The Impact of Pseudorandom Number Quality on *P-RnaPredict*, a Parallel Genetic Algorithm for RNA Secondary Structure Prediction

Track: Biological Applications

Kay C. Wiese
School of Computing Science
Simon Fraser University
Surrey, B.C., Canada
wiese@cs.sfu.ca

Andrew Hendriks, Alain Deschênes, and Belgacem Ben Youssef
InfoNet Media Center
Simon Fraser University
Surrey, B.C., Canada
ahendrik@sfu.ca, aadesche@sfu.ca, bbenyous@sfu.ca

ABSTRACT

This paper presents a parallel version of RnaPredict, a genetic algorithm (GA) for RNA secondary structure prediction. The research presented here builds on previous work and examines the impact of three different pseudorandom number generators (PRNGs) on the GA's performance. The three generators tested are the C standard library PRNG RAND, a parallelized multiplicative congruential generator (MCG), and a parallelized Mersenne Twister (MT). A fully parallel version of RnaPredict using the Message Passing Interface (MPI) was implemented. The PRNG comparison tests were performed with known structures that are 118, 122, 543, and 556 nucleotides in length. The effects of the PRNGs are investigated and the predicted structures are compared to known structures.

Categories and Subject Descriptors

J.3 [**Life and Medical Sciences**]: Biology and genetics; D.1.3 [**Programming Techniques**]: Parallel programming; G.3 [**Probability and Statistics**]: Random number generation

General Terms

Algorithms, Performance

Keywords

Bioinformatics, RNA Secondary Structure Prediction, Parallel Evolutionary Algorithms, Random Number Generators

1. INTRODUCTION

The shape of organic molecules such as RNA and proteins largely determines their function within an organic system. These biomolecules are formed from a sequence of nucleic or amino acids. The final three dimensional structure of a biomolecule forms when a sequence folds back onto itself. While physical methods such as Nuclear Magnetic Resonance and X-ray Crystallography may determine a structure, they are too time-consuming and expensive to be effective. Thus, a major goal in bioinformatics is to develop methods for computationally predicting the structure of a given biomolecule.

RNA is central in several stages of protein synthesis. Heterogeneous nuclear RNA (hnRNA) acts as the transcriber of DNA in eukaryotes. Messenger RNA (mRNA) carries the coded message to the ribosomes for synthesis. Ribosomal RNA is a component of ribosomes. Finally, transfer RNA (tRNA) combines the amino acids. Each of these types of RNA is synthesized by RNA polymerase [14].

The various algorithms that have been used for RNA structure prediction include dynamic programming (DP) [20], comparative methods [1], kinetic folding [31] and EAs [28, 30].

In the case of DP and EAs, structure prediction can be formulated as an energy minimization problem using thermodynamic models. While DP has been shown to accurately predict a structure with minimum energy within a given thermodynamic model [20], the natural fold has been shown to vary greatly from the predicted one [9]. However, van Batenburg et al [28] implemented a simple binary GA which outperformed the DP algorithm when considering true positive canonical base pairs in the structure. As the base pairs ultimately make up the secondary structure, this development is quite significant. Although other GA designs have been employed for RNA structure prediction, such as massively parallel GAs [26] there has been relatively little application of coarse-grained distributed GAs to this problem domain.

Coarse-grained distributed GAs [3] offer a number of advantages beyond the benefits of parallelization. These include the prevention of premature convergence by maintaining diversity, an increase of the selection pressure within the entire population, and also reduction of the time to convergence.

The foundation of this research is RnaPredict, a GA for RNA secondary structure prediction developed by Wiese and Glen [30] who have proposed to use permutations to encode RNA secondary structures. Later, Deschênes and Wiese [5] have shown that RnaPredict, using two stacking based thermodynamic models, can predict certain RNA secondary structures with very high accuracy and also outperform a dynamic programming algorithm [6]. Our specific work builds on [11, 10], in which a serial simulation of a distributed version of RnaPredict was implemented that predicted the secondary structure of RNA molecules from input RNA sequences. The extension that we present here is the redevelopment of the serial simulation of the distributed RnaPredict into a fully parallelized distributed GA based on the MPI standard to run on a 128 node Beowulf cluster. We shall call this fully parallel version of RnaPredict P-RnaPredict.

During development of P-RnaPredict, two major issues arose in regard to PRNGs. First, we discovered that as the RNA sequences increased in length, there was a proportional and dramatic increase in random number consumption. This dramatic consumption was further increased by our requirement to conduct 30 independent runs to gather averaged results. This can cause problems if the period of the PRNG is exceeded. The second was that the standard code library PRNG functions available in most development environments are self-contained and the user can only reseed them with a new initial seed. These challenges demanded we develop a new parallel PRNG, which in turn led us to review the impact of PRNGs on stochastic algorithms including GAs in general, and for parallel GAs in particular.

This paper has the following objectives:

- To investigate the effects of two new PRNGs, a parallelized MCG and a parallelized MT on our parallel GA
- To measure the accuracy of our predicted structures by comparing them to known structures

First, we describe the thermodynamic models in Section 2. Section 3 details the GA algorithm and design, while Section 4 offers an overview of the impact of PRNGs on GAs. Section 5 reviews our experiment design and presents the experimental results. Section 6 offers a discussion of the results, and finally Section 7 concludes the paper and offers a brief summary of future work.

2. THERMODYNAMIC MODELS

To date, four thermodynamic models have been implemented in RnaPredict. These include Individual Nearest Neighbour (INN) [25], Individual Nearest Neighbour - Hydrogen Bond (INN-HB) [32], and the models proposed by Major [17] and Matthews et al. [19].

Extensive testing with the serial GA [5] has shown conclusively that in terms of base-pairs found in known structures the INN and INN-HB stacking-energy thermodynamic models outperformed the simple Major and Matthews base-pairing models. Hence, for the experiments presented here, the thermodynamic model chosen was the INN-HB stacking-energy model.

2.1 Review of stacking-energy models

The essential idea of stacking-energy models is that the stabilizing contribution each base pair makes to its helix depends on that base pairs nearest neighbours. For example, the free energy contribution of a GC base pair would vary depending if the adjacent base pair in the helix is either an AU base pair, or its mirror a UA base pair.

2.1.1 Individual Nearest-Neighbour Hydrogen Bond Model (INN-HB)

INN-HB is largely based on INN. We will briefly review INN and INN-HB below. There are two distinct components to computing the free energy of a helix using INN. The first is initiation, or the formation of the first base pair. Initiation brings the two bases together and entails hydrogen bonding. The second component is propagation, or the continued formation of subsequent base pairs. Propagation involves nearest-neighbour or stacking interactions as well as hydrogen bonding. The nearest-neighbour thermodynamic parameters used in the INN model were initially computed at 25°C, but were later re-computed and extended at 37°C. Additional details and references on these parameters may be found in [5].

Later experimentation determined that duplexes with identical nearest neighbours but varying terminal ends also differed in their stabilities. Specifically, a duplex with one additional terminal GC pair and one less terminal AU pair is always more stable [32].

This difference is accounted for by a modification of the INN model, described via the following equation:

$$\begin{aligned}\Delta G^\circ(\text{duplex}) &= \Delta G^\circ_{\text{init}} + \sum_j n_j \Delta G^\circ_j(\text{NN}) + \\ &\quad m_{\text{term-AU}} \Delta G^\circ_{\text{term-AU}} + \Delta G^\circ_{\text{sym}} \qquad (1)\end{aligned}$$

This modification is known as INN-HB. Each $\Delta G^\circ_j(\text{NN})$ term accounts for the free energy contribution of the jth nearest neighbour with n_j occurrences in the sequence. The $m_{\text{term-AU}}$ term is the number of terminal AU pairs, and the $\Delta G^\circ_{\text{term-AU}}$ term is the free energy contribution of a terminal AU pair. The only difference from the INN model is the inclusion of the $m_{\text{term-AU}} \Delta G^\circ_{\text{term-AU}}$ term to account for the free energy penalty attributed to terminal AU pairs. Examples of this computation may be found in [32].

Although the INN-HB model only specifies a penalty for terminal AU pairs, we give terminal GU pairs the same penalty as suggested by Mathews et al in [19].

The INN-HB model as presented here is unable to account for higher-order structures such as loops, junctions, bulges and pseudoknots. A detailed review of the thermodynamic models employed in RnaPredict can be found in [5].

3. ALGORITHMS

In our model, RNA secondary structure forms as a consequence of chemical (hydrogen) bonds that form between specific pairs of nucleotides, i.e. GC, AU, and GU, and their mirrors which are collectively known as the canonical base pairs. Searching a sequence of nucleotides for all possible base pairs is rapid and straightforward; the challenge comes from attempting to predict which specific canonical base pairs will form bonds in the real structure. Different structural elements will manifest themselves in the resulting secondary structure depending on which base pairs form bonds. These include *hairpin loops* which contain one base pair, *internal loops* which contain two base pairs, and *bulges* which contain 2 base pairs with 1 base from each of its pairs adjacent in the backbone of the molecule. There are also *multi-branched* loops, which contain more than two base pairs, and *external bases* which are not contained in any loop. Stacked pairs, which form helices, provide stability in the secondary structure. A set of stacked pairs is formed by two or more base pairs, such that the ends of the pairs are adjacent, forming a ladder type structure.

In our model, a helix is specified by two constraints. First, each helix must have at least three stacked base pairs. Second, the sequence or loop connecting the two strands must be at least 3 nucleotides long. With this in mind, our objective is to generate the set of all possible helices under these constraints. This is accomplished by first finding all valid base pairs under our model within the given RNA sequence. Next, the algorithm iterates through each base pair and attempts to build a helix by stacking valid base pairs on it. If the resulting helix meets or exceeds the above requirements, it is added to the set H of possible helices. With this formulation for structure generation, the structure prediction problem becomes a combinatorial optimization problem of picking x helices from H to make the final structure.

The fitness metric employed to guide RnaPredict through this search space is energy minimization. As an RNA molecule will fold into a structure with near minimal free energy [19], RnaPredict attempts to find the combination of helices that result in the lowest energy possible. The energy function used in this implementation takes into account the stacking energies in the helices according to INN-HB.

The final solution to the secondary structure prediction problem is a subset of all possible helices that contains only the helices composing the final structure. It is key that only chemically feasible structures are predicted; thus each helix in the subset must be mutually exclusive, (i.e., must not share nucleotides with any other helix in the subset.)

When a given GA is "parallelized," there are a number of different models for implementation which largely involve how the existing single panmictic population is distributed amongst different processors. In this case, our GA employed a coarse-grained distributed model [3]. Briefly, the term coarse-grained refers to a high ratio of time spent in computation versus the time spent in communication. The essence of this distributed GA is the separation of the initial, panmictic population into segregated subpopulations which are also known as demes. These isolated demes behave essentially like miniature serial GAs, and will exchange individuals with each other on an intermittent basis.

The distributed RnaPredict should improve performance through preserving diversity in the population through multiple demes, while increasing the selection pressure through periodic migration [11]. The parallel implementation(P-RnaPredict) will offer a speedup as each deme in the distributed GA will be distributed to its own processor.

4. RANDOM NUMBER GENERATION

Random numbers are useful in many applications and we rely on them extensively in our model. They can be used in simulations, sampling, numerical analysis, decision making, and recreation. An algorithm that employs random numbers is given the name *Monte Carlo* method, in honour of the European resort town in Monaco, to which random processes are so important. It was first suggested by Stan Ulam in 1946 for modeling various neutron transport problems. Other early pioneers include Enrico Fermi and John von Neumann. In our parallel GA, random numbers are used to make coarse-grained decisions in the following functions:

1. Initialization of the random population
2. Determining if crossover occurs
3. Random selection of parents
4. Performance of crossover
5. Determining if mutation occurs on each child
6. Performance of mutation
7. Determining the random order of nodes for migration

Although there appears to be very little in the literature regarding parallel GAs and PRNGs, several studies have been done on how serial GA performance is impacted by PRNGs. These are discussed below.

4.1 PRNGs and Serial GAs

In 1997 [21], Meysenberg performed a thorough empirical study on the effect twelve PRNGs of varying quality had on a simple GA using an eleven function real-value test suite. He found that PRNGs did not significantly impact this GA's performance. In a later study, Meysenberg and Foster [22] discovered isolated cases where poor PRNG quality resulted in slightly improved GA performance. Again, better PRNG quality failed to provide better GA performance.

In 1999, Meysenberg and Foster pursued what they referred to as their *granularity hypothesis* [23]. In essence, a simple GA only requires a PRNG to make choices between several options; this requires only that the PRNG produce a uniform distribution. Since even a poor quality PRNG can accomplish this, its quality should not significantly impact GA performance. Their conclusions were that PRNG quality had no statistically significant effect on GA performance. However, this study was conducted on a generation by generation basis, and it revealed that in GA performance could vary depending on the PRNG and test function chosen. The end result was that good PRNGs could actually result in poorer GA performance, and poor PRNGs could result in slightly better GA performance in isolated cases.

In 2002, Cantù-Paz performed an "ablation" study [4] where the individual GA components of initialization, selection, crossover and mutation were separately tested. Both PRNGs and true random sources were tested. The results indicated that the PRNG used to initialize the random population are critical, whilst the other components were relatively unaffected. His conclusions were that the best PRNG available should be used to avoid misinterpretation of the results due to fortunate accidents.

With a basic notion of how PRNGS could impact GAs, the next step was determining the appropriate method for parallelizing the PRNG for our distributed GA.

4.2 Methods for Parallelization of PRNGs

There are four common methods of designing parallel random number generators [27], central server, cycle division, cycle splitting, and parameterization.

The central server method establishes one process as a central random number server for all other processes in the parallel application. The immediate problem is the tremendous inter-process communications overhead, as each process must have exclusive access during its request to avoid conflicts. Another problem is that reproducibility becomes impossible to assure, as processes may make requests in different orders due to network traffic and the application implementation.

In cycle division, the period of a serial PRNG is subdivided amongst processors in one of three basic ways: *naïve* seed selection, cycle splitting and "leap frog". In naïve seed selection, the user randomly chooses a different seed for each processor. The naïve hope here is that the portions of the PRNG period that each processor consumes are widely separated and do not overlap. Another method, cycle splitting, involves the user carefully selecting the seeds to ensure they are widely separated. In this way, a contiguous block of random numbers from the serial PRNG can be assigned to each processor. However, if the processors consume too many random numbers, the period portions could again overlap. Finally, there is the leap frog method. When given N processors, each processor gets numbers from the serial PRNG period which are N numbers apart. Here, the hazard is that long range correlations in the serial PRNG become short-range correlations within each stream.

The problem with all these methods is that the resulting PRNG is non-scalable; each additional processor takes an equal share of the finite period of the original serial PRNG. Also, reproducibility becomes an issue as each additional processor results in a different serial PRNG period partition for all processors.

The parameterization method by contrast promises to provide independent and uncorrelated random number streams for each processor. There are two basic methods of parameterization [18]: seed parameterization and iterative function parameterization. Seed parameterization works on specific PRNGs for which each initial random seed automatically selects a smaller, separate and independent period. A unique seed is assigned to each processor, ensuring each processor gets a unique period. The second method, iterative function parameterization, creates multiple independent random number streams by generating a different PRNG iteration function for each processor. The idea is that given a number i, the PRNG would generate a unique *ith* iteration function.

At the time of this writing, the best parallel PRNG available appears to be the parallel MT, named "Dynamic Creation" (DC) [16]. DC implements iterative function parameterization, accepting a number of parameters including word size, period, working memory and ID number. A small MT is then produced based on the submitted parameters. The key idea here is that the characteristic polynomial of the MT's linear recurrence encodes the specified ID number, ensuring a unique and highly independent PRNG for each ID.

4.3 PRNG Requirements of the Parallel GA

To date there appears to be little discussion in the literature on parallel GAs in regards to PRNGs. In our research PRNGs became significant for a number of reasons. First, to gain an unbiased idea of the performance of P-RnaPredict we average our results over 30 randomly seeded runs. This implicitly assumes that the random numbers generated for each run are independent of each other. Second, the GA's consumption of random numbers has been rapidly increasing as we perform structure prediction on longer RNA sequences. This has reached a point where the period of the PRNGs available in the standard C library are no longer adequate. Third, the parallelization of the initial serial GA required a corresponding parallelization of whatever PRNG we used.

Cantù-Paz's ablation study underscores the importance of independent PRNG during population initialization. An especially hazardous scenario occurs when parallel PRNG methods such as naïve seed selection or cycle splitting are used in a distributed GA. Consider an example of a distributed GA with two demes where the initial subpopulations are being generated. If each random chromosome in the initial population requires n random numbers, then the total amount of random numbers required to initialize each subpopulation is nm, where m is the population size. With a PRNG with a period of length p, each deme requires a section of that period of length nm to generate its initial population. The worst case scenario is if these sections overlap such that one section offsets the other by a multiple of the chromosome length n. Should this occur, identical chromosomes will be generated in the demes, greatly reducing the diversity within the parallel GA and resulting in diminished performance. This problem worsens with an increase in the number of demes.

Based on these observations, two parallel PRNGs were selected for evaluation in the parallel GA implementation. The first was the DC PRNG described above. The second was a parallelized version of a MCG [13]. The MCG's parameters were $m=2^{31}-1$, $c=0$, and $a=6208991$ as suggested by [7]. This MCG was parallelized by the leap-frog method suggested by [8], and was deliberately chosen to have a lower quality and shorter period than the DC.

Aside from the parallel PRNGs, we also elected to check our results against the original serial GA, which used the standard C library PRNG RAND [15]. With these three PRNGs providing a spectrum of relative quality and period length, we were able to determine the impact of PRNGs on our GA.

5. METHOD

For this set of experiments, the population was subdivided into demes, and each deme was assigned to a single processor. The pseudocode for the distributed GA model is as follows:

```
Initialize random population
Evaluate fitness of individuals
For all NUM_GENERATIONS
    For all DEME_COUNT
        For all DEME_SIZE
            Reproduce by crossover in deme
            Mutate
            Employ replacement (STDS or KBR)
            Apply Elitism
        If MIGRATION_INTERVAL, migrate
```

Given the enourmous number of possible parameter combinations, the selection of parameter sets for these experi-

Table 1: Organism Sequence Details

Organism	Length (nt)	Base Pairs in Known Structure
A. griffini	556	131
H. rubra	543	138
H. marismortui	122	38
S. cerevisiae	118	37

ments was based on previously published experimental results [11, 29]. The parameters specifically relating to the serial GA were chosen based on those which produced the best set of results in [29], and were set as follows: The global population was set to 700, with the crossover probability (P_c) set to 0.7. The mutation probability (P_m) varied as either 0.25 or 0.8. Our prior experiments showed that the standard roulette wheel selection (STDS) worked well in this domain, so all runs presented here use STDS. Similarly, 1-Elitism [12] was also applied in all experiments. We selected the INN-HB thermodynamic model and the CX [24] crossover.

The parameters specifically relating to the distributed GA were chosen based on those which produced the best set of results in [11], and were set as follows: the global population was split into two separate sets of deme sizes and deme counts: 50 and 14, and 70 and 10 respectively. The migration interval were fixed at 20 generations, and the migration rate was fixed at 10 percent. Finally, the topology was fully connected, and the migration policy was set to "best replace worst." Each parameter set was repeated with 30 random seeds and the results averaged.

Four RNA sequences were taken as test data from the Comparative RNA Web Site [2]; they were chosen to provide a good variety of sequence lengths and a variety of organisms. Each sequence chosen had a known structure available for comparison, determined by comparative methods. The three sequences used were a 556 nucleotide (nt) Acanthamoeba griffini sequence, a 543 nt Hildenbrandia rubra sequence, a 118 nt Saccharomyces cerevisiae sequence, and a 122 nt Haloarcula marismortui sequence; their relevant statistics are summarized in Table 1.

Each set of parameters was tested with one of the three PRNGs. The first two were the DC and MCG generators detailed above. The third test was performed using the serial implementation of the distributed GA, which employed the GNU C standard library PRNG RAND. All runs were done on a 128 node Beowulf cluster which supports MPI. This permitted us to perform runs with up to 128 demes in parallel, drastically reducing the time to run the experiment.

The following sections present a summary of results for each RNA sequence. Parameters which do not vary between test runs have been omitted for brevity. "Deme Size" indicates the population of an individual deme. "P_m" indicates the probability of mutation. "Deme Count" indicates the number of demes used. "PRNG" indicates the type of PRNG used. "Avg. Fitness" is the free energy measured in kcal/mol, averaged over 30 randomly seeded runs. "Avg. Base Pair %" is the percent of base pairs which match the predicted structure, averaged over 30 randomly seeded runs. Finally, "Best Base Pair %" is the percentage of matching base pairs from the run with the highest percentage out of the 30 randomly seeded runs for that specified parameter set. Each table is sorted by Deme Size, P_m and Avg. Fitness in order to group the results by parameter set. This is done to clearly delineate the performance differences between the three PRNGs, in terms of the average final free energy reached.

5.1 *Acanthamoeba griffini* - 556 nt

Table 2 indicates that the MCG PRNG performed best in two of the parameter sets based on average free energy, with the DC and RAND PRNGs performing best in one parameter set each. Overall, the MCG PRNG reached the best average free energy at -190.79 kcal/mol with the following parameters: a Deme Size of 70, a Deme Count of 10, and a P_m of 0.8. Averaged over 30 runs, the DC PRNG found the highest percentage of base pairs matching the known structure at 32.34%. The best overall structure was found with 64.88% matching base pairs with the following parameters: a MCG PRNG, a Deme Size of 70, a Deme Count of 10, and a P_m of 0.8.

Table 2: Parallel GA results using three different PRNGs on the *A. griffini* sequence

Deme Size	P_m	Deme Count	PRNG	Avg. Fitness	Avg. Base Pair %	Best Base Pair %
70	0.25	10	DC	-187.58	28.39	58.77
70	0.25	10	MCG	-187.29	30.35	56.48
70	0.25	10	RAND	-186.35	27.04	46.56
70	0.8	10	MCG	-190.79	29.79	64.88
70	0.8	10	DC	-189.35	29.26	60.30
70	0.8	10	RAND	-187.8	28.39	60.30
50	0.25	14	RAND	-186.51	26.89	52.67
50	0.25	14	DC	-184.74	32.34	58.01
50	0.25	14	MCG	-184.43	28.04	48.09
50	0.8	14	MCG	-188.85	31.67	54.96
50	0.8	14	DC	-188.29	26.92	48.85
50	0.8	14	RAND	-185.27	27.17	47.32

5.2 *Hildenbrandia rubra* - 543 nt

In Table 3 we see that based on average free energy the DC PRNG performed best in two of the parameter sets, with the MCG and RAND PRNGs performing best in one parameter set each. Overall, the MCG PRNG reached the best average free energy at -207.08 kcal/mol with the following parameters: a Deme Size of 50, a Deme Count of 14, and a P_m of 0.8. The overall best structures matched 48.55% of the base pairs in the known structure, and were found in single runs from the following two parameter sets: the first was a MCG PRNG, Deme Size of 50, Deme Count of 14, and P_m of 0.8. The second was a RAND PRNG, with a Deme Size of 70, a Deme Count of 10, and a P_m of 0.25. Finally, the highest percentage of matching base pairs averaged over the 30 runs was 29.66%, and it occurred in a run set with the following parameters: a MCG PRNG, a Deme Size of 70, a Deme Count of 10, and a P_m of 0.8.

5.3 *Haloarcula marismortui* - 122 nt

In Table 4 we can see that based on average free energy the DC and RAND PRNG tied for best performance in three

Table 3: Parallel GA results using three different PRNGs on the *H. rubra* sequence

Deme Size	P_m	Deme Count	PRNG	Avg. Fitness	Avg. Base Pair %	Best Base Pair %
70	0.25	10	DC	-200.61	25.09	41.30
70	0.25	10	RAND	-199.65	24.32	48.55
70	0.25	10	MCG	-198.76	24.44	40.57
70	0.8	10	DC	-204.17	26.28	47.82
70	0.8	10	RAND	-203.66	27.58	46.37
70	0.8	10	MCG	-203.14	29.66	44.92
50	0.25	14	RAND	-200.64	27.19	44.20
50	0.25	14	DC	-199.05	24.42	41.30
50	0.25	14	MCG	-198.83	26.81	41.30
50	0.8	14	MCG	-207.08	27.75	48.55
50	0.8	14	RAND	-202.93	26.64	45.65
50	0.8	14	DC	-199.23	22.89	38.40

out of the four parameter sets, with the RAND PRNG performing best in the fourth parameter set. Overall, the DC and RAND PRNGs both reached the best average free energy at -54.94 kcal/mol with the following identical parameters: a Deme Size of 70, a Deme Count of 10, and a P_m of 0.8. However, the DC PRNG edged out the RAND with the best Averaged Base Pair Percentage of 42.10%. For this sequence, the overall best structure was found matching 71.05% of base pairs in the known structure with the following parameter set: a MCG PRNG, a Deme Size of 70, a Deme Count of 10, and a P_m of 0.25.

Table 4: Parallel GA results using three different PRNGs on the *H. marismortui* sequence

Deme Size	P_m	Deme Count	PRNG	Avg. Fitness	Avg. Base Pair %	Best Base Pair %
70	0.25	10	RAND	-54.93	38.59	42.10
70	0.25	10	DC	-54.93	39.47	42.10
70	0.25	10	MCG	-54.88	39.56	71.05
70	0.8	10	RAND	-54.94	41.22	42.10
70	0.8	10	DC	-54.94	42.10	42.10
70	0.8	10	MCG	-54.93	39.47	42.10
50	0.25	14	RAND	-54.92	37.71	42.10
50	0.25	14	DC	-54.92	36.84	42.10
50	0.25	14	MCG	-54.91	35.08	42.10
50	0.8	14	RAND	-54.93	40.35	42.10
50	0.8	14	MCG	-54.92	35.96	42.10
50	0.8	14	DC	-54.92	37.71	42.10

5.4 *Saccharomyces cerevisiae - 118 nt*

The results in Table 5 indicate that all runs for the *Saccharomyces cerevisiae* RNA sequence converged to identical free energy values and secondary structures, regardless of parameter settings or the chosen PRNG. The prediction accuracy was very high.

Table 5: Parallel GA results using three different PRNGs on the *S. cerevisiae* sequence

Deme Size	P_m	Deme Count	PRNG	Avg. Fitness	Avg. Base Pair %	Best Base Pair %
70	0.25	10	MCG	-57.52	89.18	89.18
70	0.25	10	DC	-57.52	89.18	89.18
70	0.25	10	RAND	-57.52	89.18	89.18
70	0.8	10	MCG	-57.52	89.18	89.18
70	0.8	10	DC	-57.52	89.18	89.18
70	0.8	10	RAND	-57.52	89.18	89.18
50	0.25	14	MCG	-57.52	89.18	89.18
50	0.25	14	DC	-57.52	89.18	89.18
50	0.25	14	RAND	-57.52	89.18	89.18
50	0.8	14	MCG	-57.52	89.18	89.18
50	0.8	14	DC	-57.52	89.18	89.18
50	0.8	14	RAND	-57.52	89.18	89.18

6. DISCUSSION

After reviewing the results from the four sequences, it appears that the parallel implementation of the distributed GA (P-RnaPredict) performs comparably to the original serial version (RnaPredict). It is interesting to note that the differences in performance between the two parallel PRNGs and the original serial GA PRNG do not appear to be significant. This reinforces the findings of the previous serial GA studies discussed in Section 4.

Although we employ average matching base pairs to known structures as one of the ranking criteria for the experiment runs, it is important to note that the only value the GAs are designed to optimize is free energy. With free energy as the only applicable comparison between the algorithms, the results indicate both the serial and parallel GAs offer relatively similar performance.

One final area of interest is in the highest matching base pair count. It is important to note that there is no perfect correlation between the lowest free energy value in our models and the highest matching base pair count. Consequently, without having a known structure in advance it would be impossible to determine which particular run would have the highest count of base pair matches. This is a general limitation from which all structure prediction algorithms based on free energy models suffer.

6.1 Secondary Structure Comparison

For the *S. cerevisiae* sequence, the highest number of correctly predicted base pairs P-RnaPredict found was 33 out of 37, or 89.18%. Figure 1 shows a comparison between the known secondary structure for the *Saccharomyces cerevisiae* sequence, and the secondary structure predicted by the parallel GA. Light grey bonds indicate base pairs in the known structure not predicted by the GA. Dark grey bonds indicate base pairs predicted by the GA but not present in the known structure. Black bonds indicate base pairs present both in the known and predicted structure.

It is interesting to note that current thermodynamic models do not account for non-canonical base pairs. However, they do exist in naturally occurring structures including *S.*

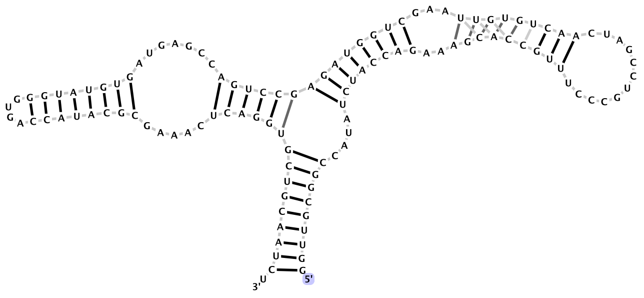

Figure 1: The above image shows a comparison between the known and predicted secondary structures for the *Saccharomyces cerevisiae* RNA sequence. Dark grey lines indicate predicted base pairs, light grey indicate base pairs in the known structure, and the black lines indicate the overlap between predicted and known base pairs. In this case, the parallel GA was able to predict 89.2% of the known base pairs.

cerevisiae. Note the two CU pairs in the structure in Figure 1. These could not have been predicted with the current thermodynamic models. This is why P-RnaPredict has predicted a different helix than what is occurring naturally (see the slight shift of the helix in the figure). Within the limits of the underlying model, P-RnaPredict has found all correct base pairs it could possibly find. *H.marismortui* has similar length as *S.cerevisiae*, however, it contains many more non-canonical base pairs. This explains why the prediction accuracy is much lower in this case than for *S.cerevisiae*.

For longer sequences, the prediction accuracy drops which can be attributed largely to limitations of the thermodynamic model which is not able to model global interactions as the structures grow larger.

While clearly the choice of PRNG has an impact on the overall results, individual differences between them can vary. There is no clear difference between DC and MCG to be observed from the current data set, however, RAND seems to display the worst performance consistently.

7. CONCLUSIONS

We have presented P-RnaPredict, a parallel GA for RNA secondary structure prediction based on the serial RnaPredict. The importance of PRNGs in parallel GAs has been discussed and methods for PRNG parallelization have been reviewed. The impact of two distinct parallel PRNGs on the performance of P-RnaPredict has been investigated. The results from four sequences of 118, 122, 543, and 556 nt indicate that PRNG quality does not have a significant effect on GA performance, which is in keeping with the previous research on serial GAs and PRNGs reviewed in Section 4. However, the serial version of RAND consistently underperformed and it cannot easily be parallelized.

Overall, prediction accuracy is good, particularly so for shorter sequences, however, non-canonical base pairs in naturally occurring structures cannot be modelled with current thermodynamic models.

In future work we will test both a broader set of parameters and a greater number of organisms to further evaluate P-RnaPredict. The speedup factor of the parallel GA will also be investigated. In addition, we plan to compare the quality of structures predicted by P-RnaPredict with other algorithms including the Nussinov DP and *mfold*.

8. ACKNOWLEDGMENTS

The first author would like to acknowledge the support of NSERC for this research under Research Grant number RG-PIN 238298 and Equipment Grant number EQPEQ 240868. The second author would like to acknowledge support from Simon Fraser University, the Advanced Systems Institute of British Columbia and a National Science and Engineering Research Council PGS-A Graduate Fellowship Award (NSERC, Canada, PGS-A). The third author would like to acknowledge support from Simon Fraser University and a National Science and Engineering Research Council PGS-A Graduate Fellowship Award (NSERC, Canada, PGS-A). The fourth author would like to acknowledge support from NSERC under Research Grant number 611407. All authors would also like to acknowledge the support of the InfoNet Media Centre funded by the Canada Foundation for Innovation (CFI) under grant number CFI-3648.

9. REFERENCES

[1] J. Abrahams, M. van der Berg, E. van Batenburg, and C. Pleij. Prediction of RNA secondary structure, including pseudoknotting, by computer simulation. *Nucleic Acids Research*, 18(10):3035–3044, May 1990.

[2] J. J. Cannone, S. Subramanian, M. N. Schnare, J. R. Collett, L. M. D'Souza, Y. Du, B. Feng, N. Lin, L. V. Madabusi, K. M. Müller, N. Pande, Z. Shang, N. Yu, and R. R. Gutell. 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.

[3] E. Cantù-Paz. *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers, 2000.

[4] E. Cantù-Paz. On random numbers and the performance of genetic algorithms. In W. B. Langdon, E. Cantú-Paz, K. Mathias, R. Roy, D. Davis, R. Poli, K. Balakrishnan, V. Honavar, G. Rudolph, J. Wegener, L. Bull, M. A. Potter, A. C. Schultz, J. F. Miller, E. Burke, and N. Jonoska, editors, *GECCO 2002: Proceedings of the Genetic and Evolutionary Computation Conference*, pages p. 311–318. Morgan Kaufmann Publishers, 2002.

[5] A. Deschênes and K. C. Wiese. Using stacking-energies (INN and INN-HB) for improving the accuracy of RNA secondary structure prediction with an evolutionary algorithm - a comparison to known structures. In *Proceedings of the IEEE Congress on Evolutionary Computation (CEC 2004)*, pages 598–606, June 2004.

[6] A. Deschênes, K. C. Wiese, and J. Poonian. Comparison of dynamic programming and evolutionary algorithms for rna secondary structure prediction. In *Proceedings of the 2004 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB'04)*, CIBCB, pages 214–222. IEEE, oct 2004.

[7] G. A. Fishman and I. Louis R Moore. An exhaustive analysis of multiplicative congruential random number

generators with modulus 2^{31}-1. *SIAM J. Sci. Stat. Comput.*, 7(1):24–45, 1986.

[8] G. Fox, M. Johnson, G. Lyzenga, S. Otto, J. Salmon, and D. Walker. *Solving Problems On Concurrent Processors, vol. 1 - General Techniques And Regular Problems.* Prentice-Hall International, 1988.

[9] R. Gutell, J. Lee, and J. Cannone. The accuracy of ribosomal RNA comparative structure models. *Current Opinion in Structural Biology*, 12:301–310, June 2002.

[10] A. Hendriks, A. Deschênes, and K. C. Wiese. A parallel evolutionary algorithm for RNA secondary structure prediction using stacking-energies (INN and INN-HB). In *Proceedings of the 2004 IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB'04)*, CIBCB, pages 223–230. IEEE, IEEE, oct 2004.

[11] A. Hendriks, K. C. Wiese, E. Glen, and A. Deschênes. A distributed genetic algorithm for RNA secondary structure prediction. In R. Sarker, R. Reynolds, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, editors, *Proceedings of the 2003 Congress on Evolutionary Computation CEC2003*, pages 343–350, Canberra, 8-12 December 2003. IEEE Press.

[12] K. D. Jong. *An Analysis of the Behaviour of a Class of Genetic Adaptive Systems.* PhD thesis, University of Michigan, 1975.

[13] D. H. Lehmer. Mathematical methods in large-scale computing units. In *Proceedings of the 2nd Symposium on Large-Scale Digital Computing Machinery*, pages 141–146. Harvard University Press, 1951.

[14] A. L. Lehninger, D. L. Nelson, and M. M. Cox. *Principles of Biochemistry.* Worth Publishers, New York, NY, second edition, 1993.

[15] Linux. *RAND Manual Page*, 1995.

[16] M. M. Matsumoto and T. Nishimura. Dynamic creation of pseudorandom number generators. In *Monte Carlo and Quasi-Monte Carlo Methods 1998*, pages 56–69. Springer, 1998.

[17] F. Major, 2003. private communication.

[18] M. Mascagni and A. Srinivasan. Parameterizing parallel multiplicative lagged-fibonacci generators. *Parallel Computing*, 30(5-6):899–916, 2004.

[19] D. H. Mathews, J. Sabina, M. Zuker, and D. H. Turner. Expanded sequence dependence of thermodynamic parameters improves prediction of RNA secondary structure. *Journal of Molecular Biology*, 288:911–940, 1999.

[20] D. H. Mathews, D. H. Turner, and M. Zuker. RNA secondary structure prediction. *Current Protocols in Nucleic Acid Chemistry*, pages 11.2.1– 11.2.10, 2000.

[21] M. M. Meysenburg. The effect of the quality of pseudo-random number generators on the performance of a simple genetic algorithm. Master's thesis, University of Idaho, Moscow, Idaho, USA, 1997.

[22] M. M. Meysenburg and J. A. Foster. The quality of pseudo-random number generators and simple genetic algorithm performance. In T. Bck, editor, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 276–282, San Francisco, CA, 1997. Morgan Kaufmann.

[23] M. M. Meysenburg and J. A. Foster. Randomness and ga performance, revisited. In W. Banzhaf, J. Daida, A. E. Eiben, M. H. Garzon, V. Honavar, M. Jakiela, and R. E. Smith, editors, *Proceedings of the Seventh International Conference on Genetic Algorithms*, pages 425–432, San Francisco, CA, 1999. Morgan Kaufmann.

[24] I. M. Oliver, D. J. Smith, and J. R. C. Holland. A study of permutation crossover operators on the traveling salesman problem. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic algorithms and their application*, pages 224–230. Lawrence Erlbaum Associates, Inc., 1987.

[25] M. J. Serra and D. H. Turner. Predicting thermodynamic properties of RNA. *Methods in Enzymology*, 259:242–261, 1995.

[26] B. A. Shapiro and J. Navetta. A massively-parallel genetic algorithm for RNA secondary structure prediction. *Journal of Supercomputing*, 8:195–207, 1994.

[27] A. Srinivasan, M. Mascagni, and D. Ceperley. Testing parallel random number generators. *Parallel Computing*, 29(1):69–94, 2003.

[28] F. H. D. Vanbatenburg, A. P. Gultyaev, and C. W. A. Pleij. An APL-programmed genetic algorithm for the prediction of RNA secondary structure. *Journal of Theoretical Biology*, 174:269–280, June 1995.

[29] K. C. Wiese, A. Deschênes, and E. Glen. Permutation based RNA secondary structure prediction via a genetic algorithm. In R. Sarker, R. Reynolds, H. Abbass, K. C. Tan, B. McKay, D. Essam, and T. Gedeon, editors, *Proceedings of the 2003 Congress on Evolutionary Computation CEC2003*, pages 335–342, Canberra, 8-12 December 2003. IEEE Press.

[30] K. C. Wiese and E. Glen. A permutation-based genetic algorithm for the RNA folding problem: a critical look at selection strategies, crossover operators, and representation issues. *BioSystems - Special Issue on Computational Intelligence in Bioinformatics*, 72:29–41, 2003.

[31] C. Woese and N. Pace. Probing RNA structure, function, and history by comparative analysis. In *The RNA World.* Cold Spring Harbor Laboratory Press, 1993.

[32] T. Xia, J. S. Jr., M. E. Burkard, R. Kierzek, S. J. Schroeder, X. Jiao, C. Cox, and D. H. Turner. Thermodynamic parameters for an expanded nearest-neighbor model for formation of RNA duplexes with watson-crick base pairs. *Biochemistry*, 37:14719–14735, 1998.