To do this, we make an integer array *b* that encodes the information about which base pairs cannot coexist.

In the proposed method, *H* is an array of base pairs different between structure *A* and structure *B*, hence $|H|$ is the Hamming distance between the two structures. For two base pairs, $H[i]$ and $H[j]$, if and only if the following two conditions are true, the *i*th bit of an integer $b[j]$ is set to 1.

- base pair $H[i]$ and $H[j]$ are contained in *A* and *B*, respectively.

Algorithm 4 ShortestPath (Unidirectional)

```
Input: (Vector<BasePair> H, Vector<Int> b,
Sequence X, Structure A, Structure B)
Output: (Vector<Real> Result, Vector<Real> Energy,
Vector<bool> Searched)
1: $Result \Leftarrow$ an array of length $2^{|H|}$ filled with $inf$s.
2: $Energy \Leftarrow$ an array of length $2^{|H|}$ filled with $inf$s.
3: $Searched \Leftarrow$ an array of length $2^{|H|}$ filled with $false$s.
4: $d \Leftarrow$ an empty priority queue of ascending order.
5: $t \leftarrow 0$
6: $Energy[0] \Leftarrow E(A)$
7: $d.push((Energy[0], t, 0))$
8: while $d \neq \emptyset$ do
9:    $t \Leftarrow t - 1$
10:   $x \Leftarrow d.top()$
11:   $d.pop()$
12:   $i \Leftarrow x[2]$
13:   if $searched[i] = true$ then
14:      $continue$
15:   end if
16:   $searched[i] \Leftarrow true$
17:   if $i = 2^{|H|} - 1$ then
18:      $break$
19:   end if
20:   for $j \in [0, |H|)$ do
21:      if $j$-th bit of $i$ is 1 then
22:         $continue$
23:      end if
24:      if $(i\, bitwiseand\, b[j]) \neq b[j]$ then
25:         $continue$
26:      end if
27:      $i' \Leftarrow i + 2^j$
28:      if $Energy[i'] = inf$ then
29:         $Energy[i'] \Leftarrow E(IndexToStructure(i', A, B))$
30:      end if
31:      $r \Leftarrow max(Result[i], Energy[i'])$
32:      if $r \geq Result[i']$ then
33:         $continue$
34:      end if
35:      $Result[i'] \Leftarrow r$
36:   $d.push((Energy[i'], t, i'))$
37: end for
38: end while
39: return $(Result, Energy, Searched)$
```

- base pair $H[i]$ and $H[j]$ cannot coexist.

The procedure for obtaining b is shown in Algorithm 3. When $H[i]$ in A cannot coexist with $H[j]$ in B, we must delete a base pair $H[i]$ before adding $H[j]$. In other words, until the transition of deleting a base pair $H[i]$ is conducted, the transition of base pair $H[j]$ is invalid. Array b enables us to efficiently determine whether each immediate transition is valid.Algorithm 4 is based on Dijkstra's algorithm (Dijkstra, 1959; Mohri, 2002). While Dotu *et al.* (2010) used the same algorithm for implementing a previous method (Morgan and Higgs, 1998), we describe it in detail because the concept of Dijkstra with respect to min–max algebraic structure is not trivial and its combination with Bit-DP is complicated. In contrast, Findpath's beam-search method can find the equivalent optimal result when beam-breadth is infinity. The novelty of our method

Algorithm 5 DirectOptimalPath

```
Input: (Sequence X, Structure A, Structure B)
Output: (Vector<Structure> Path, Real MaxEnergy)
1: $(H, b) = Preparation(X, A, B)$
2: $(R, E, S) = ShortestPath(H, b, X, A, B)$
3: return $TraceBack(H, b, X, A, B, R, E, S)$
```

includes its (i) consideration of all possible intermediate structures as a family of subsets of immediate transitions and (ii) execution of a best-first search using what is called a bit-DP (dynamic programming) technique for maintaining simplicity and efficiency.

Edges represent all possible immediate transitions. The weight of an edge is the maximum free energy of the two connected nodes. We used pairing heap (std::priority_queue of C++ and GCC 5.5.0), and the worst-case time complexity is $O(E + V \log V)$, where E is the number of edges and V is the number of nodes. Because E is $O(h2^{|H|})$ and V is $O(2^{|H|})$, the overall complexity is $O(|H|2^{|H|})$, where $|H|$ is the Hamming distance between the two input structures.In Algorithm 4, variable t has an effect associated with nodes of the same weight extracted in a Last In, First Out (LIFO) manner. With LIFO, the search manner becomes depth-first after identifying the true barrier energy structure. This is equivalent to an A-star search (Hart *et al.*, 1968) that considers the distance from the goal. Using the variable t, this concept has been implemented simply. The overall algorithm is described as Algorithm 5. This algorithm calls the *TraceBack* function detailed in Supplementary Algorithm S1.

2.6 Another algorithm with light-weight threading

The calculation time of Algorithm 4 depends heavily on which of the two endpoints is selected as the initial structure. Specifically, when the true barrier structure is very close to the goal structure, then almost all structures are explored. In this case, the solution can be obtained quickly by starting the search from the opposite end point. However, the barrier structure is not known in advance. It is therefore expected that an average speedup can be achieved by simultaneously searching from the two end points. This concept is implemented in Supplementary Algorithms S2 and S3 (in Supplementary Material). Note that, there are variants of this concept that depend on the granularity of context switching: (i) evaluating energy or searching a node at every time point ('coarse grained') or (ii) only searching a node at every time point ('fine grained'). These approaches are quite complicated because they use (i) capture by reference of a lambda expression and (ii) context switching for light-weight threading. Supplementary Algorithm S3 can switch the behavior through the use of the argument *IsFineGrained*. Although this concurrent strategy can expand to a parallel (two-threaded) approach with an appropriate mutex and other considerations, this is outside of the scope of the present research. Because parallelization is beneficial only when the energy evaluation is computationally intensive, we are interested in an arbitrary energy model. Moreover, if the parallelization is completely effective, then an exponential time algorithm would only be at most twice as fast.

2.7 A proposed algorithm for improving approximate pathways

Algorithm 6 accepts a possibly indirect pathway. For all subsequences of length MaxLen or less, if the subsequence has a barrier structure, our proposed algorithm is used to try to improve the barrier energy. If this succeeds, the subsequence is replaced and recursively computed again.

Algorithm 6 ImprovePathway

```
Input: (Sequence X, Vector<Structure> Path, int MaxLen)
Output: (Vector<Structure> NewPath)
1: $N = len(Path)$
2: $Barrier = max(E(Path))$
3: for $i \in [0, N-2]$ do
4:   for $j \in [i+2, min(i+MaxLen, N)]$ do
5:     $M \Leftarrow max(E(Path[i]), \ldots, max(E(Path[j])))$
6:     if $M = Barrier$ then
7:       $A = Path[i]$
8:       $B = Path[j]$
9:       $(P, M) = DirectOptimalPath(X, A, B)$
10:        if $M < Barrier$ then
11:          $Q = (Path[0], \ldots, Path[i-1])$
12:          $Q.Append(P)$
13:          $Q.Append(Path[j+1], \ldots, Path[N])$
14:          return $ImprovePathway(X, Q, MaxLen)$
15:        end if
16:     end if
17:   end for
18: end for
19: return $Path$
```

Algorithm 7 RNA2DFOLDIndirect

```
Input: (Sequence X, Structure A, Structure B, int K = 5)
Output: (Vector<BasePair> H, Vector<Int> b)
1: if $E(A) < E(B)$ then
2:   return Reverse(RNA2DFOLDIndirect(X, B, A, K))
3: end if
4: $F \Leftarrow$ compute RNA2DFOLD with A and B as references.
5: $G \Leftarrow$ a graph with local MFEs of $F$ as nodes and with no
   edges.
6: for $i, j \in F$ do
7:   $d \Leftarrow$ Manhattan distance between $i$ and $j$ in the space $F$
8:   if $d \leq K$ or $B \in \{i, j\}$ then
9:     $W \Leftarrow$ Compute an approximate barrier energy from $i$
       to $j$ using Findpath. The obtained pathway is kept
       and used during traceback.
10:    Add an undirected edge $(i, j)$ to $G$. The cost of the
       edge is $W$.
11:  end if
12: end for
13: Compute Dijkstra's algorithm on $G$ with min-max
    algebra.
14: Compute the traceback and obtain the lowest barrier
    pathway $H$ and barrier energy $b$.
15: return $(H, b)$
```

2.8 Experimental procedure

For the evaluations, we generated a random RNA dataset. For a given RNA of length N and Hamming distance $|H|$, the following procedure was conducted to sample one data point. At first, we sampled a uniformly random RNA sequence X of length N. Second, we sampled two secondary structures σ_1, σ_2 for RNA X using RNAsubopt in Vienna-RNA 2.4.3 (Lorenz *et al.*, 2011). Then, if the Hamming distance of the two structures is $|H|$, we added the triplet (X, σ_1, σ_2) to the dataset. The source code and datasets created and used are available at https://github.com/eukaryo/czno.

In the proposed algorithm, RNAeval in Vienna-RNA (Lorenz *et al.*, 2011) 2.4.3 was used to obtain the free energy of each secondary structure. RNAeval uses an additive loop model, and it takes $O(N)$ time to evaluate each structure, where N is the length of the RNA. Note that one can reduce this time-complexity to $O(1)$ with our algorithm by differential evaluation, as implemented by KFOLD (Dykeman, 2015). Nevertheless, this energy-model-specific improvement was beyond the scope of the present study because we aimed to construct algorithms for an arbitrary energy model. We implemented the proposed algorithms in C++14. We also re-implemented previous algorithms in C++14. All source code is available at the GitHub repository.

2.8.1 Heuristic indirect algorithm with RNA2DFOLD

Lorenz *et al.* (2009) described a method (RNA2DFOLD) to decompose the distribution of secondary structures into two-dimensional space and to calculate local MFEs of each position in decomposed space. They also proposed an algorithm to calculate indirect low-barrier folding pathways by connecting local MFEs with the Findpath method. RNA2DFOLD requires $O(N^7)$ computation, but sophisticated algorithmic techniques enabled them to calculate up to 400 nucleotides within a realistic timeframe. Additionally, they proposed a stochastic algorithm to obtain indirect approximate pathways. In this study, we introduced a deterministic algorithm for the same purpose (Algorithm 7).

Algorithm 7 can be described as shown. Without a loss of generality, we can assume that the goal structure has the same or less energy than the initial structure, because otherwise the two structures must be swapped. In Algorithm 7, we first confirm this condition. Next, we execute RNA2DFOLD and obtain local MFE structures. Then, we consider a graph G, whose nodes are all local MFEs and whose edges are (i) nearby nodes in the coordinates of RNA2DFOLD and (ii) from all other nodes to the goal structure. The cost of each edge is the approximate barrier energy obtained by Findpath, because in the coordinates of RNA2DFOLD, even two adjacent local MFEs can have a Hamming distance of two or more. We do not use the proposed optimal method here, because the distance may be very large and the Algorithm 6 can be applied later. Finally, we compute Dijkstra's algorithm on G with min–max algebra.

3 Results and discussion

3.1 Performance comparison across the detailed settings of the proposed method

First, we compared the performance of the various proposed algorithms. The results are shown in Figure 3a–e. For the concurrent searches, Figure 3a and b shows that the fine-grained version performed slightly better. In the unidirectional searches, Figure 3c shows that choosing an unstable structure of two endpoints as a start gives better results. Figure 3d and e shows that unidirectional search with a starting unstable structure often gives better results than a concurrent search. On the other hand, concurrent search was more robust. In most cases, it required twice as much computation time as either unidirectional search. Hence, we used fine-grained concurrent search in the subsequent analyses.

3.2 Comparison with Findpath, with infinite beam-breadth

As noted above, Findpath (Flamm *et al.*, 2001) can obtain the optimal direct pathway when beam-breadth parameter $k = \text{inf}$. For that reason, we compared our Algorithm 5 and Findpath

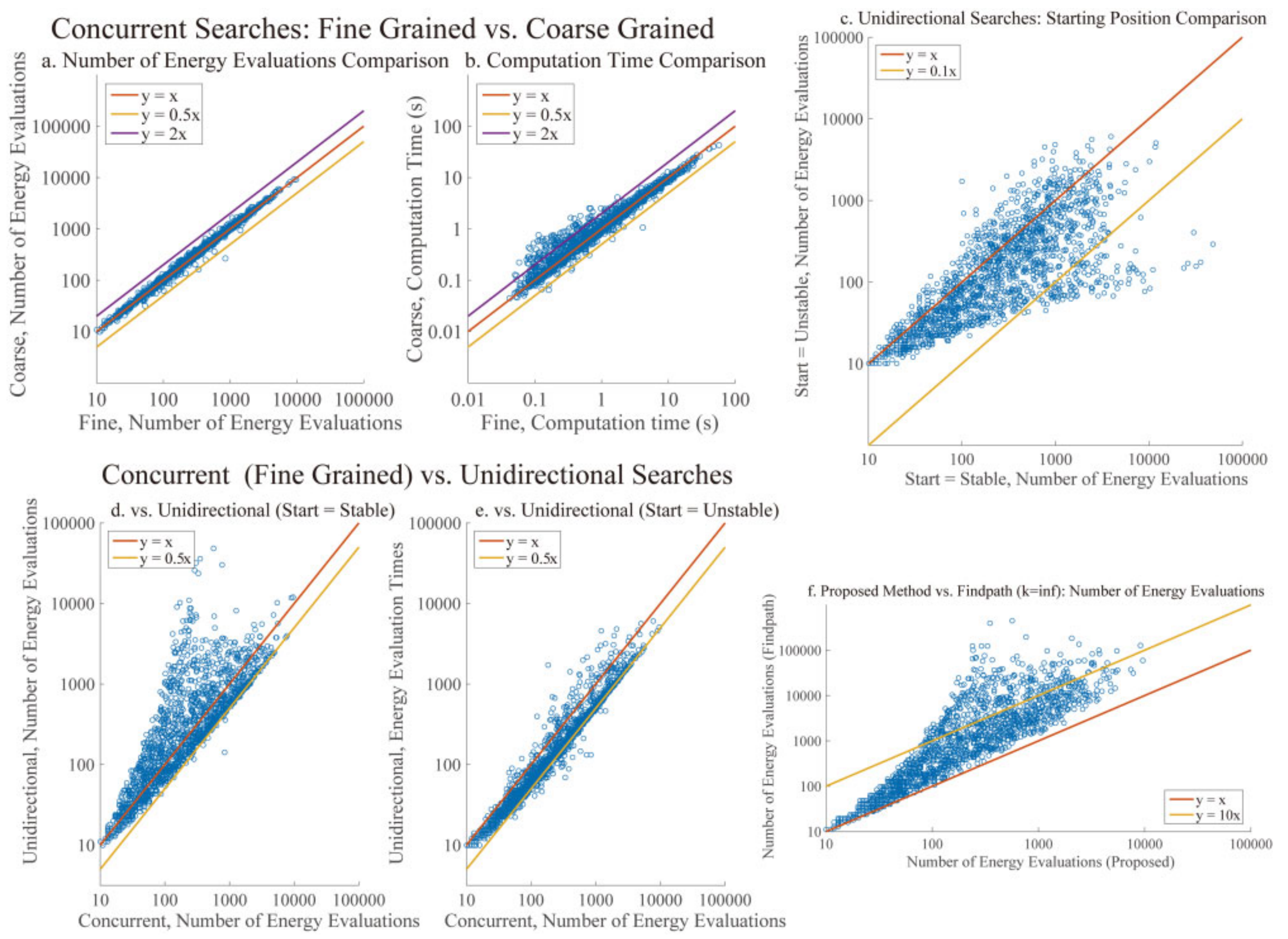

Fig. 3. Evaluation of the frequency of energy calculation and its computation time in our method using concurrent/unidirectional searches and fine/coarse grained versions. A total of 1600 data points comprising random RNA sequences and structures (100 for each, with Hamming distances ranging from 5 to 20) were evaluated. (**a**) and (**b**) Comparison between fine-grained and coarse-grained versions under the concurrent search methods. The definitions of 'fine grained' and 'coarse grained' are provided in Section 2. (**c**) Evaluation of two types of starting structures (unstable and stable) for the unidirectional search method. (**d**) and (**e**) Comparison between concurrent (fine-grained) and unidirectional searches (stable/unstable starting structures). (**f**) Comparison between the proposed method (concurrent, fine-grained) and previous method (beam-search, $k = infinity$)

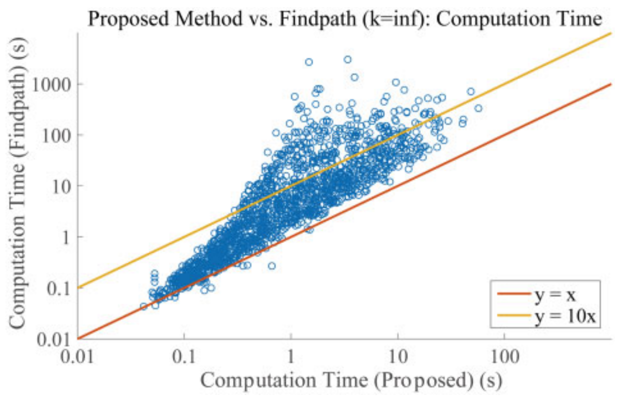

Fig. 4. The same data summarized in Figure 5a are used here. A total of 1600 data points were evaluated. The x-axis shows the computation times for the search with the proposed algorithm. The y-axis shows the computation times with the Findpath (Flamm *et al.*, 2001) algorithm for beam-breadth = infinity. Note that the two algorithms output the same barrier value; the main difference is the type of search performed

Flamm *et al.* (2001) with that setting. The results are demonstrated in Figure 3f. Our method can obtain an optimal pathway with far shorter energy evaluation times.

Figure 4 shows a comparison of real-time demand. Our re-implementation of Findpath (Flamm *et al.*, 2001) is not necessarily optimized for execution speed. However, Figure 4 suggests that our proposed method is also faster in real time.

3.3 Computation time

Figure 5a shows the computation time for a random RNA sequence (length = 100) and random secondary structures. The result roughly follows the theoretical worst-case time complexity $O(|H|2^{|H|})$. For some data points, computation finished within a very short time. As we utilized Dijkstra's algorithm, the computation may be completed quickly depending on the specific shape of the energy landscape. Figure 5b indicates that sequence length hardly affects computation time.

We expected that the energy evaluation would account for most of the total computation time. Otherwise, our attempt to reduce the number of energy evaluations would be pointless. Figure 5c was consistent with our expectation, showing that the total computation time is roughly proportional to the number of times the algorithm has evaluated the energy.

3.4 Comparison with a previous method for approximate direct pathways

It is guaranteed that the proposed method returns an exact solution. However, the previous approximate methods possibly also

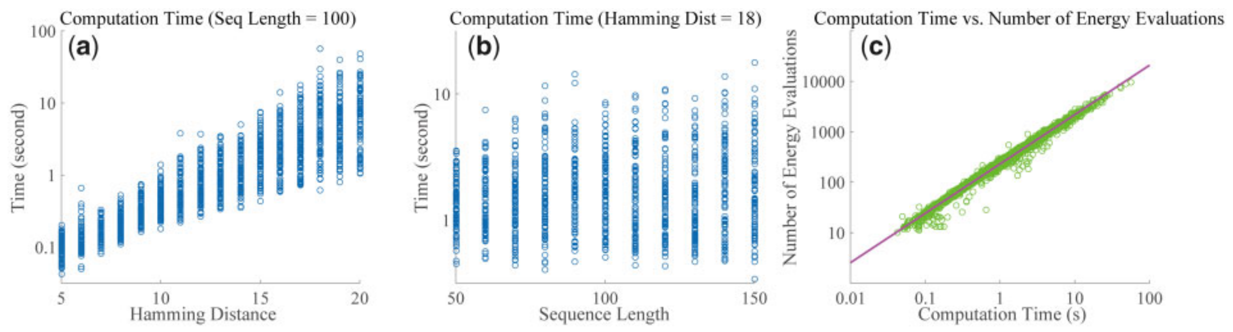

Fig. 5. Evaluation of the effect of the Hamming distance and sequence length on computation time for the proposed method. (**a**) Effect of Hamming distances ranging from 5 to 20 between the initial and the goal structures under a fixed length of RNA sequence (= 100). A total of 1600 data points were evaluated. (**b**) Effect of sequence length ranging from 50 to 150 under the fixed-Hamming distance (= 18) between the initial and the goal structure. A total of 1100 data points were evaluated. (**c**) Evaluation of a relationship between the frequency of energy calculation and its computation time. A total of 2700 data points, comprising data used in (a) and (b), are plotted

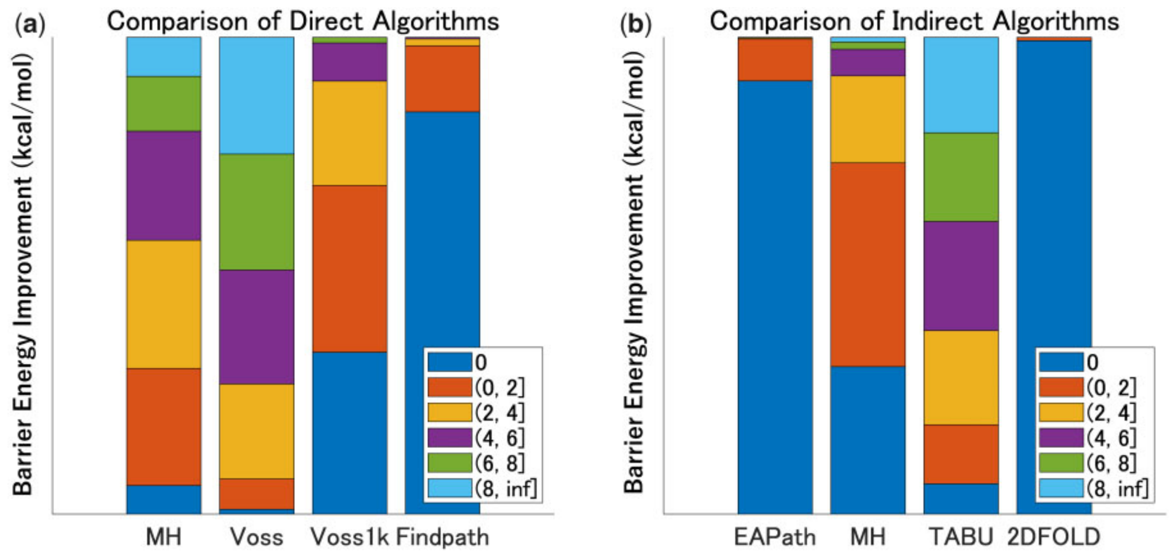

Fig. 6. Evaluation of barrier energy improvement (difference in barrier energy between an existing algorithm and our optimal algorithm). The same data used in the previous experiment (Fig. 3) are used here. (**a**) Barrier energy improvement for existing direct approximate algorithms: MH is the direct greedy method by Morgan and Higgs (1998). Voss is a semi-greedy method developed by Voss *et al.* (2004) and Dotu *et al.* (2010); semi-greedy parameter $k = 10$. Voss1k summarizes the results of the 'Voss' algorithm conducted 1000 times, with the best structure selected. Findpath is a beam-search method developed by Flamm *et al.* (2001); beam-breadth parameter $k = 10$. (**b**) Barrier energy improvement for existing indirect algorithms: RNAEAPath (Li and Zhang, 2012), MH1998Indirect (Morgan and Higgs, 1998), RNATABUPATH (Dotu *et al.*, 2010). The unit is kcal/mol, which is the RNAeval default

give an exact solution. Accordingly, we experimented with comparing the solution barrier values obtained using the previous methods and the proposed method. According to Figure 6a, in computing direct pathways, the prior approximate methods often return solutions worse than the optimal one, even at short distances of $|H| \leq 20$. This result indicates that the proposed method has a substantial advantage.

3.5 Improvement of a previous method for indirect pathways

The proposed Algorithm 6 accepts an (in)direct pathway and tries to improve it. To evaluate this method, we obtained the outcomes of three previous algorithms and input them into Algorithm 6. Figure 6b shows the results. Our algorithm improved the majority of outcomes of MH1998Indirect (Morgan and Higgs, 1998) and RNATABUPATH (Dotu *et al.*, 2010). In contrast, we were mostly unable to improve upon the results of RNAEAPath (Li and Zhang, 2012) and RNA2DFOLD (Lorenz *et al.*, 2009). As shown in Table 1, for almost all data points, RNA2DFOLD (specifically, the RNA2DFOLD-based Algorithm 7) obtained the best result. The other methods can often obtain an equivalent result, but they seldom obtain a better result than RNA2DFOLD, with the exception of only six cases.

effectiveness of the RNA2DFOLD-based Algorithm 7 can be explained more fully. In the Morgan and Higgs's indirect approximate method (specifically, the variant presented by Dotu *et al.*, 2010), 'anchor structures' are first sampled from a Boltzmann distribution and then connected them using the direct approximate method; finally, Dijkstra's algorithm is performed on the generated graph. Thus, Algorithm 7 can be interpreted as using RNA2DFOLD's local MFE structures instead of sampled ones. In that sense, it is a natural improvement upon the original design.

3.6 Real data experiment

To demonstrate our improvement to the algorithm, we analyzed the empirical dataset published by Dotu *et al.* (2010). The results are summarized in Table 2, which shows that our proposed algorithm did not improve upon the RNA2DFOLD-based method for indirect approximate pathways. Nevertheless, for one case, the original method could not finish the calculations due to its computational intensity. In such cases, combining another method and the proposed improved method is useful.

3.7 Exact direct barrier position versus RNA2DFOLD's decomposition

When we have σ_1 and σ_2 as representative structures of two meta-stable regions, there are two ways to determine the radius of each region. One way is to use RNA2DFOLD. When one uses RNA2DFOLD's decomposition into two dimensions with σ_1 and σ_2 as references, the direct pathways between σ_1 and σ_2 are associated with the shortest straight line in the decomposed space. RNA2DFOLD calculates local MFEs for each position; thus, we can divide the region at the position where the local MFE has the maximal energy. Another approach is to divide the region at the exact direct barrier position.

According to Supplementary Table S1, the two approaches above often yield different results. This result demonstrates that the former approach with local MFE cannot be substituted for the exact direct barrier approach.

Table 1. Based on the results summarized in Figure 6b, improved pathways were obtained for 1600 data points

Method name	EAPath	MH	TABU	2DFOLD
Number	1	2	3	4
Best method	only 1	only 2	only 3	only 4
Number of data points	4	1	1	633
Best methods	1 and 2	2 and 3	3 and 4	4 and 1
Number of data points	0	0	39	203
Best methods	1 and 3	2 and 4	1, 2 and 3	2, 3 and 4
Number of data points	0	61	0	39
Best methods	3, 4 and 1	4, 1 and 2	everyone	total
Number of data points	179	110	330	1600

Note: This table shows the number of data points for which the algorithm assigned the best (i.e. the lowest barrier energy) pathway. Of the two sub-tables, the first one simply shows the correspondence between method names and numbers. The second one indicate the number of data points in the data-set for which specific method(s) has (or have) produced the best result(s). For example in 61 of the 1600 data points, Methods 2 and 4 reported the best improved pathways (i.e. the barrier energies of the pathways reported by Methods 2 and 4 were the same, which was lower than those based on Methods 1 and 3). Note that whether the best improved pathway is optimal is unknown.

4 Conclusion

RNAs often have complex structure landscapes, and analysis of their kinetics is difficult. When there are multiple sub-optimal secondary structures, the minimum barrier energy and corresponding folding pathway of the two structures must be calculated.

In this study, we proposed a method for computing direct optimal barrier energy and its corresponding pathway. This is an exponential algorithm that works efficiently only for Hamming distance less than or equal to 20. However, it is far more efficient than the existing factorial algorithm examined in previous research (Morgan and Higgs, 1998). The proposed method expands the range in which the exact solution can be calculated. Furthermore, we assessed the previous approximation methods within the same range and found examples that do not agree with the exact solution. Additionally, we proposed a method for improving existing possibly indirect pathways. We applied our method to indirect pathways computed by previous methods and found that the effects of our improvement vary. For indirect pathways, the RNA2DFOLD-based method mostly achieved the best results, but our method improved upon its results in some cases.

When defining the boundaries between meta-stable regions, we found that RNA2DFOLD's local MFEs-based decision and direct optimal barrier pathway-based decision possibly generate different results. The exact solution enabled us to discover this phenomenon. While our synthetic dataset is based on random sampling, is not necessarily bi-stable, and does not necessarily contain meta-stable regions, our results demonstrate that the proposed methods are powerful tools to compute better folding pathways. We hope future work will apply this research to actual RNA kinetic analysis.

Table 2. Evaluation of barrier energy improvement using the real RNA dataset published by Dotu *et al.* (2010)

N	Gene name	Length	E (start)	E (end)	TABU	EA	MH	Findpath	RNA2DFOLD
0	rb1	148	−47.8	−27.0	−14.8 → **−27.0**	−26.5 → −26.9	−24.3 → −25.8	−19.3 → −19.3	**−27.0** → **−27.0**
1	rb2	113	−23.6	−19.8	−4.07 → −14.5	−12.1 → −13.6	−13.4 → −14.0	−9.5 → −9.9	**−15.8** → **−15.8**
2	rb3	141	−43.5	−31.3	−19.4 → −25.5	−22.8 → −22.8	−25.2 → −28.2	−28.4 → **−31.3**	**−31.3** → **−31.3**[a]
3	rb4	146	−43.8	−28.4	**−28.4** → **−28.4**	**−28.4** → **−28.4**	**−28.4** → **−28.4**	**−28.4** → **−28.4**	**−28.4** → **−28.4**
4	rb5	202	−47.8	−27.0	−14.8 → **−27.0**	−21.8 → −23.1	−23.5 → −25.7	−19.3 → −19.3	**−27.0** → **−27.0**
5	hok	395	−163.1	−138.6	−105.4 → −120.0	−109.1 → −113.3	−125.1 → −125.1	−128.4 → **−132.0**	[b]
6	SplicedLeaderAB	56	−8.6	−9.2	6.5 → **2.4**	2.5 → 2.5	8.3 → 6.3	**2.4** → **2.4**	**2.4** → **2.4**
7	attenuator	73	−21.9	−16.1	−7.0 → **−13.7**	**−13.7** → **−13.7**	−9.5 → −12.8	−11.4 → −11.4	**−13.7** → **−13.7**
8	s15	74	−18.1	−9.9	−9.7 → **−9.9**	**−9.9** → **−9.9**	−8.1 → **−9.9**	**−9.9** → **−9.9**	**−9.9** → **−9.9**
9	sbox_leader	247	−83.9	−80.8	−75.3 → **−79.6**	−79.5 → −79.5	−75.5 → −78.9	−79.5 → −79.5	**−79.6** → **−79.6**
10	thiM_leader	165	−42.3	−41.0	−23.0 → −33.2	−30.3 → −31.4	−27.6 → −29.3	−30.3 → −32.9	**−36.0** → **−36.0**[a]
11	ms2	73	−27.3	−27.4	−24.4 → **−25.8**	**−25.8** → **−25.8**	−25.3 → **−25.8**	**−25.8** → **−25.8**	**−25.8** → **−25.8**
12	HDV	153	−65.7	−67.2	−41.0 → −50.6	−44.1 → −47.0	−42.3 → −42.5	−44.2 → −44.2	**−52.1** → **−52.1**
13	dsrA	85	−31.9	−30.6	−19.7 → −22.3	−22.3 → −22.3	−16.2 → −21.7	−22.3 → −22.3	**−23.7** → **−23.7**
14	ribD_leader	304	−93.2	−90.2	−77.6 → **−85.8**	−82.1 → −83.1	−79.2 → −84.4	−81.4 → −81.9	−85.3 → −85.3
15	amv	146	−50.3	−45.8	−38.5 → −44.0	−44.0 → −44.0	−37.3 → −40.7	−43.2 → −43.2	**−45.1** → **−45.1**
16	alpha_operon	130	−35.5	−33.7	−30.6 → **−31.8**	**−31.8** → **−31.8**	−28.8 → **−31.8**	**−31.8** → **−31.8**	**−31.8** → **−31.8**
17	HIV-1_leader	280	−95.7	−94.1	−79.9 → −89.4	−88.9 → −89.5	−86.1 → −86.1	−87.2 → −87.2	**−92.3** → **−92.3**

[a]Algorithm 10 was executed with the parameter $K = 4$. For the other cases, Algorithm 10 was executed with the default parameter, $K = 5$.

[b]RNA2DFOLD could not handle the sequence.

Note: 'E(start)' and 'E(end)' indicate the energies of the starting and ending RNA secondary structures. '→' indicates that our improvement method was applied. Bold numbers are the best score for each data point. Whether the improved value is optimal is unknown in general; however, if the improved value is equal to the higher energy value between the start and goal structure, it is always the optimal value.

Acknowledgements

The authors thank Dr Shun Sakuraba for his helpful discussions. The authors also thank members of the Artificial Intelligence Research Center, AIST and Hisanori Kiryu's laboratory for helpful discussions. Computations were partially performed on the NIG supercomputer at the ROIS National Institute of Genetics.

Funding

This work was supported by MEXT/JSPS KAKENHI Grant Numbers JP16H02484 and JP16H06279 to K.A. and JP16K16143 to J.I. and JST CREST Grant Number JPMJCR18S1 to K.A.

Conflict of Interest: none declared.

References

Chen,S.-J. and Dill,K.A. (2000) RNA folding energy landscapes. *Proc. Natl. Acad. Sci. USA*, **97**, 646–651.

Dijkstra,E.W. (1959) A note on two problems in connexion with graphs. *Numer. Math*., **1**, 269–271.

Ding,Y. and Lawrence,C.E. (2003) A statistical sampling algorithm for RNA secondary structure prediction. *Nucleic Acids Res*., **31**, 7280–7301.

Dotu,I. *et al.* (2010) Computing folding pathways between RNA secondary structures. *Nucleic Acids Res*., **38**, 1711–1722.

Dykeman,E.C. (2015) An implementation of the Gillespie algorithm for RNA kinetics with logarithmic time update. *Nucleic Acids Res*., **43**, 5708–5715.

Espah Borujeni,A. and Salis,H.M. (2016) Translation initiation is controlled by RNA folding kinetics via a ribosome drafting mechanism. *J. Am. Chem. Soc*., **138**, 7016–7023.

Flamm,C. *et al.* (2001) Design of multistable RNA molecules. *RNA (New York, N.Y.)*, **7**, 254–265.

Flamm,C. *et al.* (2002) Barrier trees of degenerate landscapes. *Zeitschrift Für Physikalische Chemie*, **216**, 155.

Gerdes,K. *et al.* (1997) Antisense RNA-regulated programmed cell death. *Annu. Rev. Genet*., **31**, 1–31.

Graves,A. (2012) Sequence Transduction with Recurrent Neural Networks. *https://arxiv.org/abs/1211.3711*.

Hart,P. *et al.* (1968) A formal basis for the heuristic determination of minimum cost paths. *IEEE Trans. Syst. Sci. Cybern*., **4**, 100–107.

Li,Y. and Zhang,S. (2012) Predicting folding pathways between RNA conformational structures guided by RNA stacks. *BMC Bioinformatics*, **13**, S5.

Lorenz,R. *et al.* (2009) 2D projections of RNA folding landscapes. In: *German Conference on Bioinformatics*, Gesellschaft für Informatik Halle-Wittenberg, Germany, pp. 11–20.

Lorenz,R. *et al.* (2011) ViennaRNA package 2.0. *Algorithms Mol. Biol*., **6**, 26.

Mañuch,J. *et al.* (2011) NP-completeness of the energy barrier problem without pseudoknots and temporary arcs. *Nat. Comput*., **10**, 391–405.

Mohri,M. (2002) Semiring frameworks and algorithms for shortest-distance problems. *J. Automata Lang. Comb*., **7**, 321–350.

Morgan,S.R. and Higgs,P.G. (1998) Barrier heights between ground states in a model of RNA secondary structure. *J. Phys. A Math. Gen*., **31**, 3153–3170.

Nagel,J. *et al.* (1999) Metastable structures and refolding kinetics in hok mRNA of plasmid R1. *RNA*, **5**, 1408–1418.

Thachuk,C. *et al.* (2010) An algorithm for the energy barrier problem without pseudoknots and temporary arcs. In: *Pacific Symposium on Biocomputing 2010*, World Scientific Publishing, Hwaii, pp. 108–19.

Voss,B. *et al.* (2004) Evaluating the predictability of conformational switching in RNA. *Bioinformatics*, **20**, 1573–1582.

Wolfinger,M.T. *et al.* (2004) Efficient computation of RNA folding dynamics. *J. Phys. A Math. Gen*., **37**, 4731–4741.