BIOINFORMATICS ORIGINAL PAPER Vol. 22 no. 11 2006, pages 1317–1324
doi:10.1093/bioinformatics/btl083

Sequence analysis

Analysis of internal loops within the RNA secondary structure in almost quadratic time

Aleksey Y. Ogurtsov[1], Svetlana A. Shabalina[1], Alexey S. Kondrashov[1] and Mikhail A. Roytberg[2,*]

[1]National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health, Bethesda, MD 20894, USA and [2]Institute of Mathematical Problems in Biology, Russian Academy of Science, Pushchino, Moscow Region, 142290, Russia

Received on February 10, 2005; revised on February 15, 2006; accepted on March 3, 2006
Advance Access publication March 16, 2006
Associate Editor: Charlie Hodgman

ABSTRACT

Motivation: Evaluating all possible internal loops is one of the key steps in predicting the optimal secondary structure of an RNA molecule. The best algorithm available runs in time $O(L^3)$, L is the length of the RNA.

Results: We propose a new algorithm for evaluating internal loops, its run-time is $O(M^*\log^2 L)$, $M < L^2$ is a number of possible nucleotide pairings. We created a software tool Afold which predicts the optimal secondary structure of RNA molecules of lengths up to 28 000 nt, using a computer with 2 Gb RAM. We also propose algorithms constructing sets of conditionally optimal multi-branch loop free (MLF) structures, e.g. the set that for every possible pairing (x, y) contains an optimal MLF structure in which nucleotides x and y form a pair. All the algorithms have run-time $O(M^*\log^2 L)$.

Availability: Executables of Afold software tool, precompiled for Linux and Windows, are available at ftp://ftp.ncbi.nlm.nih.gov/pub/ogurtsov/Afold.

Contact: MRoytberg@impb.psn.ru

Supplementary information: ftp://ftp.ncbi.nlm.nih.gov/pub/ogurtsov/Afold

1 INTRODUCTION

Knowing the optimal, i.e. the one possessing the minimal possible free energy, secondary structure of an RNA molecule is crucial for understanding RNA function (Zuker and Stiegler, 1981; Zuker and Sankoff, 1984; Hofacker *et al*., 1994; Lyngsø *et al*., 1999; McCaskill, 1990). Since the pioneering works by Tinoco *et al*. (1971, 1973), methods for computational prediction of such structures (usually, pseudoknot-free) have been improved in several ways. First, more realistic energy functions become available. While early papers (Nussinov and Jacobson, 1980) estimated free energy simply by the number of paired nucleotides, a much more detailed nearest-neighbor model (NNM, Jaeger *et al*., 1989) is used now. This model treats an RNA structure as a composition of loops of different types, i.e. stacking pairs, bulges, hairpins, internal loops and multi-branch loops (Supplementary Fig. S1). NNM includes rules which assign the energy to a loop of any of these types, with the energy of the whole structure being the sum of energies of its constituent loops. Parameters of NNM have been refined experimentally (Xia *et al*., 1998; Mathews *et al*., 1999; Zuker *et al*., 1999).

Second, a variety of target objects are sought by contemporary algorithms. Among them are the set of all base pairs that occurs in suboptimal structures (Zuker, 1989), the set of suboptimal structures (Zuker, 1989; Wuchty *et al*., 1999), partition function and probabilities of specific nucleotide pairings (McCaskill, 1990; Hofacker *et al*., 2004), and the optimal structure containing no multi-branch loops (Eppstein *et al*., 1992; Larmore and Schieber, 1991). The algorithms for finding these objects are based on the appropriate versions of dynamic programming (DP) approach. Surprisingly, analysis of internal loops, i.e. loops containing only two base pairings and two regions of unpaired nucleotides between them, turned out to be the most difficult problem. The algorithm evaluating all internal loops of an RNA molecule of length L in time $O(L^3)$ was proposed by Lyngsø *et al*. only in 1999. Earlier solutions were either $O(L^4)$ or $O(L^2*D^2)$, where D is a maximal length imposed on the internal loop size.

Searching for the optimal multi-branch loop-free (MLF) structure is closely related to evaluation of all possible internal loops. Eppstein *et al*. (1992) proposed an algorithm that finds the optimal MLF structure using a sparse dynamic programming (SDP) approach. The algorithm has run-time $O(M*\log^2 L)$ and work space $O(M)$, where M is the number of possible combinations of paired nucleotides. Since $M < L^2$, this implies $O(L^2*\log^2 L)$ time bound.

However, this elegant algorithm had no significant impact on tools for predicting RNA structures because it requires the internal loop energy to be a convex or a concave function of the sum of lengths of the two unpaired regions which constitute the loop. In contrast, energy functions used in the current most popular model, NNM, depend on both the sum and the difference of these two lengths. Also, it is often necessary to find not only the optimal MLF structure, but a set of all plausible MLF structures.

*To whom correspondence should be addressed.

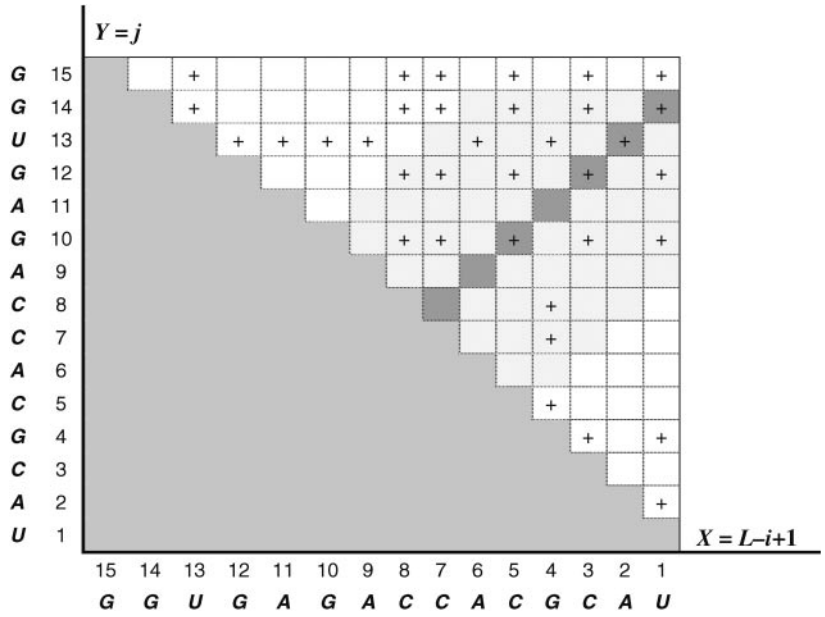

Fig. 1. Dot-matrix representation of possible base pairings within the secondary structure of an RNA molecule with sequence 'UACGCACCAGAGUGG' ($L = 15$). A putative pair (p, q) is represented by '+' at position $(L- p + 1, q)$, as if we place one copy of RNA sequence along the x-axis from right to left and the other one along the y-axis from bottom to top. The sets STRIP_r (shadowed area) and DIAG_r (dark grey cells) for $r = 15$ are highlighted. For a base pair $(A, B) =$ with $A + B = r$ the value $f_{\text{diff}}(A, B; x, y)$ differs from the base_level only if $(x, y) \in \text{STRIP}_r$.

The aim of this work is to overcome these limitations. First, we propose algorithms for evaluating internal loops. The algorithms have run-time $O(M{*}\log^2 L)$, which improves the time bound of Lyngsø *et al*. (1999) and are applicable to internal loop energy functions which conform to the NNM model, i.e. we assume that penalty F for an internal loop depends on two variables,

$$F(s, d) = f_{\text{Len}}(s) + f_{\text{Diff}}(d),$$

where length penalty $f_{\text{Len}}(s)$ is a concave function of the total number s of unpaired nucleotides in the loop and asymmetry penalty $f_{\text{Diff}}(d)$ is a function of the difference d between the lengths of two unpaired regions which constitute the loop.

The algorithms exploit the fact that $f_{\text{Diff}}(d)$ differs from a constant only at small number of points [$f_{\text{Diff}}(d) =$ const when d is large enough]. Thus, evaluation of the internal loop energy can be divided in two parts: one part corresponds to large values of d and thus we can ignore $f_{\text{Diff}}(d)$; the other relates only to narrow 'strips' corresponding to small values of d. For each of the parts we propose two algorithms calculating energy terms: one algorithm is based on SDP and implements divide and conquer procedure; the other uses candidates list paradigm and avoid divide-and-conquer procedure.

Second, we apply this approach to finding sets of conditionally optimal MLF structures and outline possible ways of using such sets.

The SDP-based algorithms are of the same order of time and space complexity as algorithm of Eppstein *et al*. (1992). For the candidate list paradigm we cannot prove the final run-time bound, however empirically it gives even better results.

The paper is organized as follows. First we describe searching for internal loops as a part of an algorithm which finds the optimal structure for an RNA molecule, according to NNM. After this, we apply our approach to develop an algorithm which finds the optimal MLF structure. Finally, we present algorithms for finding sets PAIRED and HAIRPIN of conditionally optimal MLF structures.

We shall use the following notations. We fix the RNA sequence of length L and the set U of M allowed combinations of paired nucleotides within the RNA. The paired nucleotides are represented as integer pairs (p, q); $p < q$. All sets of base pairs refer to dot-matrix representation, where the pairing (p, q) is represented by the point $(L - p + 1, q)$ on the right-upper triangle of the $L \times L$ matrix (Fig. 1). $U(A, B) \subseteq U$ denotes the set of all allowed base pairs (p, q) with $A + 3 \leq p < q \leq B - 3$. For each $B \in [1, L]$ a set ROW_B consists of all allowed base pairs (p, B). The r-th diagonal is a set of points $\text{DIAG}_r = \{(p, q) \mid (p, q) \in U, p + q = r\}$, and for some fixed parameter width the r-th strip is a union of diagonals $\text{STRIP}_r = \{(p, q) \in U \mid r - \text{width} \leq p + q \leq r + \text{width}\}$.

The optimal RNA structure within a given class of structures is the structure with the minimal possible energy. $\text{IStruct}(A, B; p, q)$ is the optimal structure in which paired nucleotides at positions A and B, p and q $(A < p < q < B)$, form an internal loop, and (A, B) is the closing pair of the structure. $\text{IStruct}(A, B)$ is the optimal structure among $\text{IStruct}(A, B; p, q)$ for all p, q.

2 ALGORITHM

2.1 Finding internal loops during construction of the optimal RNA structure

Algorithms for predicting RNA secondary structures according to NNM (Zuker and Sankoff, 1984; Hofacker *et al*., 1994; Lyngsø *et al*., 1999) follow the framework given in Supplementary algorithm S1. An algorithm looks over all possible base pairs row by row in the ascending order of their row numbers $B \in [1, L]$, calculating optimal structures of different types for each closing

base pair $(A, B) \in U, A \in [1, B-1]$. Consider the following optimal internal loop problem.

Problem 1. For all the allowed base pairs $(A, B) \in U$ find the optimal structure IStruct(A, B) with the closing pair (A, B) within the framework of the algorithm OptimalRNA, (Supplementary algorithm S1, line 2.5). We assume that energies $\Delta G(x, y)$ of optimal structures with the given closing loop pair (x, y) are already known before the call InternalLoopRow(B) for all $(x, y) \in U$ with $y < B$.

We propose an algorithm solving this problem with NNM energy functions, i.e. give an implementation of the function InternalLoopRow. We describe only how to calculate the energy of an optimal internal loop, which requires filling on the right-upper triangle of the $L*L$ matrix (Fig. 1). The subsequent reconstruction of the optimal RNA structure only involves tracing back the corresponding graph, which realistically contains only a small number of branching points, each describing a multi-branch loop, and does not significantly increase either run-time or the needed space. Thus, this reconstruction will not be described. Also, we do not consider analyses of hairpins and simple loops, i.e. stacking pairs, bulges and internal loops with small distances between the opening and closing base pairs (Mathews *et al*., 1999; see also Supplementary Fig. S1).

The energy $\Delta G_{\mathrm{IStruct}}(A, B)$ of the structure IStruct(A, B) with the given closing pair (A, B) can be determined as

$$\begin{aligned}&\Delta G_{\mathrm{IStruct}}(A,B)\\&\quad=\min\{\Delta G_{\mathrm{IStruct}}(A,B;x,y)\,|\,(x,y)\in U(A,B)\}\\&\quad=\min\{\Delta G(x,y)+\Delta G_{\mathrm{Internal\_Loop}}(A,B;x,y)\,|\,(x,y)\in U(A,B)\},\end{aligned}\tag{1}$$

where $\Delta G_{\mathrm{IStructl}}(A, B; x, y)$ is the energy for IStruct$(A, B; x, y)$. Here $\Delta G(x, y)$ is the minimal energy of all structures (including those containing multi-branch loops) with the closing pair (x, y); with the termination base pairing energy is included to $\Delta G(x, y)$. $\Delta G_{\mathrm{Internal\_Loop}}(A, B; x, y)$ is a penalty for two unpaired regions $[A+1, x-1]$ and $[y+1, B-1]$, with lengths $d_A = x - A - 1$ and $d_B = B - y - 1$, respectively. The penalty function $\Delta G_{\mathrm{Internal\_Loop}}(A, B; x, y)$ is defined as

$$\begin{aligned}&\Delta G_{\mathrm{Internal\_Loop}}(A,B;x,y)\\&\quad=f_{\mathrm{Len}}(dA+dB)+f_{\mathrm{Diff}}(dA-dB)\\&\quad=f_{\mathrm{Len}}((B-A)-(y-x+2))+f_{\mathrm{Diff}}((B+A)-(y+x)).\end{aligned}\tag{2}$$

The function $f_{\mathrm{Len}}(s)$ is convex (Supplementary Fig. S2), and can be approximated by a logarithmic function. The function $f_{\mathrm{Diff}}(d)$ is defined as

$$\begin{aligned}f_{\mathrm{Diff}}(d)&=\text{base\_level},\quad \text{if}\,|\,d\,|\,\geq\text{width};\\f_{\mathrm{Diff}}(d)&=(\text{base\_level}/\text{width})*|\,d\,|,\quad \text{if}\,|\,d\,|<\text{width}.\end{aligned}\tag{3}$$

Here, base_level determines the value $f_{\mathrm{Diff}}(d)$ for large enough values of $|\,d\,|$, and width determines the range of values of d, where $f_{\mathrm{Diff}}(d)$ is not equal to base_level. Thus, for all d, $f_{\mathrm{Diff}}(d) \leq$ base_level and $f_{\mathrm{Diff}}(d)$ differs from base_level only in $2*\text{width} - 1$ points, i.e. for $d = -\text{width}+1, \ldots, 0, \ldots, \text{width}-1$. In the current version of NNM, the values of the constants are base_level $= +3.00$, width $= 6$. Thus, $2*\text{width} - 1 = 11$. We will exploit the fact that this value is small relative to the total RNA length L. Let $\Delta G_r(x, y)$ be the sum

$$\Delta G_r(x,y)=\Delta G(x,y)+f_{Diff}(r-(y+x)).\tag{4}$$

Using (2)–(4), we can transform (1) as follows

$$\begin{aligned}&\Delta G_{\mathrm{IStruct}}(A,B)=\min\{\\&\quad\min\{\Delta G(x,y)+f_{\mathrm{Len}}((B-A)-(y-x+2))+\text{base\_level}\\&\qquad\,|\,(x,y)\in U(A,B)\},\\&\quad\min\{\Delta G(x,y)+f_{\mathrm{Len}}((B-A)-(y-x+2))+\\&\qquad f_{\mathrm{Diff}}((B+A)-(y+x))\,|\,(x,y)\in \mathrm{STRIP}_{A+B}\},\\&\}=\min\{\\&\quad\text{base\_level}+\ \min\{\Delta G(x,y)+f_{\mathrm{Len}}((B-A)-(y-x+2))\\&\qquad\qquad\,|\,(x,y)\in U(A,B)\},\\&\quad\min\{\Delta G_{A+B}(x,y)+f_{\mathrm{Len}}((B-A)-(y-x+2))\\&\qquad\,|\,(x,y)\in \mathrm{STRIP}_{A+B}\}.\\&\}\end{aligned}\tag{5}$$

Proceeding the ROW_B, our algorithm first calculates

$$\begin{aligned}\Delta G_{\mathrm{Main}}(A,B)=\min\{&\Delta G(x,y)+f_{\mathrm{Len}}((B-A)\\&-(y-x+2))\,|\,(x,y)\in U(A,B)\}\end{aligned}\tag{6}$$

for all $(A, B) \in \mathrm{ROW}_B$. After this,

$$\begin{aligned}\Delta G_{\mathrm{Strip}}(A,B)=\min\{&\Delta G_{A+B}(x,y)+f_{\mathrm{Len}}((B-A)\\&-(y-x+2))\,|\,(x,y)\in \mathrm{STRIP}_{A+B}\}\end{aligned}\tag{7}$$

is calculated, and finally the desired minima (5) are found.

Therefore, function InternalLoopRow(B) is computed as shown in Supplementary algorithm S2. InternalLoopMainRow(B) calculates values of $\Delta G_{\mathrm{Main}}(A, B)$ for all $(A, B) \in \mathrm{ROW}_B$. InternalLoopStripRow$(B)$ calculates values of $\Delta G_{\mathrm{Strip}}(A, B)$, and InternalLoopFinalRow(B) calculates $\Delta G_{\mathrm{IStruct}}(A, B)$ according to (5). InternalLoopMainRow(B) and InternalLoopStripRow(B) also support the necessary data structures described below.

Calculating (6) and (7), we exploit convexity of the function $f_{\mathrm{Len}}(s)$. Note that each base pair (x, y) belongs only to $2*\text{width} - 1$ strips. Therefore the total time to calculate values $\Delta G_{A+B}(x, y)$ for all (A, B) and $(x, y) \in \mathrm{STRIP}_{A+B}$ is

$$T_{\mathrm{DIAG}} = c_1 * \text{width} * M.\tag{8}$$

To calculate (6), we can use an appropriate modification of the SDP algorithm of Eppstein *et al*. (1992). The total run time of the algorithm is

$$T_{\mathrm{MAIN}} = c_2 * M * \log^2 L\tag{9}$$

and the additional space needed is $O(M)$.

Another way to calculate (6) is to store candidate lists, i.e. for each $x \in [1, L]$ to store values $\Delta G(x, y)$ for all pairs (x, y) that can give maximum in (6) for some (A, B) with B greater than the current row number B^*. The naive upper bound for this case is $O(M^2)$. However, our experiments show that each list contains in average 2–3 items (26 in worth case) for L from 1 500 to 17 000 that (in combination with owner technique analogous to SDP) leads to empirical run-time bound

$$T_{\mathrm{MAIN\_EMP}} = c_{2\_\mathrm{emp}} * L^2 * \log L$$

with very low constant c_{2_emp} (section 3 'Implementation', Fig. 3).

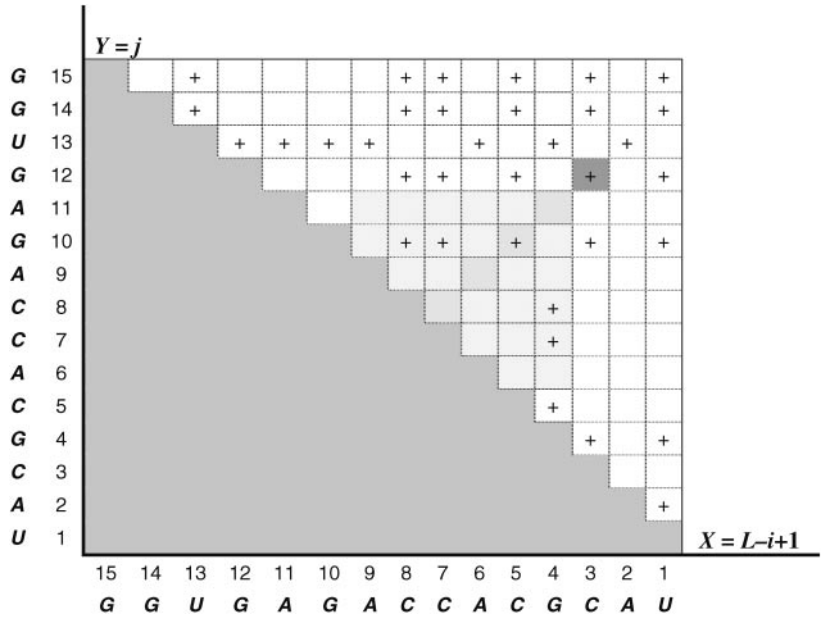

Fig. 2. The set $\mathrm{STRIPPRED}_r(B)$ (shadowed) for $B = 12$ and $r = 15$.

To calculate (7) we can also use the SDP algorithm that leads to the aggregate run-time

$$T_{\mathrm{STRIP}} = c_3 * \mathrm{width} * M * \log^2 L \tag{10}$$

Another possibility is to take advantage of the structure of the set DIAG_r of base pairs sharing the same strip STRIP_r and avoid the divide and conquer procedure. This gives a better time

$$T_{\mathrm{STRIP}} = c_4 * \mathrm{width} * M * \log L \tag{11}$$

at the cost of slightly larger used space. However, in both cases the space needed is $O(M)$. The SDP-based algorithms to calculate (6) and (7) are given in the Supplementary Section 1; the algorithms will be referred to as E-algorithm and ES-algorithms below. The candidate list based algorithms are described in the next sub-section.

Overall [(8)–(11)], we obtain run-time upper bound $O(\mathrm{width}{*}M{*}\log^2 L)$ if we use the SDP algorithms both to calculate (6) and (7), and $O(M{*}\log^2 L + \mathrm{width}{*}M{*}\log L)$ if we do not implement the divide and conquer approach to calculate (7). In both cases the space needed is $O(M)$.

2.2 Functions InternalLoopStripRow, InternalLoopMainRow and candidate lists preliminary remarks

Here we present the algorithms to calculate values $\Delta G_{\mathrm{Strip}}(A, B)$ (the G-algorithm) and $\Delta G_{\mathrm{Main}}(A, B)$ (the M-algorithm), that use global candidate lists and unlike the SDP avoid divide-and-conquer procedure.

First, we consider the algorithm calculating $\Delta G_{\mathrm{Strip}}(A, B)$, the G-algorithm. The G-algorithm is $\log L$ times faster than ES-algorithm. The work space of G-algorithm is $O(\mathrm{width}{*}M)$, i.e. of the same order as of ES-algorithm; the work space of G-algorithm exceeds that of ES-algorithm only by a term proportional to L (cf. Hirschberg, 1975). Then we consider the algorithm calculating $\Delta G_{\mathrm{Main}}(A, B)$, the M-algorithm.

Fix a row $B \in [1, L]$ and a diagonal r, $r > B$. Let us define the set of base pairs $\mathrm{STRIPPRED}_r(B)$ as follows (Fig. 2):

$$\mathrm{STRIPPRED}_r(B) = \{(x, y) \mid (x, y) \in \mathrm{STRIP}_r \ \&\ r - B < x < y < B\}.$$

The set $\mathrm{STRIPPRED}_r(B)$ consists of all base pairs that can be related to $(r - B, B)$ within the calculation of $\Delta G_{\mathrm{Strip}}(r - B, B)$ according to (7).

$$\mathrm{Let}(p, q) \in \mathrm{DIAG}_r, q \geq B, r = p + q.$$

We define $\mathrm{OWNER}_{r,B}(p, q) \in \mathrm{STRIPPRED}_r(B)$ as a minimal element according to the following ordering:

$(x, y) <_{\otimes} (x', y')$ if
(a) $\Delta G_{\mathrm{IStruct}}(p, q; x, y) < \Delta G_{\mathrm{IStruct}}(p, q; x', y')$, or
(b) $\Delta G_{\mathrm{IStruct}}(p, q; x, y) = \Delta G_{\mathrm{IStruct}}(p, q; x', y')$ & $y - x < y' - x'$, or
(c) $\Delta G_{\mathrm{IStruct}}(p, q; x, y) = \Delta G_{\mathrm{IStruct}}(p, q; x', y')$ & $y - x = y' - x'$ & $y < y'$.

We say that $(x, y) \in \mathrm{STRIPPRED}_r(B)$ is an (r, B)-candidate if

$$(x, y) = \mathrm{OWNER}_{r,B}(p, q)$$

for some $(p, q) \in \mathrm{DIAG}_r$, $q \geq B$. By definition, for any $(A, B) \in \mathrm{ROW}_B$

$$\Delta G_{\mathrm{IStruct}}(A, B) = \Delta G_{\mathrm{IStruct}}(A, B; u, v),$$

where $(u, v) = \mathrm{OWNER}_{A+B,B}(A, B)$. The idea of the G-algorithm is to store lists $\mathrm{CAND}_{r,B}$ of (r, B)-candidates and update them from $\mathrm{CAND}_{r,B}$ to $\mathrm{CAND}_{r,B+1}$ within the processing of ROW_B.

Let (r, B)-home $\mathrm{HOME}_{r,B}(x, y)$ of the (r, B)-candidate (x, y) be a set of all q such that $B \leq q < r$ and $(x, y) = \mathrm{OWNER}_{r,B}(r - q, q)$.

Statement 1. Let $(x_1, y_1), \ldots, (x_n, y_n)$ be all (r, B)-candidates, ordered by decrement of $d_i = y_i - x_i$. Then,

(1) All values $y_1 - x_1, \ldots, y_n - x_n$ are different, i.e. $y_1 - x_1 > \cdots > y_n - x_n$;
(2) $\Delta G_r(x_1, y_1) > \cdots > \Delta G_r(x_n, y_n)$;

(3) there are values $T_0 = B - 1 < T_1 < \cdots < T_{n-1} < T_n = L$ such that $\mathrm{HOME}_{r,B}(x_i, y_i) = [T_{i-1} + 1, T_i]$.

Proof. In Supplementary Section 3.

2.2.1 Description of G-algorithm Let $\mathrm{CAND}_{r,B}$ be a list of 5-tuples

$$\{C_i = < C_i.x, C_i.y, C_i.dG, C_i.Tmin, C_i.Tmax > \}, i = 1, \ldots, n,$$

where $(x_i, y_i) = (C_i.x, C_i.y)$ are all (r,B)-candidates; $y_1 - x_1 > \cdots > y_n - x_n$; $C_i.dG = \Delta G_r(x_i, y_i)$; $[C_i.T\mathrm{min}, C_i.T\mathrm{max}] = \mathrm{HOME}_{r,B}(x_i, y_i)$, i.e. $C_i.T\mathrm{min} = T_{i-1}+1$, $C_i.T\mathrm{max} = T_i$; T_i are defined in Statement 1.

G-algorithm utilizes lists $\mathrm{CAND}_{r,B}$; to store the lists the algorithm uses the array VAR_CANDIDATESTRIP of length $2*L - 1$. Before processing of $\mathrm{ROW_B}$, an element VAR_CANDIDATESTRIP[r] contains the set $\mathrm{CAND}_{r,B}$. Unlike the array VAR_ACTIVESTRIP, which is local, i.e. is reinitialized at each call of InternalLoopStrip_ES(B), the array VAR_CANDIDATESTRIP is the global data structure. This forces us to change the framework of OptimalRNA algorithm (Supplementary algorithm S1) to the modified algorithm OptimalRNA_G (Supplementary algorithm S3A). The elements of VAR_CANDIDATESTRIP are initialized with empty lists only once during the preprocessing step of the algorithm OptimalRNA_G.

The implementation of the G-algorithm consists of two functions: InternalLoopStripRow_G (Supplementary algorithm S3B) and UpdateCandidate. Analogously to InternalLoopStripRow, it stores the calculated value $\Delta G_{\mathrm{Strip}}(A, B)$ in VAR_STRIPWORK[A, B]. The function GetFirst(r) gets the first element $C = (C.x, C.y, C.dG, C.T\mathrm{min}, C.T\mathrm{max})$ of the list VAR_CANDIDATESTRIP[r] = $\mathrm{CAND}_{r,B}$. According to Statement 1, $(C.x, C.y) = \mathrm{OWNER}_{r,B}(r - B, B)$. Thus, the value $\Delta G_{\mathrm{Strip}}(A, B)$, where $A = r - B$, can be calculated as

$$\Delta G_{Strip}(A,B) = \Delta G_{A+B}(C.x, C.y) + f_{Len}((B-A) - (C.y - C.x + 2))$$

The function UpdateCandidate is presented in Supplementary algorithm S4, its subroutine IncludeCandidate is given in Supplementary algorithm S5. A call UpdateCandidate(B) updates lists VAR_CANDIDATESTRIP[r] from $\mathrm{CAND}_{r,B}$ to $\mathrm{CAND}_{r,B+1}$ according to newly calculated values $\Delta G(A, B)$ for $(A, B) \in \mathrm{ROW}_B$.

Statement 2.

(1) The function InternalLoopStripRow_G(B) correctly calculates values $\Delta G_{\mathrm{Strip}}(A, B)$ within the algorithm OptimalRNA_G.

(2) The total run-time of calls InternalLoopStripRow_G(B) and UpdateCandidate(B) within the work of the algorithm OptimalRNA_G (Supplementary algorithms S3–S5) is $O(\mathrm{width}*M*\log L)$, $f_{\mathrm{Len}}(s)$ is assumed to be a convex function, its value can be calculated in constant time.

(3) If $f_{\mathrm{Len}}(s)$ is logarithmic function, then the total run-time of calls InternalLoopStripRow_G(B) and UpdateCandidate(B) within the work of the algorithm *OptimalRNA*_G (Supplementary algorithms S3–S5) is $O(\mathrm{width}^2*M)$, the value $f_{\mathrm{Len}}(s)$ is supposed to be calculated in constant time.

(4) The workspace of the OptimalRNA_G is $O(L) + O(\mathrm{width}*M)$.

Proof. In Supplementary Sections 3 and 4.

2.2.2 The M-algorithm The M-algorithm is based on the observation, that the $\Delta G(A, B)$ has a negative correlation with $B - A$. In other words, we expect better (lower) ΔG for longer sequences. Thus, for every A ($A \in [1, B]$) we construct and maintain the list of base pairs (A, Y) that have ascending order by Y and $\Delta G(A, Y)$. Using the fact that function $f_{\mathrm{Len}}(s)$ increases monotonously, it is easy to prove the following statement.

Statement 2*M*. If $A < x < y_1 < y_2 < B$ and $\Delta G(x, y_1) > \Delta G(x, y_2)$, then $\Delta G_{\mathrm{Internal\_Loop}}(A, B; x, y_1) > \Delta G_{\mathrm{Internal\_Loop}}(A, B; x, y_2)$ for every pair (A, B). Indeed,

$$\begin{aligned}\Delta G_{\mathrm{Internal\_Loop}}(A,B;x,y_1) &= \Delta G(x,y_1) + f_{\mathrm{Len}}((B-A) - (x - y_1 + 2)) \\ &> \Delta G(x,y_1) + f_{\mathrm{Len}}((B-A) - (x - y_2 + 2)) \\ &> \Delta G(x,y_2) + f_{\mathrm{Len}}((B-A) - (x - y_2 + 2)) \\ &= \Delta G_{\mathrm{Internal\_Loop}}(A,B;x,y_2).\end{aligned}$$

We say that base pair (x, y) where $x < y < B$ is B-weak if there is $y' < B$ such that $y' > y$ and $\Delta G(x, y') < \Delta G(x, y)$; otherwise the base pair (x, y) is called B-strong. The above statement shows, that weak base pairs (i.e. base pairs with smaller lengths and higher ΔG) can be excluded from evaluation for all (A, B). For each x we construct and maintain candidate lists consisting of all B-strong pairs (x, y). Our tests show that for sequence of 1 500–17 000 nt a candidate list contains on average 2 pairs and is never longer than 26 pairs (Supplementary Table S1).

Analogous to G-algorithm, the M-algorithm works out all allowed base pairs row by row. According to Statement 2M, to proceed a row B we have to look out only $\sim 2*B$ base pairs that are members of the above candidate lists. This can be done using SDP-like technique in time $O(L*\log L)$ that leads to the total run-time $O(L^2*\log L)$.

2.3 Multi-branch loop-free structures

2.3.1 Optimal multi-branch loop-free structures The MLF structure is a structure containing only hairpins, stacking pairs, bulges and internal loops. According to the NNM, the energy $\Delta G_{\mathrm{MLF}}(A, B)$ of the optimal MLF structure with a given closing pair (A, B) can be found from the recursive equation:

$$\begin{aligned}\Delta G_{\mathrm{MLF}}(A,B) = \min\{&\Delta G_{\mathrm{Hairpin}}(A,B), \Delta G_{\mathrm{Simple}}(A,B), \\ &\min\{\Delta G_{\mathrm{MLF}}(x,y) + \Delta G_{\mathrm{Internal\_Loop}}(A,B;x,y) \\ &\quad | \, (x,y) \in U(A,B)\}\}.\end{aligned} \tag{12}$$

Here $\Delta G_{\mathrm{Hairpin}}(A, B)$ is the energy of the hairpin loop with the closing pair (A, B), $\Delta G_{\mathrm{Simple}}(A, B)$ is the minimal energy of a structure with the closing pair (A, B), where (A, B) closes a 'simple loop' i.e. a stacking pair, or a bulge, or an internal loop where one of the unpaired fragments has 1 or 2 nt. The last term in the formula (12) corresponds to internal loops of general form. The $\Delta G_{\mathrm{Internal\_Loop}}(A, B; x, y)$ is given by formula (2), the function $f_{\mathrm{Diff}}(d)$ differs from constant only if

$$|d| < \mathrm{width}, \tag{13}$$

here width is a constant. For the NNM, width = 6. Thus, we come to the Problem 2.

Problem 2 *(OMLF problem)*. Given: an RNA sequence of length L and a set of allowed base pairs U of size M.

The goal: Find the optimal MLF secondary structure according to the recursive Equation (12) and the $\Delta G_{\text{Internal\_Loop}}$ function meeting conditions (2) and (13).

Let the algorithms MLF_E and MLF_G be the algorithms obtained from the algorithms OptimalRNA (Supplementary algorithm S1) and OptimalRNA_G (Supplementary algorithm S3A) respectively by deletion of the lines related to multi-branch loops (line 2.4). The algorithm MLF_E in detail is given in the Supplementary Section 6.

The following statement generalizes the result of Eppstein *et al*. (1992).

Statement 3. The algorithms MLF_E and MLF_G solve the OMLF problem. The run-times of the algorithms are $O(M*\text{width}*\log^2 L)$ and $O(M*\log L*(\text{width} + \log L))$ respectively and the work space are $O(M*\text{width})$.

Proof follows from the previous Algorithm sections.

2.3.2 Conditionally optimal multi-branch loop-free structures In the course of their work, algorithms MLF_E and MLF_G, for each base pair $(A, B) \in U$, find the MLF structure MLF_Close(A, B) that is optimal among the structures with the closing pair (A, B). Below, we consider other types of 'conditionally' optimal MLF structures.

Let MLF_Hairpin(A, B) be the optimal MLF structure having (A, B) as the most internal loop ('hairpin' base pair), i.e. (A, B) is a closing pair of the only hairpin of the structure. Let MLF_*BP* be the optimal MLF structure containing the base pair (A, B).

Problem 3. *(OMLF_H problem)*. Given: an RNA sequence of length L and a set of allowed base pairs U of size M.

The goal: For each allowed base pair (A, B) find the conditionally optimal structure MLF_Hairpin(A, B) according to the recursive Equation (12) and the Penalty function meeting conditions (2) and (13).

Problem 4. *(OMLF_BP problem)*. Given: an RNA sequence of length L and a set of allowed base pairs U of size M.

The goal: For each allowed base pair (A, B) find the conditionally optimal structure MLF_BP(A, B) according to the recursive Equation (12) and the Penalty function meeting conditions (2) and (13).

Below we present algorithms solving Problems 3 and 4 with the same time and space complexities as that of algorithms MLF_E and MLF_G.

2.3.3. Conditionally optimal MLF structures with a given hairpin loop

Statement 4. Suppose that function $\Delta G^R(x, y)$, $(x, y) \in U$ meets the following recursive equation [definition of $\Delta G_{\text{Internal\_Loop}}(A, B; x, y)$ in (2), sub-section 2.1]

$$\begin{aligned}\Delta G^R(x,y) = \min\{0,\\ &\min\{\Delta G^R(A,B) + \Delta G_{\text{Simple}}(A,B;x,y) \mid (x,y)\\ &\in U(A,B);(A,B) \text{ and } (x,y)\text{form a simple loop}\},\\ &\min\{\Delta G^R(A,B) + \Delta G_{\text{Internal\_Loop}}(A,B;x,y) \mid (x,y)\\ &\in U(A,B);(A,B)\text{and}(x,y)\\ &\quad\text{form a general internal loop}\}\}. \qquad (14)\end{aligned}$$

Then, the energy $\Delta G^R{}_{\text{Hairpin}}(x, y)$ of the structure MLF_Hairpin(x, y) can be found from

$$\Delta G^R_{\text{Hairpin}}(x,y) = \Delta G^R(x,y) + \Delta G_{\text{Hairpin}}(x,y),$$

where $\Delta G_{\text{Hairpin}}(x, y)$ is the energy of the hairpin loop with the closing pair (x, y).

Proof. It follows from the definition of the energy of an RNA structure in the NNM model. Informally, $\Delta G^R(x, y)$ is the optimal energy of the exterior part of the MLF structure containing the pairing (x, y), i.e. of the part outside of the pairing (x, y). In turn, the value $\Delta G_{\text{MLF}}(A, B)$ is the optimal energy of the interior part. The recursion (12) considers base pairs (p, q) of a MLF structure from the pairs with the minimal distance $q - p$ to the pairs with the maximal distance. The recursion (14) goes in the opposite direction. Thus, the desired proof can be obtained by induction.

Statement 5. There are modifications MLF_EH and MLF_GH of the algorithms MLF_E and MLF_G that solve the OMLF_Hairpin problem with the same time and space complexity as initial algorithms solve the MLF problem.

Proof. For the sake of brevity, we restrict ourselves with informal explanations. The recursion (14) is the 'reversed' recursion for the recursion (12), see statement 4. Therefore, we can reduce the computation of $\Delta G^R(x, y)$ to the computation of $\Delta G^R{}_{\text{Main}}(x, y)$ and $\Delta G^R{}_{\text{Strip}}(x, y)$, which are analogous to $\Delta G_{\text{Main}}(x, y)$ and $\Delta G_{\text{Strip}}(x, y)$. The values $\Delta G^R{}_{\text{Main}}(x, y)$ and $\Delta G^R{}_{\text{Strip}}(x, y)$ in turn can be computed using appropriate modifications of E-algorithm, ES-algorithm, and G-algorithm.

The relation between recursions (12) and (14) can be illustrated with mappings of the set of allowed pairs U into the square $[1, L]*[1, L]$. The mappings map base pairs of a given MLF structure into the increasing chain of points. The recursion (12) corresponds to the mapping

$$\gamma(p,q) = (L - p + 1, q),$$

i.e. the nucleotide pair (p, q) is represented with the point $(x, y) = (L - p + 1, q)$ (Supplementary Fig. S3). All base pairs thus correspond to the points of the upper-right triangle; the distance $q - p$ corresponds to the distance from the point $\gamma(p, q)$ to the diagonal. The MLF structure, consisting of the base pairs $(p_1, q_1), \ldots, (p_n, q_n)$, $p_n < \cdots < p_1 < q_1 < \cdots < q_n$, corresponds to the increasing chain of points $\gamma(p_1, q_1) = (p_1, q_1), \ldots, \gamma(p_n, q_n) = (x_n, y_n)$, where

$$x_1 < \ldots < x_n, \text{ and } y_1 < \ldots < y_n$$

The last point of the chain corresponds to the closing pair of the MLF structure.

The recursion (14) corresponds to the mapping

$$\gamma^R(p,q) = (p, L - q + 1)$$

It maps nucleotide pairs into the bottom-left triangle. An increasing chain corresponds to a MLF structure, considered in the opposite direction, i.e. from the most distant base pairs to the closest ones. The increasing chain $\{(x_1, y_1), \ldots, (x_n, y_n)\}$ corresponds to the MLF with the closing pair $(x_1, L - y_1 + 1)$ and the hairpin pair $(x_n, L - y_n + 1)$.

Thus, in both cases we can reduce the search for the optimal MLF structure to the search for the least weight increasing chain (Eppstein *et al*., 1992).

2.3.4 Locally optimal multi-branch loop-free structures containing a given base pair Solution of the Problem 4 is based on the following statement.

Statement 6. Let $(A, B) \in U$ and $\{(x_1, y_1), \ldots, (A, B), \ldots, (x_n, y_n)\}$ be an optimal MLF structure containing the base pair (A, B), $x_1 > \cdots > A > \cdots > x_n$, and $y_1 < \cdots < B < \cdots y_n$, then $\{(x_1, y_1), \ldots, (A, B)\}$ is an optimal MLF structure with the given

closing pair (A, B) and $\{(A, B), \ldots, (x_n, y_n)\}$ is an optimal MLF structure with the given hairpin pair (A, B).

Proof. It follows from the above definitions and the recursive Equations (12) and (14).

Statement 7. There are algorithms MLF_E_BP and MLF_G_BP that solve the OMLF_BP problem with the same time and space complexity as the algorithms MLF_E and MLF_G, respectively, solve the MLF problem.

Proof. It follows from the Statements 3, 5 and 6. We first find conditionally optimal structures MLF_Close(A, B), $(A, B) \in U$ by the algorithm MLF_E (or MLF_G). Then we find structures MLF_Hairpin(A, B), $(A, B) \in U$ by the algorithm MLF_EH (or MLF_GH). Finally, we obtain the desired MLF structures MLF_BP(A, B) combining MLF_Close(A, B) and MLF_Hairpin (A, B) according to the Statement 6.

3 IMPLEMENTATION

We implemented software tool Afold, freely available as a C/C++ code, currently precompiled under Linux and Windows, which computes the optimal RNA secondary structure within the framework of NNM. To evaluate internal loops, Afold uses candidate lists-based algorithms (Subsection 2.2). We performed extensive comparison of the performance of Afold with that of Mfold (Zuker, 2003), and ZUKER (Lyngsø *et al*., 1999) and demonstrated that Afold clearly outperforms the other two. Our algorithm constructs internal loops faster than Mfold and ZUKER (Fig. 3A). Afold and Mfold require similar time to fill the whole energy matrix (Fig. 3) including multi-branch loop evaluation (ZUKER is much slower at this step). However, Mfold artificially limits the length of internal loops by 30 nt. In contrast, there is no such limitation in Afold.

Also, Afold uses the same matrix to evaluate internal and multi-branch loops and, as a result, requires memory only $\sim 2.5L^2$. Thus, on a computer with 2 GB of RAM, Afold can analyze sequences with length up to 28 000 nt (in $\sim$28 h on a regular PC), which is substantially longer than what is allowed by other software tools (Fig. 3B).

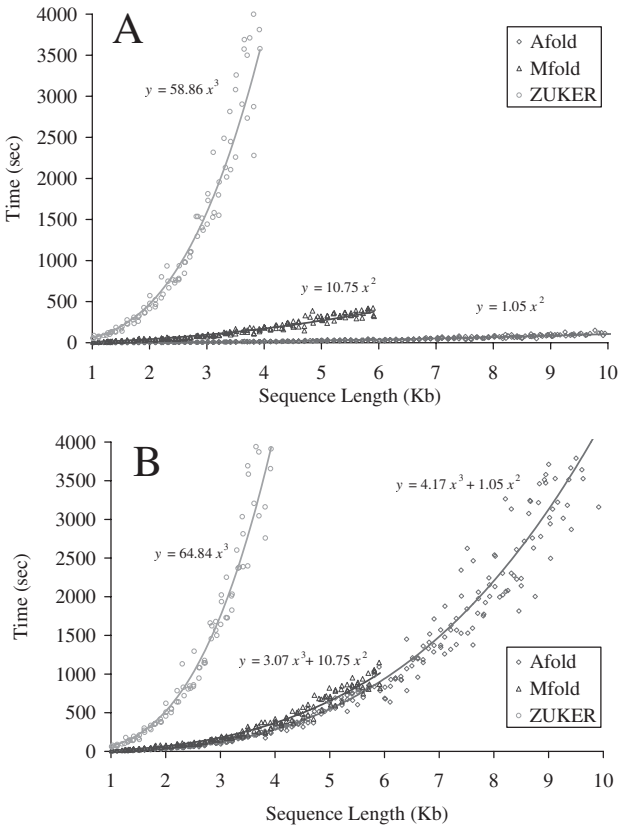

Fig. 3. Performance of Afold, Mfold (Zuker, 2003) and ZUKER (Lyngsø *et al*., 1999) on a PC with 2.6 GHz processor and 2 Gb of RAM. We determined the optimal secondary structure for 270 mRNA sequences from the human genome, of lengths from 1000 to 10 000 (30 sequences for each thousand). (**A**) displays the time required for determining MLF structures and (**B**) displays time required for the construction of optimal secondary structures which may include multi-branch loops.

4 DISCUSSION

The currently available software tools for predicting secondary structures of RNA molecules, e.g. Mfold (Zuker, 2003) and Vienna (Hofacker *et al*., 1994), are based on algorithms with run-time $O(L^3)$. While adequate to analyze individual RNAs, they may be too slow for genome-wide studies. Thus, faster algorithms are of interest, even if they can solve only restricted versions of the problem. One such algorithm, (Eppstein *et al*. 1992) exploits SDP and inspired our work.

First, we have proposed a novel method to evaluate internal loops with NNM energy functions, where the energy of an internal loop depends both on its size and its asymmetry (Mathews *et al*., 1999). The adjustment is based on the following observations: (a) the calculation of minimal possible energy $\Delta G_{\mathrm{IStruct}}(A, B)$ in (5) can be reduced to the independent calculation of $\Delta G_{\mathrm{Main}}(A, B)$ and $\Delta G_{\mathrm{Strip}}(A, B)$ according (6) and (7); (b) the value $\Delta G_{\mathrm{Main}}(A, B)$ can be found using SDP (E-algorithm) or using candidate lists (M-algorithm); (c) the value $\Delta G_{\mathrm{Strip}}(A, B)$ can be computed by the candidate-list algorithm (G-algorithm, Subsection 2.2). Our algorithm evaluates internal loops during the search for the optimal RNA secondary structure with $O(L^2 * \log^2 L)$ run-time, which improves the result of Lyngsø *et al*. (1999). Moreover, we can calculate $\Delta G_{\mathrm{Main}}(A, B)$ applying the algorithm of Larmore and Schieber (1991) and thus obtain an even better run-time bound.

Second, we proposed algorithms to compute all conditionally optimal MLF structures in RNA (section 2.3). The energy of MLF structure can be found both by classical recursive equation (12) and by a reversed equation (14). The latter equation corresponds to the computation of the energy 'outside-in' in contrast to the 'inside-out' computation according to the equation (12). This observation results in effective algorithms, allowing us to find sets of conditionally optimal MLF structures. Our algorithm is significantly faster than the currently known algorithms, and may be adequate for genome-wide studies.

Knowing the complete set of conditionally optimal MLF structures for an RNA molecule does not provide comprehensive information about its secondary structure. However, for some biological problems, such knowledge is very helpful. Two possible examples are as follows. First, the presence of a low-energy putative MLF structure within a genome fragment can serve as a sign of a non-coding RNA gene. Second, information about locally optimal MLF structures can be used to predict unpaired RNA regions. The last problem is of great interest because of the accumulating experimental evidence that support the importance of target local

secondary structure in mRNA and their accessibility for interaction with antisense oligos or siRNAs (Lee *et al*., 2002; Bohula *et al*., 2003; Vickers *et al*., 2003). Recent studies, based on computational predictions and experimental validation for accessibility, strongly suggested that the secondary structure of a target can be a useful indicator of the gene-silencing efficiency of the siRNA (Luo and Chang, 2004). Activity of siRNA is influenced by local characteristics of the target RNA, including local RNA folding (Kretschmer-Kazemi *et al*., 2003). Such observations suggest that predicting unpaired RNA regions and assessing target accessibility for siRNA can be useful for the design of active siRNA constructs.

When applied to predicting secondary structures of RNAs, SDP is substantially different from DP. In particular, SDP explicitly takes into account the number of allowed base pairs (M). This can lead to fast algorithms, especially in combination with preliminary base pair filtration and hierarchical approach (Roytberg *et al*., 2002). Our preliminary tests with in-house implementation of Zuker's algorithm showed that filtration of non-stacked base pairs implemented in Mfold (Zuker, 2003) may lead to finding of structures with substantially improved energy. For example, some wobble base pairs in tRNA secondary structures can be found or overlooked depending on filtration mode.

However, the SDP approach also has an inherent drawback. DP can be used to find both the optimal RNA structure and the partition function, and the time and space complexities are the same for both tasks. Formally speaking, DP exploits distributivity of operations in semirings (Aho *et al*., 1974; Finkelstein and Roytberg, 1993). The distributivity applies both to pair of operations (+, min) corresponding to the search of optimal structures, $\min(a+b, a+c) = a + \min(b, c)$, and to pair ($*$, +) corresponding to the computation of partition function, $a*b + a*c = a*(b + c)$.

SDP, in contrast, uses 'the owner paradigm' based on the following observation. Let $\Delta G = \min\{\Delta G_B, \Delta G_1, \Delta G_2, \ldots\}$ and $\Delta G_A > \Delta G_B$. Then we know already the value $\Delta G' = \min\{\Delta G_A, \Delta G_B, \Delta G_1, \Delta G_2, \ldots\} = \Delta G$ and, thus, can save computation time. However, this does not help if we have to compute $S = \Delta G_B + \Delta G_1 + \Delta G_2 + \cdots$ and $S' = \Delta G_A + \Delta G_B + \Delta G_1 + \Delta G_2 + \cdots$. Thus, SDP cannot be applied, in this form, to compute the partition function (this was noted by Lyngsø *et al*., 1999). Our approach to finding the set of all conditionally optimal structures to some extent obviates this obstacle and may be developed to approximate partition function on the basis of SDP.

Comparison of our software tool Afold with Mfold (Zuker, 2003), and ZUKER (Lyngsø *et al*., 1999) demonstrates that Afold clearly outperforms the other two tools (Fig. 3). Afold constructs internal loops much faster than Mfold and ZUKER (Fig. 3A). The time required for filling the whole energy matrix (Fig. 3B) including multi-branch loop evaluation is similar for our algorithm and Mfold (and much higher for ZUKER). However, Mfold artificially limits the length of internal loops (to 30 nt by default). In contrast, there is no such limitation in Afold. Also, Afold uses the same matrix to evaluate internal and multi-branch loops and, as a result, can fold, with 2 GB RAM, RNA sequences up to 28 000 nt long (in ~28 h on a regular PC).

ACKNOWLEDGEMENTS

This work was supported by the Intramural Research Program of the National Library of Medicine at National Institutes of Health/DHHS. R.M.A. acknowledges financial support by the Russian Foundation for Basic Research (project nos 03-04-49469, 02-07-90412), by grant from the RF Ministry for Industry, Science, and Technology (20/2002, 5/2003), NWO, ECO-NET and by the program of RF Ministry of Science and Education (contract No. 02.434.11.1008). Funding to pay the Open Access publication charges for this article was provided by the National Center for Biotechnology Information, National Library of Medicine, National Institutes of Health.

Conflict of Interest: none declared.

REFERENCES

Aho,A.V. *et al*. (1974) *The Design and Analysis of Computer Algorithms*. Addison-Wesley, Reading, MA.

Bohula,E.A. *et al*. (2003) The efficacy of small interfering RNAs targeted to the type 1 insulin-like growth factor receptor (IGF1R) is influenced by secondary structure in the IGF1R transcript. *J. Biol. Chem*., **278**, 15991–15997.

Eppstein,D. *et al*. (1992) Sparse dynamic programming II: convex and concave cost functions. *J. ACM*, **39**, 546–567.

Finkelstein,A.V. and Roytberg,M.A. (1993) Computation of biopolymers: a general approach to different problems. *Biosystems*, **30**, 11–19.

Jaeger,J.A. *et al*. (1989) Improved predictions of secondary structures for RNA. *Proc. Natl Acad. Sci. USA*, **20**, 7706–7710.

Hofacker,I.L. *et al*. (1994) Fast folding and comparison of RNA secondary structures. *Monatsh. Chemie*, **125**, 167–188.

Hofacker,I.L. *et al*. (2004) Conserved RNA secondary structures in viral genomes: a survey. *Bioinformatics*, **20**, 1495–1499.

Hirschberg,D.S. (1975) A linear space algorithm for computing maximal common subsequences. *Commun. ACM*, **18**, 341–343.

Kretschmer-Kazemi Far,R. and Sczakiel,G. (2003) The activity of siRNA in mammalian cells is related to structural target accessibility: a comparison with antisense oligonucleotides. *Nucleic Acids Res*., **31**, 4417–4424.

Larmore,L.L. and Schieber,B. (1991) On-line dynamic programming with applications to the prediction of RNA secondary structure. *J. Algorithms*, **12**, 490–515.

Lee,N.S. *et al*. (2002) Expression of small interfering RNAs targeted against HIV-1 rev transcripts in human cells. *Nat. Biotechnol*., **20**, 500–505.

Lyngsø,R.B. *et al*. (1999) Fast evaluation of internal loops in RNA secondary structure prediction. *Bioinformatics*, **15**, 440–445.

Luo K. and Chang D. (2004) The gene-silencing efficiency of siRNA is strongly dependent on the local structure of mRNA at the target region. *Biochem. Biopys. Res. Commun*, **318**, 303–310.

Mathews,D.H. *et al*. (1999) Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *J. Mol. Biol*., **288**, 911–940.

McCaskill,J.S. (1990) The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers*, **29**, 1105–1119.

Nussinov,R. and Jacobson,A.B. (1980) Fast algorithm for predicting the secondary structure of single-stranded RNA. *Proc. Natl Acad. Sci. USA*, **77**, 6309–6313.

Roytberg,M.A. *et al*. (2002) A hierarchical approach to aligning collinear regions of genomes. *Bioinformatics*, **18**, 1673–1680.

Tinoco,I. Jr *et al*. (1971) Estimation of secondary structure in ribonucleic acids. *Nature*, **230**, 3362–3367.

Tinoco,I. Jr *et al*. (1973) Improved estimation of secondary structure in ribonucleic acids. *Nat. New Biol*., **246**, 40–41.

Vickers,T.A. *et al*. (2003) Efficient reduction of target RNAs by small interfering RNA and RNase H-dependent antisense agents. A comparative analysis. *J. Biol. Chem*., **278**, 7108–7118.

Wuchty,S. *et al*. (1999) Complete suboptimal folding of RNA and the stability of secondary structures. *Biopolymers*, **49**, 145–165.

Xia,T. *et al*. (1998) Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with Watson–Crick base pairs. *Biochemistry*, **37**, 14719–14735.

Zuker,M. (1989) On finding all suboptimal foldings of an RNA molecule. *Science*, **244**, 48–52.

Zuker,M. (2003) Mfold web server for nucleic acid folding and hybridization prediction. *Nucleic Acids Res*., **31**, 3406–3415.

Zuker,M., Mathews, D.H. and Turner, D.H. (1999) Algorithms and thermodynamics for RNA secondary structure prediction: a practical guide. In Barciszewski,J. and Clark,B.F.C. (eds), *RNA Biochemistry and Biotechnology, NATO ASI Series*. Kluwer Academic Publishers. Dordrecht, The Netherlands, pp. 11–43.

Zuker,M. and Stiegler,P. (1981) Optimal computer folding of large RNA sequences using thermodynamics and auxiliary information. *Nucleic Acids. Res*., **9**, 133–148.

Zuker,M. and Sankoff,D. (1984) RNA secondary structures and their prediction. *Bull. Math. Biol*., **46**, 591–621.