Bull. Nov. Comp. Center, Comp. Science, 37 (2014), 149–162

SWORD: Genetic algorythm tool for protein-RNA interaction motifs recognition

D.N. Shtokalo, D.S. Miginsky, V.P. Lobanov, G.C. St.Laurent III

Abstract. Recognition of potential for protein-RNA interaction is an important problem in bioinformatics. The solution may present a clue for understanding gene regulation. Formalization of the problem leads to in silico search for a complex motif in the 15-letter UIPAC alphabet in RNA sequences considering their secondary structure. The genetic algorithm with island modification was used to solve the related discrete optimization problem. The algorithm named SWORD was implemented on GPU and CPU. The comparison has shown a significant performance advantage of the GPU implementation. The algorithm was applied for searching RNA-motifs interacting with Hu antigen R (HuR) protein. The result achieved is better than that obtained in the previous work based on fitness-function criterion.

Keywords: GPU computing, parallel genetic algorithm, protein-RNA interaction, RNA motif recognition.

1. RNA motifs search problem

Modeling of molecular biology processes is a very important problem. Nowadays a huge amount of experimental data needs to be analyzed properly to extract new knowledge for medicine and diagnostic tests development. Formalization of molecular biology processes in living cells leads to the modeling of complex digital-analogue systems. Digital information coded in DNA provides instructions to cells on metabolism, proliferation, virus response and even cell death. Spatial organization of DNA, RNA molecules and proteins is not random and serves for analogue way signal transmission.

A DNA molecule can be formally represented as a sequence in the four-letters alphabet {A,C,G,T}, denoting the types of nucleotides connected in the linear order. The total length of human DNA sequences is about 3×10^9 nucleotides. The central paradigm of molecular biology states that parts of DNA, called genes, can be transcribed to RNA molecules. During the transcription, new RNA molecules are created. They consist of four types of nucleotides A, C, G, U connected in a linear order. A nucleotide sequence of RNA copies a gene sequence of DNA, with the only exception: T nucleotide is substituted with U. As an example, here is a sequence of RNA coding pre-mature miRNA *hsa-let-7a-1*: "UGGGAU GAGGUAGUAGGUUGUAUAGUUUUAGGGUCACACCCACCACCUGG GAGAUAAUAUACAAUCUACUGUCUUUCCUA". Dozens of thousands

of genes serve as templates for the millions of RNA copies. During post-transcription regulation, RNA molecules undergo essential transformation called maturation. Mature mRNA (messenger RNA) molecules are transported from the cell nucleus to the cytoplasm, where they serve as templates for protein coding.

In a healthy organism, a cell supports the necessary level of the transcriptional activity of each gene. Transcription as well as post-transcriptional regulation of genes is implemented through protein-DNA, protein-RNA and RNA-DNA interactions with the presence of catalytic and signaling molecules. Hundreds of thousands of types of proteins and millions of types of RNA molecules interact with each other, implementing the program of cell life.

Spatial organization of RNA molecules implemented through the affinity of A to U, C to G and U to G provides the secondary structure, or "shape". Hypothetically, one RNA molecule X of length L may have many secondary structures $S(X)$. Each secondary structure $s \in S(X)$ can be described by the symmetric bit matrix $s_{L\times L}$:

$$\begin{gathered} s_{i,j} = s_{j,i}, \\ s_{i,j} = 0,\ i = j, \\ s_{i,j} = 1 \text{ if } (X(i), X(j)) \in \{(A,U),(U,A),(C,G),(G,C),(G,U),(U,G)\}, \\ 1 \le i < j \le L, \\ \sum_{j=1}^{L} s_{i,j} = 1,\ 1 \le i \le L. \end{gathered}$$

Chemical interactions between nucleotides and the tension of the geometrical form of an RNA molecule can be characterized by the free energy function. It is usually assumed that an RNA molecule chooses a secondary structure with the minimum free energy (MFE) or the most likely structure in the Boltzmann ensemble.

Parts of RNA molecules that may interact with proteins are called protein interaction motifs. It is assumed that the shape provides the analogue part of a motif. The specific nucleotide content at the proper positions of an RNA molecule provides the digital part of a motif. Well known examples of digital-analogue information coupling in protein-RNA interactions (e.g. ribosomal complexes, LIN28-DICER [1], dsRNA-ADAR [2]) allow us to assume protein-RNA interaction motifs to be a common feature of RNAs.

Knowledge about the RNA motifs structure will provide a better understanding of gene regulation mechanisms, design novel diagnostics tests and medicines. In spite of recent progress in molecular biology, many mechanisms of gene regulation are understood poorly. Direct experimental identification of protein-RNA interaction motifs is quite an expensive procedure. Theoretical prediction of motifs faces combinatorial complexity problems. There are plenty of computational tools and methods [3] aimed at motif search, like MEME, PWM, RNAMOT, RNAForester [4-6] and others. In

practice, the capabilities of the methods are limited by exponential complexity or irrelevance of mathematical solution space to physical process.

In theory, the basic knowledge of chemical and physical characteristics of nucleotides and amino-acids is sufficient to derive the properties of RNA and protein polymers. In practice, the prediction of such characteristics as spatial conformation, domain structure and affinity represent an NP-complete problem. The computation of the physical properties of RNA and protein molecules using modern computers may take days or years of computational time (depending on the molecule length). Due to its complexity, the motif search strategy does not usually require a full knowledge about the physical properties of RNA or protein molecules. Only physical principles of the first magnitude order (like the complementary law) in combination with indirect characteristics (like environmental conditions) observed in the experiment are utilized. Then the machine learning algorithms like SVM, Neural networks, Random forest [7-9] and others are used to recognize patterns in sample molecules by comparing them with the control samples.

In [10], a method was presented for efficient solving the RNA-protein motif search problem under certain assumptions. The secondary structure of RNA is assumed to provide the analogue level of affinity to a specific protein domain. The nucleotide content of RNA provides the digital-like level of affinity. The research was based on the analysis of the sequences of 2243 RNA molecules forming a complex with HuR (Human antigene R) protein. HuR protein is the key post-transcriptional regulator that takes part in RNA stabilization and transporting. It is involved in stress response and inflammation. HuR dysfunction is associated with cancer [11]. With the help of the microarray technology, RNA molecules interacting with HuR, as well as the control RNAs not interacting with HuR, were found. Locally stable secondary structures were found using the RNAFold [12] program in the sliding window 75 nt

$$\{s \in S(X_i[j, j+74]), i = 1, \ldots, 2243, j = 1, \ldots, L(X_i) - 74|$$
$$FE(s) = \min_{x \in S(X_i[j,j+74])}\{FE(x)\}\},$$

and filtered against duplicates, where FE is free energy function [12], $L(X_i)$ is the length of sequence X_i, and $S(X_i[j, j+74])$ is all possible structures of the subsequence of X_i RNA molecule.

Clustering of structures [13] revealed HuR interacting RNA molecules were enriched with two hairpin structures of 66-75 nt length. Then a specially designed parallel version of the genetic algorithm SWORD (superword) was tuned and applied to find the specific nucleotide content enriching or depleting the sequences of two hairpin structures.

The SWORD algorithm finds statistically over-represented and under-represented groups of words in the primary sequences of RNA molecules

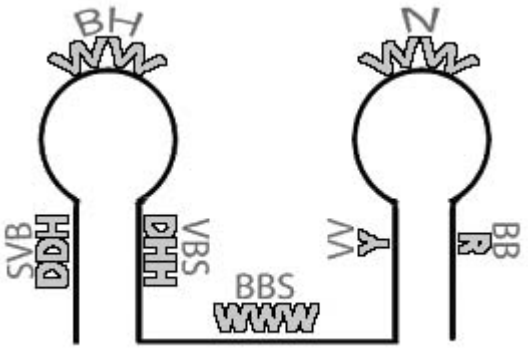

Figure 1. HuR interaction RNA motif coupling secondary structure and primary sequence content

folded in 2-hairpin structures [10]. The primary sequence of RNA is virtually split into 7 fragments according to the 2-hairpin structure (left stem, loop, right stem, gap, left stem, loop, and right stem). Each fragment is assumed to comprise a specific "word" recognized by HuR protein while interacting with RNA. A group of 7 words that simultaneously appear in the 7 fragments of the primary sequence of RNA is termed as "super-word". Figure 1 displays two hairpin structures with statistically enriched (thick letters) and depleted (thin letters) primary sequence content in 15-letter alphabet (Table 1). This motif was found by the SWORD algorithm [10] in RNA molecules interacting with HuR protein.

However, the analogue part of a signal (shape) can potentially be presented by one hairpin with or without bulges (e.g. pre-miRNA), clover (e.g. tRNA) or other shapes. In [10], the maximum length of each word was restricted to six nucleotides. Even the application of these significant restrictions allows us to narrow the search space to just nearly 15^{42} combinations, where 42 stands for 7 words of maximum 6 letters each and 15 is number of letters in alphabet. The optimized and enhanced SWORD tool based on the genetic algorithm is presented below. The advantage of GPU as in [14] was taken. The extended user interface allows an arbitrary shape of RNA to be analyzed. These new features increase the applicability of the original SWORD to different motif search problems.

2. Model

The experimental data is represented by two sets of sequences in the 4-letter alphabet {A,C,G,U}. The first one is positive, i.e. its sequences are interacting with the target protein. The second is negative or control, i.e.

Table 1. UIPAC alphabet

Letter	**Value**	**Letter**	**Value**
A	Adenosine	**M**	A or C
C	Cytosine	**S**	G or C
G	Guanine	**W**	A or U
U	Uracil	**B**	G or U or C
R	G or A	**D**	G or A or U
Y	U or C	**H**	A or C or U
K	G or U	**V**	G or C or A
		N	A or C or G or U

sequences are not interacting with the protein. We will denote these two sets as *pos* and *con*, respectively. The sequence $\boldsymbol{\alpha}$ matches the sequence $\boldsymbol{\beta}$ iff $\boldsymbol{\alpha}$ is a subsequence of the sequence $\boldsymbol{\beta}$. However, a 4-letter representation is too strict to solve the problem.

The nature of protein-RNA interactions allows different variants and combinations of nucleotides to interact with proteins. It is very unlikely to find an exact sequence that matches the most positive sequences and does not match the most control ones. So, we consider sequences in the 15-letter IUPAC alphabet that includes all possible non-empty combinations besides the concrete nucleotides (see Table 1). For example, **R** means that on a given position there should be either G or A; **D** means either G, A, or U; **N** means any nucleotide. A letter in the 15-letter alphabet matches all its possible options, so such sequences could be considered as primitive regular expressions (without repeats, i.e. with pre-determined lengths). From this point on, the match relationship will be considered in a generalized form: $\boldsymbol{\alpha}$ (15-letter alphabet) matches $\boldsymbol{\beta}$ (4-letter alphabet) iff there is a subsequence $\boldsymbol{\beta'}$ of $\boldsymbol{\beta}$ of the same length as $\boldsymbol{\alpha}$ such that $\boldsymbol{\alpha}[\boldsymbol{i}]$ matches $\boldsymbol{\beta'}[\boldsymbol{i}]$ for any $i = 1, .., L(\boldsymbol{\alpha})$. Hereafter, the terms *sequence* and *word* will be used interchangeably.

The problem could be formalized as $F(\boldsymbol{\alpha}) \to max$, where F is a fitness-function that could be defined in different ways. In this paper, the following definition is considered:

$$F(\boldsymbol{\alpha}) = \frac{N_{pos}(\boldsymbol{\alpha})}{|pos|} - \frac{N_{con}(\boldsymbol{\alpha})}{|con|}, \tag{1}$$

where $N_{pos}(\boldsymbol{\alpha})$ and $N_{con}(\boldsymbol{\alpha})$ are numbers of sequences that $\boldsymbol{\alpha}$ matches in the positive and control set respectively; $|pos|$ and $|con|$ are cordialities of sets.

This is the problem formalization for the simple motifs, where the sec-

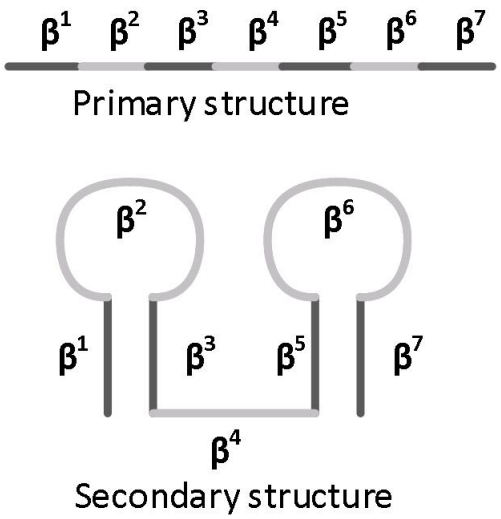

Figure 2. Primary and secondary structure of an RNA molecule

ondary structure of RNA is irrelevant. Figure 2 illustrates the possible primary and secondary structures of RNA.

In this particular case, $\boldsymbol{\beta^1}$ should be complementary to the reversed $\boldsymbol{\beta^3}$, and $\boldsymbol{\beta^5}$ should be complementary to the reversed $\boldsymbol{\beta^7}$. In other words, $\boldsymbol{\beta^1}$ and $\boldsymbol{\beta^3}$ as well as $\boldsymbol{\beta^5}$ and $\boldsymbol{\beta^7}$ form duplexes. Bulges (part of a sequence without a complement in a duplex partner) inside duplexes are allowed. To take into account the secondary structure, let us consider a "super-word" instead of a simple word. The super-word $\boldsymbol{\alpha = \alpha^1\alpha^2 \ldots \alpha^n}$ is a sequence of several words. The super-word $\boldsymbol{\alpha}$ matches $\boldsymbol{\beta}$ iff $\boldsymbol{\alpha^i}$ matches $\boldsymbol{\beta^i}$, $i = 1, \ldots, n$.

Thus, we have a discrete optimization problem with additional constraints. The constraints are the following: the structure of a super-word is fixed by the user, i.e. there is a fixed number of parts (seven in case of a two hairpin structure), and there are fixed constraints about complementary relationships between the words. The lengths of sub-words and super-words are arbitrary except that each sub-word should have a non-zero length. However, there could be additional constraints as a result of the limitations of particular optimization methods.

3. Solver

To find the sub-optimal solution in the 15^L combinatorial space, the population-based algorithm family was considered. Here

$$L = \sum_{i=1}^{n} \max_{j=1,\ldots,|pos|} \boldsymbol{\beta_j^i}$$

is the sum of the maximum lengths of the RNA fragments parts of a motif, and 15 is the number of letters in the alphabet (Table 1). In the case of HuR, the RNA length of the motif L could be 75 (the length of the 2-hairpin structure) [10]. The general idea implies all representatives of this family support the population of solutions and evolve this population through the algorithm steps. The two very important characteristics of this family are:

1. Natural by-data parallelism even if computation of a target function could not be performed in a parallel way.
2. Relative resistance to the local extrema.

The population evolving laws vary for different methods and usually relate to some kind of a biological or ethological metaphor. For example, the particle swarm optimization (PSO) algorithm [15], one of the best known algorithms of this family, is based on the behavior of a swarm of flying insects. The Bees Algorithm [16] is based on the natural foraging behavior of bees. The genetic algorithm [17,18] is based on natural organism selection and evolution on the genetic level.

To solve the problem the genetic algorithm (GA) was chosen. The main reasons are high customizability and, with proper expansion [19], the possibility of working effectively on computing with loosely coupled components like computer clusters and GPUs (see Island Expansion section below for details).

In fact, the GA gives only a kind of a meta-structure of the optimization algorithm, and many details could and must be customized regarding a particular problem. In general, the possible solution vector (or tuple, because components could be of different types) is considered as the "genotype", and the target function called the fitness-function in case of the GA is considered as the level of fitness of the organism to the environment. In ecology, better fitness means better survivability and more offsprings with similar genotypes. The subsequent generation (the population state of the next algorithm step) is more likely to contain organisms with better fitness. "More likely" means that all the processes in computing the subsequent generation are stochastic, and some organisms should be generated completely randomly to make some "noise" in the population important for the resistance to local extrema. To compute the next generation, the following techniques could be generally implemented:

1. Mutate the best part of a population (regarding the fitness-function value) to produce part of the next generation. Mutation is usually a

user-defined random modification of solution vector components. The probability of mutation greatly affects the convergence rate. In general, a higher rate means a better convergence, but in some cases like the stiff problems it could completely skip some extrema. So balancing is required regarding a particular problem to be solved.

2. (Optional) Crossover of random organisms from the best part to produce part of the next generation. Crossover is a user-defined operator of the type $SP^2 \to SP$ (where SP is solution space), but also degrees higher than 2 could be used. It is optional but is used in most implementations because it helps to avoid local extrema.
3. (Optional) Preserve the elite organisms (the best of the best), i.e. move them to the next generation without any changes. It guarantees that the last generation will always contain the best solution found.
4. (Optional) Generate random organisms for the remaining part of the next generation. It is optional and sometimes could be replaced with higher mutation probabilities.

To use the genetic algorithm, the mutation and crossover operators must be defined, in addition to the optimized function.The super-words are considered as organisms. The individual sub-words within a super-word are mutated independently except for the ones participating in complementary constraints. In RNA, the structures $\boldsymbol{\alpha^1}$ and $\boldsymbol{\alpha^3}$ as well as $\boldsymbol{\alpha^5}$ and $\boldsymbol{\alpha^7}$ are mutated synchronously, so only five independent sub-words should be taken into account. There are also limitations to the size of each sub-word. First, each sub-word must have a non-zero size, which is an obvious natural limitation. The maximum sub-word size is an artificial limitation, but it should be set to run the genetic algorithm (and possibly any other optimization method) predictably and effectively. We have used the limits of 6-8, which looks fair enough considering the results obtained. Taking into account all the constraints and limitations, a sub-word can have the following mutations:

1. Random letter replacement in a random position (in the 15-letter alphabet).
2. Remove letter in the right-most or left-most position.
3. Add a random letter to the beginning or end of a sub-word.
4. Shift left or right (remove a letter on the one side and add random one on the other side), which is technically a combination of 2) and 3).

Each mutation has its own probability that affects the algorithm convergence rate.

Crossover is crucial to avoid local maximums. The crossover operation for two organisms must be applied for the correspondent pairs of subwords individually (except for the ones participating in complementary constraints). To perform the crossover for two sub-words, they should be aligned first (in case their lengths are different). To implement aligning we have used the following encoding. Each position is represented by 4 bits, one for each of the 4 basic letters {A,C,G,U}. So U will be represented by 0100, **R** by 1010, **N** by 1111, and so on. Let $\boldsymbol{\alpha}$ be not longer than $\boldsymbol{\beta}$. For $\boldsymbol{\beta'}$, which is a subsequence of $\boldsymbol{\beta}$ with the same length as $\boldsymbol{\alpha}$, the quality of alignment is the number of bits with '1' value in $\mathbf{BITAND}(\boldsymbol{\alpha}, \boldsymbol{\beta'})$. The alignment procedure is the following:

1. Find position of the alignment $\boldsymbol{\beta'}$ in $\boldsymbol{\beta}$ with the best quality.
2. Build the extension $\boldsymbol{\alpha^{ext}}$ of $\boldsymbol{\alpha}$ such that the length of $\boldsymbol{\alpha^{ext}}$ is the same as of $\boldsymbol{\beta}$, $\boldsymbol{\alpha}$ is a subsequence of $\boldsymbol{\alpha^{ext}}$ with the same position as $\boldsymbol{\beta'}$ in $\boldsymbol{\beta}$, and all other symbols in $\boldsymbol{\alpha^{ext}}$ are **N.**

For two words of the same length, the crossover operation is rather simple. For each position of the result $\boldsymbol{\gamma_i}$, we take randomly either $\boldsymbol{\beta_i}$ or $\boldsymbol{\alpha_i^{ext}}$.

The genetic algorithm step builds a subsequent generation considering the previous one (the cardinality of population is constant). In our case, it includes the following:

1. Order the organisms by their fitness-function value, determine the elite and good organisms by two thresholds. $\boldsymbol{N_{elite}}$ organisms with the best fitness-function are considered as elite, and $\boldsymbol{N_{good}} \geq \boldsymbol{N_{elite}}$ are considered as good (the elite organism is always a good one).
2. Move elite organisms to the next generation without changes.
3. Move good but not elite organisms to the next generation with possible mutations.
4. Some organisms (their number is determined by an additional threshold) of the remaining part of the population are computed as a result of crossover between random pairs of good organisms.
5. The other organisms are computed as random super-words.

Unfortunately, there is no good convergence criterion, when an algorithm should be stopped. It can be stopped after reaching a pre-determined number of generations, a certain value of the fitness-function, or even manually.

4. Island extension

In the previous paper [10], it was shown that the computation times necessary to achieve satisfactory results are too long (hours to days) to search

for the ways to improve the performance. A possible way is to use parallel computing, which is very natural for the genetic algorithm. For most operations, by-data parallelism can be used because each organism (or a pair of them in case of crossover) can be processed independently. The only exception is sorting a generation (generation sorting), but it is irrelevant because the other parts are much more complex. Unfortunately, after each stage, the synchronization of data between processes is required, and in fact, only SMP-like systems can be used effectively without the algorithm modification.

In [19], the island modification of the genetic algorithm is described, which can be used effectively, for instance, with computer clusters. The general idea is that several relatively independent populations (islands) are being computed simultaneously (usually on different nodes of a cluster). Some synchronizing occurs only in fixed time points, usually once per several hundred generations. Islands exchange their best organisms with other islands in synchronizing points, while evolving independently all the other time.

In this paper, we used island modification to solve the problem with the genetic algorithm on a GPU. The GPU architecture is closer to MPP-systems than to SMP. There are several computing units (usually 10-20 or more for expensive GPUs), each with several computing cores and its own memory block. There can be shared memory for all the units, but it is much slower. The genetic algorithm can be implemented much more effectively on a GPU by computing a separate island on a separate computing unit.

5. Performance

In implementation OpenCL framework is used, mainly because it utilizes the GPUs of different vendors and the same code can be run on CPU utilizing all the available cores. For testing purposes, we have used GPU AMD Radeon 7870 and CPU AMD FX 8320, which is a rather common desktop machine. Figure 3 shows the results of comparison of performance on GPU and CPU depending on the population size. For larger populations, there is about ten times difference. The oscillation-like effect on GPU curve can be probably explained by the fact that it should work with the highest efficiency when the population size is divisible by the computing cores of GPU. Otherwise, some cores are not fully loaded.

6. Results and discussion

In [10], the super-word “DDH - WW - DHH - WWW - Y - WW - R” was found in 5 hours on an 8-CPU computer cluster using only CPUs. The value of the fitness-function was 0.45. The GPU-based algorithm was able to find

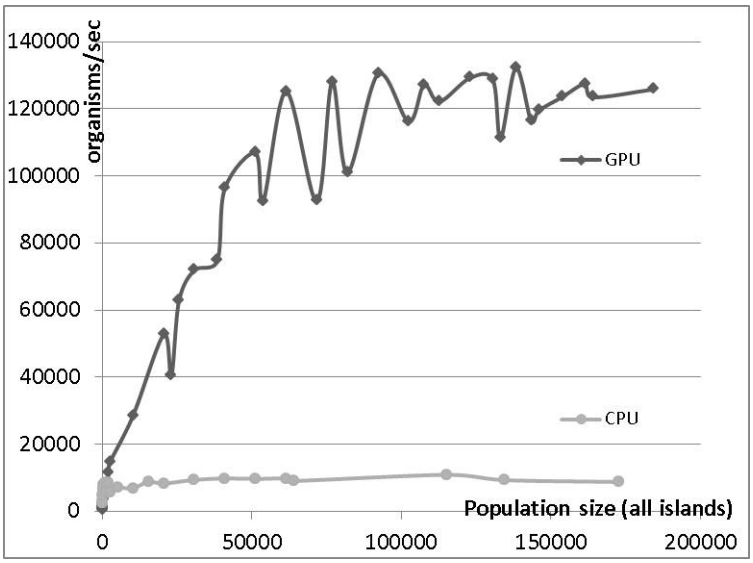

Figure 3. Comparison of performance using GPU and CPU

the super-word "DWDN - DW - HBHN - DDDNDW - KNN - DW - VNN" which delivers the value 0.489 to the fitness-function. On the same machines and with the same initial data, it required just 30 minutes of computational time. The latter motif exceeds the former within the current assumptions. However, the quality of the motif should be assessed in additional biological experiments [1], where HuR-RNA affinity will be measured after the per nucleotide mutations of the motif. Previously, several methods and different motifs of the HuR-RNA interaction have been published. In [20], a simple motif AUUU[U]*A was found. It did not consider any secondary structure of the HuR targets and its fitness-function (1) value measured in our data was 0.443. Another method [3] was able to recognize a motif consisting of one hairpin up to 17 bases in length with specific nucleotide content. This method takes into account the secondary structure, but in spite of this fact, it gives lower fitness function (1) value, that is 0.423. The result of the methods comparison is presented in summary in Table 2. The SWORD GPU algorithm provides the best result.

Further improvement of SWORD suggests the implementation of other fitness-functions for optional selection by the user. One of such functions is based on probability approach:

$$F(\boldsymbol{\alpha}) = -\frac{1}{n|pos|} \sum_{j=1|\boldsymbol{\alpha} match \boldsymbol{\beta_j}}^{|pos|} \sum_{i=1}^{n} \log p(\boldsymbol{\alpha^i} match \boldsymbol{\beta_j^i}), \tag{2}$$

Table 2. Fitness-function (1) value for motifs found with different methods

SWORD GPU (current work)	0.489
SWORD CPU [10]	0.45
AUUU[U]*A motif [20]	0.443
One-hairpin motif [3]	0.423

where the probability $p(\boldsymbol{\alpha^i} match \boldsymbol{\beta_j^i})$ is computed with the dynamic programming algorithm [21] extended for the 15-letter alphabet. This algorithm considers the first order Markov chain over the nucleotide sequences of *pos* or *con* sets. The fitness-function (2) can be defined without a *con* set, which extends the applicability of the method.

However, there is an obvious limitation of SWORD usage. The method requires a predicted secondary structure of RNA. Thus, it requires cooperation with such programs as RNAFold, Sfold, RNAduplex [12] or others. SWORD can be improved further through increasing its autonomy. The modified SWORD will automatically split an RNA sequence being analyzed into fragments according to pre-defined templates. Each fragment is assumed to be part of the functional domain of RNA and will be subjected to the search for a characteristic sub-word.

In its current modification, SWORD is a fast and high-quality software tool with a command line user interface and customizable parameters. It can be applied routinely for the whole genome search and classification of the pre-miRNA molecules, for the prediction of transcription terminators and search for motifs in any RNA molecules interacting with proteins. Exponentially growing size of experimental data makes SWORD a demandable tool for knowledge discovery.

References

[1] Thornton J.E., Gregory R.I. How does Lin28 let-7 control development and disease? // Trends in Cell Biology. – 2012. – Vol. 22(9). – P. 474–482.

[2] St. Laurent G., Tackett M., Nechkin S., Shtokalo D. Genome-wide analysis of A-to-I RNA editing by single-molecule sequencing in Drosophila // Nature Structural and Molecular Biology. – 2013. – Vol. 20(11). – P. 1333–1339.

[3] Lopez de Silanes I., Zhan M., Lal A., Yang X., Gorospe M. Identification of a target RNA motif for RNA-binding protein HuR // PNAS. – 2004. – Vol. 101(9). – P. 2987–2992.

[4] Hochsmann M., Voss B., Giegerich R. Pure multiple RNA secondary structure alignments // A Progressive Profile Approach in IEEE/ACM Transactions on Computational Biology and Bioinformatics. – 2004. – Vol. 1(1). – P. 53–62.

[5] Gautheret D., Major F., Cedergren R. Pattern searching/alignment with RNA primary and secondary structures: an effective descriptor for tRNA // Comp. Appl. Biosci. – 1990. – Vol. 6 – P. 325–331.

[6] Bailey T.L., Boden M., Buske F.A., Frith M., Grant C.E., Clementi L., Ren J., Li W.W., Noble W.S. MEME SUITE: tools for motif discovery and searching // Nucleic Acids Res. – 2009. – Vol. 37. – P. W202–W208.

[7] Breiman L. Random forests // Machine Learning. – 2001. – Vol. 45(1). – P. 5–32.

[8] Cortes C., Vapnik V. Support-vector networks // Machine Learning. – 1995. – Vol. 20(3). – P. 273–297.

[9] Hoskins J.C., Himmelblau D.M. Process control via artificial neural networks and reinforcement learning // Computers & Chemical Engineering. – 1992. – Vol. 16(4). – P. 241–251.

[10] St.Laurent G. Insights from the HuR-interacting transcriptome: ncRNAs, ubiquitin pathways, and patterns of secondary structure dependent RNA interactions // Mol. Genet. Genomics. – 2012. – Vol. 287(11-12). – P. 867–879.

[11] Topisirovic I., Siddiqui N., Orolicki S., Skrabanek L.A., Tremblay M., Hoang T., Borden K.L. Stability of eukaryotic translation initiation factor 4E mRNA is regulated by HuR, and this activity is dysregulated in cancer // Mol Cell Biol. – 2009. – Vol. 29(5). – P. 1152–1162.

[12] Gruber A.R., Lorenz R., Bernhart S.H., Neubock R., Hofacker I.L. The Vienna RNA websuite // Nucleic Acids Research. – 2008. – Vol. 36, Iss. 2. – P. W70–W74.

[13] Hofacker I.L., Fontana W., Stadler P.F., Bonhoeffer S., Tacker M., Schuster P. Fast folding and comparison of RNA secondary structures // Monatshefte fur Chemie. – 1994. – Vol. 125, Iss. 2. – P. 167–188.

[14] Vishnevsky O. Using GPUs to single out functional signals within the regulatory regions of genes // CUDA: week in review. – 2012. – Iss. 80. – Access Regime: http://www.nvidia.com/object/cuda-newsletter.html.

[15] Kennedy J., Eberhart R. Particle Swarm Optimization // Proc. of IEEE Internat. Conf. on Neural Networks. – 1995. – Vol. 4. – P. 1942–1948.

[16] Karaboga D. An Idea Based on Honey Bee Swarm for Numerical Optimization. – 2005. – (Prep. / Tech. Rep.-tr06; Erciyes University, Engineering Faculty, Computer Engineering Department).

[17] Michalewicz Z. Genetic Algorithms + Data Structures = Evolution Programs . – Springer-Verlag, 1996.

[18] Ackley D., Littman M. Interactions between learning and evolution // Artificial life II. – 1992. – Vol. 10. – P. 487–509.

[19] Whitley D., Rana S., Heckendorn R.B. The island model genetic algorithm: on separability, population size and convergence // J. of Computing and Information Technology. – 1999. – Vol. 7 – P. 33–48.

[20] Ma W.J., Cheng S., Campbell C., Wright A., Furneaux H. Cloning and characterization of HuR, a ubiquitously expressed Elav-like protein // J. Biol Chem. – 1996. – Vol. 271(14). - P. 8144–8151.

[21] Zhang J., Jiang B., Li M., Tromp J., Zhang X., Zhang M.Q. Computing exact P-values for DNA motifs // Bioinformatics. – 2007. – Vol. 23(5). – P. 531–537.