A New SLS Algorithm for RNA Secondary Structure Design

Mirela Andronescu[1], Anthony P. Fejes[2], Firas Hamze[1], Frank Hutter[1], Holger H Hoos[1*], and Anne Condon[1]

[1] Department of Computer Science
University of British Columbia
Vancouver, B.C., V6T 1Z4, Canada
{androne s,fhamze,hoos,condon}@cs.ubc.ca, mail@fhutter.de
WWW home page: http://www.cs.ubc.ca/labs/beta

[2] Department of Microbiology and Immunology
University of British Columbia
Vancouver, B.C., V6T 1Z4, Canada
fejes@interchange.ubc.ca

Abstract. Ribonucleic acids (RNAs) are amongst the most important molecular components of all biological organisms. Similar to proteins, their function often depends crucially on their structure. Hence, computational methods for design of RNA molecules with specific structural properties are of considerable interest for biological and biomedical research. In this paper, we propose a new algorithm for the RNA Secondary Structure Design Problem: Given any RNA secondary structure, i.e., a specification of base pairing interactions within an RNA strand, find an RNA strand that folds into this structure. This problem can be seen as a complex constraint satisfaction problem with interesting general properties. Our new stochastic local search (SLS) algorithm is based on a combination of problem-specific insights and state-of-the-art techniques for solving CSPs and other hard combinatorial problems. A thorough empirical evaluation shows that it substantially outperforms the best known algorithm for this problem.

1 Introduction

RNA molecules - strands composed of four possible bases, or nucleotides, denoted by A,C,G, and U - play many roles in the cell that go far beyond their role as intermediate molecules in the translation of genomic DNA into proteins. For example, an RNA-protein complex called a ribosome is central to the translation process, and transfer RNA carries free amino acids (the building blocks of proteins) to the protein assembly site. Catalytic RNA molecules, called rybozymes, can cleave other RNA molecules; these and other RNA molecules may have played a role in early evolution before the more complex proteins were evolved. The structure of RNA molecules is crucial to their function in these and many other cellular processes, and indeed computational approaches for predicting RNA secondary and tertiary structure are widely used by biologists.

* Corresponding author

In this paper, we focus on the inverse problem to that of predicting RNA secondary structure, namely design of RNA molecules with a desired secondary structure. One motivation for this work is that novel rybozymes may provide new paths to the design of drugs [**?**], or have industrial uses [**?**]. Another motivation stems from the use of carefully designed DNA structures (which are in many ways analogous to RNA structures) in DNA self-assembly computation [**?**]. Finally, the ability to design RNA molecules with specific structural features can play a crucial role in research on the function of natural RNAs.

Before describing our problem and approach in more detail, we note briefly that a secondary structure for an RNA strand is simply a set of pairing interactions between bases in the strand. Each base can be paired with at most one other base. Most base pairings occur between Watson-Crick complementary bases C and G or A and U, respectively (canonical pairs). Other pairings, such as G-U can be found occasionally. Computational approaches for prediction of RNA secondary structure are based on a thermodynamic model that associates a free energy value with each possible secondary structure for a strand. The secondary structure with the lowest possible free energy value is predicted to be the most stable secondary structure for the strand. Although there are exponentially many possible secondary structures for a given RNA strand, RNA secondary structure prediction appears to be easier than protein secondary structure prediction, at least for the class of so-called pseudoknot-free secondary structures (see Section **??** for the definition of pseudoknot-free structures). There are widely used dynamic programming algorithms that, given an RNA strand of length n, find in $\Theta(n^3)$ time the secondary structure with the lowest free energy, from the class of pseudoknot-free secondary structures. Throughout the paper, all references to secondary structures refer to pseudoknot-free secondary structures.

The RNA design problem that we consider here is as follows: given a secondary structure, find an RNA strand (if any), that folds to that structure. This can be seen as a discrete constraint satisfaction problem where the constraint variables are the positions in the desired RNA strand, the values assigned to these variables correspond to the bases at the respective positions, and the constraints capture the base pairings that define the given secondary structure. Although its complexity is unknown, this RNA secondary structure design problem appears to be computationally hard, and so a heuristic approach is appropriate. We note that for evaluating a candidate base assignment with respect to the given secondary structure constraints, the minimal energy secondary structure for the respective candidate RNA strand needs to be determined. The fact that this operation requires $\Theta(n^3)$ time poses a challenge for any heuristic search approach.

We present a new algorithm, RNA-SSD (RNA Secondary Structure Designer), that designs RNA strands for input secondary structures. At the core of our algorithm is a stochastic local search (SLS) procedure that iteratively modifies single unpaired bases or base pairs of a candidate strand in order to obtain a strand that folds into the target structure.

Since each step of the SLS algorithm requires a call to a $\Theta(n^3)$-time evaluation function, a key component of our approach is a hierarchical decomposition of the input secondary structure into small substructures. The core SLS algorithm is only applied to the smallest substructures, and the corresponding partial solutions are combined into candidate solutions for larger subproblems guided by the

decomposition tree. Since the subproblems are not independent, this does not always result in valid designs for the corresponding substructure. Consequently, multiple attempts (involving additional calls to the core SLS procedure) are often required before partial solutions can be successfully combined.

The other key component of the algorithm is a method for generating a good initial design for the RNA strand. This initialisation procedure assigns bases probabilistically to the strand, using different probabilistic models for base positions that are paired and unpaired in the target structure. In addition, the algorithm ensures that complementary stretches of bases are avoided across the design, except where desired along two sides of a stem.

We empirically evaluated our RNA-SSD algorithm on computationally predicted structures of naturally occurring sequences from the Ribosomal Database Project (RDP), as well as on randomly generated structures; and we compared its performance with a previous algorithm of Hofacker et al. [**?**] for RNA strand design which is part of the widely used Vienna RNA Secondary Structure Package. The results of this emirical evaluation shows that RNA-SSD substantially outperforms the Vienna algorithm on a broad range of structures. For example, on randomly designed structures with biological realistic features that could be solved by both algorithms, RNA-SSD was typically up to two orders of magnitude faster than the Vienna algorithm. Furthermore, when applied to 26 instances from the RDP (with lengths ranging roughly between 300 and 1,500), RNA-SSD found solutions for all but four structures, whereas the Vienna inverse folding algorithm did not find a design for any of the structures.

The rest of this paper is organized as follows. In Section **??**, we give some background on RNA secondary structure prediction, define the problem of RNA strand design formally, and describe some previous work on this problem. Our algorithm is described in detail in Section 3. In Section 4, we present our empirical analysis and performance results. Conclusions and future work are described in Section 5.

2 The RNA Secondary Structure Design Problem

An RNA strand is composed of four basic units, or nucleotides, called cytosine (C), guanine (G), adenine (A), and uracil (U). These are connected via a phosphate backbone. A strand has two chemically distinct ends, known as the $5'$ and $3'$ ends. Simple Watson-Crick basepairing involves the hydrogen bonding of C with G and A with U, forming three or two hydrogen bonds respectively. In addition to these two possible pairings, a third type of pairing between guanine and adenine (G-U), termed a "wobble pair", may also occur. For a given RNA strand, a secondary structure describes which bases are paired. Specifically, the secondary structure of a strand of length n is a set of pairs (i, j), where i and j are in the range $[1, \ldots, n]$, and (i, j) represents a pairing between the ith and jth bases in the strand where the bases in the strand are indexed from 1 to n starting at the 5' end. In a secondary structure, each base has at most one partner. Base pairs are most often found stacked onto other base pairs in substructures called *stems* or *helices.* Sometimes, unpaired bases are interspersed in stems; these are known as *internal loops* or *bulges.* Loops occurring at the ends of stems are called *hairpins*, and loops from which more that two stems originate are known as *multi-branched*

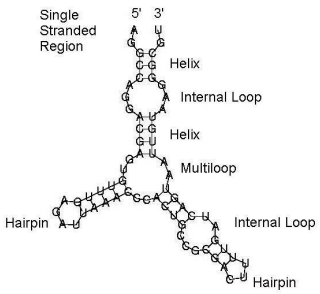

Fig. 1. A secondary structure for an RNA molecule showing loops and stems

loops, or simply *multiloops*. Figure **??** shows a standard display of a secondary structure for an RNA strand, in which the stems and loops are apparent. This particular secondary structure is *pseudoknot free*; that is, it does not have any two base pairs (i, j) and (i', j') where $i < j < i' < j'$.

Associated with a secondary structure for a strand is its *free energy*. For pseudoknot free secondary structures, this is typically calculated as the sum of the free energies of each stacked pair and each loop [**?**], and estimates for these values have been experimentally determined. Let $E(X, S)$ denote the free energy of an RNA sequence X when folded into the secondary structure S. Furthermore, let Φ denote a function that assigns to each RNA sequence X a secondary structure S^* that minimises free energy $E(X, S)$ over all possible secondary structures S of X.

The **RNA Secondary Structure Prediction Problem** can be stated as follows: Given an RNA sequence X, determine $\Phi(X)$. When the energy $E(X, S)$ of an RNA structure x folded into the secondary structure S is evaluated by the nearest neighbor thermodynamical model (not allowing for pseudoknots), the RNA folding problem is efficiently solvable using Zuker's algorithm [**?**], a dynamic programming approach that runs in time $\Theta(n^3)$.

Analogously, we can state the **RNA Secondary Structure Design Problem**: Given an RNA secondary structure S^*, find a sequence X^* s.t. $\Phi(X^*) = S^*$. In the optimisation variant of this problem, we determine the quality of a candidate solution X by comparison of the structure $S = \Phi(X)$ with the desired structure S^*; given a distance metric d, we attempt to minimise $d(S, S^*)$. In our algorithm, the distance metric will merely measure the number of bases that bond incorrectly, i.e., number of bases in X whose pairing status is different between S and S^*. It is also possible to include information on the stability of folds in the distance metric.

Hofacker et al. [**?**] have already developed an algorithm for RNA strand design, which is included in the Vienna RNA Secondary Structure Package [3]; we will refer to this approach as the *Vienna inverse folding algorithm* or short *Vienna algorithm*. The algorithm performs an "adaptive" walk search on so-called

[3] `http://www.tbi.univie.ac.at/~ivo/RNA/`

compatible sequences, i.e., sequences that can possibly form a base-pair at the required positions in the desired structure. These sequences are candidates only; there is no guarantee that they will fold into the target structure. Starting from a compatible sequence x_0, in each step the algorithm induces a mutation (while ensuring that the sequence remains compatible) and accepts it if and only if the cost function decreases. These search steps are iterated until either a solution is found or a certain number of mutations has been carried out. A drawback of this approach is that calculating the distance for each mutation involves running a folding algorithm on the sequence under observation, for which generally Zuker's $\Theta(n^3)$ algorithm is used, where n is the length of the sequence to be folded. To counter this, Hofacker et al. propose that individual substructures' subsequences be determined first and the final sequence be the one corresponding to a simple concatenation of the subsequences. The rationale is that it is likely (but not assured) that the substructures which are optimal for subsequences will also occur for the full sequence.

3 Our RNA Secondary Structure Design Algorithm

Our new algorithm for the RNA Secondary Structure Design Problem is based on a stochastic local search approach that uses a probabilistic sequence initialisation heuristic, hierarchical decomposition of the given structure, and a randomised iterative improvement method for finding sequences for the resulting substructures. The space searched by our algorithm consists of RNA sequences that correspond to complete assignments of bases $x_i \in \{\text{A,C,G,U}\}$ to all positions i of the given RNA secondary structure. We restrict this space to sequences in which positions that are paired in the desired structure are assigned complementary bases, such as C-G or A-U.

Different from many other constraint satisfaction problems, evaluating the quality of candidate solutions for the RNA Secondary Structure Design Problem is computationally quite expensive, having time complexity $\Theta(n^3)$, where n is the length of the given sequence [**?**] (see Section **??**). We use the *fold* function from the Vienna Package, the most efficient implementation of Zuker's algorithm we are aware of. Unfortunately, even a single local reassignment of a base in the sequence can result in a completely different minimal energy secondary structure. Hence, there seems to be little hope for reducing the complexity of evaluating candidate solutions by incremental updating. Consequently, an SLS algorithm for the RNA Secondary Structure Design Problem should keep the number of candidate solution evaluations minimal. In this respect, simple variants of straight-forward SLS algorithms, such as the Min-Conflicts Heuristic [**?**], can be expected to peform poorly on this problem, particularly when applied to larger problem instances.

Based on these considerations, our algorithm takes a different approach: After constructing an initial candidate sequence for the entire given RNA structure S, that structure and the initial sequence are hierarchically decomposed into smaller components corresponding to substructures of S. At the lowest level, these subproblems are independently solved using a conventional SLS algorithm. Solutions to these subproblems are then combined into candidate solutions for larger subproblems. There is no guarantee that valid solutions to subproblems can be combined into a valid solution of a larger subproblem; hence, at this stage,

```
procedure RNA-SSD
    input: target RNA secondary structure S, parameters
    output: RNA sequence X
    initialise sequence X;
    hierarchically decompose S and X;
    recursively search for sequence with minimal energy structure S;
    return X;
end RNA-SSD.
```

Fig. 2. Pseudocode for the RNA secondary structure design algorithm; details are discussed in the text.

each combination attempt has to be evaluated using Zuker's algorithm. If at any stage the respective combined candidate sequence does not fold into the required structure, new candidate solutions to the subproblems are determined using the same mechanism as described before.

Following this approach, the expensive evaluation of candidate solution happens primarily at the level of substructures which can be made small enough (by iterated decomposition) to render the $\Theta(n^3)$ complexity of Zuker's algorithm manageable. Larger candidate sequences are only evaluated after merging partial solutions, a process that happens much more rarely. It may be noted that this approach is based on the intuition that although there can be complicated dependencies between subproblems, there is a reasonable chance that solutions to subproblems can be successfully combined into solutions of the entire problem. There is substantial biological evidence that for the RNA Structure Design Problem this intuition is likely to hold; furthermore, the empirical performance of our algorithm on biological and artificial RNA structures also supports this intuition.

An outline of the algorithm is shown in Figure **??**; in the following, we will discuss its components (and parameters) in more detail.

Sequence Initialisation

The easiest way of initialising the the RNA sequence X would be to randomly and independently assign bases to the positions of X according to a uniform distribution over A,C,G,U. However, given our goal of minimising the number of candidate solution evaluations, it would be preferable to initialise the sequence in such a way that its minimal energy structure is as close as possible to the target structure S.

Firstly, the assignment should be done in such a way that paired bases in S are assigned complementary bases. This ensures that X can fold into the desired structure S, but there might be alternate conformations with lower free energy. Based on the same intuition, we make sure that the unpaired sequence positions directly following helix regions (e.g., in the loop section of a hairpin loop) are

assigned non-complementary bases; this prevents energetically favourable but undesired helix extensions.[4]

Secondly, the fact that C-G base pairings are energetically more favourable than A-U pairings suggests that by preferentially assigning C-G pairs to helix regions and other paired positions of S sequences can be obtained whose minimal energy structures tend to contain these (desired) pairings. Furthermore, by preferentially assigning A and U bases to unpaired sequence positions in X the chances of erroneous pairings should intuitively be decreased. These heuristic choices are closely related to the conditions underlying recent theoretical work on the RNA Secondary Structure Design Problem [**?**]. They are also in agreement with the fact that the average fraction of $C-G$ pairs in helices is relatively high for most organisms' RNAs.

Finally, we use a tabu-mechanism to further minimise the potential for undesired but energetically favourable interactions between subsequences of X. This mechanism is based on assigning short sequences of bases (sequence motifs) to contiguous chunks of the target structure S. A sequence motif m is only admissible if it does not form more than d_{min} consecutive base pairs with any previously used motif or with itself. (A similar mechanism is used by Seeman in design of DNA structures [**?**].) The maximal chunk size l_{max} (i.e., motif length) and the motif distance threshold d_{min} are parameters of our algorithm; obviously, d_{min} and l_{max} need to be set in such a way that sufficiently large sets of admissible motifs exist. In our experiments we somewhat arbitrarily set them as $l_{max} = 10$ and $d_{min} = 5$, but there is likely room for improvement.

Overall, our initialisation algorithm works as follows: the target structure S is partitioned into chunks of paired or unpaired bases of maximal size l_{max}. Then, the sequence of chunks is traversed from the $5'$ to the $3'$ end of the structure. For each chunk of unpaired bases we generate a corresponding sequence motif m of the same length by choosing bases A,C,G,U independently and randomly for each position with probabilities p_A, p_C, p_G, p_U with $p_A = p_U > p_C = p_G$, respectively. If base pairing between the first or last base of this chunk with the last or first base of a previously assigned unpaired chunk could lead to an undesired helix extension (as described above), the probabilistic base generation is conditioned on not producing the problematic bases.

If the motif m is admissible in terms of the tabu mechanism described above, it is assigned to the respective chunk. Chunks of paired bases are handled analogously, only that in this case probabilities p'_A, p'_C, p'_G, p'_U are used for generating the bases with $p'_C = p'_G > p'_A = p'_U$ and both the motif m as well as its complement $\bar{m}$ are checked for admissibility and assigned to the two chunks corresponding to both sides of the respective helix region.

The probabilities p_C and p'_C are parameters of our algorithm (the other probabilities can be derived from these based on the equalities stated above). For the experiments reported in Section **??**, we used $p_C = 0.1$ and $p'_C = 0.4$; these values merely reflect the underlying biological intuition and have not been tuned to optimise performance.

[4] Since we are considering non-canonical G-U pairings, it is possible (for bulge loops with one free base) that such an assignment does not exist, in which case this constraint is ignored.

Empirical evidence (not shown here) suggests that especially for larger and more complex target structures, the probabilistic base assignment as well as the tabu mechanism contribute significantly towards the strong performance of our algorithm (see Section **??**).

Hierarchical Decomposition

The purpose of the hierarchical decomposition procedure is to divide the given target structure S into small structure components; these components correspond directly to subsequences of the candidate sequence X. Like Hofacker et al. [**?**], we split the structure at multiloops. Different from their method, however, we recursively split the structure into two substructures at each stage; thus we obtain a binary decomposition tree whose root is formed by the full target structure X and whose leaves corresponds to substructures of X that do not contain any multiloops. Each non-leaf represents a substructure of X that can be obtained by merging the two substructures corresponding to its children.

In contrast to Hofacker et al.'s algorithm [**?**], our procedure attempts to construct decomposition trees that are as balanced as possible. This is done by selecting a multiloop and split point within that multiloop such that the two substructures obtained by splitting at that point results are as close as possible in size (where the size of a substructure corresponds to the length of the corresponding subsequence).

In general, such a split creates two new free ends of the underlying RNA sequence (in addition to its original $5'$ and $3'$ ends). Subsequently, we will use Zuker's RNA secondary structure prediction algorithm for evaluating candidate sequences assigned to these substructures; but this algorithm can only be applied to single RNA strands (i.e., structures with two free ends). Therefore, we always split by separating the structure such that (a) the split point never falls within a chunk of consecutive free or paired basis, and (b) by connecting the two free ends created by the split such that both resulting substructures have exactly two free ends. To create structural "boundary conditions" at the split points that are similar to those of the original structure, this connection is achieved by merging the free ends with those of a small hairpin loop of size nine (three paired, three unpaired, three paired bases); furthermore, we add two unpaired bases to the two remaining free ends of each substructure if it contains a bulge directly after the first base pair. (The added bases are generated using the sequence initialisation procedure described above.)

An example for a split as performed by our hierarchical decomposition method is shown in Figure **??**; note that there are only six split points in this structure, five of which would lead to a less balanced decomposition into two substructures.

Recursive Stochastic Local Search

The final component of our algorithm is a recursive SLS procedure. Starting at the leaves of the decomposition tree, corresponding to the smallest substructures S_k of the given target structure S, this procedure iteratively modifies single bases of the corresponding subsequences X_k that are in conflict with S_k, i.e., that are paired in the mimimal energy structure of X_k but not in S_k or vice versa. This is

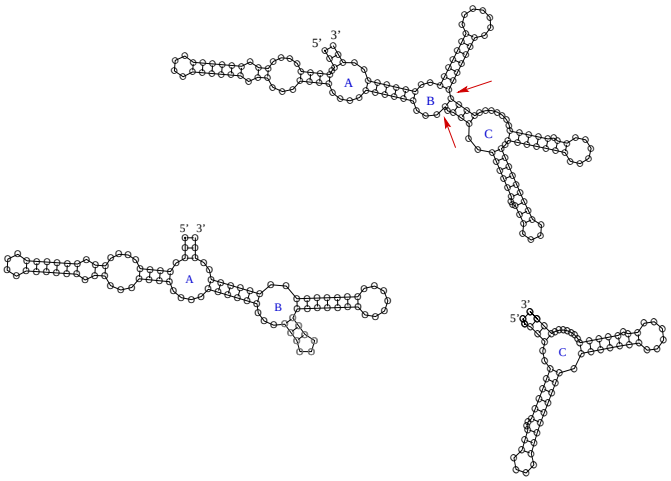

Fig. 3. An example of a structure divided into two substructures using the decomposition method described in the text. (The split point is indicated by arrows, bases shown in grey are added to enforce boundary conditions.)

done until either a subsequence X_k with minimal energy structure S_k is obtained, or until a maximal number n_L of base modifications have been performed without finding such a sequence. Note that Zuker's RNA structure prediction algorithm has to be called in each iteration of this procedure.

In its recursion step, the SLS procedure is first used to determine valid sequences X_i and X_j for the two substructures corresponding to the children of a given non-leaf node k of the decomposition tree. (If no valid sequence is found for X_i, the algorithm still continues with the previous best candidate sequence for X_i as if it were valid, and similarly for X_j.) A candidate sequence X_k for S_k, the substructure associated with node k is then determined by merging X_i and X_j (after removing any bases newly introduced in the hierarchical decomposition process). Next, the minimal energy structure S'_k of X_k (determined using Zuker's algorithm) is compared to S_k. If S'_k and S_k are identical, the recursive step has been successfully completed. Otherwise, the conflicting bases in X_i or X_j are memorised and the SLS procedure is recursively called on the subsequence with the higher relative fraction of conflicting bases, resulting in a new sequence X_i or X_j. This process is iterated until either a valid sequence for S_k has been found, or until a maximal number n_M of attempts to merge subsequences X_i and X_j into a valid sequence for S_k have been performed.

The single-base modifications at the level of smallest substructures are determined based on a randomised first-improvement strategy: With a fixed probability p_c, an arbitrary base position i in the given subsequence is selected uniformly at random; otherwise, this uniform random choice is restricted to base positions that are currently in conflict with the target structure. An alternate base as-

signment is then generated for the selected position using the same probabilistic model for that position as in the previously described sequence initialisation procedure. (In particular, the probabilities for generating particular base assignments depend on whether that base position is paired or free in the target structure.) If necessary, this process of proposing alternate bases for position i is repeated until a base x is found that is not identical with the present, conflicting base assignment to position i. If base assignments for this position previously resulted in a mispairing on higher levels in the recursion, x is also not allowed to be the base assigned on the lowest conflicting level. Replacing the current, conflicting base at position i with x leads to a new candidate sequence X'_k for this substructure. At this stage, the minimal energy structure S'_k for X'_k is determined (using Zuker's algorithm); if S'_k is closer than the minimal energy substructure of the previous candidate sequence X_k in terms of the number of conflicting bases, the search is continued from X'_k; if the number of conflicting bases is higher for X'_k than for X_k, the search is still continued from X'_k with probability p_a, and from X_k in the remaining cases.

Note that using this first-improvement strategy helps to keep the number of (expensive) candidate solution evaluations per search step small. The randomisation helps to ensure that this search does not get stuck in local optima of the evaluation function.

4 Experiments and Results

In this section, we report results from a thorough empirical evaluation of the performance of RNA-SSD as compared to the Vienna inverse folding algorithm [**?**]. Our results clearly indicate that RNA-SSD shows substantially improved performance over the Vienna algorithm; furthermore, the observed differences in performance increase with problem hardness.

Random Structure Generator

In order to allow us to evaluate our algorithm on larger sets of RNA structures with controlled properties, we designed and implemented a simple random generator for pseudoknot-free RNA secondary structures. In the following description, $[j..k]$ denotes the set of integers i with $j \leq i \leq k$; all random selections are based on a uniform distribution over all possible choices.

The probabilistic generation process starts with a single stem, whose length (in base pairs) is randomly selected from [5..14]. To one end of this helix we attach two stretches of three unpaired bases each; these form the 5' and 3' ends of the target structure and will not be further extended throughout the construction process. The other end of the stem is added to a list L which contains all open ends of stems to which further structure elements can be added later.

In each iteration of the algorithm, we choose an open stem uniformly at random from L and attach a new multiloop M to it. The number of unpaired bases in M is randomly chosen from [6..9]; likewise, the number of stems branching off M (including the "incoming" stem which was extended by adding M) is randomly selected from [2..5]. These stems are inserted in M at randomly chosen positions, and their open ends are added to L.

This process is iterated until the structure reaches a total number multiloops that is randomly selected from [1..19]. At this point, all open stems in L are closed by adding a hairpin loop consisting of a number of unpaired bases randomly chosen from [3..6]. Finally, bulge loops are created by inserting short stretches of unpaired bases at random points of the structure This is done by randomly selecting a fraction f_b of the bases positions as insertion points; the number of unpaired bases inserted at each of these positions is selected independently at random from [3..6]. Since structures that contain isolated, single base pairs between stretches of unpaired bases tend to be highly unstable and often appear to be impossible to design, we do not allow insertions that would lead to such structures.

This construction method yields structures that share many of the features of RNA secondary structures found in nature (e.g., rRNA). Salient properties of the generated structures can be controlled by the parameters of the probabilistic process.

Data Sets for Empirical Analysis

Our empirical analysis is based on three data sets. Sets 1 and 2 were created using the random structure generator described above. Set 1 consists of 81 structures without any bulge loops, with lengths ranging from 54 to 499 bases. The structures in Set 2 contain bulges (between one and five percent of the respective strand positions are located in bulges); the set consists of 50 structures ranging in length from 101 to 395.

The third data set was created from ribosomal RNA sequences (rRNAs) obtained from the Ribosomal Database Project [5]. We obtained a minimal energy structure for each of these sequences by using the same RNA structure prediction algorithm used within RNA-SSD for evaluating candidate sequences; this ensures that at least one solution exists for each structure. The rRNA sequences and substructures for which experiments were performed were chosen in an arbitrary and unbiased way.

RNA-SSD vs. Vienna Inverse Folding Algorithm

All computational experiments were carried out on PCs with dual 1GHz Pentium III processors, 256MB cache, and 1GB RAM running Red Hat Linux, Version 2.4.9-6smp. Both, RNA-SSD and the Vienna algorithm are highly randomised; consequently, we performed multiple runs of each algorithm on all problem instances whenever feasible and analysed the run-time distributions (RTDs) thus obtained [**?**]. Figure **??** shows typical cumulative RTDs for RNA-SSD and the Vienna inverse folding algorithm applied to a a randomly generated structure without bulges (left) and an rRNA fragment (right). As can be seen from these RTDs, the time required for reaching any given success probability is substantially shorter for RNA-SSD than for the Vienna algorithm. Furthermore, both algorithms show a high degree of variability in the time required for finding a solution; this is typical of many SLS algorithms for CSPs (see [**?**]).

[5] `http://rdp.cme.msu.edu/download/SSU_rRNA/unaligned/SSU_Unal.gb`

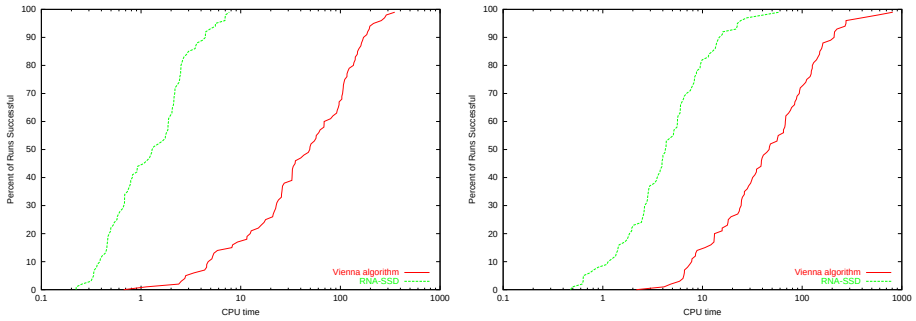

Fig. 4. Empirical run-time distributions (RTDs) for RNA-SSD and the Vienna inverse folding algorithm for a small randomly generated structures without bulges (left, length 119) and a small rRNA fragment (right, length 118). The RTDs are based on 100 runs of each algorithm in all of which a solution was found. CPU time is measured in seconds.

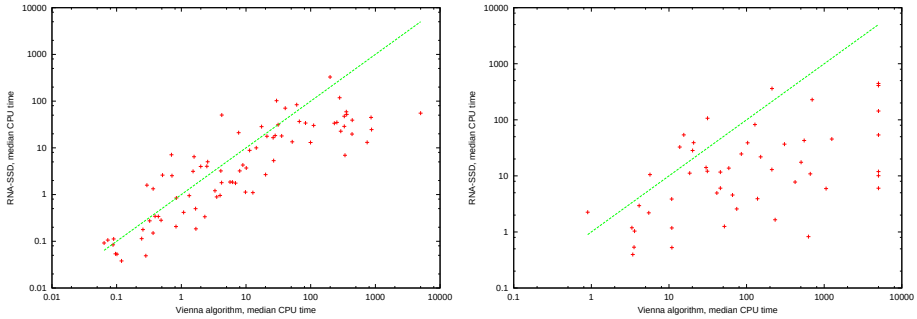

Fig. 5. Performance of RNA-SSD vs. Vienna algorithm on sets of randomly generated structures. Left: Set 1 (bulge-free structures), right: Set 2 (sequences with bulges). Reported times are median values over 10 runs per instances (in CPU seconds).

Figure **??** compares RNA-SSD and the Vienna inverse folding algorithm on the two test-sets of randomly generated structures with and without bulges. Each algorithm performed 10 runs per structure; unsuccessful runs were terminated after 5000 CPU seconds. In both cases, RNA-SSD solved all instances, while the Vienna algorithm never found a solution for one (out of 81) instances from Set 1 and for seven (out of 50) instances from Set 2. For both sets, our algorithm is faster than the Vienna algorithm on more than 3/4 of the remaining instances. The performance difference is much more pronounced for the biologically more realistic structures with bulges, where RNA-SSD if often up to two orders of magnitude faster than the Vienna algorithm. Generally, these structures appear to be more challenging for both RNA secondary structure design algorithms.

Finally, we evaluated the RNA-SSD and the Vienna inverse folding algorithm on the complete rRNA secondary structures in Set 3. For each structure, we performed 10 runs of up to five CPU hours each for RNA-SSD and one run of

up to five hours each for the Vienna algorithm. For both algorithms, if a solution attempt terminated unsuccessfully within less than 1000 CPU seconds, the run was continued by restarting the algorithm.

The Vienna algorithm did not find a solution to any of the structures in this test-set. In some cases, it returns solutions that are close to the target structure, while in other cases, no candidate solution is produced within 5 CPU hours. RNA-SSD, on the other hand, found solutions in all 10 runs for 14 of the 26 problem instances; with few exceptions, solutions were found in an average time of less than one CPU hour. Only four of the remaining structures were not solved in any of 10 runs; but in all of these cases, our algorithm produced better candidate structures in substantially less time than the Vienna algorithm.

5 Conclusions and Future Work

We have introduced a new algorithm for the RNA Secondary Structure Design Problem, an interesting and complex constraint satisfaction problem from computational biology. Our algorithm is based on a stochastic local search technique that uses a carefully designed probabilistic initialisation procedure and a hierarchical structure decomposition technique to keep the high cost of evaluating candidate solutions manageable. The empirical performance of our new algorithm on a broad range of biological and artificially generated RNA structures is substantially better than the state-of-the-art method for this problem, the Vienna inverse folding algorithm, both in terms of speed as well as with respect to the structures that can be solved. We attribute this success to a powerful combination of modern constraint-solving methods (particularly, advanced SLS techniques) and problem-specific biological knowledge, as for example embodied in the heuristic initialisation method.

This work can be extended in various directions. Firstly, we are convinced that further substantial improvement of our algorithm can be achieved. By using a customised implementation of Zuker's RNA secondary structure prediction algorithm, it should be possible to significantly speed up the evaluation of candidate sequences. This modified evaluation procedure would not apply the full dynamic programming algorithm underlying Zuker's method for each evaluation, but start from a previous dynamic programming matrix and update only those elements that have been affected by one or more local sequence changes. Another area for further improvements is the SLS algorithm used for the smallest substructure; we expect that more advanced SLS methods for constraint satisfaction problems, such as tabu-search techniques (see, e.g., [**?**]), will lead to improved performance of the overall algorithm. Also, we are currently studying recursive SLS procedures that incorporate a dynamic stochastic decomposition into even smaller substructures (which are conceptually related to the chunks currently used during sequence initialisation); preliminary results suggest that this extension leads to substantially performance improvements, especially for large RNA structures with more than 1,000 bases.

Secondly, further empirical analysis of our algorithm as well as the Vienna inverse folder could help to better understand the strengths and limitations of both approaches. We plan to study both algorithms' behaviour on a wider range of biological randomly generated sequences with controlled properties; e.g., we

Source (organism / sample)	length	RNA-SSD # sln found	RNA-SSD avg time	Vienna Inv. Folder distance	Vienna Inv. Folder time
Sargasso Sea 10m depth bacterio-plankton DNA clone BDA1-10	299	10/10	28.743	9	1390.773
Clone pM9-23	299	10/10	1254.235	12	733.410
Clone pB7-128R	299	10/10	192.437	4	1555.210
Leptospira interrogans str. 94-7997013	299	3/10	1049.805	15	1393.612
Str. NR64, bacterium expressing pSym plasmid	300	10/10	77.989	2	1272.369
Ochrobactrum BL200-8 str. BL200-8	357	10/10	1039.172	2	262.201
Methanobrevibacter FMBK1	359	7/10	787.070	—	—
Anabaena uncultured bacterium SY2-21	359	10/10	226.738	10	1548.105
Octopus Spring microbial mat DNA from Yellowstone NP clone Type E	417	8/10	524.179	10	3153.804
Prevotella albensis str. M384 DSM 11370 (T)	419	0/10 (6)	1396.816	17	2678.361
Clone 3-25	419	10/10	1402.312	8	2805.905
Clone CRE-FL72	479	7/10	1448.288	27	7226.848
Stenotrophomonas sp. str. P-9-8	534	8/10	665.520	20	9790.491
Clone vadinIA59	539	10/10	39743.329	8	8820.554
Nitrobacter Nb4 str. Nb4	659	9/10	4080.171	15	7005.243
Wolbachia pipientis	659	10/10	581.381	8	4750.398
Clone ST1-4	776	9/10	3591.275	37	9819.491
Bradyrhizobium sp. str. 283A	780	10/10	610.515	—	—
Sulfolobus acidocaldarius str. N8	899	10/10	23452.347	—	—
Spirochaeta sp	899	10/10	938.238	—	—
Str. 1200m deep water sample of Lake Baikal	960	0/10 (6)	4526.396	—	—
Dendrodoa grossularia	1019	10/10	1774.614	16	19847.821
Chitinophaga pinensis ACM 2034 (T)	1175	0/10 (468)	1476.879	—	—
Thermococcus stetteri str. K-3 DSM 5262 (T)	1282	0/10 (29)	33632.245	—	—
Methanobrevibacter sp. str. HO DSM 11977	1378	9/10	8475.299	—	—
Methanococcus jannaschii	1499	10/10	742.680	—	—

Table 1. Performance results for RNA-SSD and the Vienna inverse folding algorithm for rRNA secondary structures from the RDP database. For cases where RNA-SSD did not find a valid solution within 10 tries, the median distance from the target structure (number of incorrectly paired or unpaired bases) is shown in parentheses. Runs that did not finish within five hours of CPU time and had to be aborted without producing any result are marked with "—".

have recently begun to investigate which factors make instances of the RNA Secondary Structure Design problem hard or easy to solve. This type of investigation is facilitated by our random RNA structure generator, whose underlying probabilistic model can be easily extended to yield structures based on important structural and statistical properties (such as helix length, number and location of bulges, etc.) of various classes of biological RNAs.

Finally, it would be interesting to apply the approach used here, especially the combination of hierarchical decomposition and recursive SLS methods, to other types of structured CSPs. We expect this to work well for related problems from computational biology, such as as the design of multistable RNA molecules [**?**]. More generally, we believe that many CSPs exhibit conceptually similar structure to the problem studied here in that they can be decomposed into weakly, but critically dependent subproblems. It appears to be not unreasonable to assume that this type of structure would generally afford the successful application of recursive SLS methods such as the one underlying our new algoritm for RNA Secondary Structure Design.

References

1. Cech T.R., *Ribozyme engineering*, Current Opinions in Structural Biology, 1992, 2:605–609.
2. Flamm C., I.L. Hofacker, S. Maurer-Stroh, P.F. Stadler, M. Ziehl, *Design of multistable RNA molecules*, RNA(2001), Cambridge University Press, 2001, 7:254–265.
3. Heitsch C., A. Condon, H. Hoos, *From RNA Secondary Structure to Coding Theory: A Combinatorial Approach*, To appear in: Proceedings of the 8th International Meeting on DNA-Based Computing, 2002.
4. Hofacker I.L., W. Fontana, P.F. Stadler, L.S. Bonhoeffer, M. Tacker, P. Schuster, *Fast folding and comparison of RNA secondary structures*, Chemical Monthly, 1994, 125:167–188.
5. Hoos H.H., *Stochastic Local Search - Methods, Models, Applications*, PhD thesis, Department of Computer Science, Darmstadt University of Technology, 1998.
6. Hoos H.H., T. Stützle, *Local Search Algorithms for SAT: An Empirical Evaluation*, Journal of Automated Reasoning, 2000, 24(4):421–481.
7. Lyngso R.B., M. Zuker and C.N.S. Pedersen, *An Improved Algorithm for RNA Secondary Structure Prediction*, BRICS Report Series, 1999, RS-99-15, Department of Computer Science, University of Aarhus.
8. Minton S., M.D. Johnston, A.B. Philips, P. Laird, *Minimizing Conflicts: A Heuristic Repair Method for Constraint Satisfaction and Sceduling Problems*, Artificial Intelligence 1992, 52:161–205.
9. Seeman N.C., *De novo design of sequences for nucleic acid structural engineering*, Journal of Biomolecular Structure and Dynamics, 1990, 8(3):573–581.
10. Steinmann O., T. Stützle, A. Strohmaier, *Tabu Search vs. Random Walk*, KI-97:Advances in Artificial Intelligence, Springer Verlag, LNCS, Vol. 1303.
11. Usman, N., J.A. McSwiggen, *Catalytic RNA (ribozymes) as drugs*, Annual Reports in Medicinal Chemistry, 1995, 30:285–294.
12. Winfree E., F. Liu, L. Wenzler, N. Seeman, *Design and self-assembly of 2D DNA crystals*, Nature, 1998, 394:539–544.
13. Zucker A.M., B.D.H. Mathews, C.D.H. Turner, *Algorithms and Thermodynamics for RNA Secondary Structure Prediction: A Practical Guide*, RNA Biochem & Biotech, J. Barszewski & B.F.C. Clark, eds, Nato ASI Series, Kluwer Academic Publishers, 1999.