Finding a Common Motif of RNA Sequences Using Genetic Programming: The GeRNAMo System

Shahar Michal, Tor Ivry, Omer Schalit-Cohen, Moshe Sipper, and Danny Barash

Abstract—We focus on finding a consensus motif of a set of homologous or functionally related RNA molecules. Recent approaches to this problem have been limited to simple motifs, require sequence alignment, and make prior assumptions concerning the data set. We use genetic programming to predict RNA consensus motifs based solely on the data set. Our system—dubbed *GeRNAMo* (*Ge*netic programming of *RNA Mo*tifs)—predicts the most common motifs without sequence alignment and is capable of dealing with any motif size. Our program only requires the maximum number of stems in the motif and, if prior knowledge is available, the user can specify other attributes of the motif (e.g., the range of the motif's minimum and maximum sizes), thereby increasing both sensitivity and speed. We describe several experiments using either ferritin iron response element (IRE), signal recognition particle (SRP), or microRNA sequences showing that the most common motif is found repeatedly and that our system offers substantial advantages over previous methods.

Index Terms—Genetic Programming (GP), RNA, common motif, microRNA.

1 Introduction

Recent research in molecular biology has discovered novel RNA functions which are responsible for mediating the synthesis of proteins, regulating cellular activities, and exhibiting enzyme-like catalysis and post-transcriptional activities. It is commonly agreed upon that the RNA's secondary structure is crucial to its functionality. For example, microRNA molecules take part in translational regulation. As part of the regulation process, their secondary structure plays an important role in the degradation of the microRNA precursor to its final regulating form, which in turn hybridizes to the $3'$ UTR of a matching mRNA sequence and thus reduces protein synthesis [16], [17].

The importance of the secondary structure presents a need for tools that rely on searching for common structures rather than searching for a common alignment of sequences. Finding common motifs can provide information on RNA functionality [4], [6] and help us classify RNA families. Usually, motif finding is limited to specific parts in the RNA's structure (up to 100 nt) rather than trying to find a common structure for the complete RNA molecule, which is less conserved. This is because, sometimes, the RNA's functionality is based on a specifically conserved element that it contains and not on its global structure. Therefore, searching for a global common structure might lead to the loss of important information.

Several methods aimed at predicting RNA secondary structure have been devised over the years. Some methods try to predict the structure based on a single RNA sequence [23], [32], [49], while others are based on information gained from multiple sequences [22], [45], culminating in the comparative modeling approach [20]. Multiple sequence-based techniques for finding common motifs can be divided into three categories: those that perform a multiple sequence alignment prior to the structure predictions, e.g., SLASH [15] (SLASH combines COVE [5], a stochastic context-free grammar RNA secondary structure prediction method, and FOLDAlign [14], which is based on Sankoff's algorithm [37]), those that do not use multiple sequence alignment, e.g., comRNA [26], and those that align the sequences during the computation, e.g., RNAProfile [35] and CMFinder [47]. Alignment-dependent methods fail in case no similarity is found between the multiple sequences. In the nonalignment category, there are recent approaches that use evolutionary computation [9], [10], [42] because of the difficulty of the problem. One example is GPRM [24], [25], a genetic programming-based method that predicts a common motif while folding the input sequences. Another example is the approach developed by Fogel et al. [11], which takes RNAMotif's [31] output[1] as input and uses evolutionary computation to narrow down the list of structures. This method is powerful as it finds the correct solution in a very small fraction of the time it would have taken to scan the

- S. Michal is with Orbotech, Hemeyadim 25, Kfar Saba, 44205, Israel. E-mail: mshaha@cs.bgu.ac.il.
- T. Ivry is with Applied Materials, Hapardes 14, Tel Aviv, 64245, Israel. E-mail: ivryt@cs.bgu.ac.il.
- O. Schalit-Cohen is with Medcon, Hanesher 26, Raanana, Israel. E-mail: schalitc@gmail.com.
- M. Sipper and D. Barash are with the Department of Computer Science, Ben-Gurion University of the Negev, PO Box 653, 84105 Beer-Sheva, Israel. E-mail: {sipper, dbarash}@cs.bgu.ac.il.

Manuscript received 23 Feb. 2006; revised 24 Aug. 2006; accepted 19 Oct. 2006; published online 25 Jan. 2007.
For information on obtaining reprints of this article, please send e-mail to: tcbb@computer.org, and reference IEEECS Log Number TCBB-0025-0206.
Digital Object Identifier no. 10.1109/TCBB.2007.1045.

1. RNAMotif searches a database for RNA sequences that fold according to a given descriptor (motif). The output is a list of the structures that conform to the secondary structure specified by the descriptor, along with additional information (e.g., base pairing, position in the sequence).

entire output of RNAMotif. Furthermore, Fogel et al. performed extensive experiments in order to tune the evolutionary algorithm process by investigating the effects of population settings and selection parameters on the convergence of the algorithm [12].

Evolutionary computation was previously used for RNA structure folding [2], [8], [18], [19], [38], [39], [40], [44] and the prediction of common structures [3] using free energy calculations. Furthermore, evolutionary algorithms were applied to the identification of conserved regulatory RNA structures in prokaryotic metabolic pathway genes [29].

We believe that the problem of finding a common motif of RNA sequences, without aligning them first, is at the level of difficulty for which evolutionary computation can offer a substantial contribution in practice.

In this paper, we present *GeRNAMo* (*Ge*netic programming of *RNA Mo*tifs), an evolutionary computation system that searches for a motif which is common to a given data set of RNA sequences. GeRNAMo does not perform any sequence alignment. Furthermore, the use of genetic programming enables the system to explore the search space of motif structures in a reasonable amount of time. We tested GeRNAMo on various data sets with a known common motif, as well as on RNA data sets that do not have any known relations among them. Moreover, we compared our approach to another genetic programming-based technique that tries to solve motif-finding problems: GPRM [24].

In the next section, we describe the basic methods related to RNA folding and motifs. Section 3 describes our algorithm, followed by a presentation of experimental results in Section 4. Finally, conclusions and future work are summarized in Section 5.

2 Methods

2.1 Motif Representation

We use two types of basic elements to describe a motif: a complementary segment (i.e., Watson-Crick base-pairing of A-U and C-G or the wobble interaction G-U), represented by "h5" and "h3" ("h" represents a double helix, i.e., complementary segment, and the values 5 and 3 represent the $5'$-end and $3'$-end, respectively). A single-strand is represented by "ss." An element is composed of one of these symbols accompanied by (x:y), where x and y are the minimum and maximum lengths of the elements, respectively. Different combinations of these elements generate different motifs. An example is given in Fig. 1.

2.2 RNAsubopt

In order to make predictions based on an RNA secondary structure, we used RNAsubopt, a program that predicts all suboptimal secondary structures of a given sequence based on thermodynamics and base-pairing rules [46].

RNAsubopt, like many other RNA folding approaches, uses a Gibbs free energy[2] minimization procedure. It is hoped that the native fold of the sequence is close to the minimum free energy (mfe) structure. We are interested in all suboptimal solutions because, in nature, RNA folds into

2. Gibbs free energy is defined as the energy portion of a thermodynamic system available to do work. $G = H - TS$.

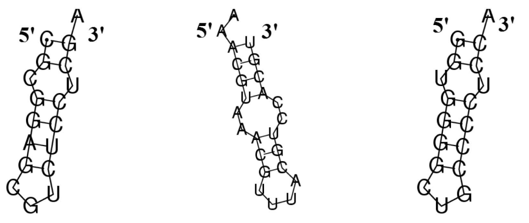

Fig. 1. The common motif of the sequences (a) CGCGGAGCGU-CUCCUCGA, (b) AAACGUAAACGUUUACGUCCACGU, and (c) GGUGGGGCUGCCCCUCCA contains an internal loop and a hairpin, and is represented by: ss(0:2) h5(2:4) ss(1:2) h5(3:4) ss(3:4) h3(3:4) ss(1:2) h3(2:4) ss(0:1).

a suboptimal structure (and also because of limitations of thermodynamic models), which may cause the mfe structure to be different from the native fold.

RNAsubopt is part of the Vienna-RNA package [21], [23]. For a given sequence, RNAsubopt calculates all suboptimal secondary structures within an energy range above the minimum free energy. It outputs the suboptimal structures —sorted by mfe—in a dot-bracket notation, followed by the energy in kcal/mol. A different method for calculating suboptimal solutions, used in [48], was described in Zuker's seminal work.

Under dot-bracket notation, a dot represents an unpaired base and a parenthesis represents a paired base. The returned dot-brackets are then compressed to new strings which represent the structures and consist of the different elements in these structures.

For example, the structure in Fig. 1b is the optimal solution (the first of RNAsubopt's output structures) for the sequence AAACGUAAACGUUUACGUCCACGU. Its dot-bracket notation is:

$$..((((..(((....)))..))))$$

and its compressed string is similar to a motif's representation, with each element having a specific length:

$$\mathrm{ss}(2)\ \mathrm{h5}(4)\ \mathrm{ss}(2)\ \mathrm{h5}(3)\ \mathrm{ss}(4)\ \mathrm{h3}(3)\ \mathrm{ss}(2)\ \mathrm{h3}(4).$$

2.3 Naive Folding

In addition to RNAsubopt, we used a naive folding algorithm, according to [7], [34]. We implemented a simple dynamic programming algorithm and used two versions that assign different scores to wobble interactions. The naive folding algorithm scheme is:

For a pair of nucleotides (i, j) choose the maximum among three options:

- i and j pair with each other according to the score function Score(i, j); try to solve for $(i-1, j+1)$.
- i and j do not pair; try to solve for $(i, j+1)$.
- i and j do not pair; try to solve for $(i-1, j)$.

Score(i, j) is based on one of two scoring methods: 1) A-U and G-C are assigned two points and G-U is assigned one point or 2) G-U is also assigned two points (as in the Nussinov algorithm). The output of this algorithm is in dot-bracket notation.

```
((((..(.....)..)))).                              .....((((......)))).

AAGUUCCAGGAAGUGACUUG                              AGUUCCAGGAAGUGACUUGC

h5(4) ss(2) h5(1) ss(5) h3(1) ss(2) h3(4) ss(1)   ss(5) h5(4) ss(6) h3(4) ss(1)

(a)                                               (b)
```

Fig. 2. Two segments of length 20 of the sequence AAAGUUCCAG-GAAGUGACUUGCUGCGACAGUGCUCGUGUAG.

2.4 Finding a Motif Occurrence in a Sequence

Finding a motif usually requires aligning it with the sequence in a manner that results in the best score and includes spaces and gaps. We present a different approach where no such alignment is required. Initially, we break down every sequence into segments of variable lengths. A subset of segments in the range of the motif minimum and maximum lengths is chosen. We then iteratively go through this subset to find segments that fold under the motif constraints. When a motif is being compared to the segment, there is no need for scanning or aligning—we simply compare the corresponding elements of the motif and the compressed string representing the segment. When elements do not match, we move on to the next suboptimal structure of the segment or to the next segment of the sequence until a match is found or until there are no more segments.

For example, suppose we are looking for the hairpin motif, **ss(0:5) h5(4:6) ss(2:6) h3(4:6) ss(1:2)**, and we wish to know if the sequence AAAGUUCCAGGAAGUGACUUG CUGCGACAGUGCUCGUGUAG contains it. Fig. 2 shows two segments of length 20. When comparing the motif with Fig. 2a, a segment starting at position 2 of the sequence, there is no single-strand element, a condition which satisfies the motif's first-element constraint (namely, ss(0)). The second and third elements also match (h5 of length 4 and ss of length 2), but the fourth element, h3, does not match and, therefore, this segment cannot fold to the aforementioned boldfaced motif. In Fig. 2b, which starts at position 3, all of the elements match the motif constraints, indicating that the sequence contains our hairpin motif. This procedure is used to evaluate individuals in the evolutionary run (more specifically, it is used to assign fitness to an individual), as will be discussed later.

2.5 ECJ

ECJ is a freely available evolutionary computation and genetic programming system written in Java [30]. It is highly flexible and nearly all classes and settings can be changed by the user. All structures in the system are arranged to be easily modifiable. We used this package to run the evolutionary simulation and added classes to support our preprocessing and main algorithm.

2.6 Base-Pairing Distance

In order to compare two secondary structures, we used RNAdistance, which is also part of the Vienna-RNA package. It reads RNA secondary structures and calculates a "base-pair distance" given by the number of base pairs present in one structure—but not the other.

This method reports a difference for secondary structures that are included in each other. For example, ...(((...)))..... is included in ...(((((...)))))..., but RNAdistance reports that the base-pairing distance is 4. We acknowledge the fact that the two bases flanking both ends may not be counted as base-pairing differences, but we use this method as a measure of success in identifying the correct occurrences of the predicted motif in the data set.

3 The GeRNAMo System

GeRNAMo is based on genetic programming [1], [27], [28], an evolutionary methodology inspired by biological evolution. Genetic programming (GP) is a machine-learning technique—an evolutionary algorithm that seeks to progressively improve a population of candidate solutions. The entire process is driven by a user-defined fitness function that assigns a score to an individual solution in the population in accordance with its ability to solve the problem at hand. In genetic programming, we evolve a population of individual LISP expressions,[3] each comprised of *functions* and *terminals*. Since LISP programs may be readily represented as program trees, the functions are internal nodes and the terminals are leaves. In GeRNAMo an individual in the population is a common motif.

The genetic programming process begins by creating an initial random set of individuals (a random population of programs referred to as generation zero). Next, each individual is tested for its ability to solve the problem and assigned a fitness score accordingly. Individuals with a high score are more likely to survive and pass on their "genetic" material to the next generation. New individuals for the next generation are created from the selected high-fitness parents by applying the pseudogenetic operators of crossover, combining the genomes of two parents, and mutation, randomly effecting a small change to a single parent. Over many generations, this stochastic process often yields good solutions. The pseudocode of the genetic programming algorithm is given in Fig. 3.

The GeRNAMo algorithm is composed of three parts: preprocessing, evolutionary run, and output generation.

3.1 System Parameters

In addition to the GP-related parameters, which will be discussed later on, the system has several other parameters. These are shown in Table 1. Except for the maximum number of stems in the motif, all parameters are optional and have default values.

3.2 Preprocessing

The system receives as input a data set containing RNA sequences that are suspected of sharing a common motif. It then performs the following actions prior to the commencement of the evolutionary run:

- Each sequence is allocated to a slot in an array of size N (N is the size of the data set).
- Each sequence is then broken down into all possible subsequences ranging between a user-defined minimum and maximum motif length (the default is between 15 to 100 nt).

3. Languages other than LISP have been used, although LISP is still by far the most popular within the genetic programming community.

```
Generate initial population at random;
Gen:=0
WHILE (Termination condition not met AND Gen<MaxGen)
 BEGIN
  Evaluate fitness of each individual;
  FOR J:=1 TO PopSize DO
   BEGIN
    Select 2 individuals based on fitness;
    Perform crossover with probability Pc;
    Perform mutation with probability Pm
                          on each offspring;
    J:=J+2;
    Insert the 2 offspring into the new population;
   END;
  Gen:=Gen+1;
 END;
Designate result for run;
END.
```

Fig. 3. Pseudocode of the genetic programming algorithm. PopSize: Population size, MaxGen: Maximal number of generations, and Gen: Generation counter.

- A query is sent to RNAsubopt (or other external RNA structure prediction programs), which is capable of predicting all suboptimal secondary structures for every subsequence. RNAsubopt's output includes the dot-bracket notation and free energy value for each of the structures. This information is stored for additional runs using an efficient filing system.
- In addition to the external folding program, the naive folding algorithm described in Section 2.3 can also be used (this is a user option; see Table 1). We recommend using this option for very short sequences (where the basic base-pairing rules give a better fold than the thermodynamic model).
- RNAsubopt's (and the naive folding's, if selected) output is then parsed in order to build a data structure containing the secondary structures' information in an efficient, compressed, and easily comparable form. The output of RNAsubopt and the data structure that is built using it were described in Section 2.2.

The preprocessing stage forms an array containing all compressed strings, ranging between the defined minimum and maximum lengths of the given input sequences.

This idea of applying a single RNA structure predictor as a preprocessor was also introduced in MARNA [41]. MARNA makes a multiple alignment of RNAs by taking into consideration both the sequence and its secondary structure (a different problem than the one discussed in this paper), but, whereas MARNA uses the secondary structure only to aid in the calculations, our approach relies on the preprocessor folding program in order to find the common motif.

TABLE 1
System Parameters

Parameter	Default Value
Folding program	RNAsubopt
Min length	15
Max length	100
Maximum no. of stems in the motif	a required parameter
Minimum no. of stems in the motif	1
Minimum stem size	1
No. of suboptimal results	11
Use naive folding?	false
Generality level	specific

3.3 Evolutionary Algorithm

To set up a genetic programming run, one needs to define the individual representation, function and terminal sets, fitness function, and genetic operators and parameters. The rest of the implementation is based on ECJ. In addition, we expanded ECJ's classes, added classes of our own, and set up the required parameters.

3.3.1 Individual Representation

An individual in the population is a motif, much like the one described in Section 2.1. The representation of a motif in the simulation is a tree, where "h5," "h3," and "ss" are used as functions (i.e., internal nodes of the tree) and the element lengths are terminals (i.e., tree leaves).

The terminal set includes:

- The end of the motif: "no element."
- Length: "no length constraints." The default is a random range, proportional to the expected motif length.
- Numerical values: "0," "1," "2," "3."

The function set includes:

- Complementary segments: "h3," "h5"; and nonpairing segments: "ss."
- Lengths: "min length" and "max length."
- Numerical values: "0," "1," "2," "3."

Note that numerical values are considered functions as well as terminals since an integer number is represented as the sum of values in the binary tree of small numbers (0-3). This was done to increase sensitivity and diversity (for example, a mutation of such a value will change a subtree and not the whole number).

The tree is made up of two types of basic trees:

- A motif starting with h5 can contain a submotif (or elements) which ends with h3 and can be followed by a submotif, as shown in Fig. 4a.
- A motif starting with ss can only be followed by a submotif, as shown in Fig. 4b.

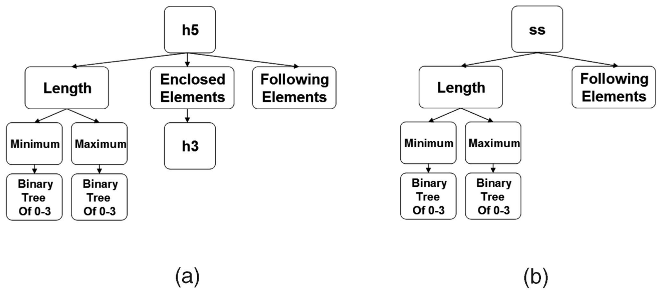

Fig. 4. (a) An "h5" basic tree. (b) An "ss" basic tree.

Each basic tree has a minimum and a maximum length which makes up its "Length" property (i.e., (x:y) in Section 2.1). The values are calculated as the sum of the numbers in the binary tree that compose the minimum or the maximum branch of "Length."

An individual is a tree made up of a combination of these two basic trees, where each basic tree is an element in the motif representation. An example is given in Fig. 5.

3.3.2 Genetic Operators and Parameters

We experimented with different parameter settings and evaluated the convergence rate of the algorithm as a function of these parameters (mainly Pm, Pc, K, and population size). We finally settled on those shown in Table 2. The population was generated using the ramped-half-and-half method, with minimum and maximum sizes proportional to the motif size (the default settings were 4 and 7, respectively), and grow probability of 0.5 (these parameters are further described in [27]).

Genetic operators are applied as follows:

- Selection: Individuals are chosen from the population by tournament selection, which picks the individual with the highest fitness value out of a small group of randomly chosen individuals (in our case, seven). It is efficient since the competing individuals are randomly chosen and, therefore, the algorithm is able to prevent domination of one individual. At the same time, it allows individuals that are not optimal in relation to the whole population to survive, thereby enriching the genetic diversity in the population and increasing the ability of the population solving the given problem.
- Crossover is performed between two individuals with probability Pc. The crossover is done in STGP (Strongly Typed Genetic Programming [33]) style, thus allowing only compatible parts to be crossed over.
- Mutation is applied to each individual with probability Pm by choosing a random node in the tree representing the individual. The subtree stemming from that point is deleted and a new one is grown instead.
- The offspring are inserted into the population, thereby comprising the next generation.

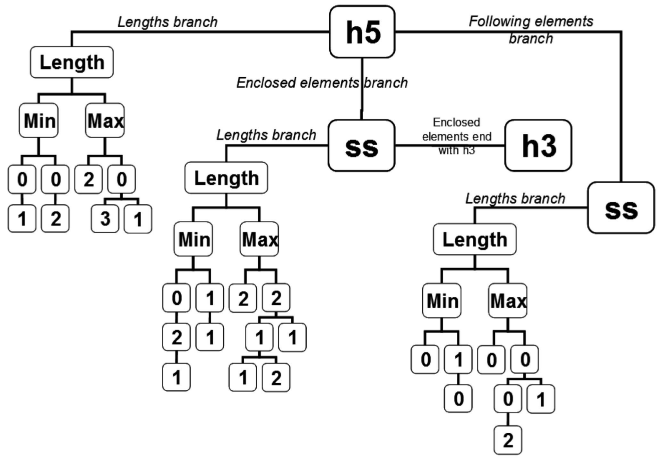

Fig. 5. An example individual in the evolving population, representing the motif h5(3:6) ss(5:9) h3(3:6) ss(1:3). Note that the h3 range implicitly equals the h5 range; otherwise, they cannot base-pair with each other.

TABLE 2
Genetic Programming Parameters

Parameter	Value
Population size	1024
Number of generations	51
Selection method	tournament, with K = 7
Pc, Crossover rate	0.95
Pm, Mutation rate	0.05
Maximum tree depth	11-17
Terminals probability	0.1
Functions probability	0.9

3.3.3 Fitness Function

The fitness function assigns a score to an individual based on how well it solves the problem. In our case, a high score is given to a motif common to the majority of the sequences in the data set. Higher fitness leads to a higher probability of the individual's being selected for reproduction. Our goal is to find the motif that is the most common and preferably has the most elements. At the end of the evolutionary run, the individual with the highest fitness encountered throughout the run is returned as the result.

The fitness for a motif m is assigned according to the following formulas:

$$Fitness(m) = Main(m) + Supplements(m), \qquad (1)$$

$$Main(m) = \frac{\sum_{i=1}^{N} Major(m,i)}{N}, \qquad (2)$$

$$Supplements(m) = \sum_{i=1}^{N} \frac{Minor(m,i) + MC(m,i)}{TS(i)}, \qquad (3)$$

$$Major(m,i) = Modifier(m) \times Length(i,j,k,l), \qquad (4)$$

$$Modifier(m) = Elements(m) \times Stems(m) - General(m), \qquad (5)$$

where

- i: The index of the sequence.
- j: The length of a subsequence of sequence i.
- k: The position of a subsequence of length j in sequence i.
- l: The index of the suboptimal secondary structure (in RNAsubopt's output) of a subsequence in position k and length j of the sequence i.
- $Major(motif, i)$: The most meaningful occurrence of the motif in sequence i. An occurrence is discovered using the method described in Section 2.4. We define the most meaningful occurrence as the **longest** subsequence of sequence i that folds according to the motif constraints, with the **most negative** free energy. Special cases, like a motif that is only a single strand or a very general motif, are given a penalty. The formulas for the calculation of Major(motif, i) are given in (4) and (5).
- $Minor(motif, i)$: Number of all other occurrences of the motif in all segments of all positions in sequence i.
- $MC(motif, i)$: Length of the longest consecutive segment of the motif that is contained in sequence i. This score is used in the early stages of the evolutionary run, where it is unlikely that randomly generated motifs will be common to all of the sequences in the data set. It prevents the situation where most of the individuals in the population receive zero fitness, thereby hampering evolution.
- $Elements$: Number of elements in a motif.
- $Stems$: Number of complementary segments in a motif. We can provide a bonus to motifs containing specific numbers of stems when we have prior knowledge of the common motif.
- $Length(i, j, k, l)$: Length of segment checked (the lth suboptimal solution of a segment in position k and length j of sequence i).
- $General$: Reflects the generality of the motif. The user can specify the level of generality and, according to the level chosen, a different penalty function is calculated. We currently support four levels:
    1. maximum generality: no penalty is given,
    2. moderate generality 1: the penalty is calculated based on the differences between the motif's range and the lengths of the sequences in which it occurs, by calculating the minimum and maximum length over all of the occurrences,
    3. moderate generality 2: the penalty is calculated based on the differences between the motif's range and the lengths of the sequences in which it occurs, by calculating the average length over all of the occurrences, and
    4. high specificity: the penalty is calculated based on the lengths of the elements in the motif.
- $Modifier$: The number of elements in the motif multiplied by the number of stems minus the penalty (if one exists) for the generality of the motif, according to the generality level.
- N: Total number of sequences in the data set.

```
( ( ( ( ( . [ [ [ [ [ . . . . . . ] ] ] ] ] ) ) ) ) )

( ( ( ( ( . [ [ [ [ [ . . . . . . . ] ] ] ] ] ) ) ) ) )
```

Fig. 6. IRE consensus structure motif of the IRE-like data set.

- $TS(i)$: Total number of segments of all lengths and all indexes of a sequence.

3.4 Output

At the end of the evolutionary run, when the best solution (a motif) has evolved, the system returns every occurrence of the motif in all the sequences of the data set. When the data set contains many sequences or very long sequences, or when the motif is very general, we use the following filters to reduce the size of the output file:

- When an occurrence is returned, the surrounding occurrences within several nt are not returned. This is because, had they indeed been folded according to the motif, they would have had the same secondary structure and very similar sequences.
- Using thermodynamics as a filter, return only occurrences with free energy below a certain value.

4 Results

We experimented with four data sets. The first was used in SLASH [15] and GPRM [24], the second was used by Fogel et al. [11], the third was used in RNAProfile [35], and the fourth was constructed by us. GeRNAMo's results were compared to GPRM in all cases, in addition to a comparison with the work from which the data set was taken. All of our experiments were done using the default system parameters, except for the parameter "Maximum number of stems in the motif," which was set to four for the microRNA data set and two for all other data sets. We specifically mention any other parameter change (e.g., use of naive folding) in the appropriate section.

Two aspects of the results are presented:

- The evolved common motif, found by GeRNAMo.
- The occurrence of the evolved motif in each sequence in the data set. Each sequence contains a segment (or more) that folds according to the motif constraints (in terms of element types and lengths). The occurrence is the secondary structure of this segment.

Supplemental data is available at http://www.cs.bgu.ac.il/~mshaha/GeRNAMo.html.

4.1 Ferritin IRE-Like 1

This data set contains 56 Ferritin IRE-like (Iron Response Element [13], [43]) sequences, as described in [15], [24]. The motif presented in [15] is **h5(5:5) ss(1:1) h5(5:5) ss(6:7) h3(5:5) h3(5:5)** and it will be regarded as the known motif. The corresponding secondary structures in dot-bracket notation are given in Fig. 6.

We ran a total of 50 runs, of which most produced good common motifs, representing the IRE-like motif. The best

```
124. . . . 130. . . . . . . . . . 142. . . 147. . . .
g a u g c c c c a u u c a c g a g u a g u g g g u a u u
( ( ( ( ( . [ [ [ [ [ . . . . . . . ] ] ] ] ] ) ) ) ) )   (ΔG = -6.5)
```

(a)

```
70. . . . . . 77. . . . . . . . . . 88 . . 92 . . . .
g a u u g c u g g g a c a c g a g c u u c u g c g g u u
( ( ( ( ( ( . [ [ [ [ . . . . . . . ] ] ] ] ) ) ) ) ) )   (ΔG = -4.4)
```

(b)

```
3. . . . . .89 . . . . . . . . . . . 21 . . . 26 . . . . .
a a u g u c a u u c a c a c g a g c u g a a u a u a u u
( ( ( ( ( . [ [ [ [ [ . . . . . . . ] ] ] ] ] ) ) ) ) )   (ΔG = -2.9)
```

(c)

Fig. 7. A sample output of the occurrences of the motif predicted by GeRNAMo for the IRE-like data set. (a) seqM60170.1 (no. 3). (b) seqAF161708.4 (no. 47). (c) seqJ04716.4 (no. 49).

motif predicted by GeRNAMo is: **h5(4:6) ss(1:2) h5(4:6) ss(6:7) h3(4:6) h3(4:6)**.

Out of the 56 sequences in the data set, this motif occurs in 43 sequences in the right positions (i.e., the occurrence is the IRE motif segment). According to RNAsubopt and Mfold, the other 13 sequences do not contain the segments with the secondary structure that was described in [15], [24] (the IRE consensus structure motif). Because we used RNAsubopt as our folding program, these occurrences were not found, but the evolved motif is correct. A sample output is given in Fig. 7 and a typical secondary structure of IRE is given in Fig. 8.

Because of the simplicity of IRE, we used the naive folding algorithm described in Section 2.3 in addition to RNAsubopt. With the added folding possibilities of the sequences, the motif occurred (in the correct positions) in all sequences but two.

We compared the base-pairing distances (as described in Section 2.6) of the occurrences of the motif predicted by GeRNAMo with the correct occurrences of the known motif (from [15]). We also compared the base-pairing distances of GPRM's [24] results to the occurrences of the known motif. The motif predicted by GPRM is: **h5(5,8) ss(1,1) h5(5,5) ss(6,7) h3(5,5) ss(0,0) h3(5,8)**.

As can be seen from Table 3, our results are closer to the known occurrences than GPRM results (despite the fact that the base-pairing distance measure reports a difference for included structures, it still reflects how similar our occurrences are to known occurrences versus GRPM's results). Table 4 displays the average base-pairing distances of both GPRM and GeRNAMo versus the known secondary structure of the IRE motif (note that GPRM has only one set of results that is compared to both sets of GeRNAMo—with and without naive folding).

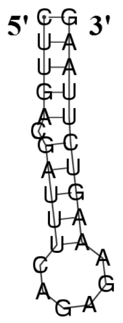

Fig. 8. A typical secondary structure of IRE (seq D28463.1 in the data set).

4.2 Signal Recognition Particle (SRP)

This data set was used in experiment 5 in Fogel et al. [11]. It contains five full-length sequences for 4.5S/7S rRNA (obtained from GenBank) from the following species, with the indicated NCBI "gene-info" (gi) sequence identifiers: *Archaeoglobus fulgidus* (38795), *Bacillus subtilis* (216348), *Escherichia coli*, (42758); *H.sapiens*, (177793); *Methanococcus voltae* (150042).

The problem addressed by Fogel et al. is different than the one addressed in this paper. Unlike RNA secondary structure predictors, given a descriptor, RNAMotifs aims at identifying all RNA structure elements that comply with the descriptor's constraints. Fogel et al.'s method was designed to screen the numerous candidate motifs produced by RNAMotifs and to find the correct solution in a fraction of the time it would have taken for exhaustive comparisons. This method does not perform any prediction, whereas GeRNAMo predicts the common motif based on the sequences in the data set.

To test our approach we used GeRNAMo to discover the common motif in the data set and compared it to the descriptor that was used by Fogel et al. The goal was to use GeRNAMo to predict the descriptor used by Fogel et al.—a predesigned motif—without any prior knowledge (other than the maximum number of stems: two).

We ran a total of 50 runs, of which most produced good common motifs, representing the SRP motif. The best motif predicted by GeRNAMo is:

$$\mathbf{h5(3:4)}\ \mathrm{ss}(4:5)\ \mathbf{h5(3:4)}\ \mathrm{ss}(4:5)\ \mathbf{h3(3:4)}\ \mathrm{ss}(4:5)\ \mathbf{h3(3:4)}.$$

The descriptor that was used by Fogel et al. is given in Fig. 9 and, in our representation,

$$\mathbf{h5(3:3)}\ \mathrm{ss}(3:5)\ \mathbf{h5(3:3)}\ \mathrm{ss}(4:6)\ \mathbf{h3(3:3)}\ \mathrm{ss}(3:5)\ \mathbf{h3(3:3)}.$$

One can easily see that the motif and the descriptor are very similar.

We also compared Fogel et al.'s top bin structures with GeRNAMo's output—the occurrences of the motif. As can be seen from Table 5[4] and Fig. 10, the results are identical.

In addition, we submitted this data set to GPRM, with the default parameters which should allow discovery of the

4. The results described in Table 5 are identical to the results described in Fogel et al. for the SRP data set. The original table is given in the supplemental data.

TABLE 3
Base-Pairing Distances

Index in data set	Sequence name	GPRM	GeRNAMo	Index in data set	Sequence name	GPRM	GeRNAMo
1	AJ251148.1	4	**0**	52	X51395.4	**0**	2
3	M60170.1	2	**0**	54	AF068224.4	**0**	2
4	D28463.1	4	**0**	2	L37082.1	2	2
5	Y15629.1	4	**0**	7	AF161708.1	0	0
8	M12120.1	2	**0**	9	J04716.1	0	0
11	D86626.1	2	**0**	12	X51395.1	0	0
13	AF161710.1	6	**0**	14	AF068224.1	0	0
17	M60170.2	2	**0**	15	AJ251148.2	0	0
18	D28463.2	2	**0**	16	L37082.2	0	0
21	AF161708.2	4	**0**	20	D15071.2	0	0
22	M12120.2	2	**0**	23	J04716.2	0	0
27	AF161710.2	2	**0**	24	S77386.2	0	0
32	Y15629.3	6	**0**	25	D86626.2	0	0
33	D15071.3	6	**0**	26	X51395.2	0	0
34	AF161708.3	6	**2**	30	M60170.3	0	0
40	AF161710.3	2	**0**	31	D28463.3	0	0
41	AF068224.3	4	**0**	35	M12120.3	0	0
44	M60170.4	6	**2**	36	J04716.3	0	0
49	J04716.4	4	**0**	37	S77386.3	2	2
50	S77386.4	2	**0**	38	D86626.3	0	0
6	D15071.1	**0**	2	42	AJ251148.4	0	0
10	S77386.1	**0**	2	43	L37082.4	0	0
19	Y15629.2	**0**	2	45	Y15629.4	0	0
28	AJ251148.3	**0**	2	47	AF161708.4	2	2
29	L37082.3	**0**	2	48	M12120.4	0	0
39	X51395.3	**0**	2	51	D86626.4	0	0
46	D15071.4	**0**	2	53	AF161710.4	0	0

The occurrences of the motif predicted by GeRNAMo and GPRM were compared to the known secondary structures of the segments containing the IRE motif. This table shows a comparison of the base-pairing distances, where results in boldface indicate a smaller base-pairing distance, which are, hence, favorable to the method. (The sequences AF068224.2 and D28463.4, not shown in the table, do not contain our motif according to RNAsubopt and our naive folding algorithm implementation. The naive method used by GPRM apparently discovers these occurrences, but, as will be discussed later, at the expense of failing to find logical motifs in more complicated data sets (see Fig. 17).)

motif described above (Pair number: 2; Base pairing size: min 3, max 8; Nonpairing size: min 0, max 10). The motif that was predicted by GPRM is: **h5(3,5) ss(0,3) h5(7,9) ss(0,3) h3(3,5) ss(5,7) h3(7,9)**. This motif is very different from both the descriptor of the SRP motif (from Fogel et al.) and GeRNAMo's predicted motif. Furthermore, GPRM's results (given in Fig. 11) are very different from Fogel et al.'s and GeRNAMo's top occurrences.

We also used Mfold to predict the secondary structure of the sequences in GPRM's results and these structures do not comply with the SRP motif or with GPRM's predicted motif (full structures are given in the supplemental data). These results indicate that GPRM fails to discover the SRP motif in this data set.

TABLE 4
Base-Pairing Average Distances

	GPRM	GeRNAMo
Without naive folding	1.44	0.75
With naive folding	1.44	0.52

4.3 Ferritin IRE 2

This data set was used in RNAProfile [35] in the discovery test. It was built by retrieving the full mRNA sequences of human ferritin (light and heavy chain) and of aminolevulinate synthase 2, as well as their mouse homologs from GenBank. In addition, the mRNA sequences of three human ferritin pseudogenes were added to the data set. The sequence lengths ranged between 800-2,000 nt.

Fig. 12 shows the highest-scoring motif occurrences output by RNAProfile on the IRE data set. These results

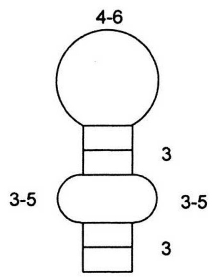

Fig. 9. The descriptor used by Fogel et al. [11] to screen the SRP data set for structures using RNAMotif.

TABLE 5
Fogel et al.'s Top Bin Structures of the SRP (Signal Recognition Particle) Motif

SeqID(gi)	start	length	sequence	structure
38795	192	24	GCCCAGGCCCGGAAGGGAGCAGGC	(((....(((....)))....)))
216348	153	24	TGTCAGGTCCGGAAGGAAGCAGCA	(((....(((....)))....)))
42758	204	24	GGTCAGGTCCGGAAGGAAGCAGCC	(((....(((....)))....)))
177793	308	24	GCCCAGGTCGGAAACGGAGCAGGT	(((....(((....)))....)))
150042	310	26	CCGCCAGGCCCGGAAGGGAGCAACGG	(((.....(((....))).....)))

show that RNAProfile was capable of identifying the IRE in the correct sequences. Moreover, the motif instances reported in the pseudogenes have a much lower score and are thus very unlikely to be real IRE instances.

We ran a total of 50 runs, of which most produced good common motifs, representing the IRE motif. The best motif predicted by GeRNAMo is: **ss(0:1) h5(3:4) ss(0:1) h5(4:5) ss(3:6) h3(4:5) h3(3:4) ss(0:1)**. A full output is given in Fig. 13. The results for all six real genes are identical between GeRNAMo and RNAProfile. These results show that GeRNAMo, like RNAProfile, found the IRE in all six genes.

Regarding the pseudogenes, GeRNAMo did not find occurrences of the predicted motif in these sequences because of the specificity of the motif (for instance, the motif's constraints do not allow a single unpaired base between the h3s, as in the last two sequences in the data set; see Fig. 12). This is a reasonable result because a functional IRE motif may not occur in these particular pseudogene sequences as the RNAProfile results also suggest.

In addition, we submitted this data set to GPRM for several runs, with the default parameters which should allow discovery of the motif described above (Pair number: 2; Base pairing size: min 3, max 8; Nonpairing size: min 0, max 10). Because GPRM only accepts sequences with length smaller than 1,000 nt., we shortened the original input at the end of each sequence at a position far beyond the occurrence of the motif (each sequence still contained the motif) and removed the three pseudogenes. Therefore, the input given to GPRM

```
>Archaeoglobusfulgidus
192........199.... ...206........213..
GCCCAGGCCCGGAAGGGAGCAGGC
(((....(((....)))....)))          (ΔG = -11.7)

>Bacillussubtilis
153........160........167.........174..
TGTCAGGTCCGGAAGGAAGCAGCA
(((....(((....)))....)))          (ΔG = -6.8)

>Escherichiacoli
204.......211........218........225..
GGTCAGGTCCGGAAGGAAGCAGCC
(((....(((....)))....)))          (ΔG = -8.5)

>H.sapiens
308........315........322........329..
GCCCAGGTCGGAAACGGAGCAGGT
(((....(((....)))....)))          (ΔG = -7.1)

>Methanococcusvoltae
310.........318........325..........333..
CCGCCAGGCCCGGAAGGGAGCAACGG
(((.....(((....))).....)))        (ΔG = -10.5)
```

Fig. 10. GeRNAMo's results of the SRP data set.

```
[3,5] (0,3) [7,9] (0,3) [3,5] (5,7) [7,9]

>Archaeoglobusfulgidus ,gi|38795
228  230231          239    243  245           253          261
(  ( (  (  ( ( ( ( ( ( ( (  . . . )  ) )  . . . . . . . )  ) ) ) ) ) ) ) )
a  c c  g  g c g c u u c c  g g g g  g u  g c g g g g a g  g a g g u g c c

>Bacillussubtilis ,gi|216348
75      7980             88     92     96         102         110
( ( ( ( ( ( ( ( ( ( ( ( ( ( . . . ) ) ) ) ) . . . . . )  ) ) ) ) ) ) ) )
a g g u u c g u u u a c u u u a a g g u c u g c c u u a  a g u a a g u g

>Escherichiacoli ,gi|42758
138   141  143          150151   154          161          168
(  ( ( (  . (  ( ( ( ( ( ( (  )  ) ) )  . . . . . . )  ) ) ) ) ) ) )
g  c g u  u g  g u u c u c a  a  c g c  u c u c a a u  g g g g g c u

>H.sapiens ,gi|177793
13  15      19              2627  29          35          42
( ( ( . . . ( ( ( ( ( ( ( ( ) ) ) . . . . . ) ) ) ) ) ) ) )
c g c c c a g u g u g g g u g u g u c c a a g c u c a c g u

>H.sapiens ,gi|177793
494  496      500          506  508  510         517        523
(  ( (  . . . (  ( ( ( ( ( (  . )  ) )  . . . . . . )  ) ) ) ) ) )
u  u c  a u a c  g u a u u a  a g  a a  a c a u a c u  a a u g u g

>Methanococcusvoltae,gi|150042
379   382    385         391      395    398          405         411
(  ( ( (  . . (  ( ( ( ( ( (  . . . )  ) ) )  . . . . . . )  ) ) ) ) ) )
a  a u u  a a g  g c a a a a  u g a g  a u u  u g g u g c u  u u u g u c
```

Fig. 11. GPRM's results for the SRP data set.

was less complicated (shorter, no noise from the pseudogenes). We also tested this input with GeRNAMo and our results were unchanged from the full input case.

GPRM predicted two types of different motifs over these runs, represented by the following motifs:

1. **h5(5,9) ss(4,8) h5(5,6) ss(2,6) h3(5,6) ss(0,4) h3(5,9)**,
2. **h5(5,10) ss(2,7) h3(5,10) ss(6,10) h5(4,8) ss(0,2) h3(4,8).**

Both of GPRM's predicted motifs are very different from the IRE motif discovered by RNAProfile and predicted by GeRNAMo; GPRM fails to discover this motif. Fig. 14 displays an example of GPRM's results (corresponding to the second motif; GPRM's full results are given in the supplemental data).

4.4 microRNA

The fourth data set contains 25 microRNA precursor sequences from the human genome. This data set was derived from the Sanger microRNA database [16], [17], [36]. The database was searched for human microRNA precursors and reduced to 25 precursors containing four double-stranded elements. Each sequence in the data set is the microRNA surrounded by its flanking genomic sequence. Although containing the same number of complementary segments, the range of the elements' lengths is very large, a fact that leads to a general common motif. This was done in order to make it difficult for GeRNAMo to find a common motif. The sequences are given in Table 6.

The motif of the data set is:

ss(**0:4**) **h5**(**2:15**) ss(**0:4**) **h5**(**1:24**) ss(**0:3**) **h5**(**1:23**) ss(**0:3**) **h5**(**2:11**) ss(**3:14**) **h3**(**2:11**) ss(**0:9**) **h3**(**1:23**) ss(**0:3**) **h3**(**1:24**) ss(**0:5**) **h3**(**2:15**) ss(**0:5**).

We ran a total of 50 runs, of which most produced good common motifs. The best motif predicted by GeRNAMo is:

```
Iron Responsive Element

>NM_009653.1| Mus musculus aminolevulinic acid synthase 2
gGTTcGTCCTcagtgcAGGGCAACa
.(((.(((((......)))))))).                              (E: -7.2 Fitness: 5.7)

>NM_010240.1| Mus musculus ferritin light chain 1 (Ftl1)
cTTGcTTCAAcagtgtTTGAACGGa
.(((.(((((......)))))))).                              (E: -1.8 Fitness: 8.7)

>NM_010239.1| Mus musculus ferritin heavy chain (Fth)
cCTGcTTCAAcagtgcTTGAACGGa
.(((.(((((......)))))))).                              (E: -4.4 Fitness: 9.5)

>NM_000146.2| Homo sapiens ferritin, light polypeptide (FTL)
cTTGcTTCAAcagtgtTTGGACGGa
.(((.(((((......)))))))).                              (E: -1.3 Fitness: 8.5)

>NM_000032.1| Homo sapiens aminolevulinate, delta-, synthase 2
cGTTcGTCCTcagtgcAGGGCAACa
.(((.(((((......)))))))).                              (E: -7.5 Fitness: 7.3)

>L20941.1|HUMFERRITH Human ferritin heavy chain mRNA
cCTGcTTCAAcagtgcTTGGACGGa
.(((.(((((......)))))))).                              (E: -3.9 Fitness: 9.4)
________
>gi|806340:c862-1 H.sapiens (24) Ferritin H pseudogene
GTCAcTCAAttctTTGATGGC
((((.((((....))))))))                                  (E: -4.3 Fitness: -10.3)

>J04755.1|HUMFERHX Human ferritin H processed pseudogene
GAGacATTCTTcaccAAGAGTcCTC
(((..((((((....)))))).)))                              (E: -3.4 Fitness: -25.1)

>gi|806342|emb|X80336.1|HS5FERHPE H.sapiens (5) Ferritin H pseudogene
cTGAAtctTCCTtccttcGGGAcTTCAa
.((((...((((......)))).)))).                           (E: -4.2 Fitness: -13.8)
```

Fig. 12. RNAProfile's results for the IRE(2) data set.

```
>NM_009653.1| Mus musculus aminolevulinic acid synthase 2, erythroid (Alas2), mRNA
. 18 . . . 21. . . . . . . . . . . 32 . . . . 37 . . .
GGTTCGTCCTCAGTGCAGGGCAACA
.(((.(((((......)))))))).   (ΔG = -7.2)
------------------------------------------------------------
>NM_010240.1| Mus musculus ferritin light chain 1 (Ftl1), mRNA
. 33 . . . 37. . . . . . . . . . . 48. . . . . 53 . . .
CTTGCTTCAACAGTGTTTGAACGGA
.(((.(((((......)))))))).   (ΔG = -1.8)
------------------------------------------------------------
>NM_010239.1| Mus musculus ferritin heavy chain 1 (Fth1), mRNA
 . 33 . . .37 . . . . . . . . .  . .48 . . . . 53 . . .
CCTGCTTCAACAGTGCTTGAACGGA
.(((.(((((......)))))))).   (ΔG = -4.4)
------------------------------------------------------------
>NM_000146.3| Homo sapiens ferritin, light polypeptide (FTL), mRNA
. 30 . . . 34 . . . . . . . . . . . 45 . . . . 50 . . .
CTTGCTTCAACAGTGTTTGGACGGA
.(((.(((((......)))))))).   (ΔG = -1.3)
------------------------------------------------------------
>NM_000032.1| Homo sapiens aminolevulinate, delta-, synthase 2 (ALAS2), mRNA
. 13 . .  .17 . . . . . . . . . . . 28 . . . . 33 . . .
CGTTCGTCCTCAGTGCAGGGCAACA
.(((.(((((......)))))))).   (ΔG = -7.5)
------------------------------------------------------------
>L20941.1|HUMFERRITH Human ferritin heavy chain mRNA, complete cds
. 34 . .  .38 . . . . . . . . . . . 49 . . . . 54 . . .
CCTGCTTCAACAGTGCTTGGACGGA
.(((.(((((......)))))))).   (ΔG = -3.9)
```

Fig. 13. GeRNAMo's results for the IRE(2) data set (results for the three pseudogenes are not shown).

```
[5,10] (2,7) [5,10] (6,10) [4,8] (0,2) [4,8]

>NM_009653.1| Mus musculus aminolevulinic acid synthase 2, erythroid (Alas2), mRNA
21  25          32  36              45      5051    56
( ( ( ( ( . . . . . . ) ) ) ) ) . . . . . . . . ( ( ( ( ( ( ) ) ) ) ) )
g u c c u c a g u g c a g g g c a a c a g g a c u u u g g g c u c a g g

>NM_010240.1| Mus musculus ferritin light chain 1 (Ftl1), mRNA
232       238   241         247                   258   261   264   267
(  ( ( ( ( ( (  . . )  ) ) ) ) ) )  . . . . . . . . . . (  ( ( (  . . )  ) ) )
u  u c c a c c  g a g  g u g g a a  g c u g c c g u g a a  c c g  c c u  g g u

>NM_010239.1| Mus musculus ferritin heavy chain 1 (Fth1), mRNA
285               294   297                306         313     317   320     324
(  ( ( ( ( ( ( ( ( (  . . )  ) ) ) ) ) ) ) ) ) )  . . . . . . (  ( ( ( (  . . )  ) ) ) )
u  g u u a u u u u g  a c c  g a g a u g a u g  u g g c u c u  g a a g  a a c  u u u g

>NM_000146.3| Homo sapiens ferritin, light polypeptide (FTL), mRNA
17  21            29  33                42      4647    51
( ( ( ( ( . . . . . . . ) ) ) ) ) . . . . . . . . ( ( ( ( ( ) ) ) ) )
g c g g g u c u g u c u c u u g c u u c a a c a g u g u u u g g a c g

>NM_000146.3| Homo sapiens ferritin, light polypeptide (FTL), mRNA
17  21            29  33                    44    47    50    53
( ( ( ( ( . . . . . . . ) ) ) ) ) . . . . . . . . . . ( ( ( ( . . ) ) ) )
g c g g g u c u g u c u c u u g c u u c a a c a g u g u u u g g a c g g a

>NM_000032.1| Homo sapiens aminolevulinate, delta-, synthase 2
124   128             136     140           147           154155          162
(  ( ( ( (  . . . . . . . )  ) ) ) )  . . . . . . (  ( ( ( ( ( ( (  )  ) ) ) ) ) ) )
g  g u g g  u u a a g a c u  c a c c  a g u u c c u  g u u u g g u  a  u u g g a c g

>L20941.1|HUMFERRITH Human ferritin heavy chain mRNA, complete cds
38  42          49  53              62    65    68    71
( ( ( ( ( . . . . . . ) ) ) ) ) . . . . . . . . ( ( ( ( . . ) ) ) )
u u c a a c a g u g c u u g g a c g g a a c c c g g c g c u c g u u
```

Fig. 14. GPRM's results for the reduced IRE(2) data set.

TABLE 6
MicroRNA Data Set, Taken from the Sanger MIRbase [36]

Sequence name	Accession number	Genomic coordinates [a]
hsa-mir-1-1	MI0000651	20+ 60561907
hsa-leu-7a-3	MI0000062	22+ 44829097
hsa-mir-20b	MI0001519	X- 133029478
hsa-mir-29b-1	MI0000105	7-130019604
hsa-mir-30a	MI0000088	6- 72170096
hsa-mir-30c-2	MI0000254	6- 72143456
hsa-mir-92-1	MI0000093	13+ 90801518
hsa-mir-136	MI0000475	14+ 100420741
hsa-mir-155	MI0000681	21+ 25868112
hsa-mir-186	MI0000483	1- 71245471
hsa-mir-198	MI0000240	3- 121597317
hsa-mir-302a	MI0000738	4- 113927062
hsa-mir-362	MI0000762	X+ 49476557
hsa-mir-368	MI0000776	14+ 100575729
hsa-mir-371	MI0000779	19+ 58982690
hsa-mir-374	MI0000782	X- 73290264
hsa-mir-378	MI0000786	5+ 149092530
hsa-mir-380	MI0000788	14+ 100561056
hsa-mir-429	MI0001641	1+ 1144257
hsa-mir-493	MI0003132	14+ 100405099
hsa-mir-502	MI0003186	X+ 49482206
hsa-mir-519a-1	MI0003178	19+ 58947412
hsa-mir-520f	MI0003146	19+ 58877174
hsa-mir-520c	MI0003158	19+ 58902468
hsa-mir-527	MI0003179	19+ 58949033

[a] *Genomic coordinates are the location of each sequence in the genome.*

$$\mathrm{ss}(\mathbf{0{:}16})\ \mathbf{h5}(\mathbf{1{:}15})\ \mathrm{ss}(\mathbf{0{:}5})\ \mathbf{h5}(\mathbf{2{:}17})\ \mathrm{ss}(\mathbf{1{:}3})\ \mathbf{h5}(\mathbf{3{:}23})\ \mathrm{ss}(\mathbf{0{:}16})\ \mathbf{h5}(\mathbf{3{:}11})\ \mathrm{ss}(\mathbf{3{:}14})\ \mathbf{h3}(\mathbf{3{:}11})\ \mathrm{ss}(\mathbf{0{:}6})\ \mathbf{h3}(\mathbf{3{:}23})\ \mathrm{ss}(\mathbf{0{:}6})\ \mathbf{h3}(\mathbf{2{:}17})\ \mathrm{ss}(\mathbf{0{:}6})\ \mathbf{h3}(\mathbf{1{:}15})\ \mathrm{ss}(\mathbf{0{:}20}).$$

This motif has the same structure as the known motif and similar elements' length ranges (the main differences are in the length of the single strand elements and the length of the third stem).

Out of the 25 sequences in the data set, this motif occurs in 20 sequences at the correct positions. The reason that the motif does not occur in all of the sequences is that this data set contains sequences with a very wide range of lengths and GeRNAMo tries to reduce the generality of the motif (general individuals are penalized by the fitness function). As a consequence, it misses some marginal sequences with extreme length values. Out of the 20 occurrences, 17 occurrences completely match the known secondary structure (in terms of exact position in the sequence, element types, and lengths equal to those in the data set) and three occurrences have a base-pairing distance of 4. A sample output with comparison to the known secondary structure is given in Fig. 15 (for more detailed results, see the supplemental data). A typical secondary structure of four complementary segments is given in Fig. 16.

When we reduced the penalty for the motif's generality, GeRNAMo predicted a more general motif that occurred correctly in all 25 sequences. In the reduced-penalty case, GeRNAMo predicted the correct motif:

(1) .((((.[[[[[[[[[[[[[[[[..((((([[[.....]]]...))))).]]]]]]]]]]]]]]]].)))).

(2) .((((.[[[[[[[[[[[[[[[[..((((([[[.....]]]...))))).]]]]]]]]]]]]]]]].)))).

(a)

(1) ((((((((((((((.[[[.((((((((.[[[....]]]..))))))))]]]))))))))))))))

(2) ((((((((((((((.[[[.((((((((.[[[....]]]..))))))))]]]))))))))))))))

(b)

(1) (((.[[.((((((((((((((...[[[[[[........]]]]]].)))))))))))))).]].)))

(2) (((.[[.((((((((((((((...[[[[[[........]]]]]].)))))))))))))).]].)))

(c)

Fig. 15. MicroRNA sample output. Comparison between 1) the known secondary structure and 2) the occurrence of the evolved motif in the sequence. (a) hsa-mir-1-1. (b) hsa-mir-155. (c) hsa-mir-378.

$$\mathbf{ss(0:15)\ h5(1:21)\ ss(0:15)\ h5(2:25)\ ss(0:13)\ h5(0:25)\ ss(0:16)\ h5(0:13)\ ss(2:24)\ h3(0:13)\ ss(0:22)\ h3(2:25)\ ss(0:15)\ h3(2:25)\ ss(0:10)\ h3(1:21)\ ss(0:20)}.$$

This motif retains the same secondary structure as the more specific motif, but it varies in the length ranges of its elements, thus allowing it to occur at the correct position in all of the sequences in the data set.

We submitted this data set to GPRM, trying several parameter combinations to achieve good performance. Two of the motifs that were generated by GPRM were:

- **h5(3:20) ss(2:12) h5(9:20) ss(0:14) h3(3:20) ss(0:15) h5(3:7) ss(0:13) h5(3:6) ss(0:13) h3(9:20) ss(0:7) h3(3:7) ss(0:6) h3(3:6)**.
- **h5(6:14) ss(0:19) h3(6:14) ss(0:15) h5(16:35) ss(0:20) h5(6:13) ss(0:16) h3(16:35) ss(0:5) h5(5:14) ss(6:15) h3(6:13) ss(0:2) h3(5:14)**.

These motifs represent a pseudoknot secondary structure, which is a false positive. Fig. 17 shows a secondary structure of a pseudoknot, according to the first result generated by GPRM. These results show that GPRM fails to detect a common motif in the microRNA data set, whereas GeRNAMo succeeds convincingly.

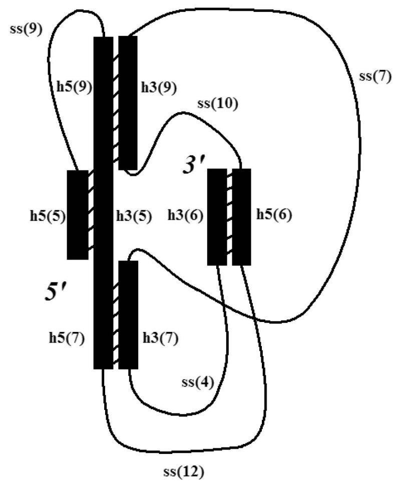

Fig. 17. A graphic presentation of the secondary structure given by: **h5(5) ss(9) h5(9) ss(0) h3(5) ss(0) h5(7) ss(12) h5(6) ss(10) h3(9) ss(7) h3(7) ss(4) h3(6)**, which folds to the motif that was predicted by GPRM (a single strand element that does not exist in the drawing, but complies with the motif's constraints—"ss(0:x)" is marked by "ss(0)" in the above structure). This is a pseudoknot secondary structure which is not known to exist in microRNAs.

In addition, we submitted this data set to RNAProfile [35]. We used several sets of parameters which included the default parameters and length parameters that comply with the microRNA motif length (maximum 100 nt.). RNAProfile was not successful at building a profile of the microRNA motif and the profiles it returned were of much smaller length. RNAProfile's results on this data set can be found in the supplemental data.

These results indicate that GPRM and RNAProfile are not capable of identifying long motifs in microRNA. GeRNAMo outperforms these methods in this regard, although RNAProfile is successful at building a profile for shorter motifs.

5 Discussion and Future Work

We described a genetic programming approach for finding a motif that is common to a set of given RNA sequences. Our experiments have shown that GeRNAMo is able to find common motifs of any size for a given data set. We compared our results with other approaches that deal with the same problem on the same data sets. We also compared our method with approaches that deal with a different problem and explained how the comparison was made.

Section 4.1, involving the Ferritin IRE-like data set (that was used in SLASH [15] and GPRM [24]), shows that the motif predicted by our approach is similar to the known motif. However, the motif occurs only in 43 out of 56 sequences without naive folding. This is due to the fact that we use RNAsubopt to fold the subsequences and the remaining sequences' secondary structure (according to RNAsubopt and also according to Mfold [49]) are different from this motif. RNAsubopt provides a good estimate for the best folding of a sequence by giving suboptimal solutions in the proximity of the minimum free energy, thereby providing a

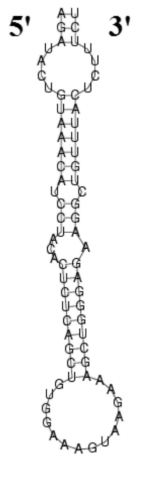

Fig. 16. The secondary structure of hsa-mir-30c-2 (MI0000254).

variety of candidates to choose from. Using thermodynamic values is important because naive folding does not always provide good estimates for the secondary structure. We chose ViennaRNA's RNAsubopt because it is a widely used folding program that was easier for us to utilize. However, our system is capable of using any folding program in the preprocessing stage. For this data set, we used naive folding in addition to RNAsubopt's results. The results were better with naive folding: 54 occurrences out of 56 in the Ferritin IRE-like example because the motif is short and simple.

Section 4.2, involving the SRP motif, shows that GeRNAMo was able to predict the motif that was designed by Fogel et al. without prior knowledge except for the maximum number of stems. Furthermore, GeRNAMo's results are identical to Fogel et al.'s top bin structures. GPRM failed to discover this motif.

Section 4.3, involving the ferritin IRE data set that was used in RNAProfile [35], shows that GeRNAMo was able to discover the common motif of the six real genes in the data set, despite the fact that this data set contained three more sequences that did not contain the IRE motif. GeRNAMo's motif and motif occurrences in the six genes are identical to RNAProfile's results. The results obtained by GPRM are very different from those of RNAProfile and GeRNAMo and it failed to discover this motif.

Both Sections 4.2 and 4.3 show that short sequences do not always fold according to the naive folding algorithm. RNAsubopt provided the correct results while the naive Watson-Crick base pairing—also used in GPRM—failed.

Section 4.4, involving the microRNAs data set, also shows that the motif predicted by our approach is similar to the known motif. Here, it is apparent that using a robust folding program is necessary. RNAsubopt provided similar results to the known secondary structure [36] and the reason that not all occurrences of the motif were found is that the data set contains a very broad range of stem-lengths, while GeRNAMo tries to increase the specificity of the motif. Despite the fact that not all occurrences were found and that this data set is general, a correct motif was found. When we reduced the penalty for the generality of the motif, a more general motif was found which occurred in all of the sequences in the data set. This demonstrates a notable trade-off between a general and more common motif and a less common but more specific motif.

In our present work, we have tried to improve upon previous works, especially that of GPRM [24]. Although we do not yet support pseudoknots, as does GPRM, GeRNAMo has the following advantages:

- GPRM uses only Watson-Crick base pairing to fold the sequences, while GeRNAMo can use any folding prediction method, in addition to an internal implementation of naive folding based on Watson-Crick base pairing and wobble interaction. Section 4.4 illustrates the significance of thermodynamic considerations in more complicated sequences as GPRM fails in the microRNA motif prediction (as well as the SRP and IRE2 motifs).
- GeRNAMo allows the user an option to choose between levels of generality of the motif by using different penalty functions for general motifs or not using a penalty function at all.
- GeRNAMo does not restrict the length of the RNA sequences in the data set. It does not limit the number of stems or each element's length in the motif. Thus, we take advantage of the flexibility that evolutionary computation allows.
- With regard to prior knowledge, both GPRM and GeRNAMo use the maximum number of stems in the motif. In addition, GPRM requires some knowledge of the length of the complementary and noncomplementary elements in the motif, while GeRNAMo only requires the range of lengths of the whole motif and uses the default of (15:100) when such input is not given.

There are a number of avenues we propose to pursue in the future for GeRNAMo's extension. The first is to support pseudoknots, which involves making some changes to the representation. The second is to improve the filtering methods made on the output, based on biological experience.

References

[1] W. Banzhaf, P. Nordin, R.E. Keller, and F.D. Francone, *Genetic Programming – An Introduction: On the Automatic Evolution of Computer Programs and Its Applications.* Morgan Kaufmann, 1997.

[2] G. Benedetti and S. Morosetti, "A Genetic Algorithm to Search for Optimal and Suboptimal RNA Secondary Structures," *Biophysical Chemistry,* vol. 55, pp. 253-259, 1995.

[3] J.H. Chen, S.Y. Le, and J.V. Maizel, "Prediction of Common Secondary Structures of RNAs: A Genetic Algorithm Approach," *Nucleic Acids Research,* vol. 28, pp. 991-999, 2000.

[4] R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids.* Cambridge Univ. Press, 1998.

[5] S. Eddy and R. Durbin, "RNA Sequence Analysis Using Covariance Models," *Nucleic Acids Research,* vol. 22, pp. 2079-2088, 1994.

[6] S.R. Eddy, "Computational Genomics of Noncoding RNA Genes," *Cell,* vol. 13, pp. 137-140, 2002.

[7] S.R. Eddy, "How Do RNA Folding Algorithms Work?" *Nature Biotechnology,* vol. 22, pp. 1457-14588, 2004.

[8] I. Titov et al., "A Fast Genetic Algorithm for RNA Secondary Structure Analysis," *Russian Chemistry Bull.,* vol. 51, pp. 1135-1144, 2002.

[9] D.B. Fogel, *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence.* IEEE Press, 2000.

[10] G.B. Fogel, "The Application of Evolutionary Computation to Selected Problems in Molecular Biology," *Proc. Sixth Int'l Conf. Evolutionary Programming,* pp. 23-33, 1997.

[11] G.B. Fogel, V.W. Porto, D.G. Weekes, D.B. Fogel, R.H. Griffey, J.A. McNeil, E. Lesnik, D.J. Ecker, and R. Sampath, "Discovery of RNA Structural Elements Using Evolutionary Computation," *Nucleic Acids Research,* vol. 30, no. 23, pp. 5310-5317, 2002.

[12] G.B. Fogel, D.G. Weekes, R. Sampath, and D.J. Ecker, "Parameter Optimization of an Evolutionary Algorithm for RNA Structure Discovery," *Proc. 2004 IEEE Congress Evolutionary Computation,* pp. 607-613, June 2004.

[13] Z. Gdaniec, H. Sierzputowska-Gracz, and E.C. Theil, "Iron Regulatory Element and Internal Loop/Bulge Structure for Ferritin mRNA Studied by Cobalt(iii) Hexamine Binding, Molecular Modeling and nmr Spectroscopy," *Biochemistry,* vol. 37, pp. 1505-1512, 1998.

[14] J. Gorodkin, L.J. Heyer, and G.D. Stormo, "Finding the Most Significant Common Sequence and Structure Motifs in a Set of RNA Sequences," *Nucleic Acids Research,* vol. 25, pp. 3724-3732, 1997.

[15] J. Gorodkin, S.L. Stricklin, and G.D. Stormo, "Discovering Common Stem-Loop Motifs in Unaligned RNA Sequences," *Nucleic Acids Research,* vol. 29, pp. 2135-2144, 2001.

[16] S. Griffiths-Jones, "The microRNA Registry," *Nucleic Acids Research,* vol. 32, pp. D109-D111, 2004.

[17] S. Griffiths-Jones, R.J. Grocock, S. can Dongen, A. Bateman, and A.J. Enright, "miRBase: microRNA Sequences, Targets and Gene Nomenclature," *Nucleic Acids Research,* vol. 34, pp. D140-D144, 2006.
[18] A.P. Gultyaev, F.H.D van Batenburg, and C.W.A. Pleij, "The Computer Simulation of RNA Folding Pathways Using a Genetic Algorithm," *J. Molecular Biology,* vol. 250, pp. 37-51, 1995.
[19] A.P. Gultyaev, F.H.D van Batenburg, and C.W.A. Pleij, "Dynamic Competition between Alternative Structures in Viroid RNAs Simulated by an RNA Folding Algorithm," *J. Molecular Biology,* vol. 276, pp. 43-55, 1998.
[20] R.R. Gutell, J.C. Lee, and J.J. Cannone, "The Accuracy of Ribosomal RNA Comparative Structure Models," *Current Opinion in Structural Biology,* vol. 12, pp. 301-310, 2002.
[21] I.L. Hofacker, "Vienna RNA Secondary Structure Server," *Nucleic Acids Research,* vol. 31, no. 13, pp. 3429-3431, 2004.
[22] I.L. Hofacker, M. Fekete, C. Flamm, M.A. Huynen, S. Rauscher, P.E. Stolorz, and P.F. Stadler, "Automatic Detection of Conserved RNA Structure Elements in Complete RNA Virus Genomes," *Nucleic Acids Research,* vol. 26, pp. 3825-3836, 1998.
[23] I.L. Hofacker, W. Fontana, P.F. Stadler, L.S. Bonhoeffer, M. Tacker, and P. Schuster, "Fast Folding and Comparison of RNA Secondary Structures," *Monatshefte Fur Chemie,* vol. 125, pp. 167-188, 1994.
[24] Y.J. Hu, "Prediction of Consensus Structural Motifs in a Family of Coregulated RNA Sequences," *Nucleic Acids Research,* vol. 30, no. 17, pp. 3886-3893, 2002.
[25] Y.J. Hu, "GPRM: A Genetic Programming Approach to Finding Common RNA Secondary Structure Elements," *Nucleic Acids Research,* vol. 31, no. 13, pp. 3446-3449, 2003.
[26] Y. Ji, X. Xu, and G.D. Stormo, "A Graph Theoretical Approach for Predicting Common RNA Secondary Structure Motifs Including Pseudoknots in Unaligned Sequences," *Bioinformatics,* vol. 20, pp. 1591-1602, 2004.
[27] J.R. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, 1992.
[28] J.R. Koza, M.A. Keane, M.J. Streeter, W. Mydlowec, J. Yu, and G. Lanza, *Genetic Programming IV: Routine Human-Competitive Machine Intelligence.* Kluwer Academic, 2003.
[29] E.A. Lesnik, G.B. Fogel, D. Weekes, T.J. Henderson, H.B. Levene, R. Sampath, and D.J. Ecker, "Identification of Conserved Regulatory RNA Structures in Prokaryotic Metabolic Pathway Genes," *Biosystems,* vol. 80, pp. 145-154, 2005.
[30] S. Luke, *ECJ: A Java-Based Evolutionary Computation and Genetic Programming Research System,* http://www.cs.umd.edu/projects/plus/ec/ecj/, 2007.
[31] T. Macke, D. Ecker, R. Gutell, D. Gautheret, D.A. Case, and R. Sampath, "RNAMotif—A New RNA Secondary Structure Definition and Discovery Algorithm," *Nucleic Acids Research,* vol. 29, pp. 4724-4735, 2001.
[32] D.H. Mathews, J. Sabina, M. Zuker, and D.H. Turner, "Expanded Sequence Dependence of Thermodynamic Parameters Improves Prediction of RNA Secondary Structure," *J. Molecular Biology,* vol. 288, pp. 911-940, 1999.
[33] D.J. Montana, "Strongly Typed Genetic Programming," *Evolutionary Computation,* vol. 3, no. 2, pp. 199-230, 1995.
[34] R. Nussinov and A. Jacobson, "Fast Algorithm for Predicting the Secondary Structure of Single Stranded RNA," *Proc. Nat'l Academy of Sciences USA,* vol. 77, no. 11, pp. 6309-6313, 1980.
[35] G. Pavesi, G. Mauri, M. Stefani, and G. Pesole, "RNAProfile: An Algorithm for Finding Conserved Secondary Structure Motifs in Unaligned RNA Sequences," *Nucleic Acids Research,* vol. 32, no. 10, pp. 3258-3269, 2004.
[36] *Sanger MIRbase,* http://microrna.sanger.ac.uk/sequences/index.shtml, 2007.
[37] D. Sankoff, "Simultaneous Solution of the RNA Folding, Alignment and Protosequence Problems," *SIAM J. Applied Math.,* vol. 45, pp. 810-825, 1985.
[38] B.A. Shapiro, D. Bengali, W. Kasprzak, and J.C. Wu, "RNA Folding Pathway Functional Intermediates: Their Prediction and Analysis," *J. Molecular Biology,* vol. 312, pp. 27-44, 2001.
[39] B.A. Shapiro and J. Navetta, "A Massively Parallel Genetic Algorithm for RNA Secondary Structure Prediction," *J. Supercomputing,* vol. 8, pp. 195-207, 1994.
[40] B.A. Shapiro, J.C. Wu, D. Bengali, and M.J. Potts, "The Massively Parallel Genetic Algorithm for RNA Folding: MIMD Implementation and Population Variation," *Bioinformatics,* vol. 17, pp. 137-148, 2001.
[41] S. Siebert and R. Backofen, "Marna: Multiple Alignment and Consensus Structure Prediction of RNAs Based on Sequence Structure Comparisons," *Bioinformatics,* vol. 21, pp. 3352-3359, 2005.
[42] M. Sipper, *Machine Nature: The Coming Age of Bio-Inspired Computing.* McGraw-Hill, 2002.
[43] E.C. Theil, "The Iron Responsive Element (IRE) Family of mRNA Regulators. Regulation of Iron Transport and Uptake Compared in Animals, Plants and Microorganisms," *Metal Ions in Biological Systems,* vol. 35, pp. 403-434, 1998.
[44] K.C. Wiese and A. Hendriks, "Comparison of p-rnapredict and mfold Algorithms for RNA Secondary Structure Prediction," *Bioinformatics,* vol. 22, pp. 934-942, 2006.
[45] C. Witwer, S. Rauscher, I.L. Hofacker, and P.F. Stadler, "Conserved RNA Secondary Structures in Picornaviridae Genomes," *Nucleic Acids Research,* vol. 29, pp. 5079-5089, 2001.
[46] S. Wuchty, W. Fontana, I.L. Hofacker, and P. Schuster, "Complete Suboptimal Folding of RNA and the Stability of Secondary Structures," *Biopolymers,* vol. 49, pp. 145-165, 1999.
[47] Z. Yao, Z. Weinberg, and W.L. Ruzzo, "Cmfinder a Covariance Model Based RNA Motif Finding Algorithm," *Bioinformatics,* vol. 22, pp. 445-452, 2006.
[48] M. Zuker, "On Finding All Suboptimal Foldings of an RNA Molecule," *Science,* vol. 244, pp. 48-52, 1989.
[49] M. Zuker, "Mfold Web Server for Nucleic Acid Folding and Hybridization Prediction," *Nucleic Acids Research,* vol. 31, no. 13, pp. 3406-3415, 2003.

Shahar Michal received the BSc degree from the Department of Computer Science at Ben-Gurion University in 2005. Since October 2005, he has been a graduate student in the Department of Computer Science at Ben-Gurion University. His research interests include genetic programming, bioinformatics, RNA structure predictions, and scientific computing. Currently, he is working at Orbotech, Israel, as a software engineer.

Tor Ivry received the BSc degree from the Department of Computer Science at Ben-Gurion University in 2005. Since October 2005, he has been a graduate student in the Department of Computer Science at Ben-Gurion University. His research interests include bioinformatics, RNA structure predictions, and scientific computing. Currently, he is working at Applied Materials, Israel, as a software engineer.

Omer Schalit-Cohen received the BSc degree from the Department of Computer Science at Ben-Gurion University in 2005. Currently, he is working at Medcon, Israel, as an application engineer.

Moshe Sipper received the BA degree from the Technion-Israel Institute of Technology and the MSc and PhD degrees from Tel Aviv University, all in computer science. He is currently an associate professor in the Department of Computer Science, Ben-Gurion University, Israel. During the years 1995-2001, he was a senior researcher at the Swiss Federal Institute of Technology in Lausanne. Dr. Sipper's major research interests include evolutionary computation, bio-inspired computing, and artificial life; minor interests include cellular computing, cellular automata, and artificial self-replication, along with a smidgen of evolutionary robotics, artificial neural networks, and fuzzy logic. Dr. Sipper has published close to 120 scientific papers and is the author of two books: *Machine Nature: The Coming Age of Bio-Inspired Computing* and *Evolution of Parallel Cellular Machines: The Cellular Programming Approach*. He is an associate editor of the *IEEE Transactions on Evolutionary Computation* and an editorial board member of *Genetic Programming and Evolvable Machines*.

Danny Barash received the PhD degree in applied science in 1999 from the University of California at Davis. From 1999 to 2001, he was employed at Hewlett Packard Laboratories at the Technion, Israel, pursuing research on image processing and computer vision. From 2001 to 2003, he was a Howard Hughes Medical Institute postdoctoral fellow at New York University and a research fellow at the Institute of Evolution at the University of Haifa, Israel, where he made a transition to computational biology. Since 2004, he has been with the Department of Computer Science at Ben-Gurion University, where he is currently an assistant professor in bioinformatics. His secondary affiliation is with the Institute of Evolution at Haifa University. His research interests include computational biology, RNA structure predictions, computational imaging, and numerical analysis. He is a member of the IEEE.