SIMARD: A Simulated Annealing Based RNA Design Algorithm with Quality Pre-Selection Strategies

Sinem Sav*, David J. D. Hampson†, and Herbert H. Tsang†
*Faculty of Engineering & Computer Science, Bilkent University, Ankara, Turkey
†Applied Research Lab, Trinity Western University, Langley, British Columbia, Canada
Email: herbert.tsang@twu.ca†

***Abstract*—Most of the biological processes including expression levels of genes and translation of DNA to produce proteins within cells depend on RNA sequences, and the structure of the RNA plays vital role for its function. RNA design problem refers to the design of an RNA sequence that folds into given secondary structure. However, vast number of possible nucleotide combinations make this an NP-Hard problem. To solve the RNA design problem, a number of researchers have tried to implement algorithms using local stochastic search, context-free grammars, global sampling or evolutionary programming approaches. In this paper, we examine SIMARD, an RNA design algorithm that implements simulated annealing techniques. We also propose QPS, a mutation operator for SIMARD that pre-selects high quality sequences. Furthermore, we present experiment results of SIMARD compared to eight other RNA design algorithms using the Rfam datset. The experiment results indicate that SIMARD shows promising results in terms of Hamming distance between designed sequence and the target structure, and outperforms ERD in terms of free energy.**

I. Introduction

The RNA sequence design problem is an emerging research topic motivated by applications such as customized drug design. The goal of RNA design is to determine RNA primary structure given its secondary structure. Although it is possible to find the RNA primary structure from the secondary structure using brute force methods, our computational technology does not allow us to return a feasible answer within a reasonable amount of time. This is due to the size of the solution space which refers to the combinations of nucleotides. Researchers in the field are currently using other methods, such as heuristic methods, in order to reduce the time complexity.

The most important applications of RNA design would be in molecular biology, biotechnology and medicine fields. As the function and the secondary structure of RNA sequences are tightly coupled, knowing the secondary structure of the RNA is important to decide on its function. Thus, solving the RNA design problem would allow researchers to assemble RNA primary sequences, according to their need for secondary structures or specific functions.

In this paper, we will introduce a new approach for RNA design problem that was based on the Simulated Annealing (SA) framework. Part of the algorithm was built upon an evolutionary algorithm ERD [1]. Our research showed that using SA as a meta-heuristic approach to find the global optimum on top of evolutionary algorithm has an outstanding performance in terms of Hamming distance between folded structure of designed sequence and target structure, and our preselection strategy can help to improve energy.

In the subsequent sections, we consider major existing approaches on RNA design problem and then demonstrate our SA based algorithm, with and without QPS.

II. RNA Design

Currently, the major approaches to the RNA design research are: local stochastic search, global sampling algorithms, and evolutionary algorithms.

A. Local Stochastic Search for RNA Design Problem

A viable approach to solve the RNA Design problem can be found by employing stochastic methods. Most of the time, however, the target sequence is too long which forces researchers to use local stochastic search to speed up the process. RNA-SSD [2] and INFO-RNA [3] are two major programs using local stochastic search for the RNA design problem. Both algorithms use the principle of minimizing the structural distance between the target structure and the minimum free energy (MFE) structure of the designed sequence.

In RNA-SSD, the primary sequence is first initialized using three principles (see [2] for details). However, it is not guaranteed that the initialized sequence can directly fold into the target structure. Therefore, RNA-SSD decomposes the target structure and initializes the sequence hierarchically and constructs a tree where the root is the whole structure and the leaves are the sub-sequences. Finally, it performs a recursive local search. Consequently, decomposition plays an important role in this algorithm as the complexity of the prediction algorithm is $O(n^3)$.

INFO-RNA has a different initialization approach from RNA-SSD. In the initialization part of INFO-RNA, dynamic programming is used to make sure that the initial sequence is the one with lowest possible energy while folding into the target structure [3]. Again, local stochastic search is used to improve initial sequence in this algorithm.

Using local stochastic search for the RNA design problem is feasible, however it has some drawbacks. With local stochastic search, an exact solution may not be found, even if one

exists. Another drawback is most of the local stochastic search algorithms stagnates after a while and the ones which does not stagnate are too slow. Lastly, local stochastic search algorithms promise to find a local optimum but not necessarily the global optimum if simulated annealing is not used.

B. Global Sampling Algorithms

In general, local search algorithms algorithms use an initial sequence and mutate it to find final sequence which folds into target structure. In these kind of algorithms, the initial sequence is very important to find a stable and appropriate solution. With local search approaches, it may be difficult to find the best or good-enough solution with chosen initial sequence. However, in 2012, Alex Levin et al. introduced a new approach and tried to overcome disadvantages of local search algorithms [4]. Their package is called RNA-ensign and it uses global sampling methods [4].

RNA-ensign is an ensemble-based approach which starts with a random sequence and computes the probability of folding into target structure of all k-mutants of this random sequence (Boltzmann distribution is used to find this) and samples from this group of sequences [4]. Finally, starting from first mutant sequence, it tries to find the sequence whose minimum free energy (MFE) structure is the target structure. In the sampling step, they use RNAmutants [5] algorithm which enables them to sample in polynomial time and space [4]. If the algorithm can not find a mutant-sequence which folds into target structure, it reports as a failure [4]. Results show that RNA-ensign has more stable solutions compared to local search algorithms. However, it has $O(n^5)$ complexity which implies that it has significant drawback about time.

In 2013, RNA-ensign is improved and a new algorithm called IncaRNAtion [6] is implemented to decrease complexity of RNA-ensign. Although RNA-ensign has $O(n^5)$ complexity, IncaRNAtion runs in linear complexity. In this algorithm, they use modified version of RNAmutants [5] to get linear complexity in sampling process [6].

C. Evolutionary Algorithms

In general, evolutionary algorithms use meta-heuristic and optimization techniques to find an optimal solution. Evolutionary algorithms takes its name from Darwinian principles. Their behaviour is similar to a natural processes: natural selection. Evolutionary algorithms usually consists of four main parts: reproduction, mutation, variation(or recombination), and selection [7].

The genetic algorithms used to solve RNA design problem are MODENA [8], Frnakestein [9], GGI-FOLD [10] and ERD [1]. All of these algorithms apply four main steps of evolutionary algorithms but the way they implemented these steps is different. In general, they initiate the population to select the sequence, they mutate the selected one and after finding the fitness of the final solution, they terminate.

Although they look similar to each other, using different techniques, generally in the calculation of fitness of solutions, distinguishes one from the other. Both MODENA and GGI-FOLD use multiple objectives for determining the fitness of the solutions but the number and characteristics of these objectives differ [8] [10]. Frnakenstein is another algorithm differing from others mainly by fitness calculation method: It uses Boltzmann probabilities and find positional fitness [9]. As it is written in Python and using Boltzmann probabilities is computationally costly, Frnakenstein suffers from high run-time compared to others [9].

Last but not least, ERD is the only evolutionary algorithm and it uses mutation instead of recombination [1]. One advantage of ERD is that one can specify both energy and structural constraints. Another advantage of ERD is the initial decomposition scheme used before evolutionary part of the algorithm starts. The structure given is decomposed using multiloop occurrences and compatible sub-sequences are gathered from STRAND database to form the initial sequence for evolutionary algorithm while preserving the energy and structural constraints. This makes the initial sequence closer to the ones in the nature as database of natural sequences (STRAND) is used in this step. Lastly, ERD replaces the sub-structures in mutation steps; thus, the mutation is not on the nucleotide level but on the sub-structure level to maintain the consistency between structures [1].

The objectives of this paper are as follows:

- To examine an algorithm for RNA secondary structure design based on simulated annealing techniques (SIMARD).
- To measure the designability of our RNA structure design algorithm by design by comparing the structure from Rfam and The RNA secondary STRucture and statistical ANalysis Database (STRAND).
- To compare the result of SIMARD to eight other RNA design algorithms.
- To observe the impact that QPS has on the quality of SIMARD results.

III. Method

A. Simulated Annealing in RNA Structure Prediction

Simulated Annealing (SA) was first applied to problems involving optimization by Kirkpatrick et al [11]. Instead of finding the local optimum with traditional deterministic or local approaches, SA is a robust probabilistic approach to find global optimum in the presence of large solution space.

SA utilize the iterative search optimization approach, based on successive update steps (either random or deterministic) where each update step is proportional to an arbitrarily set parameter which can play the role of a temperature. This is where the analogy with the annealing process of metals play a role.

In contrast to other evolutionary algorithms, SA is superior because Geman and Geman has proved a necessary and sufficient condition for the convergence of the algorithm to the global minimum [12]. This is the big advantage for SA over other evolutionary algorithms.

In the past, our lab has done extensive research in using SA for RNA secondary structure prediction. Our algorithm,

SARNA-Predict, has shown superior performance (in terms of prediction accuracy when compare to native structure) over other state-of-the-art algorithms in predicting structures both with and without pseudoknots [13] [14] [15]. As a result, we are now proposing to use the SA paradigm for RNA structure design.

B. SIMARD

SIMARD (Simulated Annealing for RNA Design) is a heuristic algorithm for the RNA inverse-folding problem and it was first proposed by Erhan *et al.* [16]. Algorithm 1 shows the pseudo-code for SIMARD under the simulated annealing framework [17].

Algorithm 1 Structure of the simulated annealing algorithm in RNA secondary structure design

```
Sequence = InitialSequence;
Temperature = InitialTemperature;
Distance = HammingDistance(Sequence, Structure);
while (Temperature > FinalTemperature) do
    for (i = 1 to NumberOfIterations) do
        NewSequence = Mutate(Sequence);
        NewDistance = HammingDistance(NewSequence, Structure);
        Δ Distance = NewDistance - Distance;
        if (Δ Distance ≤ 0) OR (with Probability[Accept] = e^(−ΔDistance / Temperature)) then
            Distance = NewDistance;
            Sequence = NewSequence;
        end if
    end for
    decrease Temperature;
end while
FreeEnergy = FoldAndEvaluate(Sequence);
```

The basic structure of a SA search optimization algorithm consists of several main parts:

A. State Representation
B. Perturbation / Mutation Function
C. Evaluation / Cost Function
D. Scheduler

In the following sub-sections, we will describe these main parts in more detail.

1) RNA Secondary Structure Representation: We use ERD to decompose an RNA secondary structure to create several pools of RNA secondary sub-structures [1].

Pool construction is done using STRAND sequences such that the resulting pools contain sub-sequences that are similar to the natural ones [1]. To construct the pools of sub-sequences, *fold* method from Vienna package [18] is employed and resulting structures are then decomposed into sub-structures.

The target structure is also decomposed into structural components so that we can randomly choose a sub-sequence with a compatible type and length from our database of natural sub-structures. In decomposition scheme, the structural components are stored three arrays: (*starray* for stems, *hparray* for hairpin loops, and *mlarray* for multiloops) [1]. As in other algorithms, the initialized sequence is not promised to fold into target structure. The overall goal of SIMARD is to improve the initial sequence.

Once we have the pools of RNA sequences, we can view the problem of designing RNA structure as one of picking the sub-set S of sub-sequences from the set of all possible sub-sequences H, such that the Hamming distance $D(S)$ between the structure of the designed sequence and the goal structure is minimized.

2) Perturbation / Mutation Function: The main goal of the perturbation function is to modify the design in a controlled and intuitive fashion. We tested two methods of mutation: without pre-selection and with pre-selection.

A. Without pre-selection

In each iteration, a random selection will be made from the pool of stems, hairpins and multiloops that make up the working solution. This selection is replaced with a sub-structure of the same type and length, taken from the corresponding pool of natural sub-structures. The resulting sequence is returned and evaluated.

B. With pre-selection (QPS)

We developed QPS, or *Quality Pre-selection Strategy* to use as a mutation operator that generates multiple RNA primary sequences, each evolved from the same parent using the same technique as above. This allows us to choose and return a solution based on our optimization criteria. Algorithm 2 shows the Pseudocode for QPS. For these experiments, a solution's quality was judged in regards to its free energy (lowest being of the highest quality).

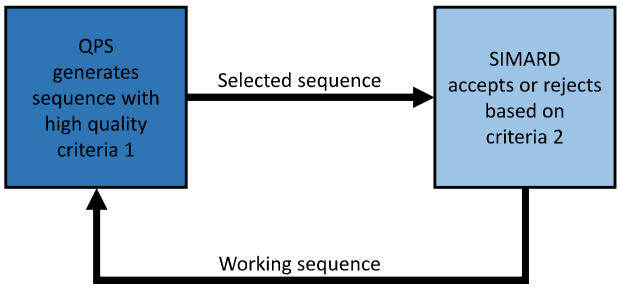


Fig. 1. The relationship between QPS and SIMARD.

The number of sequences to generate and compare at each step was on decision we had to make. To decide on this number, we ran trials in which we generated two, three, four, and five solutions in pre-selection on the sequence AF107506, a partial sequence of a ribosomal RNA gene from bacterium SY2-21 16S. The two factors we accounted for when making our decision were coverage of the Pareto front and time cost.

Figure 2 shows a map of solutions generated by test runs with pre-selection pool sizes of two, three, four, and five. This is useful because we can see the general solution space that

each variation inhabits. Based on these results, we decided on a pre-selection pool size of three, as four and five had too high of a time cost and two did not show enough of an improvement to energy.

Algorithm 2 High quality mutation selection with QPS

```
1: function QPS(Sequence)
2:     BestSequence = Sequence;
3:     initialize BestQuality;
4:     for (i = 1 to SequencesToGenerate) do
5:         NewSequence = Modify(Sequence);
6:         NewQuality = Evaluate(NewSequence);
7:         if (NewQuality > BestQuality) then
8:             BestSequence = NewSequence;
9:             BestQuality = NewQuality;
10:        end if
11:        Reset Sequence;
12:    end for
13:    return BestSequence
14: end function
```

3) ost Function: The objective of the cost function is to evaluate the appropriateness of the current design or state. In our implementation, we minimize the Hamming distance between our working solution and the target structure. Hamming distance is defined as the number of character differences (number of mismatches) between two strings (A and B) of equal length (N).

$$Hamming\_Distance = \sum_{i=o}^{N-1} |A_i - B_i| \qquad (1)$$

Since the designed sequence is a primary structure and the target is a secondary structure, we fold it into its lowest free energy state using an RNA prediction method (Vienna package [18] is currently used for the prediction). The Hamming distance is then calculated between our working structure and the target structure using dot-bracket format. We use this Hamming distance determine wether to accept or reject the sequence with a specific probability function (see line 9 of Algorithm 1).

4) Cooling and Annealing Schedule: A discrete-time implementation of SA can be realized by generating Markov chains of finite length for a finite sequence of descending values of temperature. The parameters that control these descending values of temperature are the cooling or annealing schedules [19].

In general, the classes of annealing schedules that are most effective have the following characteristics [19] [20]:

- A high starting acceptance probability.
- A very low terminating acceptance probability.
- A slow cooling rate, $\alpha \in [0, 1[$ i.e. where α between 0.8 and 0.99 is a recommended value, $T_{new} = \alpha T_{old}$.
- The number of iterations is equal to the number of neighboring solutions. Neighboring solutions are defined as two adjacent states that can be reached by a single move (i.e. $(s, s') \in M$).

In SIMARD, the decreasing temperature is performed using geometric scheduler. Thus temperature is multiplied by the geometric scheduler factor (α) in each iteration ($T_{new} = \alpha T_{old}$).

In the current implementation, we have chosen 0.90 to be our geometric scheduler factor. Also, final temperature has 0.001 as a default value. Lastly, the initial temperature is set automatically with a SA warm up procedure developed by Aarts *et al.* [21].

In the following sections, we will describe the results we have obtained using SIMARD, both with and without QPS.

IV. Data

To benchmark SIMARD, we employed the Rfam [22] [8] set of structures. As in Lyngs et al. [9], RF00023 was excluded from the dataset for the ease of comparison with the employed table. It is because RF00023 has lots of pseudo-knotted base pairs [9]. Thus, Rfam dataset includes 29 structures from 29 Rfam families with lengths varying between $54 - 451$nt inclusive. Two additional columns were added to Table I to indicate results for SIMARD and ERD [1].

V. Results and Discussion

Experiments showed that SIMARD outperforms the existing RNA Design algorithms in terms of success count where Hamming distance is used to indicate success rate. According to Rfam dataset experiments, for a target structure with length $80 - 300$ nt, an initialized sequence in ERD or SIMARD (without pre-selection) has approximately $40 - 100$ Hamming distance to target structure when folded, and this number is reduced to $0 - 3$ Hamming distance with SA optimization. Since SA performs well in large solution spaces, the change is more dramatic when the length increases. In the experiments, we found out that solution space affects the quality of the designed sequence. The larger pools we have for subsequences, it is more likely that initial sequence is improved to a sequence having zero-distance to the target structure.

Figure 3 illustrates an example run of RF00030 from the Rfam dataset using SIMARD with QPS, and without QPS. Figure 3(a) presents the graph which indicates the domain of distances that were accepted during two runs of SIMARD: with and without QPS. Notice that during the initial stage where the temperature values were high, the probability to accept a configuration that produced a higher distance was higher. As the temperature decreased, less configurations with higher distance were accepted. Hence, the convergence behavior of the algorithm can be observed.

We can see the change in energy over time in Figure 3(b). Without QPS, energy is not considered within SA moves, so we do not observe decrease in the energy over SA iterations. On the contrary, it slightly increases at last stages of SA algorithm where distance dramatically decreases. The benefit of QPS can be seen here: the algorithm finds a better solution space and the energy is lower overall.

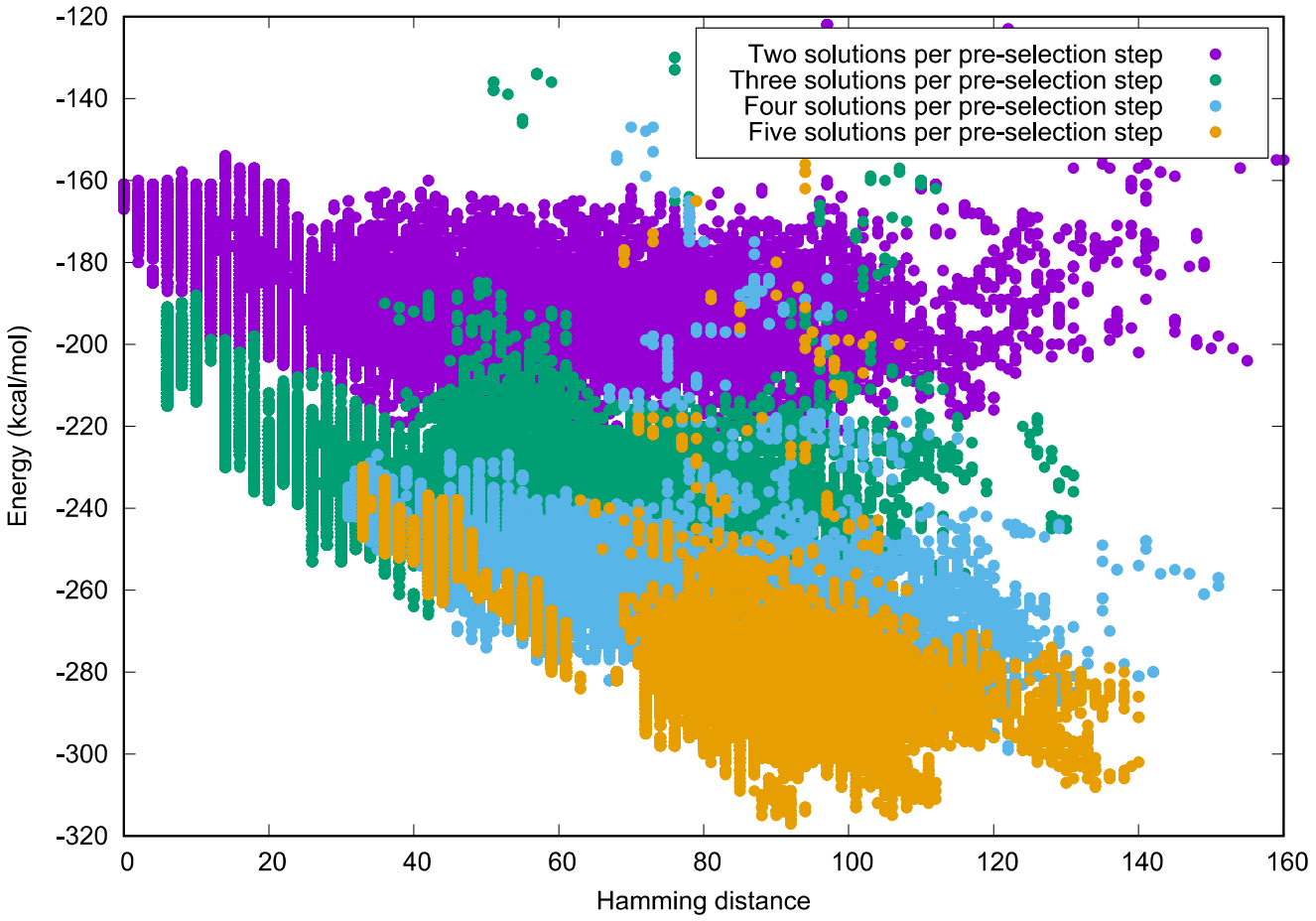


Fig. 2. The horizontal axis describe the difference between the generated solutions and the target structure in terms of Hamming distance. Vertical axis describe the free energy of these structures.

Both with and without QPS, While distance is decreasing dramatically, energy is going up to some extent. Similarly, when minimizing energy is used as a main goal in SA method, the distance is going up as expected. This shows that there is a negative correlation between Hamming distance and energy in terms of solutions found by the algorithm. This could be improved by trying a different method of generating new sequences.

A. *Rfam Dataset Results*

Table I shows the result of SIMARD against eight other algorithms (Frnakenstein, MODENA, RNA-SSD, INFO-RNA, RNAInverse, NUPACK, Inv, and ERD) using the Rfam dataset. The column labeled *Structure* indicates the accession number of the target structures in the given dataset. *Len.* indicates length of the target structure. Each entry represents the success fraction over the conducted experiments. If a given algorithm designed and returned a compatible sequence, it is counted as success. For example, if we do ten experiments and out of nine times the algorithm returned a result, then table entry is $9/10 = 0.9$. This is the general practice in the literature to report the results. The fields having asterisk under Inv indicates that the target structure is found as invalid in this algorithm [9].

For each target structure, 10 experiments were performed for Frnakenstein, and *NUPACK*; and 500 experiments were performed for the other five algorithms (*MODENA*, *RNA-SSD*, *INFO-RNA*, RNAInverse, and Inv). These results were reported by Lyngso et al [9]. Experiments for SIMARD and ERD also run 10 times for each target structure. If an algorithm does not give any result after a set time limit (e.g. several days), it is counted as unsuccessful [9].

To be consistent with the literature (employed Frnakenstein table [9]) we also counted total successes depending on the numbers that are bigger than zero in each column. To be clear, if an algorithm designed a sequence at least once within conducted experiments, it is counted as success. ERD and SIMARD outperforms other algorithms by successfully designing a compatible sequence for all 29 tested Rfam structures. MODENA and Frnakenstein then follow by designing 23 compatible sequences out of 29 (79.3%).

In terms of energy, ERD gives better (minimum) energies than SIMARD without QPS about 60% of the time (see Table II). With this in mind, we ran the sequences with

TABLE I

RFAM DATASET RESULTS. FIRST COLUMN INDICATES THE ACCESSION NUMBER OF THE TARGET STRUCTURES IN THE GIVEN DATASET. SECOND COLUMN INDICATES THE LENGTH OF THE TARGET STRUCTURE. EACH ENTRY REPRESENTS THE SUCCESS FRACTION OVER CONDUCTED EXPERIMENTS. SEQUENCES ARE ORDERED BY LENGTH

Structure	Len.	Frnakenstein	MODENA	RNA-SSD	INFO-RNA	RNAinverse	NUPACK	Inv	SIMARD	ERD
RF0008	54	1.00	1.00	0.96	1.00	0.95	1.00	0.22	1.00	1.00
RF00029	73	1.00	1.00	0.82	0.67	0.20	0.72	⋆	1.00	1.00
RF0005	74	1.00	1.00	1.00	0.99	0.87	1.00	0.29	1.00	1.00
RF00027	79	1.00	1.00	1.00	1.00	0.82	1.00	0.86	1.00	1.00
RF00019	83	1.00	1.00	0.40	0.98	0.57	1.00	⋆	1.00	1.00
RF00014	87	1.00	1.00	0.94	1.00	1.00	1.00	⋆	1.00	1.00
RF0006	89	1.00	1.00	0.98	0.66	0.06	0.99	⋆	1.00	1.00
RF00026	102	1.00	1.00	0.00	0.04	0.02	1.00	⋆	1.00	1.00
RF0001	117	1.00	1.00	0.01	0.95	0.01	0.29	⋆	1.00	1.00
RF00021	118	1.00	1.00	0.99	1.00	0.96	1.00	⋆	1.00	1.00
RF00020	119	0.00	0.00	0.00	0.00	0.00	0.00	⋆	1.00	1.00
RF00016	129	0.00	0.00	0.00	0.00	0.00	0.48	⋆	1.00	1.00
RF00015	140	1.00	1.00	0.00	0.51	0.05	1.00	⋆	1.00	1.00
RF00022	148	1.00	1.00	0.00	0.15	0.02	1.00	⋆	1.00	1.00
RF0002	151	1.00	1.00	0.00	0.00	0.00	0.00	⋆	1.00	1.00
RF0007	154	1.00	1.00	0.05	0.85	0.08	0.95	⋆	1.00	1.00
RF0003	161	0.10	1.00	0.00	0.01	0.00	0.00	⋆	1.00	1.00
RF00013	185	1.00	1.00	0.00	0.37	0.09	1.00	⋆	1.00	1.00
RF0004	193	1.00	1.00	0.00	0.27	0.10	1.00	⋆	1.00	1.00
RF00025	210	1.00	1.00	0.00	0.06	0.00	1.00	⋆	1.00	1.00
RF00012	215	1.00	1.00	0.00	0.01	0.01	0.98	⋆	1.00	1.00
RF00017	301	1.00	1.00	0.00	0.94	0.23	1.00	⋆	1.00	1.00
RF00030	340	1.00	1.00	0.00	0.01	0.00	0.00	⋆	1.00	1.00
RF00028	344	0.30	0.00	0.00	0.01	0.00	0.09	⋆	1.00	1.00
RF0009	348	1.00	1.00	0.00	0.00	0.01	0.78	⋆	1.00	1.00
RF00010	357	0.00	0.00	0.00	0.00	0.00	0.00	⋆	1.00	1.00
RF00018	360	1.00	1.00	0.00	0.01	0.01	0.00	⋆	1.00	1.00
RF00011	382	0.00	0.00	0.00	0.00	0.00	0.00	⋆	1.00	1.00
RF00024	451	0.00	0.00	0.00	0.00	0.00	0.23	⋆	1.00	1.00
Total successes		24	23	10	22	19	22	3	29	29

SIMARD using QPS as a mutation operator. SIMARD with QPS outperformed ERD on 29 out of 29 sequences in terms of final result free energy, as can be see in Table II. To make up for the runtime difference, the shown energies are the best solution found out of 10 runs for ERD, the best solution found out of 2 runs for SIMARD with QPS, and the best solution found out of 1 run for SIMARD without QPS. SIMARD without QPS outperformed SIMARD with QPS on five sequences. This could be because the sequences are shorter and the design space is more limited, meaning that the additional constraints that QPS involves forced SIMARD into a poor design space. It is also possible that, as luck can play a big factor in SA, and we have a fairly small sample size, the initial sequence for the non QPS run was of very high quality.

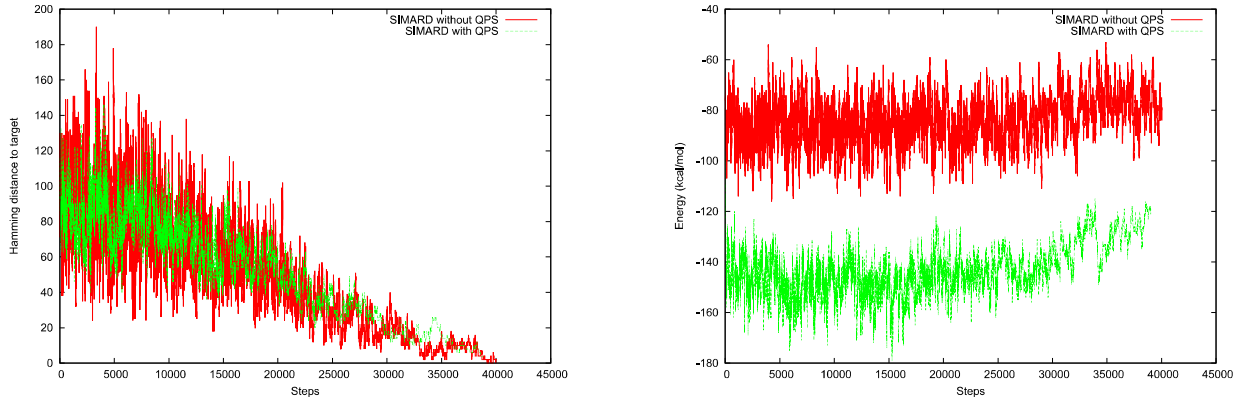
B. Length Dependency

Experiments showed that when we compare length dependency of success rate over mentioned algorithms, the general scheme is that success count of the algorithms reduce and distance of the designed sequence to the target structure increase when length of the structure increases. However, one advantage of our algorithm is, as it employs Simulated Annealing, increase in length does not affect the quality of the designed sequence. Contrarily, when length increases, as our solution space (pools of sub-sequences) also increases SA performs better compared to short sequences. For the short sequences, the SA loop stagnates over same solutions as our solution space is not big enough.

To our knowledge, regarding length dependency, ERD outperforms the existing algorithms so far. However, when length becomes more than 900nt, SIMARD starts to give better distances. When a typical sequence with length 953nt from RNASTRAND dataset (393th sequence) is used in experiments, ERD gives 14 as an average Hamming distance whereas SIMARD gives 6. This specific example is chosen because the structures with length over 953nt in RNASTRAND dataset could not be successfully designed which makes this sequence the longest sequence in our dataset.

VI. CONCLUSION

In this paper, we introduced SIMARD with QPS for RNA design problem which uses simulated annealing as an optimization technique to minimize Hamming distance, and preselection to optimize energy. Using Rfam dataset, we have showed that our algorithm yields better results for the long RNA structures in terms of Hamming distance compared to eight other RNA design algorithms. It is found that both ERD and SIMARD has superior results over the existing packages. In addition, we have shown that with QPS, we can outperform ERD in terms of energy. One drawback of SIMARD is the long computational times compared to ERD. However, we have found that with the long sequences, SIMARD performs better in terms of resulting Hamming distance and energy



(a) (b)

Fig. 3. SIMARD run with and without QPS of Rfam dataset sequence, RF00030. The horizontal access represents algorithmic steps and the vertical axis represents (a) Hamming distance to target structure (b) Free energy

TABLE II
COMPARING FREE ENERGY OF FINAL SOLUTIONS BETWEEN ALGORITHMS. THE BEST ENERGY FOR EACH SEQUENCE IS BOLDED. THE SEQUENCES ARE ORDERED BY LENGTH.

Sequence name	Length	Energy (kcal/mol) SIMARD with QPS	SIMARD no QPS	ERD
RF0008	54	-27	**-53**	-15
RF00029	73	**-31**	-16	-21
RF0005	74	**-31**	-26	-23
RF00027	79	-50	**-51**	-48
RF00019	83	**-66**	-29	-24
RF00014	87	**-61**	-31	-38
RF0006	89	-35	**-124**	-18
RF00026	102	**-28**	-2	-4
RF0001	117	**-220**	-16	-34
RF00021	118	**-75**	-40	-47
RF00020	119	**-58**	-29	-35
RF00016	129	**-65**	-62	-22
RF00015	140	**-109**	-36	-33
RF00022	148	**-128**	-36	-38
RF0002	151	-47	**-63**	-21
RF0007	154	**-77**	-47	-53
RF0003	161	**-80**	-50	-44
RF00013	185	**-110**	-28	-58
RF0004	193	**-94**	-5	-55
RF00025	210	**-86**	-46	-40
RF00012	215	-84	**-135**	-48
RF00017	301	**-217**	-130	-126
RF00030	340	**-106**	-90	-69
RF00028	344	**-115**	-58	-61
RF0009	348	**-105**	-36	-60
RF00010	357	**-206**	-28	-126
RF00018	360	**-122**	-68	-64
RF00011	382	**-188**	-52	-110
RF00024	451	**-226**	-109	-124

(with QPS) than the ERD and the other algorithms, hence SIMARD provides more accurate results than ERD in general.

Our algorithm employs *fold* function from Vienna Package [18] and pool construction scheme in ERD. In the future, trying another function for secondary structure prediction and another scheme for the evolutionary part of the algorithm could increase the accuracy of the designed sequence.

Lastly, there are many future directions for SIMARD. SIMARD is very flexible and it allows researchers to design their own SA approach for RNA design problem using evolutionary techniques. In the future, the algorithm can be easily extended to use constraints (to specify nucleotides to specific positions or GC content). QPS also adds an element of multiple optimization, and there are a number of combinations of SIMARD optimization and preselection optimization that we can try.

ACKNOWLEDGMENT

The authors would like to acknowledge support from Trinity Western University and Simon Fraser University. In addition, the authors would like to acknowledge the support from Mitacs Globalink project and a grant from the M.J. Murdock Charitable Trust Fund is also gratefully acknowledged. Special thanks to Emre Erhan for his help in the early part of this project.

REFERENCES

[1] A. Esmaili-Taheri, M. Ganjtabesh, and M. Mohammad-Noori, "Evolutionary Solution for the RNA Design Problem," *Bioinformatics*, vol. 30, no. 9, pp. 1250–1258, 2014.

[2] M. Andronescu, A. P. Fejes, F. Hutter, H. H. Hoos, and A. Condon, "A new algorithm for rna secondary structure design." *Journal of Molecular Biology*, vol. 336, no. 3, pp. 607–624, February 2004.

[3] A. Busch and R. Backofen, "INFO-RNAa server for fast inverse RNA folding satisfying sequence constraints," *Nucleic Acids Research*, vol. 35, pp. 310–313, 2007.

[4] A. Levin, M. Lis, Y. Ponty, C. W. O'Donnel, S. Devadas, B. Berger, and J. Waldispuhl, "A global sampling approach to designing and reengineering rna secondary structures," *Nucleic Acids Research*, vol. 40, 2012.

[5] J. Waldispuhl, S. Devadas, B. Berger, and P. Clote, "Efficient Algorithms for Probing the RNA Mutation Landscape," *PLoS Computational Biology*, vol. 4, 2008.

[6] V. Reinharz, Y. Ponty, and J. Waldispuhl, "A weighted sampling algorithm for the design of RNA sequences with targeted secondary structure and nucleotide distribution," *ICMB/ECCB*, vol. 29, pp. i308–i315, 2013.

[7] T. Baeck, D. Fogel, and Z. Michalewicz, *Handbook of Evolutionary Computation*. Taylor & Francis, 1997. [Online]. Available: https://books.google.ca/books?id=n5nuiIZvmpAC

[8] A. Taneda, "Multi-objective genetic algorithm for pseudoknotted RNA sequence design," *Frontiers in Genetics*, vol. 3, pp. 1–9, 2012.

[9] R. B. Lyngs, J. W. Anderson, E. Sizikova, A. Badugu, T. Hyland, and J. Hein, "Frnakenstein: multiple target inverse RNA folding," *Frontiers in Genetics*, vol. 3, pp. 1–12, 2012. [Online]. Available: http://www.biomedcentral.com/1471-2105/13/260

[10] A. N.-D. M. Ganjtabesh, F. Zare-Mirakabad, "Inverse RNA Folding Solution Based on Multi-Objective Genetic Algorithm and Gibbs Sampling Method," *EXCLI*, vol. 2013, pp. 546–555, 2013.

[11] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science, Number 4598, 13 May 1983*, vol. 220, 4598, pp. 671–680, 1983.

[12] S. Geman and D. Geman, "Stochastic Relaxation, Gibbs Distributions, andthe Bayesian Restoration of Images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 721–741, 1984.

[13] H. H. Tsang and K. C. Wiese, "SARNA-Predict: Accuracy improvement of RNA secondary structure prediction using permutation based simulated annealing," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 99, no. 1, pp. 727–740, 2010.

[14] P. Grypma, J. Babbitt, and H. H. Tsang, "A study on the effect of different thermodynamic models for predicting pseudoknotted RNA secondary structures," in *IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology*, 2013, pp. 52–59.

[15] P. Grypma and H. H. Tsang, "SARNA-Predict: Using adaptive annealing schedule and inversion mutation operator for RNA secondary structure prediction," in *IEEE Symposium Series on Computational Intelligence*, 2014, pp. 150–156.

[16] H. E. Erhan, S. Sav, S. Kalashnikov, and H. H. Tsang, "Examining the Annealing Schedules for RNA Design Algorithm," in *IEEE Congress on Evolutionary Computation*, 2016.

[17] H. H. Tsang and K. C. Wiese, "The significance of thermodynamic models in the accuracy improvement of RNA secondary structure prediction using permutation based simulated annealing," in *Proceedings of the IEEE Congress on Evolutionary Computation*, 2007, pp. 3879–3885.

[18] I. L. Hofacker, W. Fontana, F. S. Peter, L. S. Bonhoeffer, M. Tacker, and P. Schuster, "Fast Folding and Comparison of RNA Secondary Structures," *Monatshefte fur Chemie*, vol. 125, pp. 167–188, 1994. [Online]. Available: http://fontana.med.harvard.edu/www/Documents/WF/Papers/vienna.rna.pdf

[19] Y. Li, "Directed annealing search in constraint satisfaction and optimisation," Ph.D. dissertation, University of London, Imperial College of Science, Technology and Medicine, London, 1997.

[20] E. Aarts and J. Korst, *Simulated Annealing and Boltzmann Machines: a stochastic approach to combinatorial optimization and neural computing*. Chicester: John Wiley & Sons Ltd., 1989.

[21] E. H. L. Aarts, F. M. J. de Bont, J. H. A. Habers, and P. J. M. van Laarhoven, "Parallel implementations of the statistical cooling algorithm," *Integration, the VLSI Journal*, vol. 4, no. 3, pp. 209–238, 1986.

[22] S. Griffiths-Jones, A. Bateman, M. Marshall, A. Khanna, and S. R. Eddy, "Rfam: an RNA family database," *Nucleic Acids Research*, vol. 31, pp. 439–441, 2003.