Noname manuscript No.
(will be inserted by the editor)

Particle Swarm Optimization for Finding RNA Secondary Structures

Michael Geis · Martin Middendorf

Received: date / Accepted: date

Abstract This paper proposes a Particle Swarm Optimization (PSO) algorithm called HelixPSO for finding RNA secondary structures that have a low energy and are similar to the native structure. HelixPSO is compared to the recent algorithms RnaPredict, SARNA-Predict, SetPSO, and RNAfold. For a set of standard RNA test sequences it is shown that HelixPSO obtains a better average sensitivity than SARNA-Predict and SetPSO and is as good as RNA-Predict and RNAfold. When best values for different measures (e.g., number of correctly predicted base pairs, false positives, and sensitivity) over several runs are compared, HelixPSO performs better than RNAfold, similar to RNA-Predict, and is outperformed by SARNA-Predict. It is shown that HelixPSO complements Rna-Predict and SARNA-Predict well since the algorithms show often very different behavior on the same sequence. Furthermore, a parallel version of the HelixPSO is proposed and it is shown that good speedup values can be obtained for small to medium size PC clusters.

Keywords Particle Swarm Optimization · RNA secondary structure · Genetic algorithms · Simulated annealing · minimum free energy · parallel PSO

The authors would like to acknowledge the support of the EMBIO project, which is funded within the European "New and emerging science and technology NEST - PATHFINDER" program.

M. Geis
Bioinformatics Group, Department of Computer Science and Interdisciplinary Center for Bioinformatics, University of Leipzig, Haertelstr. 16-18, D-04107 Leipzig, Germany
Tel.: +49-341-9716672
Fax: +49-341-9716679
E-mail: michael@bioinf.uni-leipzig.de

M. Middendorf
Parallel Computing and Complex Systems Group, Department of Computer Science, University of Leipzig, Johannisgasse 26, D-04103 Leipzig, Germany
E-mail: middendorf@informatik.uni-leipzig.de

1 Introduction

Ribonucleic acid (RNA) plays — beyond their function in protein coding — an important role for many cell functions, such as controlling gene expression, catalyzing chemical reactions or complementing protein enzyme based activity [8]. The secondary structure of RNA can be used to infer and explain its function (e.g., [32,22]). Determining the secondary structure of an RNA often constitutes an essential step towards predicting the tertiary structure.

Different algorithmic approaches exist for RNA secondary prediction. Sequence information alone is sufficient for comparative sequence analysis ([45]) as well as thermodynamic optimization ([48]) and combinations of both [7]. Thermodynamic folding algorithms rely on an energy model that assumes additive contributions from stacked base pairs and various types of loops ([21,38]). The corresponding energy values can be obtained, e.g., with measuring absorbance melting curves or with micro calorimetry ([18]).

The prediction of RNA secondary structures is particularly difficult when pseudoknots are involved. One reason for this is that little is known about energy models involving pseudo-knots [16]. Another reason is that thermodynamic structure prediction involving pseudoknots is an NP-hard problem for the standard energy model [10].

While the benchmark for RNA folding algorithms mfold [47], uses Dynamic Programming (DP), a number of attempts have been made to apply metaheuristics to the domain. Most of the proposed algorithms are genetic algorithms (GAs) and it has been argued that GAs can simulate the actual folding process of an RNA sequence and therefore achieve higher prediction rates of base pairs than DP [13]. Among those authors proposing GAs for predicting RNA secondary structure are [29], [3], [12], and [4]. Initial approaches were rather crude, but continuous refinement achieved greater prediction accuracy than DP [30]. Massively parallel implementations [31] as well as efficient algorithms for small architectures exist [34]. RnaPredict and its parallelized version P-RnaPredict are GAs for predicting RNA secondary structure that have evolved to levels of prediction quality comparable to mfold ([44,42]). Recently, the Simulated Annealing algorithm SARNA-Predict has been introduced ([35,36]). The first attempt to predict RNA secondary structures with PSO (a method called SetPSO) was proposed by Neethling and Engelbrecht [24].

In this paper we propose the PSO algorithm HelixPSO for finding RNA secondary structures that have a low free energy and a high number of correctly predicted base pairs when compared to known native structures. A preliminary version of HelixPSO that was used only for finding secondary structures with low free energy was described in [20]. The optimization behavior of HelixPSO is compared to the algorithms RnaPredict ([44,42]), SARNA-Predict ([35,36]), and SetPSO ([24]) on standard sets of RNA test molecules. A parallel version of HelixPSO is also proposed in this paper and it is shown that is efficient for small and medium size PC Clusters.

A short overview on RNA secondary structure is given in Section 2. A description of related approaches for RNA secondary structure prediction is given in Section SecRelated. HelixPSO is introduced in Section 4. The parallel version of HelixPSO is described in Section 5. The experiments are described in Section 6 and results are presented in Section 7. Conclusions and future work are given in Section 8.

2 RNA Secondary Structure

An RNA molecule is basically a sequence of four different types of nucleotides. Each nucleotide has one of the bases adenine (A), guanine (G), cytosine (C), or uracil (U). The sequence or chain of nucleotides is called the primary structure of the RNA molecule. The secondary structure is the result of hydrogen bonds between nucleotides that are not neighbors in the chain. Typically these hydrogen bonds occur only between G and C, or A and U, or G and U (or vice versa). The so connected nucleotide bases are called base pairs. In this paper we only consider canonical base pairs, which are GC, CG, AU, UA, or GU, or UG (the first base is the one with the smaller index in the chain). A main element of secondary RNA structures are helices, which are sets of two or more adjacent base pairs that form a ladder like structure. Thus, a helix of length $k \geq 2$ consists of k base pairs with indices (i, j),$(i + 1, j - 1) \ldots, (i + k, j - k)$ where $1 \leq i < i + k < j - k < j \leq n$ and n is the length of the RNA sequence. An RNA secondary structure is defined by a set of base pairs, where each base can pair to no more than one base. Furthermore, pairing bases must be at least three bases apart and two base pairs must not cross, i.e.: i) $j - i > 3$ for each base pair (i, j) and ii) for any two different base pairs (i, j), (i', j') either $i < i' < j' < j$ or $i' < i < j < j'$ holds. When ii) does not hold, the structure is referred to as a pseudo knot. Pseudo knots do occur in nature, but energy models exist only for few types of pseudo knots and corresponding computations are significantly more complex. As has been done by many authors, we consider here only secondary structures without pseudo knots.

Several algorithms for secondary structure prediction start by computing the set H of all potential helices of an RNA molecule. This is done by iterating over all pairs of bases, checking if they can pair, and if so, whether they can be extended to a helix. If this is the case, that helix is added to H. These algorithms then try to find a subset of H that defines an optimal (in some sense) secondary structure. This is also done by algorithms RnaPredict and HelixPSO. These secondary structures are then evaluated to yield a free energy. The energies attributed to RNA secondary structures are free energies because they comprise both enthalpic and entropic contributions (arising from summing over different spatial conformations of the unpaired loop regions) and are referred to as ΔG (in Kcal/mol). The native structure usually has a free energy of about 5-10% from the minimum free energy of the sequence. In general, the free energy of an RNA secondary structure decreases with helix size. But the energy also depends on the type of base pairs. There exist several functions to compute the free energy of an RNA secondary structure. In this paper we use the RNAeval algorithm from the well known ViennaRNA package ([14,15]).

3 Related Approaches

In this section we describe three algorithms for secondary structure prediction that are used for comparison in this paper.

3.1 RnaPredict

The genetic algorithms RnaPredict and P-RnaPredict by Wiese and coworkers ([42]) find low energy RNA conformations by applying selection, mutation and crossover op-

Algorithm 1 RnaPredict

```
Initialize population with random permutations
for i = 1 to number of iterations do
  for j = 1 to (population size/2) do
    Select chromosomes c, c' from population
    if random() < p_c then
      offspring o, o'=CX(c, c')
      if random() < p_m then
        Mutate(o)
        Mutate(o')
      Add best child and best parent to new_population
    else
      Add c and c' to new_population
  population=new_population
  global_best=FindGlobalBest(population)
return global_best
```

erators to a population of chromosomes. Each chromosome corresponds to an ordering of the elements in the set H and is a permutation of the indices of the elements in H (with respect to some (arbitrary but fixed) enumeration of the helices in H). However, not all of these helices can be incorporated into the same secondary structure. The base pairing rules in 2 then determine which of the helices in H appear in the secondary structure corresponding to the permutation. A subset of H that defines a secondary structure is derived from a chromosome as follows. Starting with the open chain that has no base pairings as secondary structure, the helix that corresponds to the first element in the permutation is added to the open chain, thus forming a new secondary structure. Then the helix that corresponds to the second element in the permutation is added to the secondary structure if the combined structure still satisfies the base pairing rules in Section 2. The entire permutation is traversed in this manner and the secondary structures updated every time a helix can be added subject to the base pairing rules. The secondary structure obtained at the end is then associated with the permutation that produced it. As far less helices than the size of H fit into a secondary structure, more than one permutation can produce the same secondary structure. RnaPredict can be used in conjunction with a multitude of energy models such as the Nussinov-Jacobson energy model, the individual nearest neighbor (INN) model [28], and the individual nearest neighbor-hydrogen bond (INN-HB) model [46].

Mutation is done in RnaPredict by swapping two random indices in the permutation. Wiese et al. ([43, 41]) have investigated the optimization behavior of RnaPredict with a variety of crossover operators (such as edge recombination crossover (ERC) [39], OX2 [33], cycle crossover (CX) [25], and Partially Mapped Crossover (PMC)[11]) and different selection operators (Keep-Best-Reproduction (KBR), Standard Roulette Wheel Selection (STDS) [40]). Some versions of RnaPredict use 1-elitism. A pseudo code of RnaPredict is given in Algorithm 1, where p_c is the crossover probability and p_m is the mutation probability.

3.2 SetPSO

PSO is an iterative optimization heuristic for function optimization where a swarm of m particles searches in a multidimensional space ([19]). Typically in PSO, each particle i has a position x_i and a velocity v_i. The velocity is updated in each iteration t according

to $v_i(t) = w \cdot v_i(t-1) + c_1 \cdot r_1 \cdot (y_i(t-1) - x_i(t-1)) + c_2 \cdot r_2 \cdot (\hat{y}_i(t-1) - x_i(t-1))$ where: i) the inertia weight $0 < w < 1$ controls the influence of the previous velocity, ii) parameter $c_1 > 0$ controls the impact of the personal best position found so far (denoted y_i and called *pbest* solution), iv) parameter $c_2 > 0$ determines the impact of the best position that has been found so far by any of the particles in neighborhood of particle i (denoted $\hat{y}_i$ and called *lbest* solution), v) random values r_1 and r_2 are drawn with uniform probability from $[0, 1]$. After velocity update every particle i moves with the new velocity to the new position according to $x_i(t) = x_i(t-1)+v_i(t)$. Then for each particle i the objective function f is evaluated at its new position. If $f(x_i(t+1)) < f(y_i)$ (assuming the function has to be minimized) the personal best position y_i is updated, i.e. y_i is set to $x_i(t)$.

The first PSO algorithm for finding RNA secondary structures with minimum free energy has been proposed by Neethling and Engelbrecht and is called SetPSO ([24]). As other PSO algorithms for discrete optimization problems, SetPSO differs from the standard PSO scheme. Similar to RnaPredict, SetPSO searches on the set of helices of a given RNA sequence and represents a secondary structure as a set of helices (which must be feasible according to the rules given in Section 2). Hence, the position of a particle is characterized by a set of helices.

Movement of a particle is defined in terms of addition and removal of helices from the particle's current set of helices. A set O of helices which is removed from the particle's position is computed from the empty set by adding a helix that is neither in the *pbest* solution nor in the *lbest* solution with probability p_I. A candidate set C of helices which might be added to the particle's position is computed from the empty set by adding each helix of a target solution with probability p_C and each helix from the set of all helices with probability p_R. The target solution is a combination of the particle's *pbest* and *lbest* solution. To avoid base pair conflicts, all helices in the set O are removed before those of the set C are added (if feasible). The computation of the sets O and C is called the velocity update and the actual computation of the new solution is the position update. For more details of SetPSO see [24].

3.3 SARNA-Predict

SARNA-Predict is a Simulated Annealing algorithm. It employs the paradigm of cooling a substance, in which relatively free movement of particles is possible at high temperatures, while particle movement is gradually restricted until they are frozen in place once the substance has been cooled. SARNA-Predict is characterized by its state representation, perturbation/mutation function, evaluation function and decision mechanism as well as the annealing schedule. The annealing schedule significantly influences the evaluation function and the decision mechanism. SARNA-Predict uses the same permutation based state representation as RnaPredict. The mutation typically executes more than one swap of elements of the permutation. The number of swaps is proportional to the current temperature. The evaluation function computes the difference in free energy between the current structure and the structure of the previous iteration. For more sophisticated energy models, this value is also temperature based. The decision mechanism, which determines whether a new structure is accepted uses probabilities from the Boltzmann distribution. That means secondary structures of lower free energy are always accepted, while structures with higher energy than the previous structure are accepted with a probability that declines exponentially with the

energy difference but is scaled by the temperature. This has the effect that at high temperatures early in the algorithm, the likelihood to accept worse solutions is relatively high. AS the temperature drops to zero, so does the likelihood to accept worse solutions in latter stages of the algorithm. This implements the trade off between exploration and exploitation controlled by temperature that is characteristic for Simulated Annealing. SARNA-Predict has been implemented with the INN-HB energy model and the efn2 energy model used by mfold.

4 HelixPSO

Since the proposition of the initial Nussinov algorithm, many approaches have been employed to address the RNA secondary structure prediction problem. In the field of evolutionary and nature inspired algorithms GAs make up the vast majority of such algorithms. In particular, the consistent improvement of versions of RnaPredict over the years raised its level of prediction quality, so that it now compares quite favorably with mfold. Therefore it is interesting to apply other nature-inspired approaches in this problem domain. Even though PSO is typically applied for continuous function optimization there exists some mostly very recent applications of PSO for discrete problems. To the best of our knowledge only two papers ([24, 20]) exists where PSO has been applied to RNA secondary prediction. Both of these papers concentrate on minimum free energy secondary structures.

In this section, we define an extended version HelixPSO that is suitable to find secondary structures which have a low free energy and a high number of correctly predicted base pairs with respect to the native structure. The source code of the HelixPSO algorithm is available online [1]. Similar to RnaPredict, HelixPSO encodes a secondary structure as a permutation of the index set of all helices. The permutation determines which elements of the set H of all helices of the input RNA sequence are included in the secondary structure as explained in Section 3.1. The free energy of a secondary structure is computed with the RNAeval algorithm of the ViennaRNA package ([14, 15]). Its energy and stacking parameters can be found in [21] and [38].

The fitness function of HelixPSO mostly takes into account free energy, but can be adjusted to also consider agreement with the centroid structure, C. Each component is scaled to [0,1]. The centroid is the structure with the smallest base pair distance to the structures in the thermodynamic ensemble of a sequence, of which the mfE is the member with the highest probability. It contains the base pairs which have a probability of more than 50 % to be present in any structures the given sequence can fold into. We used the RNAfold algorithm from the ViennaRNA package to compute the centroid. The centroid contribution to the fitness is the ratio of matching base pairs in a structure S and in the centroid divided by the number of base pairs in the centroid. In other words, if the centroid were the native structure and S tried to predict C, the fitness contribution due to matching with the centroid would be the sensitivity of S. The energy component of the score is the ratio of the free energy of S to the minimum free energy of the considered sequence, but at least 0 (in the case that the free energy of S is positive). As shown in Equation 1, the obtained partial scores are multiplied by λ and $1 - \lambda$, where λ is in [0,1], and added, again yielding a fitness score in [0,1].

[1] http://www.bioinf.uni-leipzig.de/Software/HelixPSO/

$$Fitness(S) = \lambda * \frac{|S \cap C|}{|C|} + (1 - \lambda) * Min(0, \frac{E(S)}{mfE}), 0 <= \lambda <= 1 \quad (1)$$

In HelixPSO, particles move with respect to target positions. For this each particle i has an associated set of candidate target positions T_i and for each $t \in T_i$ a weight $0 < w(t) < 1$. The wight corresponds to the inertia weight in standard PSO algorithms. The relative weight of a position in T_i determines the probability that it is chosen as a target. T_i is initialized with a single random position, i.e., a permutation that is generated randomly and that has weight 1.0. After each iteration of HelixPSO the weight of each target solution is decreased by multiplication with a parameter ρ, $0 < \rho < 1$. A position that has a weight less than the value of a threshold parameter τ is removed from T_i. The reason is that elements with a very small weight are unlikely to be chosen as a target but require much memory space. Then the personal position and either the global best or the cluster best position (details are given later) are added to T_i with probability $c_1 \cdot r_1$ and $c_2 \cdot r_2$, respectively where r_1 and r_2 are random numbers that are chosen uniformly from $[0, 1]$. The constants $c_1 > 0$ and $c_2 > 0$ refer to the cognitive and social component respectively. Note, that this is similar to the impact of the personal best and global best values for the standard PSO scheme The initial weight of each position that is added to T_i is 1.0.

HelixPSO is a multi-swarm algorithm where the swarm of particles is partitioned into several subswarms. Subswarms are used to encourage the swarm to search in different areas of the search space. These subswarms are called clusters. All particles in a cluster (with one exception) use their personal best position and the cluster best position to update their set of candidate target positions. Only the cluster best particle (i.e., the particle which has the best personal position within the cluster) does the update with respect to its personal best and the global best position. The idea behind this is to ensure that particles in a cluster stay close to each other, while the clusters in their entirety should converge towards the global best solution.

When a particle i has chosen its target position from the set T_i the particle moves towards the target as follows. The positions of some helices in the particle's own permutation vector are swapped to make it more similar to the permutation vector of the target. As not every such swap causes a change in the corresponding secondary structure each particle performs a series of α swaps, where α is a parameter of the algorithm. A swap is done with probability $\beta > 0$ in direction of the target position as described in the following. To find the first helix of the swap the permutation vector of the particle is scanned from the beginning. Helices which are at the same place in the particles and the targets permutation are skipped. Otherwise an index is chosen with probability $p_S > 0$. If an index j has been chosen the helix h of the target permutation at place j is determined. Then the helix h' at place j in the particle's permutation is swapped with helix h. Thus, after the swap, the particle and the target have the same helix h' at place j. To increase diversity, a random swap is performed with probability $1 - \beta > 0$. That is, for two randomly chosen helices their positions in the permutation of the particle are exchanged.

When searching for minimum free energy structures the series of swaps is done greedily in HelixPSO, that is a swap is accepted only when the new secondary structure that is generated by the series of swaps improves over the secondary structure before the swaps were done. The chance of random swapping is set to 10%. HelixPSO uses 1-elitism, i.e., after each iteration the particle at the worst position is reset to the position of the global best particle. This, or a better value, also becomes the particle's

Algorithm 2 HelixPSO

```
Initialize the swarm by creating the particles, partition them into clusters and choose randomly permutation vector for each particle.
Personal best position for each particle is set to the current position.
Calculate cluster best position for each cluster.
Calculate global best position.
j = 1
while (j < maximum number of solutions) do
  for all particles p do
    Let c denote the secondary structure of particle p.
    for i = 1 to α do
      Chose a target from T_p.
      Chose an index idx in the permutation of particle p.
      if rand() < 1 − β or if no index was chosen then
        Perform a random swap of indices in the permutation vector and let c' be the corresponding secondary structure.
      else
        In the permutation of the particle swap the helix at idx with the helix that is at idx in the target permutation and let c' be the corresponding secondary structure.
      j++
      Compute fitness of c'.
      if fitness of c' < fitness of c then
        c = c'
  Update pbest, cbest, and gbest.
  Apply 1-elitism, i.e., reposition the worst particle to the global best.
  For each particle p update T_p.
return gbest
```

personal best at the end of the next iteration, since the algorithm is greedy, i.e., a new position is only accepted if it is better than the previous one. The movement vector of this particle is not changed. Due to the fact that a particle in HelixPSO only moves to a new position if that position has a lower free energy than the particle's previous position, significant aspects of the search space may remain unexplored for sequences whose native structures are not to be found at a level that is close to the minimum free energy. In particular, the algorithm tends to report a very limited set of structures on repeated runs for some sequences.

When searching for the native sequence an alternative to the greedy acceptance rule that is an annealing controlled acceptance function. The function that is used is based on the successful exploration technique of SARNA-Predict. The rate of acceptance for a structure depends on the energy difference between the previous and new position of a particle under the Boltzmann distribution. If the energy of the new position is an improvement, it is accepted. If the energy is worse, it is accepted with a probability that depends on the current temperature and the energy difference between the new structure and the old structure:

$$P(Accept) = e^{\frac{-(E_{new}-E_{old})}{T}} \tag{2}$$

The free energy is measured in Kcal/mol and temperature is in Kelvin. The temperature starts high (500 K) at the beginning of the algorithm, but is gradually lowered through multiplication with a constant factor given by the cooling rate whenever the number of generated solutions hits an integer multiple of the sub chain size. Following SARNA-Predict, where the value yielded good results for the majority of sequences, we have chosen the sub chain size as 1800. The cooling rate is usually chosen to lie

in $[0.8, 0.99]$ for Simulated Annealing algorithms. For the start temperature, a lower value means that less solution generations are needed until the cooling is completed, which saves significant computation time. For this reason, we have chosen a starting temperature of 500 K, which is ten times lower than the value for SARNA-Predict. Tests have shown that in addition to this classical form of annealing, a version with the unusual parameter value of 0.1 for the cooling rate leads to good results for some long sequences. This causes a rapid cooling, such that exploration of higher energy structures in the search space is only permitted at the early stages of the algorithm but very quickly evolves towards the greedy acceptance heuristic of the default version of HelixPSO.

Another difference between the version of HelixPSO that searches for minimum free energy structures and the version that searches for the native structure is that at the end of a run the most frequent structure in the population is reported as the result (instead of reporting the structure with the lowest free energy). This is motivated by the fact that the native structure of a sequence must be at a comparatively accessible location of the folding landscape of the secondary structure space. Thus, the frequency of a structure among particles gives a good indication not only of its quality (since many particles were attracted to it) but also its accessibility. If a low lying structure is not accessible, few particles would be able to get there without traveling over paths leading over intermediate conformations whose highest point is too high to be accepted with the annealing heuristic. A low energy, difficult to access structure is indeed not very likely to be the native structure, since in vivo, the RNA molecule would fold into such a structure only with a low probability, precisely because of its unfavorable location in the energy landscape. For short sequences of less than 200 nucleotides in length, at least one structure occurs more than once among the particles. For longer structures, however, it is possible that this is not the case. In this case, the positions of the particles are clustered with respect to their base pair distance, and a random structure from the biggest cluster of particles is returned. Thus the notion of the most frequent particle is approximated for sequences with large folding spaces by the notion of a neighborhood of particles that lie close to each other. The clustering algorithm is implemented in the AnalyzeDists program of the ViennaRNA package. The computational effort to perform the clustering makes this approach unwieldy for large numbers of particles. For this reason, the swarm size has been reduced from 500 to 200 for sequences of size greater than 200 nucleotides, where the clustering is employed. Furthermore, an undistorted assessment of the accessibility of structures in the space can only be achieved if the elitism implemented in the original version of HelixPSO is removed. Otherwise, the lowest lying structure will receive an additional particle every iteration for which it is global best, which makes it very likely that it will accumulate more particles than any other structure by the end of the algorithm irrespective of the energy difference that has to be crossed for particles to get there.

5 Parallel Version of HelixPSO

Metaheuristics in general and PSO algorithms in particular are typically well suited for parallelization (for a general overview see [2], for parallel variants of PSO see [9, 23]). Several parallel PSO algorithms have been proposed for different applications (examples can be found in [17, 26, 27, 37]).

Length	Organism(Accession Number)	RNA class
117	*Geobacillus stearothermophilus* (AJ251080)	5S rRNA
118	*Saccharomyces cerevisiae* (X67579)	5S rRNA
120	*Escherichia coli* (V00336)	5S rRNA
122	*Haloarcula marismortui* (AF034620)	5S rRNA
123	*Thermus Aquaticus* (X01590)	5S rRNA
124	*Deinococcus radiodurans* (AE002087)	5S rRNA
394	*Metarhizium anisopliae var. anisopliae(3)* (AF197120)	Group I intron, 23S rRNA
454	*Chlorella saccharophila* (AB058310)	Group I intron, 16S rRNA
456	*Metarhizium anisopliae var. anisopliae(2)* (AF197122)	Group I intron, 23S rRNA
468	*Aureoumbra lagunensis* (U40258)	Group I intron, 16S rRNA
543	*Hildenbrandia rubra* (L19345)	Group I intron, 16S rRNA
556	*Acanthamoeba griffini* (U02540)	Group I intron, 16S rRNA
605	*Porphyra leucosticta* (AF342746)	Group I intron, 16S rRNA
697	*Caenorhabditis elegans* (X54252)	16S rRNA
784	*Drosophila virilis* (X05914)	16S rRNA
940	*Acomys cahirinus* (X84387)	16S rRNA
945	*Xenopus laevis* (M27605)	16S rRNA
954	*Homo sapiens* (J01415)	16S rRNA
964	*Ailurus fulgens* (Y08511)	16S rRNA
1492	*Sulfolobus acidocaldarius* (D14876)	16S rRNA

Table 1 Test RNA sequences of different lengths from different species were tested and are listed with their comparative RNA website accession number (Accession)

As the only information that needs to be shared by all particles in the swarm are the cluster best and global best positions, HelixPSO is a promising candidate for parallelization. This is particularly so, as dividing a swarm into subsets of particles and assigning these to different processors reduces the amount of memory required for each processor. Thus, even though memory requirements increase with sequence size, longer sequences remain computable in reasonable time with a parallel version of HelixPSO. For the parallelization that is proposed in this paper, the LAM/MPI implementation ([5]) of the Message Passing Interface was used. In that implementation the original HelixPSO algorithm for energy minimization that uses elitism was used. Inter processor communication was performed after every iteration of the algorithm to globally update the most recent cluster and global best values. As the available PC cluster consisted of a homogeneous set of up to 10 processors, the particle swarm was subdivided into subsets of particles of equal size and each was assigned to one processor. These subsets of particles are not equivalent to the clusters, as the number of subsets equals the number of processors used and the number of clusters is a parameter set by the user which (the default value used in this paper is seven).

After each iteration, the processors exchange information about their best particles to determine a global best particle for the entire swarm. Similarly, all processors that are responsible for the particles of the same cluster exchange their respective best values of particles in that cluster to determine which particle performed best in the whole cluster. At first glance this may appear ineffective, as more data are exchanged with this setup than when all particles in a cluster share the same processor. However, the main communication overhead is caused by processors that have to wait until each processor has finished its iteration., This is necessary because the new global best position has to be known on each processor before the next iteration can start.

Exchanging information in order to determine the cluster best values does then not add much overhead.

Elitism is performed by the processor that has the worst performing particle by placing that particle at the position of the global best particle of the previous iteration. Since the exchange of a permutation takes relatively long, it is only performed when an improvement occurred since the last exchange. This is achieved as follows for the global best (for the cluster best of each cluster this is done analogously): The free energy of the best particles is obtained from each processor. Then, the global best particle and the processor that computed it are determined. If the best energy value is not better than the global best energy value of the previous iteration, the global best position remains unchanged for the next iteration. Otherwise the processor on which the new global best position was obtained broadcasts the permutation corresponding to that position to all other processors. These processors then update their global best position. Since the global best solution improves (usually) only rarely during the later part of a run of HelixPSO, often only the exchange of the global best energy values for each processor needs to be performed. The cluster best positions are updated similarly, with the exception that processors which do not have particles of the cluster under consideration send a dummy value.

Elitism is executed as follows. Each processor sends the energy of its worst particle, from which the processor with the globally worst particle is determined. The processor with the particle having the worst energy moves this particle to the global best positions of the previous iteration. This effectively increases the influence of the global best position on the swarm by placing additional particles to it.

6 Experiments

For the experiments we used RNA sequences of different lengths that have already been used by other authors so that we can compare HelixPSO with RnaPredict, SARNA-Predict and SetPSO. The 20 test sequences are available from the comparative RNA website [6]. Table 1 lists the RNA sequences with their size, organism, RNA class and accession number. The comparative RNA website also lists inferred native structures for these sequences, which have been used by RnaPredict and SARNA-Predict to compare their predictions. We will use the inferred native structures in our evaluation of the experiments as well. As the name 'inferred' suggests, it is not known with certainty that these structures are indeed the native structures.

We compare our results to those of RnaPredict in [42], where the best values obtained for 30 random seeds and 12 operator combinations are listed. Results for SARNA-Predict are taken from [35,36], where 100 runs were executed for various parameter combinations on each test sequence. For SARNA-Predict best values as well as average values are given. In these publications, the results of both RnaPredict and SARNA-Predict are compared to mfold.

HelixPSO was run 30 times on each test sequence for these comparisons. The weight λ of the centroid structure was equidistantly varied over [0,1]. For the calculation of average values, the greedy heuristic was used and the frequency of random particle movement was at 10 % or 50 %. For the tests reporting best values, all the enhancements encouraging greater diversity were used. That means the chance of random movement was either 10 % (the default) or 50 % (to encourage diversity). Also, all three acceptance heuristics were used, i.e. greedy, as well as two forms of annealing. For regular annealing,

the cooling rate was set to 0.9 for short sequences of less than 200 nucleotides and to 0.97 for all other sequences. For all sequences longer than 200 nucleotides, also a rapid form of annealing was performed with cooling rate set to 0.1. These variations in parameter settings are analogous to the use of multiple operators for RnaPredict and different parameter sets for SARNA-Predict in the above publications. For the other parameters the following standard values were used in HelixPSO: $\alpha = 25$, $\beta = 0.9$, $\lambda = 0$, $\rho = 0.95$, $c_1 = c_2 = 3.0$, $p_S = 0.01$. The number of solutions was 50000 for sequences of length less than 200 nucleotides, 100000 for sequences longer than that but less than 1000 nucleotides and finally 200000 for the *S. acidocaldarius* sequence with 1492 nucleotides. This deviation from the original 280000 solutions is due to the fact that for the annealing based acceptance function to lead to greater diversity, the process must not go on for too long. Otherwise a stage is reached, where only structures with very low free energy are returned. As the structure space grows with sequence length, shorter sequences attain this stage earlier then longer sequences, hence less solutions are needed. The second parameter that is different is that the number of particles has been lowered from 500 to 200 for sequences longer than 200 nucleotides. This is due to the fact that the returned structure is the one with the greatest frequency among the particles. For long sequences, it may not be the case that two or more particles are located at the same structure. In this case, the found structures are clustered and a random structure from the greatest cluster is returned. Since the calculation of the clusters takes too long to perform for 500 particles, the swarm size has been lowered for longer sequences. HelixPSO was used on the set H of all maximal helices, i.e., helices that can not be extended by adding further base pairs. For the parallelized version of HelixPSO, we performed 10 runs each for 1, 2, 4, 8, and 10 processors on the RNA sequences of *A. lagunensis*, *D. virilis*, *X. laevis*, and *S. acidocaldarius* with the standard parameter set.

For the comparison with a Dynamic Programming algorithm, we the RNAfold algorithm from the ViennaRNA package was used instead of mfold because this library also used for energy evaluations and centroid information. Conveniently, RNAfold also is faster than mfold. Since both mfold and the RNAfold algorithm implement the same DP procedure to find the minimum free energy structure and both algorithms use almost identical energy parameters, the comparison to RNAfold should give a good indication for the comparison between HelixPSO and mfold as well. In these tests, the values for SetPSO as published in [24] were used. The parameter values were: $p_C = 0.6$, $p_R = 0.5$, p_I was decreased linearly over the iterations from 0.9 to 0.1, swarm size 50, and each run was done over 700 iterations for a total of 35000 solutions.

We investigated the influence of different parameters on the optimization behavior of the HelixPSO variant that uses centroid information. One parameter was varied at a time, the remaining values were as in the standard value set. The investigated parameters were population size, inertia weight ρ, probability of random movement and the cooling rate. Population values were only tested up to size 1000, as the clustering algorithm that computes the output takes a very long time for large swarms. The *A. griffini* sequence was used as an example, as it lies approximately in the middle of the ranges featured by the sequences in the test set and yielded quite good predictions. Each run was repeated 30 times. Because the standard error was quite high in comparison to the differences in mean values, the runs for probability of random movement had to be repeated 300 times. Table 2 shows the parameter ranges used in the experiments.

Variable	Value
particles	$50, 100, 200, 500, 1000$
ρ	$0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.98, 0.99$
cooling rate	$0.80, 0.85, 0.9, 0.93, 0.95, 0.97, 0.99$
random	$0.1, 0.2, 0.3, 0.4, 0.5$

Table 2 HelixPSO parameter variations

7 Results

In this section we present results on the influence of some parameters on the optimization behavior of HelixPSO and compare the behavior of HelixPSO with other algorithms on the test RNAs.

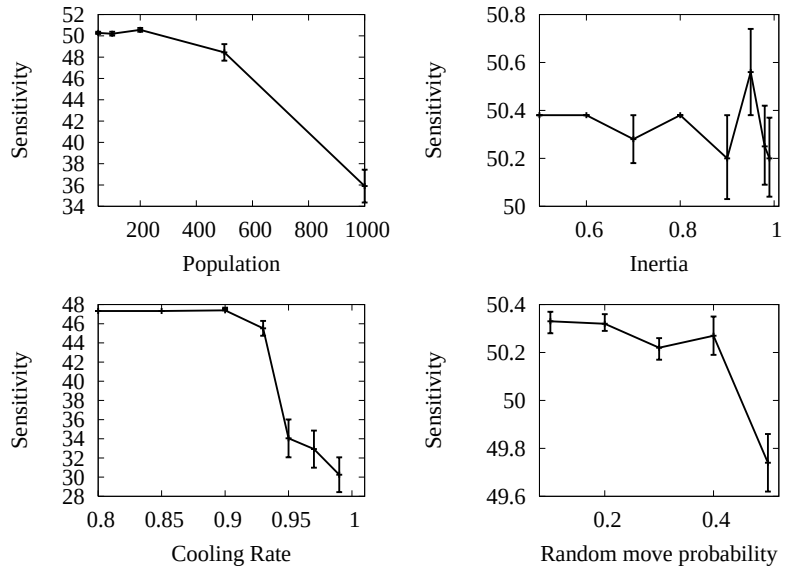

Fig. 1 Sensitivity of best solution found by HelixPSO for the *A. griffini* sequence after generation of 100000 solutions when different parameter values are varied and all other parameter values are as in the standard value set; bars indicate the standard error

7.1 Influence of Parameters

The results of the test runs with different parameter values are displayed in Figure 1. The results show that the population sizes in the range between 50 and 200 particles are good. While 200 particles achieve the best solution quality, the difference is only 0.2-0.3%. For population sizes of 500 and more the prediction quality decreases significantly and also the runtime increases. Different values for the inertia weight cause sensitivity variations of up to about 0.4%. The default value of 0.95 performed best with an average sensitivity of 50.6%. Due to the size of the standard error, the improvement of an inertia

weight of 0.95 over the remaining values is not very large. Figure 1 shows also that different values for the cooling rate yield similar results for a range of values until a cut off point at about 0.925. For values higher than the cut off point, the solution quality decreases rapidly. But cooling rates between 0.8 and 0.95 produce average sensitivities of about 47.5%, which is already significantly lower than sensitivity obtained by the default parameter set, which is around 50%. It should be noted that the annealing acceptance rule is advantageous for obtaining good best (i.e., measured over several runs) solutions by introducing greater diversity in the explored solutions. But caution should be used when an annealing schedule is used instead of the original greedy acceptance function when good average (i.e., over several runs) solutions are sought. The impact of the random movement probability parameter on the average performance is much less significant than that of the annealing schedule. The difference between best and worst sensitivities is about 0.5 only. A movement probability of 10% performs best and 50% is the worst probability of the tested values.

7.2 Comparison to other algorithms

For a comparison of HelixPSO to other algorithms, we are reporting 6 prediction measures that have been used before in the literature for the evaluation of SARNA-Predict and RnaPredict. Three measures are predicted base pairs (TP), false positives (FP), and false negatives (FN). Based on these statistics the other three measures can be calculated: the ratio between TP and count of base pairs in the natural fold (sensitivity, abbreviated SE), the ratio between TP and predicted base pairs (specificity, abbreviated SP) and finally the F-measure (abbreviated FM), which combines SE and SP. The value of the F-measure is 2*SE*SP/(SE+SP). The values for SE, SP, and FM are reported in percent in this section.

Organism	Length	HelixPSO	RnaPredict	SA	DP	SetPSO
S. cerevisiae	118	78.8	75.7/89.2	89.2	75.7	75.7
H. marismortui	122	76.3	-	71.1	76.3	42.1
H. rubra	543	31.6	22.9/31.7	27.2/33.8	37.7	-
A. griffini	556	48.8	-	31.3/35.0	47.3	-
D. virilis	784	18.7	11.5/17.7	11.8/12.9	15.5	12.8
X. laevis	945	30.4	-	24.0/27.0	24.3	23.0
H. sapiens	954	25.5	-	17.4/18.7	33.1	-
S. acidocaldarius	1492	38.6	-	24.9/29.7	54.5	-
Average	-	43.6	-	37.1/39.7	45.6	-
Av., $length > 500$	-	32.3	-	22.7/26.2	35.4	-

Table 3 Comparison of average sensitivity of HelixPSO, RnaPredict, SARNA-Predict(SA), RNAfold(DP) and SetPSO. The ranges indicate the best and worst reported values for a sequence, which are due to the use of multiple operators for RnaPredict and various parameter settings for SARNA-Predict. Averages are given for all sequences and also for the longer sequences (length > 500 nucleotides) only.

For the performance with respect to average values taken over several runs only the sensitivity values have been published for SARNA-Predict and RnaPredict. Table 3 shows the average sensitivity values for HelixPSO, RnaPredict, SARNA-Predict, and SetPSO. RNAfold is deterministic, so the prediction accuracy of the minimum free energy structure is given. The range of sensitivity values for SARNA-Predict and

RnaPredict are due to the variation of operator types for RnaPredict and varying parameter values for SARNA-Predict. They indicate the lowest and the highest value reported for any of the given parameter combinations.

Compared to RnaPredict, the HelixPSO algorithm has a sensitivity in the lower range of the values obtained by RnaPredict for the shortest sequence *S. cerevisiae* and has almost exactly the same sensitivity as the highest value reported for RnaPredict for the *H. rubra* sequence with length > 500. For the other sequence of length > 500 (*D. virilis*) HelixPSO has a higher sensitivity than the highest value given for RnaPredict. On average over the two sequences of length > 500 HelixPSO achieves a higher sensitivity of 25.2 than the average best values for the sensitivity of RnaPredict with 24.7.

HelixPSO scores higher than the best values reported for SARNA-Predict for all but the *S. cerevisiae* and *H. rubra* sequences. For the *A. griffini* sequence, HelixPSO outperforms the highest values given by SARNA-Predict by more than 10%. Averaging the SARNA-Predict values over the highest of values given by SARNA-Predict, HelixPSO performs better with an average sensitivity of 43.6% compared to 39.7% over all test sequences. If only sequences of length > 500 are considered, the difference in sensitivity values increases from 3.9% to 6.1%.

HelixPSO makes significantly better predictions than SetPSO for all four sequences for which values have been published for SetPSO. For the *H. marismortui* sequence, the difference in sensitivity is more than 30%.

HelixPSO and RNAfold yield very similar sensitivity on the sequences of *S. cerevisiae*, *H. marismortui*, *A. griffini*, and *D. virilis*. On average over all sequences RNAfold achieves a slightly higher sensitivity of 45.6% than HelixPSO with an average sensitivity of 43.6%.

For SARNA-Predict and RnaPredict most results that have been reported in the literature are not average results over several runs but best results over several runs. In the following we present such best values for HelixPSO and compare them to the results of SARNA-Predict, RnaPredict, and RNAfold. One problem with best values is that they depend on the number of runs. But a more serious problem is that in practice it is not possible to tell which of the structures that are returned by different runs is best (unless the natural fold is already known). Hence, it is not clear which of the predicted structures should be used in order to obtain the best one. This is clearly different from, say, predicting low energy structures, in which case it is obvious which structure in a set performs best (since energy models and evaluation functions exist). Another problem is that only algorithms can profit from best values that are designed to create very different solution over different runs. For a discussion of why reporting an empirical maximum of a distribution is problematic, see Biratarri and Dorigo [1].

Tables 4 and 5 list the results of HelixPSO and RnaPredict for 19 sequences, taken from [42]. On average HelixPSO has a slightly lower number of correctly predicted base pairs (56.3 versus 58.7), a significantly smaller number of incorrectly predicted base pairs (62.1 versus 78.8), and a slightly higher number of base pairs from the native structure that where not predicted (78.5 versus 76.1). Note that average absolute numbers have to be interpreted with care since longer sequences typically have a greater influence. Furthermore, HelixPSO has on average a slightly lower sensitivity (51.0% versus 52.0%), a higher specificity (58.7% versus 52.1%), and a higher F-measure (54.1% versus 51.8%). All this holds also when only the sequences of length > 500 are considered.

Organism	Length	TP		FP		FN	
		PSO	GA	PSO	GA	PSO	GA
G. stearothermophilus	117	23	23	7	10	15	15
S. cerevisiae	118	30	33	6	6	7	4
E. coli	120	28	10	3	29	12	30
H. marismortui	122	29	27	0	3	9	11
T. Aquaticus	123	26	33	7	3	14	7
D. radiodurans	124	24	25	0	8	16	15
M. anisopliae (3)	394	59	75	44	46	61	45
C. saccharophila	454	85	86	50	51	41	40
M. anisopliae (2)	456	41	55	80	80	74	60
A. lagunensis	468	67	68	36	63	46	45
H. rubra	543	64	79	81	82	74	59
A. griffini	556	89	81	72	80	42	50
P. leucosticta	605	67	63	82	90	54	58
C. elegans	697	30	55	115	147	159	134
D. virilis	784	54	65	139	177	179	168
A. cahirinus	940	81	74	118	154	179	186
X. laevis	945	101	93	103	147	150	158
H. sapiens	954	84	89	123	161	182	177
A. fulgens	964	87	82	113	160	178	183
Average	-	56.3	58.7	62.1	78.8	78.5	76.1
Av.,$length > 500$	-	73.0	75.7	105.1	133.1	133.0	130.3

Table 4 Comparison of best values of HelixPSO (PSO) and RnaPredict (GA), measuring correctly predicted base pairs (TP), incorrectly predicted base pairs (FP) and base pairs in native structure that were not predicted (FN).

Organism	Length	SE		SP		FM	
		PSO	GA	PSO	GA	PSO	GA
G. stearothermophilus	117	60.5	60.5	76.7	69.7	67.6	64.8
S. cerevisiae	118	81.1	89.2	83.3	84.6	82.2	86.8
E. coli	120	70.0	25.0	90.3	25.6	78.9	25.3
H. marismortui	122	76.3	71.1	100.0	90.0	86.6	79.4
T. Aquaticus	123	65.0	82.5	78.8	91.7	71.2	86.8
D. radiodurans	124	60.0	62.5	100.0	75.8	75.0	68.5
M. anisopliae (3)	394	49.2	62.5	57.3	62.0	52.9	62.2
C. saccharophila	454	67.5	68.3	63.0	62.8	65.1	65.4
M. anisopliae (2)	456	35.7	47.8	33.9	40.7	34.7	44.0
A. lagunensis	468	59.3	60.2	65.0	51.9	62.0	55.7
H. rubra	543	46.4	57.2	44.1	49.1	45.2	52.8
A. griffini	556	67.9	61.8	55.3	50.3	61.0	55.5
P. leucosticta	605	55.4	52.1	45.0	41.2	49.6	46.0
C. elegans	697	15.9	29.1	20.7	27.2	18.0	28.1
D. virilis	784	23.2	27.9	28.0	26.9	25.4	27.4
A. cahirinus	940	31.2	28.5	40.7	32.5	35.3	30.3
X. laevis	945	39.3	37.1	49.5	38.8	43.8	37.9
H. sapiens	954	31.3	33.5	40.6	35.6	35.4	34.5
A. fulgens	964	32.8	30.9	43.5	33.9	37.4	32.3
Average	-	51.0	52.0	58.7	52.1	54.1	51.8
Av.,$length > 500$	-	38.2	39.8	40.8	37.3	39.0	38.3

Table 5 Comparison of best values of HelixPSO (PSO) and RnaPredict (GA), measuring sensitivity in % (SE), specificity in % (SP), and F-measure in % (FM).

When considering the behavior of both algorithms on the different sequences it is interesting that their results often differ greatly. For example, for C. elegans RnaPredict has 55 correctly predicted base pairs, a sensitivity of 29.1%, and an specificity of 27.2% whereas HelixPSO has only 30, respectively 15.9% and 20.7%. This is different for E. coli where HelixPSO has 28 correctly predicted base pairs, a sensitivity of 70% and a specificity of 90.3%, whereas RnaPredict has only 10, respectively 25% and 25.6%.

		TP		FP		FN	
Organism	Length	PSO	SA	PSO	SA	PSO	SA
S. cerevisiae	118	30	33	6	6	7	4
H. marismortui	122	29	27	0	3	9	11
H. rubra	543	64	79	81	83	74	59
A. griffini	556	89	87	72	81	42	44
D. virilis	784	54	80	139	159	179	153
X. laevis	945	101	112	103	141	150	139
H. sapiens	954	84	116	123	128	182	150
S. acidocaldarius	1492	236	226	187	226	232	242
Averages	-	85.9	95.0	88.9	103.4	109.4	100.3
Av.,$length > 500$	-	104.7	116.7	117.5	136.3	143.2	131.2

Table 6 Comparison of best values of HelixPSO (PSO) and SARNA-Predict (SA) with INN-HB and INN energy models, measuring correctly predicted base pairs (TP), incorrectly predicted base pairs (FP) and base pairs in native structure that were not predicted (FN).

		SE		SP		FM	
Organism	Length	PSO	SA	PSO	SA	PSO	SA
S. cerevisiae	118	81.1	89.2	83.3	84.6	82.2	86.8
H. marismortui	122	76.3	71.1	100.0	90.0	86.6	79.4
H. rubra	543	46.4	57.2	44.1	48.8	45.2	52.7
A. griffini	556	67.9	66.4	55.3	51.8	61.0	58.2
D. virilis	784	23.2	34.3	28.0	33.5	25.4	33.9
X. laevis	945	39.3	44.6	49.5	44.3	43.8	44.4
H. sapiens	954	31.3	43.6	40.6	47.5	35.4	45.5
S. acidocaldarius	1492	50.4	48.3	55.8	50.0	53.0	49.1
Averages	-	52.0	56.8	57.1	56.3	54.1	56.3
Av.,$length > 500$	-	43.1	49.1	45.6	46.0	44.0	47.3

Table 7 Comparison of best values of HelixPSO (PSO) and SARNA-Predict (SA) with INN-HB and INN energy models, measuring sensitivity in % (SE), specificity in % (SP), and F-measure in % (FM).

Tables 6 and 7 compare HelixPSO with SARNA-Predict as it is described in [35]. Both algorithms differ quite widely in their results. SARNA-Predict tends to predict more base pairs correctly than HelixPSO, performing better on 5 out of 8 sequences with an average number of 95.0 correctly predicted base pairs compared to a number of 85.9 for HelixPSO. However, HelixPSO has a smaller number of false positives on all sequences with the exception of the shortest sequence where both algorithms have the same number of false positives. On average HelixPSO has 88.9 false positives compared to 103.4 of SARNA-Predict. With respect to sensitivity and F-measure SARNA-Predict performs better on 5 of the 8 sequences and HelixPSO performs better on the other 3 sequences. The average sensitivity of SARNA-Predict 56.8% and the avreage F-measure

of 56.3%. HelixPSO has slightly lover average values of 52.0%, respectively 54.1%. With respect to specificity, each of the two algorithm performs better than the other on 4 of the 8 eight sequences. On average, HelixPSO is slightly better with a specificity of 57.1% compared to a value of 56.3% for SARNA-Predict.

Considering the results it seems advantageous to use both algorithms — HelixPSO and SARNA-Predict — instead of only one of them alone because they give good results on different sequences. The example of sequence *H. sapiens*, for example, shows that SARNA-Predict can correctly predict 116 correct base pairs but HelixPSO has only 84 correct base pairs. On the other hand, for the *X. laevis* sequence, HelixPSO has 103 false positives whereas the number for SARNA-Predict is 141.

		TP		FP		FN	
Organism	Length	PSO	SA	PSO	SA	PSO	SA
S. cerevisiae	118	30	33	6	6	7	4
M. anisopliae (3)	394	59	70	44	64	61	45
A. lagunensis	468	67	84	36	46	46	29
H. rubra	543	64	89	81	68	74	49
A. griffini	556	89	97	72	68	42	34
D. virilis	784	54	98	139	134	179	135
X. laevis	945	101	116	103	121	150	135
H. sapiens	954	84	119	123	125	182	147
Averages	-	68.5	88.3	75.5	79.0	92.6	72.3
Av.,$length > 500$	-	78.4	103.8	103.6	103.2	125.4	100.0

Table 8 Comparison of best values of HelixPSO (PSO) and SARNA-Predict (SA) with efn2 energy model, measuring correctly predicted base pairs (TP), incorrectly predicted base pairs (FP), base pairs in native structure that were not predicted (FN).

		SE		SP		FM	
Organism	Length	PSO	SA	PSO	SA	PSO	SA
S. cerevisiae	118	81.1	89.2	83.3	84.6	82.2	86.8
M. anisopliae (3)	394	49.2	60.9	57.3	52.2	52.9	56.2
A. lagunensis	468	59.3	74.3	65.0	64.6	62.0	69.1
H. rubra	543	46.4	64.5	44.1	56.7	45.2	60.3
A. griffini	556	67.9	74.0	55.3	58.8	61.0	65.5
D. virilis	784	23.2	42.1	28.0	42.2	25.4	42.2
X. laevis	945	39.3	46.2	49.5	48.9	43.8	47.5
H. sapiens	954	31.3	44.7	40.6	48.8	35.4	46.7
Averages	-	49.7	62.0	52.9	57.1	51.0	59.3
Av.,$length > 500$	-	41.6	54.3	43.5	51.1	42.2	52.4

Table 9 Comparison of best values of HelixPSO (PSO) and SARNA-Predict (SA) with efn2 energy model, measuring sensitivity in % (SE), specificity in % (SP), and F-measure in % (FM).

Tables 8 and 9 compare HelixPSO with an advanced version of SARNA-Predict using the efn2 energy model as presented in [36]. The sequences that were used in that context differ slightly from those in [35]. The *H. marismortui* and *S. acidocaldarius* sequences have been replaced by *M. anisopliae (2)* and *A. lagunensis*. With the efn2 energy model, the prediction quality of SARNA-Predict improves strongly. SARNA-Predict predicts more base pairs correctly than HelixPSO for all the listed sequences,

with an average of 88.3 to 68.5 correctly predicted base pairs. However, HelixPSO still outperforms SARNA-Predict with respect to false positives with an average value of 75.5 versus 79.0. Overall, SARNA-Predict scores higher in sensitivity, specificity and F-measure, the latter with a value of 59.3% compared to 51.0% of HelixPSO.

		TP		FP		FN	
Organism	Length	PSO	Fold	PSO	Fold	PSO	Fold
G. stearothermophilus	117	23	25	7	9	15	13
S. cerevisiae	118	30	28	6	14	7	9
E. coli	120	28	10	3	28	12	30
H. marismortui	122	29	29	0	5	9	9
T. Aquaticus	123	26	8	7	29	14	32
D. radiodurans	124	24	27	0	6	16	13
M. anisopliae (3)	394	59	67	44	54	61	53
C. saccharophila	454	85	93	50	50	41	33
M. anisopliae (2)	456	41	25	80	116	74	90
A. lagunensis	468	67	44	36	79	46	69
H. rubra	543	64	52	81	119	74	86
A. griffini	556	89	62	72	113	42	69
P. leucosticta	605	67	67	82	119	54	54
C. elegans	697	30	40	115	175	159	149
D. virilis	784	54	36	139	204	179	197
A. cahirinus	940	81	41	118	189	179	219
X. laevis	945	101	61	103	191	150	190
H. sapiens	954	84	88	123	168	182	178
A. fulgens	964	87	48	113	194	178	217
S. acidocaldarius	1492	236	255	187	242	232	213
Averages	-	65.3	55.3	68.3	105.2	86.2	96.2
Av.,$length > 500$	-	89.3	75.0	113.3	171.4	142.9	157.2

Table 10 Comparison of best values of HelixPSO (PSO) and RNAfold (Fold), measuring correctly predicted base pairs (TP), incorrectly predicted base pairs (FP) and base pairs in native structure that were not predicted (FN).

In Table 10 and 11, the results for HelixPSO are compared with those of the minimum free energy structure as calculated by the RNAfold algorithm of the ViennaRNA package. HelixPSO performs significantly better. It has more correctly predicted base pairs on 11 sequences whereas RNAfold is better on only 7 sequences (both algorithms are equal on two sequence). Furthermore, HelixPSO has fewer incorrectly predicted base pairs on all sequences except *C. saccharophila*, where both algorithms are tied. On average, HelixPSO is leading 50.9 % to 42.2 % in sensitivity, 58.6 % to 40.6 % in specificity, and 54.0 % to 41.1 % with respect to the F-measure. Altogether, the results show that HelixPSO clearly outperforms RNAfold on the test sequences with respect to the best values over several runs.

The run times of the serial version of HelixPSO are shown in Table 12. The run times were measured on eight core 64 Bit PCs with Intel Xeon 2.33GHz processors and 8 to 16 GB RAM. The variations in runtime due to differing parameter settings are to be explained by two causes. When solution acceptance is greedy, no resources are used to arrive at the decision whether to accept or reject a new structure. When the annealing method of solution acceptance is employed, two energy evaluations are required, one each for the previous and the new structure of the particle in question. The costs of energy evaluations grow with sequence length. Furthermore, a high rate of random particle movement decreases the runtime. This is due to the fact that a random

		SE		SP		FM	
Organism	Length	PSO	Fold	PSO	Fold	PSO	Fold
G. stearothermophilus	117	60.5	65.8	76.7	73.5	67.6	69.4
S. cerevisiae	118	81.1	75.7	83.3	66.7	82.2	70.9
E. coli	120	70.0	25.0	90.3	26.3	78.9	25.6
H. marismortui	122	76.3	76.3	100.0	85.3	86.6	80.6
T. Aquaticus	123	65.0	20.0	78.8	21.6	71.2	20.8
D. radiodurans	124	60.0	67.5	100.0	81.8	75.0	74.0
M. anisopliae (3)	394	49.2	55.8	57.3	55.4	52.9	55.6
C. saccharophila	454	67.5	73.8	63.0	65.0	65.1	69.1
M. anisopliae (2)	456	35.7	21.7	33.9	17.7	34.7	19.5
A. lagunensis	468	59.3	38.9	65.0	35.8	62.0	37.3
H. rubra	543	46.4	37.7	44.1	30.4	45.2	33.7
A. griffini	556	67.9	47.3	55.3	35.4	61.0	40.5
P. leucosticta	605	55.4	55.4	45.0	36.0	49.6	43.6
C. elegans	697	15.9	21.2	20.7	18.6	18.0	19.8
D. virilis	784	23.2	15.5	28.0	15.0	25.4	15.2
A. cahirinus	940	31.2	15.8	40.7	17.8	35.3	16.7
X. laevis	945	39.3	24.3	49.5	24.2	43.8	24.3
H. sapiens	954	31.3	33.1	40.6	34.4	35.4	33.7
A. fulgens	964	32.8	18.1	43.5	19.8	37.4	18.9
S. acidocaldarius	1492	50.4	54.5	55.8	51.3	53.0	52.8
Averages	-	50.9	42.2	58.6	40.6	54.0	41.1
Av.,$length > 500$	-	39.4	32.3	42.3	28.3	40.4	29.9

Table 11 Comparison of best values of HelixPSO (PSO) and RNAfold (Fold), measuring sensitivity in % (SE), specificity in % (SP), and F-measure in % (FM).

swap is simpler to execute than the selection process of indices to be swapped when a reference structure is used. The results show that for 6S RNAs of length up to 124 HelixPSO takes between 5 and 10 seconds for a run (which produces 50000 solutions). For sequences longer than that but of length shorter than 1000 nucleotides HelixPSO take approximately between 1 and 10 minutes (when producing 100000 solutions). For the longest sequences of *S. acidocaldarius* HelixPSO took between half and hour and 4 hours to produce 200000 solutions, depending on the parameter settings used. It should be noted that it is possible to reduce the number of iterations performed by HelixPSO without suffering a dramatic loss in prediction quality.

Figure 2 shows the speedup (i.e., the runtime of the single processor HelixPSO divided by the runtime of the multi-processor HelixPSO) achieved through parallelization in a network of Intel Xeon 3.0 GHz dual core units with 4GB RAM. The results are similar for all four test sequences. The speedup values correspond to an efficiency (i.e. the speedup divided by the number of processors) of about 0.7 for the *D. virilis* and *X. laevis* sequences and 0.8 for the *A. lagunensis* and *S. acidocaldarius* sequences when using ten processors. It is notable that using two processors for the *S. acidocaldarius* sequence takes less than half of the time of using one processor only. This super linear speedup can be explained as a consequence of the increased memory requirements for long sequences. It requires a single processor to swap out memory while a set of processors can split up the particles and therefore distribute the memory requirements. Altogether, the speedup curves show that parallel version of HelixPSO is interesting for solving longer RNA sequences because it runs efficiently on small to medium size PC clusters as they are available in many labs.

Figure 3 (left) shows the best secondary structure of the *A. lagunensis* RNA that were found by HelixPSO during 30 runs with the standard value set with free energy

Length	Organism	Runtime
117	*G. stearothermophilus*	7.1-10.7
118	*S. cerevisiae*	4.8-7.8
120	*E. coli*	7.0-10.6
122	*H. marismortui*	7.1-11.1
123	*T. Aquaticus*	7.6-11.6
124	*D. radiodurans*	6.3-10.0
394	*M. anisopliae (3)*	48.7-303.1
454	*C. saccharophila*	89.9-427.2
456	*M. anisopliae (2)*	84.8-430.1
468	*A. lagunensis*	61.0-349.2
543	*H. rubra*	118.2-531.5
556	*A. griffini*	89.6-405.4
605	*P. leucosticta*	135.0-562.4
697	*C. elegans*	52.5-249.2
784	*D. virilis*	69.9-281.8
940	*A. cahirinus*	116.5-453.2
945	*X. laevis*	146.8-529.0
954	*H. sapiens*	154.9-549.2
964	*A. fulgens*	123.0-483.6
1492	*S. acidocaldarius*	1958.3-14293.7

Table 12 Runtime of *HelixPSO* in seconds for different parameter settings (as explained in the text)

as only scoring criterion, i.e., $\lambda = 0.0$. Clearly, when compared to the native structure of the RNA that is shown in the right part of Figure 3 there are many differences. Two other secondary structures that were the best found in two of 30 runs of HelixPSO, with free energy and centroid information weighed equally, i.e.,$\lambda = 0.5$, are shown in Figure 4. This figure illustrates that with $\lambda = 0.5$ the structures become more similar to the native structure than for $\lambda = 0.5$.

In conclusion, we state that HelixPSO tends to make more accurate predictions than SetPSO, though only results for four sequences have been published. HelixPSO furthermore outperforms SARNA-Predict as far as average numbers are concerned. There are not enough data for average results published on RnaPredict to allow a meaningful comparison. HelixPSO also performs about 5% better than RnaPredict on a comparison set of 19 sequences with respect to the F-measure, which is based on all 5 of the other statistics. In turn, SARNA-Predict performs about 5% better than HelixPSO on the 8 sequences for which SARNA-Predict was tested with the INN and INN-HB energy models. When SARNA-Predict is used with the efn2 energy model, it scores clearly higher than HelixPSO both with respect to correctly predicted base pairs and false positives. Since RnaPredict compares quite well to mfold in [42], HelixPSO also does on the 19 sequences that were tested.

8 Conclusions

In this paper we have proposed the Particle Swarm Optimization (PSO) algorithm HelixPSO for the prediction of RNA secondary structures. HelixPSO uses thermody-

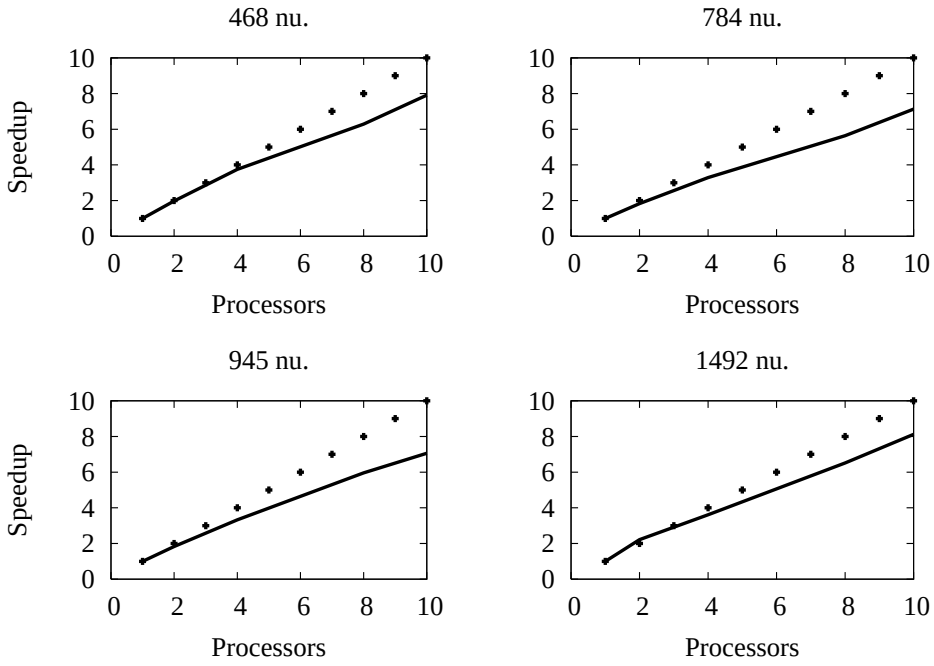

Fig. 2 Speed-up of the HelixPSO algorithm as a function of the number of processors used for the sequences *A. lagunensis* of length 468, *D. virilis* of length 784, *X. laevis* of length 945, and *S. acidocaldarius* of length 1492; the dotted line indicates the ideal linear speedup

namic information as well as the centroid as a reference structure and is based on a multiple swarm approach. Each particle has a target set of reference positions are used to define the direction of movement of a particle. As an extension of the classical PSO principle an acceptance function is used to decide whether a particle moves to the new position.

It was shown experimentally fore 20 RNA test sequences from a variety of taxa, different RNA classes and different lengths (ranging from 117 to 1492) that HelixPSO performs on average better than the simulated annealing algorithm SARNA-Predict and at least as good as the genetic algorithm RNA-Predict. With respect to best values found over several runs, HelixPSO outperforms RnaPredict, but does not perform as well as SARNA-Predict, in particular when it is used with the efn2 energy model. HelixPSO also predicts native secondary structures more accurately than the dynamic programming algorithm RNAfold and the other existing PSO algorithm SetPSO. The results show that HelixPSO complements RnaPredict and SARNA-Predict well since the algorithms perform very differently on several test sequences.

We also proposed a parallel version of HelixPSO and it was shown that it works efficiently for smaller to medium size PC cluster. For future work it is planned to implement HelixPSO as a multi-objective algorithm for optimizing free energy as well as maximizing the conformance with a reference structure, such as the centroid used in this paper. It is also planned to design a version of HelixPSO that delivers solutions of a larger diversity over several runs in order to increase the performance with respect to best results.

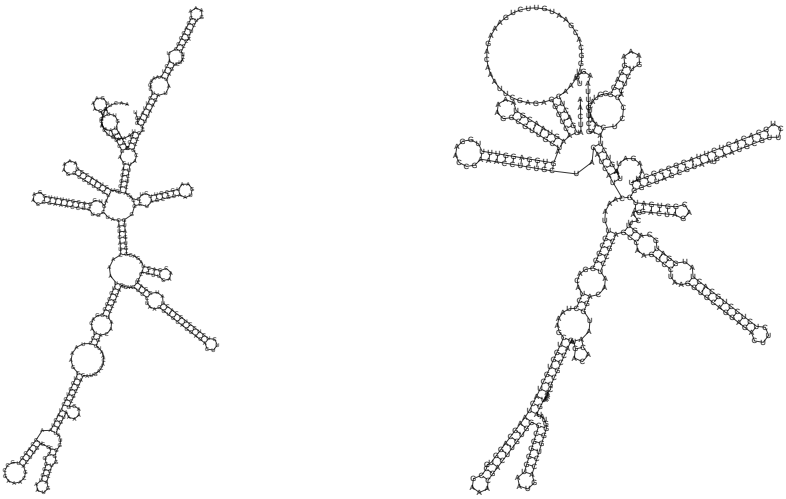

Fig. 3 Best RNA secondary structure with respect to minimum free energy computed by HelixPSO (left) and inferred native structure (right) for the *M. anisopliae(3)* RNA of length 394

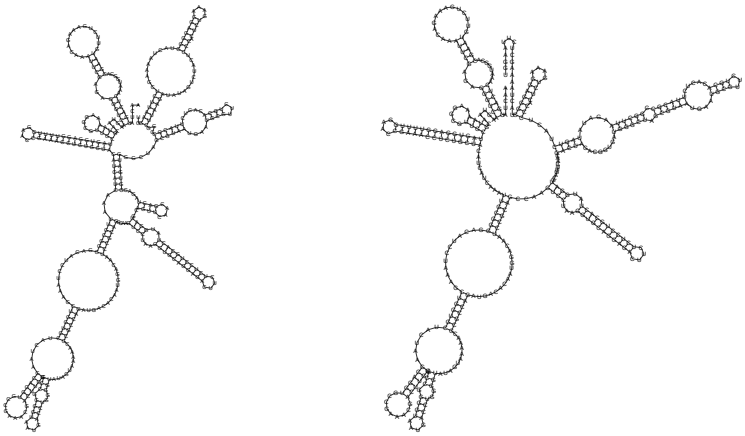

Fig. 4 Two best found RNA secondary structures of HelixPSO for the *M. anisopliae(3)* RNA of length 394

References

1. Birattari, M. and Dorigo, M.: How to assess and report the performance of a stochastic algorithm on a benchmark problem: mean or best result on a number of runs? Optimization Letters **1 3**,309–311 (2007)
2. Alba, E.: Parallel Metaheuristics: A New Class of Algorithms. John Wiley & Sons (2005)
3. van Batenburg, F.H.D., Gultyaev, A.P., Pleij, C.: An apl-programmed genetic algorithm for the prediction of RNA secondary structure. J. theor. Biol. **174**, 269–280 (1995)
4. Benedetti, G., Morosetti, S.: A genetic algorithm to search for. optimal and suboptimal RNA secondary structures. Biophys. Chem. **55**, 253–259 (1995)
5. Burns, G., Daoud, R., Vaigl, J.: LAM: An Open Cluster Environment for MPI. In: Proceedings of Supercomputing Symposium, pp. 379–386 (1994). URL

http://www.lam-mpi.org/download/files/lam-papers.tar.gz
6. Cannone, J.J., et al.: The comparative RNA web (crw) site: an online database of comparative sequence and structure information for ribosomal, intron, and other RNAs. BMC Bioinformatics **3** (2002)
7. Chen, J.H., Le, S.Y., Maizel, J.V.: Prediction of common secondary structures of RNAs: a genetic algorithm approach. Nucleic Acids Res. **28**(4), 991–999 (2000)
8. Doudna, J.A.: Structural genomics of RNA. Nature Struct. Biol. **7**, 954–956 (2000)
9. Engelbrecht, A.P.: Fundamentals of Computational Swarm Intelligence. John Wiley & Sons (2006)
10. Giegerich, R., Reeder, J.: From RNA folding to thermodynamic matching including pseudoknots. Tech. Rep. Technical Report 2003-03, Universitt Bielefeld (2003)
11. Goldberg, D.E., Lingle, R.J.: Alleles, loci and the travelling salesman problem. Proceedings of the First International Conference on Genetic Algorithms pp. 154–159 (1985)
12. Gultyaev, A.P., van Batenburg, F.H.D., Pleij, C.W.A.: The computer-simulation of RNA folding pathways using a genetic algorithm. J. Mol. Biol. **250**, 37–51 (1995)
13. Gultyaev, A.P., et al.: Dynamic competition between alternative structures in viroid RNAs simulated by an RNA folding algorithm. J. Mol. Biol. **276**, 43–55 (1998)
14. Hofacker, I.L., Fontana, W., Stadler, P.F., Bonhoeffer, S., Tacker, M., Schuster, P.: Fast folding and comparison of RNA secondary structures. Monatshefte f. Chemie **125**, 167–188 (1994)
15. Hofacker, I.L., Stadler, P.F.: Memory efficient folding algorithms for circular RNA secondary structures. Bioinformatics **22**(10), 1172–1176 (2006)
16. Isambert, H., Siggia, E.D.: Modeling RNA folding paths with pseudoknots: Application to hepatitis delta virus ribozyme. Proc Natl Acad Sci USA **97**(12), 6515–6520 (2000)
17. Jin, N.B., Rahmat-Samii, Y.: Parallel particle swarm optimization and finite-difference time-domain(pso/fdtd) algorithm for multiband and wide-band patch antenna designs. IEEE Transactions on Antennas and Propagation **53**, 3459–3468 (2005)
18. Jr., J.S., Turner, D.H.: Measuring the thermodynamics of RNA secondary structure formation. Biopolymers **44**, 309–319 (1997)
19. Kennedy, J., Eberhart, R.C.: Particle swarm optimization. In: Proc. of IEEE International Conference on Neural Networks (ICNN), pp. 1942–1948. Perth, Australia (1995)
20. M., G., Middendorf, M.: A particle swarm optimizer for finding minimum free energy RNA secondary structures. Proceedings of the IEEE Swarm Intelligence Symposium (2007)
21. Mathews, D.H., Sabina, J., Zuker, M., Turner, H.: Expanded sequence dependence of thermodynamic parameters provides robust prediction of RNA secondary structure. JMB **288**, 911–940 (1999)
22. Mills, D., Priano, C., Merz, P., Binderow, B.: Q RNA bacteriophage: mapping cis-acting elements within an RNA genome. J. Virol. **64**, 3872–3881 (1990)
23. Mostaghim, S., Branke, J., Schmeck, H.: Multi-objective particle swarm optimization on computer grids. In: GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation, pp. 869–875. ACM Press, New York, NY, USA (2007). DOI http://doi.acm.org/10.1145/1276958.1277127
24. Neethling, M., Engelbrecht, A.P.: Determining RNA secondary structure using set-based particle swarm optimization. In: IEEE Congress on Evolutionary Computation(CEC2006), pp. 1670–1677 (2006)
25. Oliver, I.M., Smith, D.J., Holland, J.R.C.: A study of permutation crossover operators on the traveling salesman problem. In: Proceedings of the Second International Conference on Genetic Algorithms (ICGA-87), pp. 224–230. Lawrence Erlbaum Associates, Inc. (1987)
26. Sahina, F., Yavuza, M., Arnavut, Z., Uluyol, .: Fault diagnosis for airplane engines using bayesian networks and distributed particle swarm optimization. Parallel Computing **33**, 124–143 (2007)
27. Schutte, J., Reinbolt, J., Fregly, B., Haftka, R., George, A.: Parallel global optimization with the particle swarm algorithm. Int. J. Numer. Meth. Engng **61**, 2296–2315 (2004)
28. Serra, M.J., Turner, D.H.: Predicting thermodynamic properties of RNA. Meth. Enzymol **259**, 242–261 (1995)
29. Shapiro, B.A., Navetta, J.: A massively-parallel genetic algorithm for RNA secondary structure prediction. J. Supercomput. **8**, 195–207 (1994)
30. Shapiro, B.A., Wu, J.C.: An annealing mutation operator in the genetic algorithms for RNA folding. Comput. Appl. Biosci. **12**, 171–180 (1996)
31. Shapiro, B.A., et al.: The massively parallel genetic algorithm for RNA folding: mimd implementation and population variation. Bioinformatics **17**, 137–148 (2001)

32. de Smit, M., van Duin, J.: Control of prokaryotic translation initiation by mRNA secondary structure. Progress in Nucleic Acid Research in Molecular Biology **38**, 1–35 (1990)
33. Starkweather, T., McDaniel, S., Mathias, K.E., Whitley, L.D., Whitley, C., Belew, R., Booker, L.: A comparison of genetic sequencing operators. In: Proceedings of the Fourth International Conference on Genetic Algorithms, pp. 69–76. Morgan Kaufman, San Mateo, CA (1991)
34. Titov, I.I., et al.: A fast genetic algorithm for RNA secondary structure analysis. Russ. Chem. Bull. **51**, 1135–1144 (2002)
35. Tsang, H.H., Wiese, K.C.: Sarna-predict: A study of RNA secondary structure prediction using different annealing schedules. Proceedings of the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology pp. 239–246 (2007)
36. Tsang, H.H., Wiese, K.C.: The significance of thermodynamic models in the accuracy improvement of RNA secondary structure prediction using permutation-based simulated annealing. Proceedings of the IEEE Congress on Evolutionary Computation pp. 3879–3885 (2007)
37. Venter, G., Sobieszczanski-Sobieski, J.: A parallel particle swarm optimization algorithm accelerated by asynchronous evaluations. In: 6th World Congresses of Structural and Multidisciplinary Optimization Rio de Janeiro, p. 10pp. (2005)
38. Walter, A., Turner, D., Kim, J., Lyttle, M., Mller, P., Mathews, D., Zuker, M.: Coaxial stacking of helixes enhances binding of oligoribonucleotides and improves predictions of RNA folding. Proc. Natl. Acad. Sci. **91**, 9218–9222 (1994)
39. Whitley, D., Starkweather, T., Shaner, D.: The traveling salesman and sequence scheduling: quality solutions using genetic edge recombination. Handbook of Genetic Algorithms pp. 350–372 (1991)
40. Wiese, K., Goodwin, S.D.: Keep-best reproduction: a local family competition selection strategy and the environment it flourishes in. Constraints **6**, 399–422 (2001)
41. Wiese, K.C., Deschenes, A., Glen, E.: Permutation based RNA secondary structure prediction via a genetic algorithm. In: Proceedings of the 2003 Congress on Evolutionary Computation (CEC2003), pp. 335–342. IEEE Press (2003)
42. Wiese, K.C., Deschnes, A., Hendriks, A.: Rnapredict - an evolutionary algorithm for RNA secondary structure prediction. IEEE/ACM Transactions on Computational Biology and Bioinformatics, in press (2007)
43. Wiese, K.C., Glen, E.: A permutation-based genetic algorithm for the RNA folding problem: a critical look at selection strategies, crossover operators, and representation issues. BioSyst. Comput. Intel. Bioinformatics **72**, 29–41 (2003)
44. Wiese, K.C., Hendriks, A.: Comparison of p-rnapredict and mfold–algorithms for RNA secondary structure prediction. Bioinformatics **22**(8), 934–942 (2006)
45. Woese, C.R., Pace, N.R.: The RNA World. Cold Spring Harbor Laboratory Press, Cold Spring Harbor, NY (1993)
46. Xia, T., et al.: Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with watson-crick base pairs. Biochemistry **37**, 14,719–14,735 (1998)
47. Zuker, M.: Mfold web server for nucleic acid folding and hybridization prediction. Nucleic Acids Res **31**, 3406–3415 (2003)
48. Zuker, M., Sankoff, D.: RNA secondary structures and their prediction. Bull. Math. Biol. **46**, 591–621 (1984)