Cofolga: a genetic algorithm for finding the common folding of two RNAs

Akito Taneda *

Department of Electronic and Information System Engineering, Faculty of Science and Technology, Hirosaki University, Hirosaki, 036-8561, Japan

Abstract

In order to predict non-coding RNA genes and functions on the basis of genome sequences, accurate secondary structure prediction is useful. Although single-sequence folding programs such as mfold have been successful, it is of great importance to develop a novel approach for further improvement of the prediction performance. In the present paper, a secondary structure prediction method based on genetic algorithm, Cofolga, is proposed. The program developed performs folding and alignment of two homologous RNAs simultaneously. Cofolga was tested with a dataset composed of 13 tRNAs, seven 5S rRNAs, five RNase P RNAs, and five SRP RNAs; as a result, it turned out that the average prediction accuracies for the tRNAs, 5S rRNAs, RNase P RNAs, and SRP RNAs obtained by Cofolga with an optimal weight factor and default parameters were 83.6%, 81.8%, 73.5%, and 67.7 %, respectively. These results were superior to those obtained by a single-sequence folding based on free-energy minimization in which corresponding average prediction accuracies were 52.4%, 47.4%, 57.7%, and 52.3 %, respectively. Cofolga has a post-processing in which a single-sequence folding is performed after fixation of a predicted common structure; this post-processing enables Cofolga to predict a structure that is present in one of two RNAs alone. The executable files of Cofolga (for Windows/Unix/Mac) can be obtained by an e-mail request.

Key words: secondary structure prediction, free-energy minimization, sequence alignment, covariation, simulated annealing

* Tel./fax: +81-172-39-3662.
Email address: taneda@si.hirosaki-u.ac.jp (Akito Taneda).

1 Introduction

Non-coding RNAs (ncRNAs), which are the RNAs not translated to a protein, usually have their own characteristic secondary structures in accordance with their functions. Since the secondary structures play an important role in the analysis and prediction of the genes and functions of ncRNAs, various methods and programs for predicting the secondary structures have been developed. For example, free-energy minimization (FEM) (Zuker, 2003; Mathews et al., 1999) and covariation analysis (Chiu and Kolodziejcza, 1991; Gutell et al., 1992) have been widely used for the purpose.

Combination of different approaches is an attractive idea since it has a possibility to drastically improve the prediction accuracy. Sankoff algorithm is a dynamic programming approach for obtaining the simultaneous solution to the secondary structure prediction and the alignment of RNAs (Sankoff, 1985). In Sankoff algorithm, FEM and alignment (including covariation) are simultaneously taken into account by optimizing a hybrid objective function that is defined by a linear combination of an alignment-score term and a free-energy term. In the present paper, we call a secondary structure prediction which predicts the common secondary structure of a set of RNAs on the basis of FEM 'common folding prediction' (of course, including Sankoff algorithm). Since Sankoff algorithm is $O(N^4)$ in time and $O(N^6)$ in space, other common folding programs have been developed for more practical use: FOLDALIGN (Gorodkin et al., 1997) and RNAGA (Chen et al., 2000) are those for multiple sequences; Dynalign (Mathews and Turner, 2002) and CARNAC (Perriquet et al., 2003) are common folding programs developed for pairwise comparison. The common folding programs can be divided into two types in accordance with their purpose: Dynalign and RNAGA predict the whole structure of RNAs; FOLDALIGN and CARNAC are the programs for finding the local structures such as hairpin loops. In addition to the methods mentioned above, alignment folding, e.g. alifold (Hofacker et al., 2002) and the method included in X2s package (Juan and Wilson, 1999), have been proposed for predicting the common structure of a set of RNAs with a pre-defined sequence alignment of the RNAs. Any common folding program has advantages and disadvantages. For example, while Dynalign is the algorithm closest to Sankoff's one and is a very accurate algorithm, it needs a large memory proportional to N^2, where N is a sequence length (e.g. Dynalign needs 256 MB to perform the common folding of sequences of 218 nt and 234 nt) (Mathews and Turner, 2002); although alignment folding is fast, it cannot improve the alignment when the given alignment is not accurate.

In the present paper, we propose a genetic algorithm (GA), Cofolga (Common folding by genetic algorithm), for finding the common folding of two homologous RNAs. Cofolga predicts the whole common secondary structure of given

RNAs by optimizing a function that has a hybrid form. In addition, by using a post-processing, Cofolga can also predict a non-common structure that exists in one of the RNAs alone. In the present study, pairwise common folding is carried out for tRNAs, 5S rRNAs, RNase P RNAs, and SRP RNAs to evaluate the performance of the proposed method; from the results of the tests, it is shown that the proposed algorithm improves the accuracy of the secondary structure prediction compared with a single-sequence folding method.

2 Algorithm

In Cofolga algorithm, the common folding prediction problem of two RNAs is solved by GA. Our GA is derived from RAGA (Notredame et al., 1997) that is based on the simple GA described in (Goldberg, 1989). GA searches for the solution with the highest objective function (OF) by iteratively updating a population of individuals (solutions) with various GA operators such as crossovers and mutations. In the present GA, alignments are used as an individual in the population and the size of the population is fixed through a run. The OF, f, used in Cofolga is as follows:

$$f = s + \frac{w}{N} \sum_{i=1}^{N} \Delta G_i, \tag{1}$$

where s is an alignment score, w is a weight factor, N is the total number of RNA sequences ($N = 2$ in the present study), and ΔG_i is the free energy of the ith RNA. It is noted that the w must be negative, since lower free energy is more favorable. The schematic flowchart of Cofolga algorithm is shown in Figure 1. As shown in Figure 1, Cofolga comprises three GA steps (initialization, evaluation, reproduction) and a post-processing step. The detail of each step will be described in the next sections. The evaluation and reproduction steps repeats until the number of generations reaches the pre-defined maximum number (the population generated at nth GA step is called nth generation). As a maximum iteration number of the GA loops, 20 or 50 is used in accordance with the sequence length of the input sequences. A user can change the parameters of Cofolga such as the maximum iteration number and the population size.

2.1 Initialization

In initialization step, an initial population is generated by using dynamic programming with added noise (DPAN) (Gerstein and Levitt, 1996). First,

individuals (alignments) are randomly generated by DPAN to fill the initial population. Then, an individual which is not unique in the population is removed. If there is a vacancy in the initial population after the procedure with DPAN, the following three-step procedure is invoked to fill the vacancy: (i) two sequences are randomly aligned without gap opening (terminal gap only); (ii) a mutation operator (partial alignment by DPAN) is applied to the alignment (for the detail of this operator, see 'partial alignment with DPAN' subsection); (iii) if the generated alignment is unique, it is added to the population. This procedure repeats until the number of accepted individuals reaches a population size. The default value of the population size that ranges from 50 to 200 is automatically selected in accordance with the sequence length.

2.2 Evaluation

In evaluation step, alignment scoring and alignment folding (free-energy minimization for a given alignment) are performed to evaluate the OF of each individual (for the detail of the alignment folding used in the present study, see next subsection). The alignment score parameters used to evaluate the s in Equation 1 are as follows: match = 2, mismatch = 0, gap open = -3, and gap elongation = -1.

In standard GA, the fitness of each individual is defined and used to determine which individual is reproduced in next generation. In Cofolga, the fitness, t_k, of an individual k is calculated by $t_k = f_k - f^{\text{lowest}}$, where f_k is the OF of individual k and f^{lowest} is the lowest OF in the generation. From the t_ks, the reproduction probability of each individual is obtained by assuming that reproduction of each individual occurs with a probability proportional to t_k. After the calculation of t_ks, the expected offspring (EO) of each individual is obtained. EO is a small integer and is used as the number of virtual slots for the roulette-wheel selection in reproduction step.

2.2.1 Alignment folding by simulated annealing

For each individual, Cofolga performs alignment folding to obtain a common secondary structure with the lowest sum of the free-energies. In Cofolga, the alignment folding is done by using a simulated annealing (SA). SA is one of the stochastic combinatorial-optimization techniques like GA (Kirkpatrick et al., 1983). In SA, the objective function, E, to be optimized is called 'energy' and the minimum-energy state is searched for by accumulating small random conformational changes. In our algorithm, the energy is expressed by $E = (\Delta G_1 + \Delta G_2)/2$. Each conformational change is accepted or rejected in accordance with the following rule: if $\Delta E \leq 0$, the conformational change is

accepted (where ΔE is an energy change due to the conformational change); when $\Delta E > 0$, if $\exp(-\Delta E/\Theta) < r$, the conformational change is accepted, if not it is rejected (where Θ is a distribution parameter and r is a random number $\in [0,1]$). The Θ is controlled by a scheduling function. Our free-energy minimization for the alignment folding is done by the SA successfully applied to the single-sequence folding of RNA, where a conformational change is defined by formation or disruption of a single base pair (Schmitz and Steger, 1996). In the SA, the Θ is controlled by the following hyperbolic scheduling function:

$$\Theta(m) = \Theta_{\text{final}} + \Theta_{\text{init}} \left\{ \frac{n_{\text{half}} + 1}{n_{\text{half}} + m} - \frac{n_{\text{half}} + 1}{n_{\text{half}} + n_{\text{max}}} \right\}, \tag{2}$$

where m is an iteration number of SA steps, Θ_{final} and Θ_{init} are a final and initial Θ, respectively, and n_{max} is the maximum iteration number of the SA. n_{half} is an approximate half-life, i.e. $\Theta(n_{\text{half}}) \approx \Theta_{\text{final}} + \Theta_{\text{init}}/2$ for $n_{\text{half}} \ll n_{\text{max}}$. An application of the SA proposed by Schmitz and Gerhard to alignment folding is straightforward except for the point such that a base pair is defined between alignment columns (column pair) in alignment folding while it is defined between base positions in the case of single-sequence folding. In the base pairs included in a column pair, only C:G, A:U, and G:U pairs are allowed; in a column pair, a column with a gap is not allowed.

We adopt the lowest-energy structure through one SA run as the final result of the SA run. In our alignment folding, $\Theta_{\text{init}} = 6.0$ (kcal/mol), $\Theta_{\text{final}} = 1.0$ (kcal/mol), $n_{\text{max}} = 20 \times$ (the total number of possible column pairs) and $n_{\text{half}} = 0.05\, n_{\text{max}}$ are used; these values are determined based on (Schmitz and Steger, 1996) and our experiments. Free-energy parameters used are 'version 3.0 free-energy parameters' downloaded at mfold web site (Mathews et al., 1999; Zuker, 2003).

2.3 Reproduction

In this step, the population of next generation is created by copying, selecting and modifying the individuals of current generation. The reproduction procedure is composed of the following four steps. (i) The fittest half of the individuals in current population is copied to next generation. (ii) A GA operator is chosen at random. Each operator is invoked with an equal probability. (iii) The parents for the operator determined at step 2 are randomly selected. A crossover needs two parents and a mutation does only one. This parent selection is done by weighted roulette-wheel selection in which the EO obtained in evaluation step is used as the number of slots for each individual. If the newly generated individual is unique in the next generation, the individual is

inserted into the next generation and the EO of the parent(s) is reduced. If not, the generated individual is rejected. (iv) Step 2 and 3 repeat until the vacancies of the next generation is filled with new individuals. The GA operators used in Cofolga are a modified version of those used in (Notredame et al., 1997). The six GA operators (two crossovers, random and semi-greedy gap-block shuffling, partial alignment with DPAN, and local Cofolga) used in Cofolga are described below.

2.3.1 Crossover operators

We use a random uniform crossover taken from RAGA (Notredame et al., 1997) and its semi-greedy version developed for Cofolga. In uniform crossover, first, we find identical alignment blocks between the parents and divide parent A and B into L alignment blocks $B_1^{\mathrm{A}}...B_l^{\mathrm{A}}...B_L^{\mathrm{A}}$ and $B_1^{\mathrm{B}}...B_l^{\mathrm{B}}...B_L^{\mathrm{B}}$, respectively, where all even-numbered blocks or all odd-numbered blocks are the identical blocks. Then, for all blocks we count the number, c_l^{A} and c_l^{B}, of the columns included in the predicted column pairs, where the column pairs obtained in the previous evaluation step are used as the predicted column pairs (an example is shown in Figure 2). Finally, to generate an alignment that differs from both parents, a child individual is constructed by concatenating the blocks taken from one of the parents: $B_1^{v(1)}...B_l^{v(l)}...B_L^{v(L)}$, where $v(l) = \{\mathrm{A}, \mathrm{B}\}$. The random and semi-greedy version of the uniform crossover differ in terms of the method for determining $v(l)$. The random version determines $v(l)$ at random, while the semi-greedy version does in such a way that $\sum_{l=1}^{L} c_l^{v(l)}$ is maximized (when $c_l^{\mathrm{A}} = c_l^{\mathrm{B}}$, the one with higher alignment score is adopted). As can be known from the name, the semi-greedy operator is not 'greedy', since it is designed to increase a probability to find a common structure with low free energy and does not guarantee to increase the OF.

2.3.2 Anchor point for mutation operators

Mutation operators generate a new alignment by modifying an alignment. In the mutations of Cofolga, we introduce an anchor point to accelerate the convergence of the GA by avoiding destruction of a highly conserved region. The anchor point is defined as a continuous occurrence of matching of nucleotides in the alignment. As a default value, the smallest size for the anchor point is 8 nt (this value can be changed by a user). It is noted that the anchor point is not a perfect one, since a mutation (local Cofolga) can destroy it.

2.3.3 Random gap-block shuffling

This mutation shifts a randomly selected gap block (continuous gaps). The direction and the maximum size of the shift is randomly determined. If the gap block meets other gap block, anchor point, or the edge of the alignment during the shift, the shift is stopped.

2.3.4 Semi-greedy gap-block shuffling

This operator also shifts a gap block. The shift size is determined in accordance with the following five steps. (i) We find large helices ($\geq h$ bp) from the common structure obtained in the previous evaluation step (the default value of the h is 7 bp; the h can be changed by a user). (ii) We select a gap block at random. (iii) We find a loop ('virtual loop') including the gap block in 'virtual common secondary structure' which is defined as the secondary structure formed by only the large helices found at step 1 (an example is shown in Figure 3). (iv) The gap block is shifted (the direction and maximum size of the shift is randomly determined). During the shift, all possible column pairs within the virtual loop are examined and the following values are stored as a function of a shift size, x: $\alpha(x)$ = (the number of the possible column pairs included in the common helix longer than 3 column pairs), $\beta(x)$ = (the size of the longest common helix), and $\gamma(x)$ = (the alignment score of the gap-shifted alignment). The alignment score for the $\gamma(x)$ is calculated with the parameters used to calculate the s in Equation 1. The shift is stopped if the gap block meets the large helix found at step 1, anchor point, or the edge of the alignment. (v) We obtain the final shift size in accordance with the flowchart shown in Figure 4. By using this operator, we can search for locally stable structures beyond accidental small common helices.

2.3.5 Partial alignment with DPAN

When this operator is invoked, first an alignment region with a randomly determined size and position is chosen. Then, the selected region is re-aligned by DPAN. The parameters for the DPAN are as follows: match = 2.0, mismatch = -1.0, gap open = -2.5, gap elongation = -0.8, and terminal gap = -0.3. A random noise $\in [-0.8, 0.8]$ is independently added to each parameter. It is noted that each match (A, C, G, or U) has an independent noise. This operator causes gap insertion or deletion in the region.

2.3.6 Local Cofolga

To find a local common structure, this operator applies Cofolga algorithm to an alignment region with randomly determined size and position. After the

determination of the alignment region, this operator performs common folding of the alignment region in accordance with the algorithm shown in Figure 1 (except for constrained folding). This operator uses a population size of 20 and a GA maximum iteration number of 20. Only three operators (random uniform crossover, random gap-block shuffling, and partial alignment with DPAN) are used. Since local Cofolga is performed for a region with a few tens of nucleotides, it is suitable for finding a common hairpin structure.

2.4 Constrained folding by simulated annealing

As a post-processing, 'constrained folding' is carried out to predict a non-common structure. In this procedure, single-sequence FEM of each RNA is performed after the fixation of all base pairs that is predicted as a common structure by the GA. This FEM is carried out with a single-sequence version of the SA used in the evaluation step; in the SA used in constrained folding, we use re-annealing in which the lowest-energy structure through five SA runs is adopted as the final result. All prediction results described below were obtained by using constrained folding.

3 Software availability

The executable files of Cofolga (version 0.9.00) for Windows 2000/XP, Red Hat Linux 9, and Mac OS X can be obtained by an e-mail request to the author.

4 Results and Discussion

To test the present method, we applied Cofolga to 13 tRNAs, seven 5S rRNAs, five RNase P RNAs, and five SRP RNAs. The RNA sequences used in the present study have a wide range of sequence identity (tRNA, 26% - 79%; eubacteria 5S rRNA, 60% - 80%; archaea 5S rRNA, 49% - 82%; RNase P RNA, 54% - 83%; SRP RNA, 66% - 82%). The sequences, secondary structures (including G:U and other non-canonical base pairs), and multiple alignments of these RNAs were taken from the following databases: 'compilation of tRNA sequences' (Sprinzl et al., 1998), 5S ribosomal RNA database (Szymanski et al., 2002), the RNase P RNA database (Brown, 1999), and SRPDB (Rosenblad et al., 2003). The performance of Cofolga was analyzed with a prediction accuracy that is defined by 100 $\times$ (the number of correctly predicted base pairs)/(the total number of the base pairs in a reference structure). To compare

the results of Cofolga with the global minimum structures obtained by single-sequence folding, we performed predictions for the datasets by mfold quickfold server (Zuker, 2003).

4.1 The w dependence of the prediction accuracy

Figure 5 shows the average prediction accuracy for the datasets of tRNA and 5S rRNA as a function of w ($-1 \leq w \leq -100$). The average was taken for five runs with different initial random numbers. This averaging is necessary to analyze the performance of Cofolga since different initial random numbers may give different results when a stochastic optimization method like GA is used. As can be seen from Figure 5, the average prediction accuracies for the tRNAs and 5S rRNAs were approximately 83 % and 80 %, respectively, except for the ws near $w = 0$. These results indicate that Cofolga works well in relatively wide range of ws. In addition, it can be seen from the figure that w = -10 gives the highest prediction accuracy for both tRNA and 5S rRNA. This optimal w is adopted as a default value of w and the all results shown below were obtained using the optimal w.

4.2 tRNA

The prediction results for the tRNA dataset are shown in Table 1. These calculations were performed with w = -10 and default parameters. In the table, the results of a single-sequence folding (mfold) are also shown. As can be seen from the table, Cofolga improved the prediction accuracy of the tRNAs whose structure are poorly predicted by the single-sequence folding. As a result, the average prediction accuracy (83.6 %) obtained by Cofolga is much better than that by single-sequence folding (52.4 %). It is noted that Cofolga improved the accuracy not only in such a pair that one of the pair is successfully predicted and another one is poorly predicted by the single-sequence folding (e.g. RD1140 and RE4800), but also in such a pair that both RNAs are poorly predicted by the single-sequence folding (e.g. RD0260 and RE4800). In the present dataset, the tRNAs both with and without a variable stem are included. The average prediction accuracy for the pairs that is composed of a tRNA with a variable stem and that without a variable stem was 76.3 %, while the average prediction accuracy for the other pairs (pairs without a non-common structure) was 89.8 %. This result indicates that the presence of a non-common structure lessens the prediction accuracy; it is noted that it also indicates that Cofolga can improve the prediction accuracy of the tRNA pair with a non-common structure compared with prediction by the single-sequence folding.

4.3 5S rRNA

The prediction results for the 5S rRNA dataset obtained by Cofolga and the single-sequence folding are presented in Table 2. We can see from the table that in the 5S rRNA dataset Cofolga improved the prediction accuracy compared with prediction by the single-sequence folding; average prediction accuracy for the 5S rRNAs obtained by Cofolga was 81.8 %, while that by the single-sequence folding was 47.4 %.

4.4 RNase P RNA

In the beginning, we used a population size of 100 to predict the structure of the RNase P RNAs by Cofolga, obtaining an average prediction accuracy of 69.9 %; the average prediction accuracy for the same dataset by single-sequence folding is 57.7%. One reason why the improvement of the prediction accuracy of the RNase P RNAs was not so large as those of the tRNAs and 5S rRNAs is that the RNase P RNA dataset contains diverse sequences in length. The present RNase P RNA dataset is composed of five purple bacteria RNase P RNAs; three of those are taken from delta subdivision (*D. desulfuricans*, *D. vulgaris*, and *G. sulfurreducens*) and two of those belong to epsilon subdivision (*C. jejuni* and *H. pylori*). The delta-subdivision RNAs are from 359 to 367 nt in length, while the epsilon-subdivision RNAs have a length from 316 to 318 nt. The average prediction accuracy of intra-subdivision pairs (three pairs in delta subdivision and one pair in epsilon subdivision) was 76.7 %, while that of inter-subdivision pairs (six pairs) was 65.4 %. One possible cause of this low accuracy in the inter-subdivision pairs is an increase of complexity around the solution with the highest OF, because it is possible that the number of combinations around the best solution increases as the number of gaps rises. To test the hypothesis, we performed prediction of the RNase P RNAs by Cofolga with a population size of 200 to search wider conformational space. The result is shown in Table 3. By using a population size of 200, the average prediction accuracy of the inter-subdivision pairs increased to 72.0 %; as a result, the average prediction accuracy for the present RNase P RNA dataset increased to 73.5 %. Since large population size is essential for finding a good solution, we have adopted 200 as the default population size for the sequences longer than 300 nt, while the calculation with a population size of 200 needs a computational time about two times as long as the calculation with a population size of 100.

4.5 SRP RNA

The prediction results for the SRP RNA dataset are shown in Table 4. The SRP RNA dataset was composed of plant SRP RNAs (*A. thaliana* and *H. lupulus*). The reference secondary structures in the SRP RNA dataset were composed of only canonical and G:U pairs (other base pairs were excluded). As can be seen from Table 4, Cofolga outperfomed the single-sequence folding in most cases; as a result, the average prediction accuracy obtained by Cofolga was 67.7 %, while that by single-sequence folding was 52.3 %.

4.6 Pairwise alignment by Cofolga

As part of the results from Cofolga, a pairwise alignment of RNAs is obtained. In the present study, the pairwise alignments obtained by Cofolga were evaluated with a measure, m1, used in (Notredame et al., 1997); m1 = (the number of correctly aligned columns)/(the number of columns in a reference alignment), where an alignment taken from the databases was used as a reference alignment and columns with a gap were ignored. Average m1, $\overline{\text{m1}}$, for each dataset is shown in Table 5, where the average was taken over all pairwise alignments within a dataset. As can be seen from Table 5, Cofolga generated accurate alignments, i.e. those with a high m1 (> 80 %). It is noted that Cofolga generated good alignments even in the dataset (tRNA) composed of evolutionarily distant homologs (i.e. RNAs with a low sequence identity; see Figure 6 for the example of such an alignment).

4.7 Computational time and memory usage

Computational times of the tRNAs, 5S rRNAs, RNase P RNAs, and SRP RNAs are summarized in Table 6. As can be seen from the table, the RNase P RNAs and SRP RNAs needed particularly long computational times (approximately 7 and 5 h on average, respectively). This is chiefly due to the following reasons: (i) the SA in alignment folding is approximately $O(N^2)$ in time, where N is an alignment length; (ii) the population size and maximum iteration number used in the RNase P RNAs and SRP RNAs were larger than those of the tRNAs and 5S rRNAs. In the current version of Cofolga, the length of the input sequences is limited to shorter than 400 nt due to the slow computational speed of Cofolga.

Some calculations of the tRNAs and 5S rRNAs needed much longer computational times compared with the others in each RNA. This is mainly due to initialization and/or reproduction steps in our GA process. Since in these

steps only unique individuals are allowed to be added to a new population, it becomes a time-consuming process when many rejections occur. Such many rejections can occur if a user uses too large population size in order to process a RNA pair having relatively high sequence identity.

To measure the memory usage of Cofolga, we performed the common folding of two RNase P RNAs, *D. desulfuricans* (360 nt) and *G. sulfurreducens* (367 nt), with a population size of 200; this is one of the largest calculation in the present study. As a result, it tuned out that this calculation needs approximately 5 MB. This memory usage is very small compared with the memory usage of Dynalign, since Dynalign has a complexity of $O(N^2)$ in space and needs 256 MB to carry out the common folding of two R2 3' UTRs (218 nt and 234 nt), as described in (Mathews and Turner, 2002).

4.8 Comparison with Dynalign

Dynalign is a common folding program for predicting the whole secondary structure of two RNAs like Cofolga (Mathews and Turner, 2002). Dynalign is different from Cofolga in terms of the following five points. (i) The OF of Dynalign does not utilize sequence identity but uses only gap-penalty in its alignment-score term. (ii) The gap penalty of Dynalign is a linear gap, while Cofolga uses an affine gap penalty. (iii) Dynalign explores a limited conformational space to reduce computational complexity in time and space. (iv) Dynalign gives a rigorous solution (the solution with the best OF in the model). Cofolga does not guarantee to find the best solution since Cofolga is a stochastic method. (v) Dynalign does not have a post-processing to predict a non-common structure. In (Perriquet et al., 2003), Perriquet et al.have performed structure predictions by Dynalign for four RNase P RNA pairs (*D. desulfuricans* and one of the other four RNase P RNAs in the present dataset); they found that prediction accuracies of two intra-subdivision pairs are 79 % and 85 % (80.4 and 79.1 % by Cofolga) and that prediction accuracies of two inter-subdivision pairs are 40 % and 36% (68.1 and 76.2 % by Cofolga). The difference between the results of Dynalign and Cofolga seen in the inter-subdivision pairs could be due to 'difference 3' mentioned above. This is because when the best solution is located at out of the conformational space accessible by Dynalign, Dynalign fails finding the best solution. Cofolga, however, has a possibility to find a good solution near the best one, since the conformational space to be searched by Cofolga has no artificial constraints.

5 Conclusion

In the present paper we have proposed Cofolga algorithm that predicts the common secondary structure shared by two RNAs. Cofolga algorithm was tested with tRNAs, 5S rRNAs, RNase P RNAs, and SRP RNAs and it turned out that Cofolga can markedly improve the prediction accuracy of the secondary structures compared with the single-sequence folding. From the comparison with the alignments taken from the databases it was shown that Cofolga can generate accurate pairwise alignment, even when applied to evolutionarily distant homologs. Memory usage and computational speed of Cofolga were also examined and we concluded that Cofolga is a memory-efficient algorithm while its computational speed is slow. Although current version of Cofolga is a slow program, the computational speed can be improved by introducing some strategies, such as parallelized GA and alignment folding by dynamic programming (because FEM by SA could be slower than that by dynamic programming in many cases). Development of new versions in these directions is currently in progress. Cofolga needs only two homologous RNAs to predict a structure, therefore it suitable for performing a comparative study of the organisms with few related species whose genome sequence is available. Since reduction of the number of species required is a great advantage, Cofolga will be a useful tool for the comparative genomics on RNA such as functional non-coding RNA prediction.

Acknowledgements

Part of the computation for the present study was performed at Center for Computational Materials Science of the Institute for Materials Research, Tohoku University under the inter-university cooperative research program of the Institute for Materials Research, Tohoku University.

References

Brown, J. W., 1999. The Ribonuclease P Database. Nucleic Acids Res. 27, 314.

Chen, J. H., Le, S. Y., Maizel, J. V. 2000. Prediction of common secondary structures of RNAs: a genetic algorithm approach. Nucleic Acids Res. 28, 991-999.

Chiu, D. K. Y., Kolodziejczak, T., 1991. Inferring consensus structure from nucleic acid sequences. Comput. Appl. Biosci. 7, 347-352.

Gerstein, M., Levitt, M., 1996. Using iterative dynamic programming to obtain

accurate pairwise and multiple alignments of protein structures. Proc. Int. Conf. Intell. Syst. Mol. Biol. 4, 59-67.

Goldberg, D. E., 1987. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, NewYork.

Gorodkin, J., Heyer, L. J., Stormo, G. D., 1997. Finding the most significant common sequence and structure motifs in a set of RNA sequences. Nucleic Acids Res. 25, 3724-3732.

Gutell, R. R., Power, A., Hertz, G. Z., Putz, E. J., Stormo, G. D., 1992. Identifying constraints on the higher-order structure of RNA: continued development and application of comparative sequence analysis methods. Nucleic Acids Res. 20, 5785-5795.

Hofacker, I. L., Fekete, M., Stadler, P. F., 2002 Secondary structure prediction for aligned RNA sequences. J. Mol. Biol. 319, 1059-1066.

Juan, V., Wilson, C., 1999. RNA secondary structure prediction based on free energy and phylogenetic analysis. J. Mol. Biol. 289, 935-947.

Kirkpatrick, S., Gelatt Jr., C. D., Vecchi, M. P., 1983. Optimization by simulated annealing. Science 220, 671-680.

Mathews, D. H., Turner, D. H., 2002. Dynalign: an algorithm for finding the secondary structure common to two RNA sequences. J. Mol. Biol. 317, 191-203.

Mathews, D. H., Sabina, J., Zuker, M., Turner, D. H., 1999. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. J. Mol. Biol. 288, 911-940.

Notredame, C., O'Brien, E. A., Higgins, D. G., 1997. RAGA: RNA sequence alignment by genetic algorithm. Nucleic Acids Res. 25, 4570-4580.

Perriquet, O., Touzet, H., Dauchet, M., 2003. Finding the common structure shared by two homologous RNAs. Bioinformatics 19, 108-116.

Rosenblad, M. A, Gorodkin, J., Knudsen, B., Zwieb, C., Samuelsson, T., 2003. SRPDB (Signal Recognition Particle Database). Nucleic Acids Res. 31, 363-364.

Sankoff, D., Simultaneous solution of the RNA folding, alignment and protosequence problems. 1985. SIAM J. Appl. Math. 45, 810-824.

Schmitz, M., Steger, G., 1996. Description of RNA folding by "simulated annealing". J. Mol. Biol. 255, 254-266.

Sprinzl M., Horn, C., Brown, M., Ioudovitch, A., Steinberg, S., 1998. Compilation of tRNA sequences and sequences of tRNA genes. Nucleic Acids Res. 26, 148-153.

Szymanski, M., Barciszewska, M. Z., Erdmann, V. A., Barciszewski J., 2002. 5S Ribosomal RNA Database. Nucleic Acids Res. 30, 176-178.

Zuker, M., 2003. Mfold web server for nucleic acid folding and hybridization prediction. Nucleic Acids Res. 31, 3406-3415. Quick fold server and free-energy parameters are available at `http://www.bioinfo.rpi.edu/~zukerm/`.

FIGURE CAPTIONS

Fig. 1 A schematic flowchart of Cofolga algorithm.

Fig. 2 An example of the semi-greedy version of the uniform crossover. B_2^{A} and B_2^{B} are identical blocks. The number of the paired columns in each blocks is as follows: $c_1^{\mathrm{A}} = c_3^{\mathrm{B}} = 4$, $c_2^{\mathrm{A}} = c_2^{\mathrm{B}} = 0$, and $c_3^{\mathrm{A}} = c_1^{\mathrm{B}} = 2$. A child is constructed in such a way that $\sum_{l=1}^{3} c_l^{v(l)}$ $(v(l) = \{\mathrm{A}, \mathrm{B}\})$ is maximized (for detail, see subsection 2.3.1).

Fig. 3 An example of the procedure for finding a 'virtual loop' in semi-greedy gap-block shuffling. (a) a gap block is randomly chosen in an alignment with a predicted common structure. The selected gap block is indicated by a box. (b) A 'virtual common secondary structure' is found, where $h = 4$ bp is used (the helix of 2 bp is eliminated). The 'virtual loop' region including the selected gap block is indicated by a box. For more detail, see subsection 2.3.4.

Fig. 4 The flowchart for determining the shift size of the gap block in semi-greedy gap-block shuffling. The functions, α, β, and γ, are described in subsection 2.3.4.

Fig. 5 The w dependence of the average prediction accuracy. The data for tRNA and 5S rRNA are plotted with a solid and open circle, respectively. All results of five different runs for each w were distributed within ± 2.9 % around each average (data not shown). Solid and dashed lines are drawn for the guide to the eyes.

Fig. 6 A typical example of the alignment of tRNAs obtained by Cofolga (w = -30 and default parameters were used). The top two lines are aligned tRNA sequences. In 'pred', 'ref-0', and 'ref-1' lines, brackets indicate a predicted common secondary structure, the reference structure for RE2140, that for RL1141, respectively. These two tRNAs are successfully aligned in spite of their very low sequence identity.

Table 1
The percent prediction accuracy of 78 tRNA pairs. These results were obtained with w = -10 and default parameters. The tRNAs are taken from archaea (RD0500, RL0503, RS0380), eubacteria (RA1140, RA1660, RD1140, RE2140, RL1141, RS1141), chloroplast (RD2640), mitocondoria of animal (RD4800, RE4800), and bacteriophage (RD0260). The IDs in (Sprinzl et al., 1998) are used as tRNA name. In the table, the prediction results of the lowest energy structures obtained by single-sequence folding (mfold) are also shown. One data in the table corresponds to the prediction accuracy of the tRNA mentioned at the leftmost column; another tRNA of the pair can be found at the topmost row (except for the column of 'mfold'). The entries with a variable stem are denoted by *. In the table, the initial 'R' of the tRNA IDs is omitted.

	mfold	A1140	A1660	D0260	D0500	D1140	D2640	D4800	E2140	E4800	L0503*	L1141*	S0380*	S1141*
A1140	31.8	-	95.5	95.5	95.5	95.5	95.5	95.5	95.5	68.2	31.8	95.5	95.5	90.9
A1660	95.5	95.5	-	95.5	95.5	95.5	95.5	95.5	95.5	95.5	54.5	77.3	95.5	72.7
D0260	31.8	95.5	95.5	-	77.3	95.5	95.5	77.3	95.5	77.3	54.5	95.5	95.5	31.8
D0500	50.0	81.8	81.8	81.8	-	95.5	81.8	81.8	81.8	72.7	54.5	95.5	81.8	54.5
D1140	95.5	95.5	95.5	95.5	95.5	-	95.5	95.5	95.5	95.5	95.5	95.5	95.5	72.7
D2640	36.4	95.5	95.5	95.5	95.5	95.5	-	95.5	95.5	95.5	95.5	95.5	95.5	95.5
D4800	40.9	95.5	95.5	77.3	77.3	95.5	95.5	-	95.5	95.5	77.3	95.5	77.3	95.5
E2140	90.9	95.5	95.5	95.5	77.3	95.5	95.5	95.5	-	95.5	90.9	54.5	77.3	72.7
E4800	0.0	71.4	95.2	95.2	23.8	95.2	95.2	76.2	95.2	-	33.3	76.2	76.2	76.2
L0503*	52.0	28.0	84.0	48.0	48.0	48.0	92.0	84.0	92.0	40.0	-	92.0	92.0	92.0
L1141*	38.5	88.5	65.4	88.5	88.5	88.5	88.5	88.5	88.5	88.5	88.5	-	88.5	76.9
S0380*	92.3	92.3	92.3	92.3	92.3	92.3	92.3	92.3	46.2	92.3	92.3	80.8	-	92.3
S1141*	25.9	85.2	55.6	25.9	59.3	51.9	88.9	70.4	40.7	70.4	70.4	88.9	74.1	-

Table 2
The percent prediction accuracy of 21 5S rRNA pairs. These results were obtained with w = -10 and default parameters. The 5S rRNAs are taken from archaea and eubacteria (those taken from archaea are denoted by †.). The prediction results of the lowest energy structures by single-sequence folding (mfold) are also shown.

	mfold	*A. globiformis*	*D. mobilis*†	*H. morrhuae*†	*H. volcanii*†	*M. phlei*	*P. gingivalis*	*R. henselae*
A. globiformis	37.5	-	75.0	87.5	81.2	81.2	87.5	81.2
D. mobilis†	95.7	87.0	-	80.4	95.7	28.3	95.7	71.7
H. morrhuae†	21.2	84.8	78.8	-	84.8	97.0	84.8	90.9
H. volcanii†	29.0	87.1	74.2	87.1	-	100.0	54.8	74.2
M. phlei	100.0	93.5	100.0	100.0	100.0	-	100.0	100.0
P. gingivalis	24.0	92.0	60.0	84.0	68.0	100.0	-	76.0
R. henselae	24.2	81.8	60.6	33.3	63.6	93.9	78.8	-

Table 3
The percent prediction accuracy of 10 RNase P RNA (delta/epsilon purple bacteria) pairs. These results were obtained with w = -10 and default parameters. The prediction results of the lowest energy structures by single-sequence folding (mfold) are also shown. The RNAs taken from epsilon subdivision are denoted by ‡.

	mfold	*D. desulfuricans*	*D. vulgaris*	*G. sulfurreducens*	*C. jejuni*‡	*H. pylori*‡
D. desulfuricans	48.1	-	81.5	81.5	88.9	81.5
D. vulgaris	69.2	80.4	-	79.4	75.7	62.6
G. sulfurreducens	79.1	79.1	71.8	-	62.7	75.5
C. jejuni‡	64.8	68.1	68.1	65.9	-	59.3
H. pylori‡	27.4	76.2	72.6	66.7	72.6	-

Table 4
The percent prediction accuracy of 10 SRP RNA pairs. These results were obtained with w = -10 and default parameters. The prediction results of the lowest energy structures by single-sequence folding (mfold) are also shown.

	mfold	*A. thaliana*-A	*A. thaliana*-C	*A. thaliana*-D	*A. thaliana*-G	*H. lupulus*-A
A. thaliana-A	47.5	—	79.2	53.5	73.3	71.3
A. thaliana-C	42.3	71.2	—	70.2	56.7	51.9
A. thaliana-D	67.0	56.0	75.0	—	81.0	68.0
A. thaliana-G	52.0	75.5	63.3	75.5	—	66.3
H. lupulus-A	52.5	68.7	62.6	68.7	66.7	—

Table 5
The average accuracy, $\overline{\text{m1}}$, of the alignments obtained by Cofolga. The 5S rRNA dataset is divided into those of archaea (denoted by 5S rRNA-1) and eubacteria (5S rRNA-2) in accordance with the alignments taken from the database. Average sequence identity based on reference alignments, and the number of sequences in each dataset is also shown.

	tRNA	5S rRNA-1	5S rRNA-2	RNase P RNA	SRP RNA
$\overline{\text{m1}}$ (%)	83.1	86.9	89.1	83.4	91.0
seq. id (%)	48.0	61.6	66.7	63.6	70.8
#sequences	13	3	4	5	5

Table 6
Average and the longest computational times (CPU times). The computational times were measured by a PC with Intel Xeon CPU (2.4GHz) and RedHat Linux 9.

RNA type	average	longest	ave. seq. length (nt)
tRNA	2 min 37 s	9 min 50 s	79.4
5S rRNA	5 min 54 s	17 min 35 s	121.0
RNaseP RNA	7 h 16 min 52 s	10 h 26 min 34 s	344.0
SRP RNA	4 h 36 min 54 s	5 h 27 min 17 s	304.4

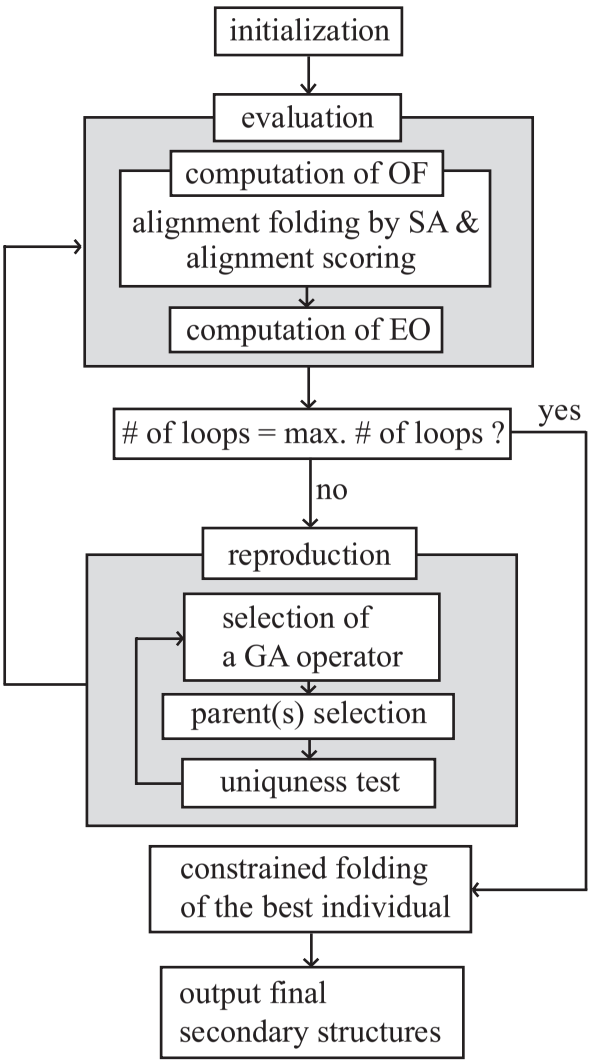
initialization
evaluation
computation of OF
alignment folding by SA &
alignment scoring
computation of EO
# of loops = max. # of loops ?
yes
no
reproduction
selection of
a GA operator
parent(s) selection
uniquness test
constrained folding
of the best individual
output final
secondary structures

Fig. 1.

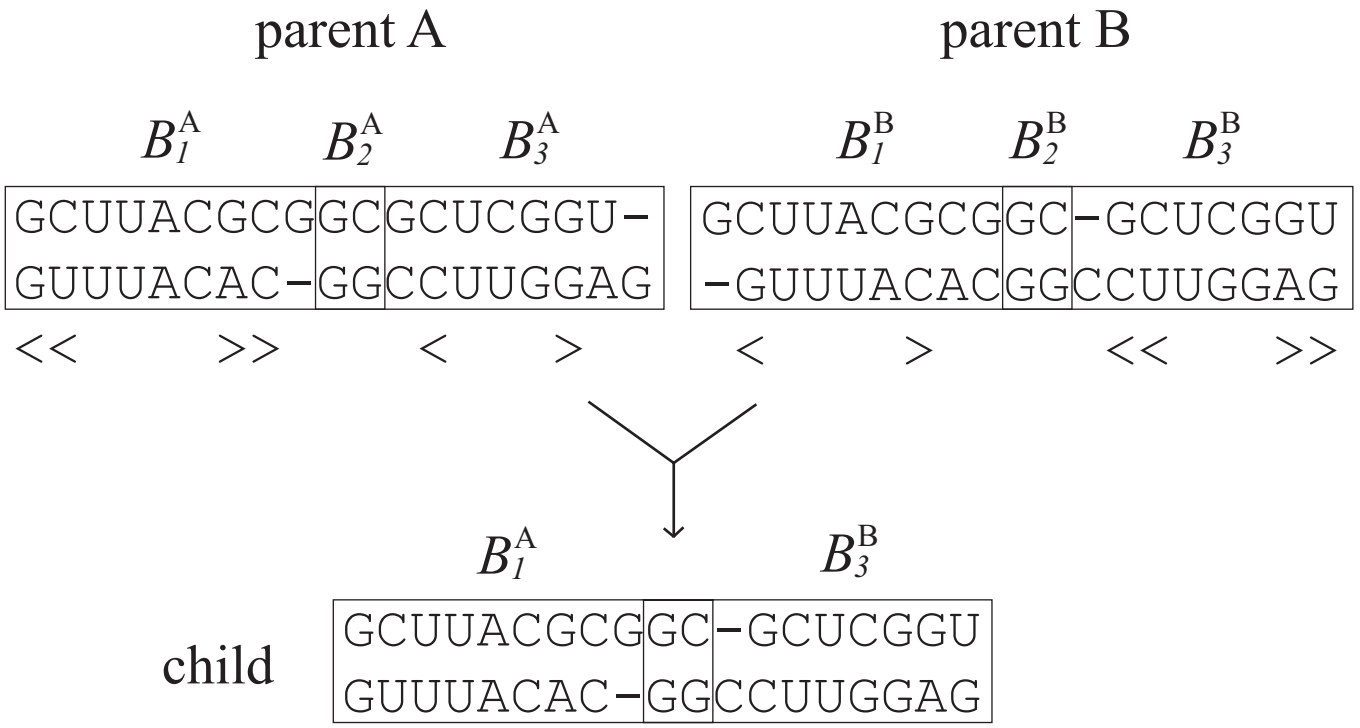
parent A
parent B
B_1^A B_2^A B_3^A
B_1^B B_2^B B_3^B
GCUUACGCGGCGCUCGGU-
GUUUACAC-GGCCUUGGAG
GCUUACGCGGC-GCUCGGU
-GUUUACACGGCCUUGGAG
<< >> < > < > << >>
B_1^A B_3^B
child
GCUUACGCGGC-GCUCGGU
GUUUACAC-GGCCUUGGAG

Fig. 2.

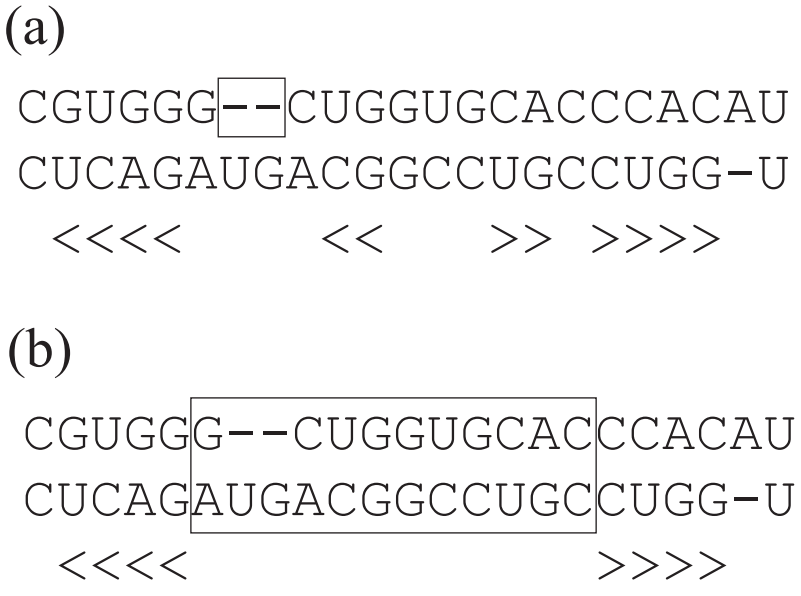
(a)
CGUGGG--CUGGUGCACCCACAU
CUCAGAUGACGGCCUGCCUGG-U
<<<< << >> >>>>
(b)
CGUGGG--CUGGUGCACCCACAU
CUCAGAUGACGGCCUGCCUGG-U
<<<< >>>>

Fig. 3.

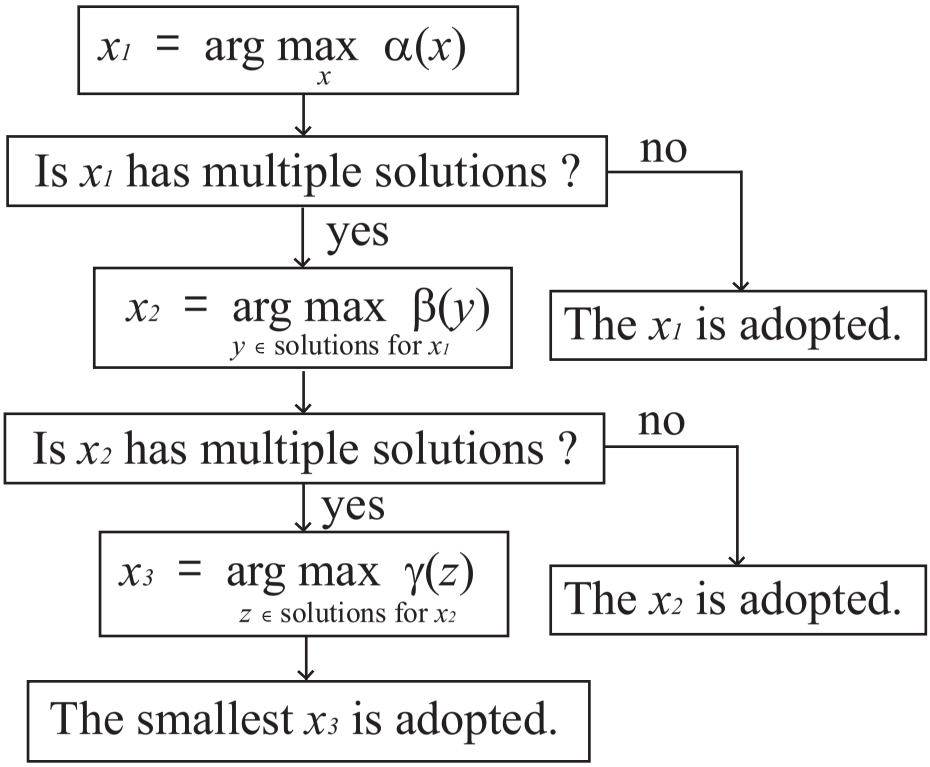
$x_1 = \arg\max_x \alpha(x)$
Is x_1 has multiple solutions ?
no
yes
The x_1 is adopted.
$x_2 = \arg\max_{y \in \text{solutions for } x_1} \beta(y)$
Is x_2 has multiple solutions ?
no
yes
The x_2 is adopted.
$x_3 = \arg\max_{z \in \text{solutions for } x_2} \gamma(z)$
The smallest x_3 is adopted.

Fig. 4.

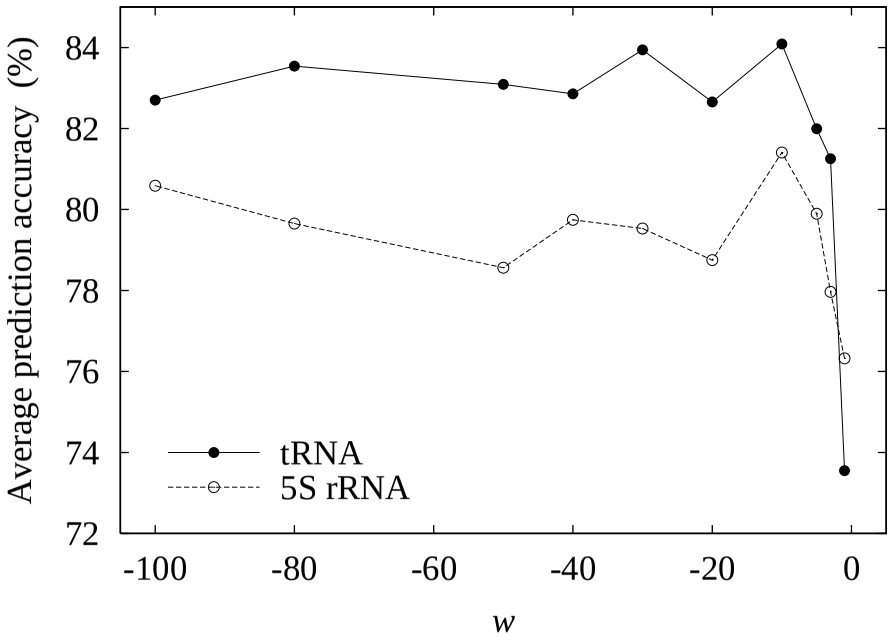
Average prediction accuracy (%)
84
82
80
78
76
74
72
-100
-80
-60
-40
-20
0
w
tRNA
5S rRNA

Fig. 5.

```
tRNA (without constrained folding)

5GCCCCCAUCG UCUAGAGGCC UAGGACACCU CCCUUUCACG GAGGCGA---  RE2140
5CCCCAAGUGG CGGAAUAGGU AGACGCAUUG GACUUAAAAU CCAACGGGCU  RL1141
  ***   * *    *   *        **      ***  *       **      cons
 <<<<<<<  < <<           >>> <<< <<       > >>>>         pred
 (((((((  ( (((          ))))){((( ((       ) )))))       ref-0
 (((((((  ( (({          }))){((( ((       ) )))))}((({   ref-1

---------- CAGGGAUUCG AAUUCCCUUG GGGGUACCA3  RE2140
UAAUAUCCUG UGCCGGUUCA AGUCCGGCCU UGGGGACCA3  RL1141
              * ***  * * *       *** ****   cons
            <<<<<        >>>>>>> >>>>>      pred
            (((((        ))))))) )))))      ref-0
    }}}}    (((((        ))))))) )))))      ref-1
```

Fig. 6.