RNA secondary structure design

Bernd Burghardt* and Alexander K. Hartmann†
Institut für Theoretische Physik, Universität Göttingen, Friedrich-Hund-Platz 1, D-37077 Göttingen, Germany
(Received 15 September 2006; published 28 February 2007)

We consider the inverse-folding problem for RNA secondary structures: for a given (pseudo-knot-free) secondary structure we want to find a sequence that has a certain structure as its ground state. If such a sequence exists, the structure is called designable. We have implemented a branch-and-bound algorithm that is able to do an exhaustive search within the sequence space, i.e., gives an exact answer as to whether such a sequence exists. The bounds required by the branch-and-bound algorithm are calculated by a dynamic programming algorithm. We consider different alphabet sizes and an ensemble of random structures, which we want to design. We find that for two letters almost none of these structures are designable. The designability improves for the three-letter case, but still a significant fraction of structures is undesignable. This changes when we look at the natural four-letter case with two pairs of complementary bases: undesignable structures are the exception, although they still exist. Finally, we also study the relation between designability and the algorithmic complexity of the branch-and-bound algorithm. Within the ensemble of structures, a high average degree of undesignability is correlated with a long time to prove that a given structure is (un-)designable. In the four-letter case, where the designability is high everywhere, the algorithmic complexity is highest in the region of naturally occurring RNA.

DOI: 10.1103/PhysRevE.75.021920 PACS number(s): 87.15.Aa, 87.14.Gg, 87.15.Cc

I. INTRODUCTION

RNA plays an important role in the biochemistry of all living systems [1,2]. Like DNA, it is a linear chain-molecule built up from four types of bases—i.e., adenine (A), cytosine (C), guanine (G), and uracil (U). It does not only transmit pure genetic information. RNA works, e.g., as a catalyst in the ribosome. While for the former only the primary structure—i.e., the sequence of the bases—is relevant, for the latter the kind of higher order structures—i.e., secondary and tertiary structures—is essential. We mention the three following examples: (i) For successful protein synthesis three-dimensional structures of rRNA [3,4] and tRNA [5] molecules are necessary. (ii) The catalytic properties of ribozymes depend on their three-dimensional structures [6]. (iii) The function of the internal ribosome entry site (IRES) of picornaviruses which directs binding of ribosomal subunits and cellular proteins in order to accomplish translation initiation, is based on higher order structures [7].

Like in the double helix of the DNA, complementary bases within RNA molecules can build hydrogen bonds between each other. Compared to DNA, where the bonds are built between two different strands, RNA builds bonds between bases located on the same RNA strand. The *secondary structure* is the information about which bases of the strand are paired, while the spatial structure is called the *tertiary structure*. The tertiary structure is stabilized by much weaker interactions compared to the interactions which are responsible for the secondary structure. This leads to a separation of energy scales between secondary and tertiary structure, and justifies neglecting the latter in many cases to obtain a first fundamental understanding of the behavior of RNA [8]. Therefore, although the tertiary structure is often important for the functionality of a RNA, here it is sufficient to deal with the secondary structure only.

One crucial point for the calculation of the secondary structure is the energy model used: on the one hand, if one aims to get minimum structures close to the experimentally observed ones, one uses energy models that take into account many different structural elements [9–12]—e.g., hair-pin loops or bulges, each being described by a different set of experimentally obtained parameters. On the other hand, if one is interested in the qualitative behavior, one uses models as simple as possible while conserving the general behavior—e.g., in the simplest case a model which exhibits only one kind of base [13] or models where the energies depend only on the number and on the type of paired bases [14–17]. Here we will consider only models with the latter kind of interaction energy.

The standard procedure when dealing with RNA secondary structures is that one starts with a given sequence and calculates quantities like the ground-state structure into which the RNA will fold for low temperatures. In this paper we look at the inverse problem: for a given secondary structure, does a sequence exist that has the given structure as its ground state? If this is the case, we call the structure *designable*. We answer this question for different alphabet sizes—i.e., different numbers of complementary bases. As an ensemble of structures we choose a set of random structures of given length and determine the fraction of structures that is designable. In a related study, Mukhopadhyay *et al.* [18] also considered different alphabet sizes, but by using a probabilistic algorithm—i.e., approximately, they determined how many different *other* sequences have the same structure as the ground state of a given sequence. Hence, by definition, all structures encountered are designable. In contrast, we generate structures randomly from scratch, and determine whether there is at least one sequence that has this structure as a ground state. Hence we can generate structures which

*Electronic address: burghard@physik.uni-goe.de
†Electronic address: hartmann@physik.uni-goe.de

might not be designable at all. The basic idea behind this approach is that nature needs as many different structures as possible to perform many different tasks and as it turns out, four letters is the minimum number necessary for this. Furthermore, we use an exact branch-and-bound algorithm to verify (un)designability. In another earlier work Hofacker *et al.* [11] (with improvements by Ref. [19]) looked at the same question as to whether a given structure is designable. In contrast to our work, they used only a probabilistic approach, hence in some cases solutions may have been missed. Furthermore, they studied a very restricted ensemble of structures, where the structures are assembled from substructures found in nature already, which implies by definition a high degree of designability. Also they did not study the dependence on the alphabet size. Another difference between our work and previous literature is that we also study the relation between the designability and the algorithmic complexity—i.e., the running time of our exact algorithm.

The paper is organized as follows. In Sec. II, we define our model—i.e., we formally define secondary structures and introduce our energy model and state the design problem. In Sec. III, we explain how to calculate a bound for the ground state with a dynamic programming algorithm and how to solve the design problem with a branch-and-bound algorithm augmented with a randomized algorithm. We also present in detail in Sec. III C how we generate the ensemble of random structures. Finally, in Sec. IV we show the results of our numerical studies.

II. THE SECONDARY STRUCTURE MODEL AND DESIGN PROBLEM

A. RNA secondary structure model

Since RNA molecules are linear chains of bases, they can be described as a (quenched) sequence $\mathcal{R}=(r_i)_{i=1,\ldots,L}$ of bases $r_i \in \mathcal{A}$. We denote the length of the sequence by L and $\mathcal{A}$ is the alphabet, which contains the underlying base types that build up the RNA sequence. Typically $\mathcal{A}=\{\mathsf{A},\mathsf{C},\mathsf{G},\mathsf{U}\}$ is used, but we also consider here alphabets with two or three letters. Within this single stranded molecule some bases can pair and build a secondary structure. The Watson-Crick base pairs—i.e., A-U and C-G—have the strongest affinity to each other, they are also called complementary base pairs. Each base can be paired at most once. For a given sequence $\mathcal{R}$ of bases the secondary structure can be described by a set $\mathcal{S}$ of pairs (i,j) (with the convention $1 \leqslant i < j \leqslant L$), meaning that bases r_i and r_j are paired. For convenience of notation we further define a Matrix $(S_{i,j})_{i,j=1,\ldots,L}$ with $S_{i,j}=1$ if $(i,j) \in \mathcal{S}$, and $S_{i,j}=0$ otherwise. Two restrictions are used:

(1) *(Noncrossing condition)* Here we exclude so-called *pseudoknots*, that means, for any $(i,j),(i',j') \in \mathcal{S}$, either $i < j < i' < j'$ or $i < i' < j' < j$ must hold—i.e., we follow the notion of pseudoknots being more an element of the tertiary structure [20].

(2) *(Min-distance condition)* Between two paired bases a minimum distance is required $|j-i| \geqslant h_{\min}$, due to the bending rigidity of the molecule. Our main results presented below will be for $h_{\min}=2$, but for comparison we discuss the unphysical case $h_{\min}=1$ as well. Larger—and more realistic—$h_{\min}$ values do not change the qualitative results compared to the $h_{\min}=2$ case, but are computationally more demanding.

In the following we assume that each structure $\mathcal{S}$ fits to all considered sequences $\mathcal{R}$—i.e., for all pairs $(i,j) \in \mathcal{S}$ the indices i and j are smaller or equal to the length L of the sequence $(1 \leqslant i,j \leqslant L)$. By $\mathcal{S}^{m,n}$ we denote a *substructure* of S between the mth and nth letter—i.e., $\mathcal{S}^{m,n} := \{(i,j) \in \mathcal{S} | m \leqslant i < j \leqslant n\}$. Similar, a *subsequence* between the mth and nth letter is denoted by $\mathcal{R}^{m,n}=(r_i)_{i=m,\ldots,n}$.

B. Energy models

In this section we define an energy model, which gives for every secondary structure $\mathcal{S}$ and every sequence $\mathcal{R}$ an energy $E(\mathcal{S},\mathcal{R})$. For a given sequence $\mathcal{R}$ the minimum

$$E(\mathcal{R}) = \min_{\mathcal{S}} E(\mathcal{S},\mathcal{R}) \tag{1}$$

is the ground-state energy of the sequence $\mathcal{R}$.

Motivated by the observation that the secondary structure is due to the formation of many base pairs via hydrogen bonds, one assigns each pair (i,j) a certain energy $e(r_i,r_j)$ depending only on the kind of bases. The total energy is the sum over all pairs

$$E_p(\mathcal{S},\mathcal{R}) = \sum_{(i,j)\in\mathcal{S}} e(r_i,r_j), \tag{2}$$

e.g., by choosing $e(r,r')=+\infty$ for noncomplementary bases r, r', pairings of this kind are suppressed. In our numerical studies we restrict ourself to the energy model

$$e(r,r') = \begin{cases} E_p & \text{if } r \text{ and } r' \text{ are complementary bases,} \\ +\infty & \text{otherwise,} \end{cases} \tag{3}$$

with a pair energy $E_p \leqslant 0$ independent of the kind of bases.

Another possible model is to assign an energy E_s to a pair $(i,j) \in \mathcal{S}$ if also $(i+1,j-1) \in \mathcal{S}$. This *stacking energy* can be motivated by the fact that a single pairing gives some gain in the binding energy, but also reduces the entropy of the molecule, because through this additional binding it loses some flexibility. Formally, the total stacking energy of a structure can be written as

$$E_s(\mathcal{S},\mathcal{R}) = \begin{cases} \displaystyle\sum_{(i,j)\in\mathcal{S}} E_s S_{i+1,j-1} & \text{if for all } (i,j) \in \mathcal{S}\text{: } r_i,r_j \\ & \text{are complementary bases} \\ +\infty & \text{otherwise.} \end{cases} \tag{4}$$

Real RNAs cannot be described by just one energy parameter, because the free energy depends on the type and the size of the structural elements—e.g., hair pin loops. Here, we examine the sum of both models—stacking energy and pair energy—

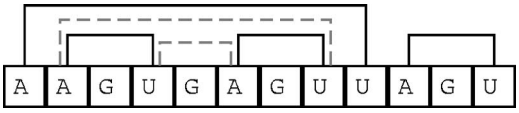

FIG. 1. In the case $E_s=0$ the structure can be easily designed—e.g., by building (A,U)-pairs for the paired bases, and assigning G to the unpaired bases. However, this is not necessarily a solution for the $E_s<0$ case: in this example two pairs could be repaired (dashed lines) giving a lower overall energy.

$$E(\mathcal{S},\mathcal{R}) := E_p(\mathcal{S},\mathcal{R}) + E_s(\mathcal{S},\mathcal{R}), \tag{5}$$

where the parameters E_s and $e(r,r')$ can be freely adjusted, including both models discussed above. For real RNA both parameters, E_p and E_s, are of the same order of magnitude, namely about $1-10$ kcal mol^{-1} [9,21,22], therefore we choose $E_p=-2$ and $E_s=-1$ in our simulations.

A sequence $\mathcal{R}$ is called *compatible* with a structure $\mathcal{S}$, if $e(r_i,r_j)\leqslant 0$ for all $(i,j)\in\mathcal{S}$.

Furthermore, we define for a structure $\mathcal{S}$ (independent of $\mathcal{R}$) the energy

$$E(\mathcal{S}) := E_{\min}|\mathcal{S}| + \sum_{(i,j)\in\mathcal{S}} E_s S_{i+1,j-1}, \tag{6}$$

with $E_{\min}=\min_{r,r'\in\mathcal{A}} e(r,r')$. For the energy model of Eq. (3) it is $E_{\min}=E_p$. Thus, $E(\mathcal{S})$ is a lower bound of $E(\mathcal{S},\mathcal{R})$ for any $\mathcal{R}$.

C. Designing RNA secondary structure

The energy model (5) has been previously studied [23], in the standard way—i.e., by calculating ground states for given sequences. In this paper we take, as already mentioned in the Introduction, a different point of view: we choose a random *structure* $\mathcal{S}$ and ask whether there exists any sequence $\mathcal{R}$ that has this structure as its ground state.

The *design problem* can be more formally stated as follows: For a given structure $\mathcal{S}$ find a sequence $\mathcal{R}$ such that $E(\mathcal{S},\mathcal{R})=E(\mathcal{R})$ holds. If such a sequence exists, the structure $\mathcal{S}$ is called designable. However, we do not require that $\mathcal{S}$ is the unique ground state of this sequence, since this issue has been addressed previously [18] for sequences which are designable (in our sense) by definition. It would also be interesting to consider a definition of designability of random structures, which involves being a ground state of a sequence and uniqueness of the ground state at the same time. Nevertheless, this would require much more computational effort, since these structures are harder to find. Hence, such a study is beyond the scope of this work and should be considered in the future.

The design problem for an energy model *without* stacking energy—i.e., which exhibits only a pair energy according to Eq. (3), can be solved easily as follows (Fig. 1): assign to any pair $(i,j)\in\mathcal{S}$ the letters A at position i and U at position j, and for every unpaired position a base of type G (in the two letter case use A again). There are exactly $|\mathcal{S}|$ pairs of bases, therefore the ground-state energy cannot be below $E_p|\mathcal{S}|$, which is just the energy of the structure $\mathcal{S}$.

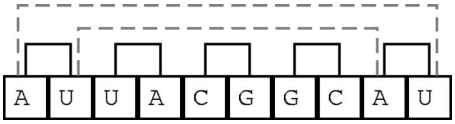

FIG. 2. In the case of $h_{\min}=1$ and $E_s<0$ this is an example of an undesignable structure. There is only a finite number of different 2-tuples (r_1,r_2). Whenever there are more than this number of neighboring pairs paired in a structure, at least two of them must be of the same kind—e.g., (A,U)—these two can be repaired (dashed lines), which leads to a gain of some stacking energy, rendering the structure undesignable.

For the case $E_s\lneqq 0$ this construction scheme might fail as one can see in the example shown in Fig. 1: regrouping of the enclosed base pairs leads to the formation of two adjacent pairs—i.e., a stack of size two. This results in an energy of the regrouped structure below the energy of the given structure, hence the given structure is not a ground state of the given sequence. Nevertheless, the structure shown in the example is in fact designable, the slightly modified sequence—positions 2 and 4 are swapped—AUGAGAGUUAGU—has the given structure as a ground state.

The case $h_{\min}=1$—i.e., neighboring bases can be paired—is of little interest: both, from the physical point of view—the RNA molecule cannot be bent into arbitrary forms—as well as from the point of view of the design problem. For an undesignable example, look at the structure sketched in Fig. 2: for any alphabet size there is only a finite number of different 2-tuples (r_1,r_2), whenever there are more than this number of neighboring pairs paired in a structure, at least two of them must be of the same kind—e.g., (A,U)—these two can be repaired, which leads to a gain of some stacking energy, rendering the structure undesignable.

III. ALGORITHMS

In principle the design problem can be solved by calculating the ground-state energy $E(\mathcal{R})$ of every compatible sequence $\mathcal{R}$ and testing whether this is equal to $E(\mathcal{S},\mathcal{R})$, but, because the number of sequences grow exponentially with the sequence size L (roughly as $|\mathcal{A}|^{L-|\mathcal{S}|}$), this is impractical.

Therefore we use a branch-and-bound algorithm, where one tries to find an upper bound $E^{\mathrm{B}}(Q) := \max_{\mathcal{R}\in Q} E(\mathcal{R})$ for the ground-state energies for a (large) set Q of sequences compatible with the structure $\mathcal{S}$. If this bound is below the energy $E(\mathcal{S})$ of the structure—i.e., $E^{\mathrm{B}}(Q)<E(\mathcal{S})$—then none of the sequences in Q can be a solution of the design problem.

Here, we consider particular sets of sequences, where at some positions all sequences of the set have the same letter (but possible different ones for the different positions), and where for all other positions all possible combinations of letters occur, which are compatible with the sequence. Hence, these positions can be described by a *joker* letter. For a more formal definition of Q, see below. In Sec. III A an algorithm is explained, which calculates an upper bound for the ground-state energy of such sequences.

This algorithm is used within the bound step of the branch-and-bound algorithm, which is explained in Sec. III B 1.

A. Calculating a bound for the ground-state energy

In this section we introduce a modification of the algorithm presented in Ref. [23] which allows us to calculate an upper bound for the ground-state energy of sequences, where some bases are still unassigned, i.e., which are represented by the joker letter.

Thus, for a formal description of the algorithm we extend the alphabet $\mathcal{A}$ by the joker-letter $*$, where $*$ represents any letter in the original alphabet. Note that $*$ is complementary to any $r \in \mathcal{A}$. The new alphabet is denoted by $\mathcal{A}^* := \mathcal{A} \cup \{*\}$. Sequences $\mathcal{R}^* = (r_i^*)_{i=1,\ldots,L}$, $r_*^i \in \mathcal{A}^*$, over this extended alphabet $\mathcal{A}^*$—we call $\mathcal{R}^*$ a *generalized sequence*—represent a set Q of sequences over the original $\mathcal{A}$: $Q = \{(r_i)_{i=1,\ldots,L} | r_i \in \mathcal{A}, r_i = r_*^i \text{ if } r_*^i \in \mathcal{A}\}$. For a given structure $\mathcal{S}$ and a generalized sequence $\mathcal{R}^*$, the scheme explained in the following can be used to calculate a bound for the ground-state energy. Note that for a sequence without a $*$-letter this bound is equal to the ground-state energy.

We start the explanation of the algorithm by considering the contribution to the bound arising from a single pair (i,j). If the letters in the sequence are fixed—i.e., $r_i, r_j \in \mathcal{A}$—then the energy contribution is simply $e(r_i, r_j)$, since there is no choice. If at least one of the two letters is the joker letter $*$, then we have different choices. First, if $(i,j) \in \mathcal{S}$, then the energy contribution must be negative, because otherwise, since we are considering ground states, bases i and j would not be paired leading to an energy contribution zero. On the other hand, we are looking for a maximum over all sequences described by the generalized $\mathcal{R}^*$, hence we have to take the maximum over all possible negative contributions, either over all possible combinations of two letters (two $*$ symbols), or, over all possible letters at the one position with a $*$ symbol. Second, if $(i,j) \notin \mathcal{S}$, then the energy contribution should be positive if bases i,j get paired nevertheless. This ensures that within the ground-state calculation, automatically the case is selected where bases i,j are not paired. We assume that for all possible cases with one or two $*$ symbols, combinations of letters are always available, such that the pair energy is positive. Since in this case, the ground-state requirement will automatically disregard the pair (i,j), instead of maximizing over all energies, we can simply assume the energy contribution $+\infty$ here. This leads to the energy contribution $e^*_{\mathcal{R}^*,\mathcal{S}}(i,j)$ for a pair (i,j) which depends on the given generalized sequence $\mathcal{R}^*$ and the given structure $\mathcal{S}$,

$$e^*_{\mathcal{R}^*,\mathcal{S}}(i,j) = \begin{cases} e(r_i,r_j) & \text{if } r_i,r_j \in \mathcal{A} \wedge |i-j| \geqslant h_{\min}, \\ E^{*,*}_{\max} & \text{if } r_i = *, r_j = *, (i,j) \in \mathcal{S}, \\ E^{r_i,*}_{\max} & \text{if } r_i \in \mathcal{A}, r_j = *, (i,j) \in \mathcal{S}, \\ E^{*,r_j}_{\max} & \text{if } r_i = *, r_j \in \mathcal{A}, (i,j) \in \mathcal{S}, \\ +\infty & \text{else} \end{cases} \tag{7}$$

with the largest possible negative pair energies

$$E^{*,*}_{\max} := \max\{e(r,r') < 0 | r,r' \in \mathcal{A}\},$$

$$E^{r,*}_{\max} := \max\{e(r,r') < 0 | r' \in \mathcal{A}\},$$

$$E^{*,r'}_{\max} := \max\{e(r,r') < 0 | r \in \mathcal{A}\} \tag{8}$$

and for the maximum of the empty set: $\max \emptyset := -\infty$. For alphabets, where each base has a complementary base—e.g., the two- and four-letter cases discussed below—with the energy $e(r,r')$ from Eq. (3) $e^*_{\mathcal{R}^*,\mathcal{S}}$ has the form

$$e^*_{\mathcal{R}^*,\mathcal{S}}(i,j) = \begin{cases} e(r_i,r_j) & \text{if } r_i,r_j \in \mathcal{A}, \\ E_p & \text{if } r_i = * \vee r_j = *, (i,j) \in \mathcal{S}, \\ +\infty & \text{else}. \end{cases} \tag{9}$$

For alphabets with letters that have no complementary counterpart—e.g., letter G, in the three-letter alphabet of Sec. IV B—the sets in Eq. (8) might be empty leading to an energy contribution $-\infty$—i.e., resulting in an upper bound $E^{\mathrm{B}}(\mathcal{R}^*) = -\infty$. In our implementation of the algorithm we do not consider (generalized) sequences, where at a position of a paired base such a letter appears, because this would lead to noncompatible sequences, i.e., sequences which are not compatible with the given structure. Note that for the case that also the pair $(i-1, j+1)$ is present, additionally to $e^*_{\mathcal{R}^*,\mathcal{S}}(i,j)$ a stacking-energy contribution E_s arises. This is handled by the following recursive equations, which perform the ground-state calculation. They are slightly modified compared to Ref. [23]. We denote by $N_{i,j}$ the maximum ground-state energy over the set of compatible subsequences given by the generalized subsequence $r_i^*, r_{i+1}^*, \ldots, r_{j-1}^*, r_j^*$. $\hat{N}_{i,j}$ is defined in the same way, only that additionally it is assumed that letters r_{i-1}^* and r_{j+1}^* are paired, which leads simply to an additional stacking-energy contribution. The basic idea is that for the ground state of subsequence $r_i^*, \ldots, r_j^*$ either the last letter j is not paired, or it is paired to another letter $k \in \{i, i+1, \ldots, j-1\}$ [the requirement $j - i \geqslant h_{\min}$ is treated through suitable choices of the energy $e^*_{\mathcal{R}^*,\mathcal{S}}(i,j)$]. The ground state is the minimum over all these cases, where in each case, due to the exclusion of pseudoknots, the ground-state calculation decomposed into the calculation for shorter subsequences. The recursion equations for $N_{i,j}$ and $\hat{N}_{i,j}$ read as follows:

$$N_{i,j} = \min\{N_{i,j-1}, \min_{k=i}^{j-1}[N_{i,k-1} + e^*_{\mathcal{R}^*,\mathcal{S}}(k,j) + \hat{N}_{k+1,j-1}]\}$$

$$\text{for } j - i > 0,$$

$$\hat{N}_{i,j} = \min\{N_{i,j-1}, e^*_{\mathcal{R}^*,\mathcal{S}}(i,j) + E_s + \hat{N}_{i+1,j-1}, \min_{k=i+1}^{j-1}[N_{i,k-1} + e^*_{\mathcal{R}^*,\mathcal{S}}(k,j) + \hat{N}_{k+1,j-1}]\} \quad \text{for } j - i > 0,$$

$$N_{i,j} = \hat{N}_{i,j} = 0 \quad \text{for } j - i \leqslant 0. \tag{10}$$

The values of $N_{i,j}$ and $\hat{N}_{i,j}$ are calculated "bottom up"—i.e., in a *dynamic programming* fashion, starting at small

values of $j-i$ until one arrives at $j-i=L-1$. The wanted bound is $E^B(\mathcal{R}^*)=N_{1,L}$, and within our energy model this bound is never larger than $E(\mathcal{S})$. In general, $N_{i,j}$ is the bound for the ground-state energy of the subsequence $(r_k^*)_{k=i,\ldots,j}$.

It is worthwhile to note that it is not necessary to recalculate the whole matrix $(N_{i,j})_{1\leqslant i\leqslant j\leqslant L}$ if only one letter in $\mathcal{R}^*$ has been changed—e.g., if base r_k has been modified this only influences subsequences which contain this base—therefore it is sufficient to recalculate all $N_{i,j}$ and $\hat{N}_{i,j}$ with $i\leqslant k\leqslant j$. This reduces the numerical effort for calculating $N_{1,L}$, but it is still of order $O(L^3)$.

B. Algorithms for solving the design problem

In this section we describe two algorithms which we used to solve the design problem stated above. The first one is deterministic—i.e., it guarantees to either successfully find a solution or to prove that no solution exists. In general, the algorithm has to consider exponentially many sequences (in the length L). In the case that the problem has a solution, a randomized algorithm is often faster in finding a solution, therefore we also implemented such an algorithm [19,24], and combined both algorithms.

1. Branch-and-bound algorithm

Our deterministic algorithm follows the branch-and-bound approach (e.g., in Ref. [25], pp. 499 or in Ref. [26] Chap. 12). Here, it finds a sequence $\mathcal{R}$—if such a sequence exists—that has the structure $\mathcal{S}$ as one ground state.

The idea of the algorithm is that it constructs a tree, where each node represents a generalized sequence $\mathcal{R}^*$—i.e., a set Q of sequences—and all children of a node represent a partition of Q. The root node stands for the set of all sequences of length L—i.e., which is described by the generalized sequence $(r_i^*)_{i=1,\ldots,L}, r_i^*=*$. For every node (r_i^*) in the tree with at least one $r_j^*=*$, its children are constructed by replacing r_j^* with one letter from $\mathcal{A}$. Sequences with no $*$-letters are the leaf nodes of the tree (sets containing exactly one element/sequence).

In Fig. 3 a pseudocode of the algorithm is shown. There, $\mathcal{T}$ contains all nodes of the tree which have not been treated yet. Initially $\mathcal{T}$ contains only the root node. New nodes are generated from existing nodes by selecting a node—i.e., a generalized sequence—selecting one position where a $*$ appears, and generating $|\mathcal{A}|$ new nodes by replacing this $*$ by all possible letters $\alpha\in\mathcal{A}$. In this way the algorithm traverses the tree from the root towards the leaves, calculating an upper bound of the ground-state energies of the sequences represented by this node. Within the algorithm, two functions appear, GROUND-STATE ENERGY $(\mathcal{R}_\alpha^*)$ and GROUND-STATE BOUND $(\mathcal{R}_\alpha^*)$, which essentially use Eq. (10) to calculate the ground-state energy and the upper bound for it, respectively. The former function called in the case $\mathcal{R}_\alpha^*$ does not contain the $*$ symbol. In the case $\mathcal{R}_\alpha^*$ contains the $*$ symbol, if the upper bound obtained by the latter function is below the energy $E(\mathcal{S})$ of the structure $\mathcal{S}$, none of the sequences represented by this node has this structure as a ground state, and the descent towards the children of this node can be stopped

```
1:  bound ← STRUCTURE-ENERGY(S) {see Eq. (6)}
2:  insert R* = (r_i = *)_{i=1...L} to T
3:  while T ≠ ∅ do
4:      select R* = (r_i) ∈ T and delete R from T
5:      select one i ∈ [1...L] with r_i = *
6:      for all α ∈ A do {Branch step}
7:          generate R*_α from R* by replacing r_i by α
8:      end for
9:      for all α ∈ A do {Bound step}
10:         if |R*_α| = 1 and
                GROUND-STATE-ENERGY(R*_α) = E(S, R)
            then
11:             return R*_α {solution found}
12:         else if GROUND-STATE-BOUND(R*_α) ≥ bound
            then
13:             insert R*_α to T
14:         else {Bound for ground state smaller than bound}
15:             do nothing (optionally do something)
16:         end if
17:     end for
18: end while
19: return nil {no solution exists}
```

FIG. 3. Pseudocode of the branch-and-bound algorithm. In line 15 the algorithm can be augmented, e.g., with a randomized algorithm—see Sec. III B 2.

here: the algorithm ignores this node by not setting it into $\mathcal{T}$. On the other hand, if a leaf node is reached and its ground-state energy is equal to the energy of the structure, a solution is found and the algorithm terminates successfully.

The selection steps in lines 4 and 5 require further explanation: we use a stacklike data structure to implement $\mathcal{T}$, so the last inserted sequence in line 13 is used here first (depth-first search). The selection step of a joker-letter in line 5 is more difficult: we tried some strategies in which the next inserted base can be chosen. All these strategies were static ones, that means the order of insertion was chosen based on the concrete structure given, but the order was fixed before starting with the algorithm. At the end we found the following strategy to be the best [33]: We first insert paired bases, and we choose the base pair (i,j) first that encloses other bases the most—i.e., $\mathcal{S}^{i,j}$ is the largest substructure of any $(i,j)\in\mathcal{S}$. The procedure continues with the substructure $\mathcal{S}^{i+1,j-1}$, if it is not empty, or continues with a pair $(i',j')\in\mathcal{S}^{i+1,j-1}$ enclosing the next largest substructure. At the end we insert the unpaired bases.

2. Randomized steepest-descent algorithm

We furthermore have implemented a randomized algorithm for finding a solution of the design problem for a given structure $\mathcal{S}$ similar to Ref. [24]. Note that in Ref. [19] a much more sophisticated method is explained, which we do not need here, since we use a combination of the complete branch-and-bound approach with the randomized algorithm. We start with a compatible sequence—e.g., every pair of the structure is assigned a A-U pair and all unpaired bases are assigned to G (again A if the alphabet contains only two letters). Either this already solves the design problem or we modify the sequence at one position as follows: for the given sequence we calculate a ground-state structure $\mathcal{S}_0$, then we choose a pair $\wp$, which is in exactly one of the structures $\mathcal{S}$ and $\mathcal{S}_0$ (i.e., $\wp\in\mathcal{S}\Delta\mathcal{S}_0$) and randomly modify one of these

two bases. If $\wp \in \mathcal{S}$ we keep the other base complementary. We accept this step, if the ground-state energy is not below that of the previous sequence. The procedure is repeated until a sequence is found that solves the design problem, or until a certain number of random steps has been executed, in this case, the algorithm stops unsuccessfully.

Of course, this method can never prove that a certain structure is undesignable. However, we have combined this strategy with the branch-and-bound algorithm above: whenever a rejection step takes place—i.e., the condition in line 14 of algorithm in Fig. 3 is reached—one random step with an independently stored sequence is done. This can be quite efficient in the designable case, because on average it requires much less steps than the deterministic branch-and-bound algorithm. On the other hand it doubles the efforts in the undesignable case. This pays off in particular for the four-letter case discussed in Sec. IV C, because there almost all structures are designable. Especially, for design times much larger than the sequence length—i.e., $T \gtrsim 10L$—the random method is almost always faster than the deterministic algorithm. This is different for the two- and three-letter case, where the deterministic algorithm requires less steps.

C. Generating random secondary structures

Below we examine the designability of randomly generated secondary structures for a given sequence length L. We parametrize this ensemble of random structures by the probability p that a certain base in the sequence is paired (for rRNA p is typical in the range 0.6–0.8 [27]). We construct each sample in two steps: First, we draw the number of pairs P of the structure from a binomial distribution between 0 and $\lfloor L/2 \rfloor$ centered at $pL/2$. Then, among all possible structures of length L having P pairs, we select one randomly, such that each structure has the same probability of being chosen. To achieve this, we have to perform a preprocessing step:

In the preprocessing step, we calculate the number $S(P,L)$ of possible structures of a sequence of length L and with P pairs. The number $S(P,L)$ is the number of possible structures $S(P,L-1)$ of the smaller sequence plus the number of possible structures, where base L is paired with base $L-k$, for all feasible values of k. Hence, the value $S(P,L)$ can be calculated by the following recursion relation [28]:

$$S(P,L) = S(P,L-1) + \sum_{k=h_{\min}}^{L-1} \sum_{q=0}^{P-1} S(q,k-1)S(P-q-1,L-k-1),$$

$$S(P=0,L)=1, \quad S(P<0,L)=S(P>L/2,L)=0. \quad (11)$$

The first sum is over all possible distances between these two bases; the second sum is over the number of pairs enclosed by the pair $(L-k,L)$. The product is the number of possible structures having q pairs enclosed by $(L-k,L)$ and the remaining $P-q-1$ pairs in the range from 1 to $L-k-1$. The construction of the matrix $S(P,L)$ requires $O(L^4)$ calculation steps, but this is required only once for all lengths up to a maximum length L. Note that for $h_{\min}=1$ the number of structures can be calculated explicitly

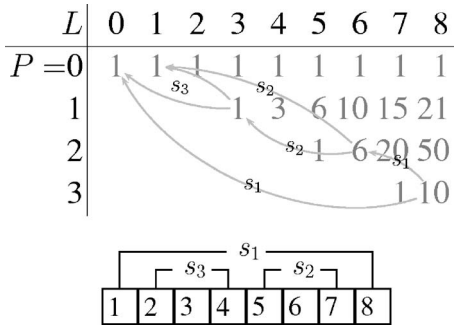

FIG. 4. Example of the structure generation $h_{\min}=2$. Construction of a random structure with $P=3$ and $L=8$. The way of constructing a (random) structure from this is indicated in the table by the arrows. There are 10 possibilities to construct a structure of length 8 with three pairs of bases. In step s_1 we choose to link base 1 and 8, which leaves a structure of length 6 with two pairs enclosed and a (trivial) structure of length 0 outside this pair. In step s_2 we choose base 5 and 7 to be paired, leaving a trivial structure of length 1 enclosed and structure of length 3 with one pair outside. For the latter there is only one choice, namely to connect base 2 and 4 (step s_3). The resulting structure is shown in the figure.

$$S(P,L) = \frac{1}{P+1}\binom{2P}{P}\binom{L}{2P}.$$

.

Now, for each sample to be generated, where the number P of pairs has been randomly chosen as explained above, the actual structure is selected in the following way. First, note that depending on $h_{\min}$ there are values of P and L, where no structures exist, i.e., $S(P,L)=0$, these cases are rejected immediately. Otherwise, the random structure is constructed with a backtracing algorithm: starting at $S(P,L)$, we choose one of the summands in (11) with a probability proportional to its value, then we insert the corresponding pair to the structure and recur into the subsequences. As an example we show the random generation of a structure of length $L=8$ with $P=3$ pairs (see Fig. 4). It is

$$S(3,8) = S(3,7) + S(0,1)S(2,5) + S(1,3)S(1,3) + S(2,5)S(0,1) + S(2,6)S(0,0).$$

Each of the summands represents a possible pairing of base number 8 with another base—with the exception of the first summand, which counts the number of possible structures, where base number 8 is not paired at all. Let us assume that by chance the last summand was chosen within the random selection, meaning that base 8 is paired with base 1. Leaving two pairs which must be distributed between the bases from 2 to 7; here we assume that base 7 with base 5 is then paired, leaving only one possibility for the remaining pair, namely base 4 paired with base 2.

Finally, note that the average number of structures available for p and L is given by

$$s(p,L) = \sum_{P=0}^{\lfloor L/2 \rfloor} \binom{\lfloor L/2 \rfloor}{P} p^P (1-p)^{\lfloor L/2 \rfloor - P} S(P,L). \quad (12)$$

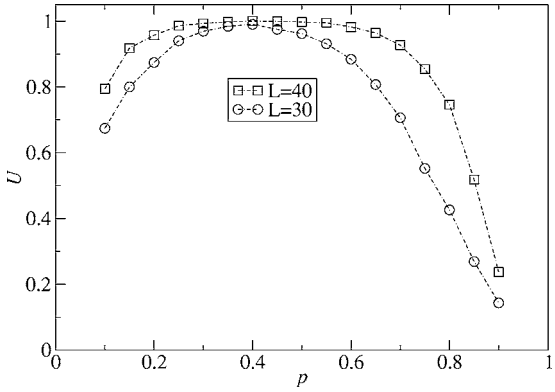

FIG. 5. The undesignability U of random structures for an underlying two-letter alphabet is shown as a function of the probability p that a base is paired. Even for small sequences and low probabilities of bases being paired, almost all structures are undesignable. Missing error bars are of the size of the symbols or smaller, and omitted for legibility. (Parameter used: $E_p=-2$, $E_s=-1$, $h_{\min}=2$, 1000 samples.)

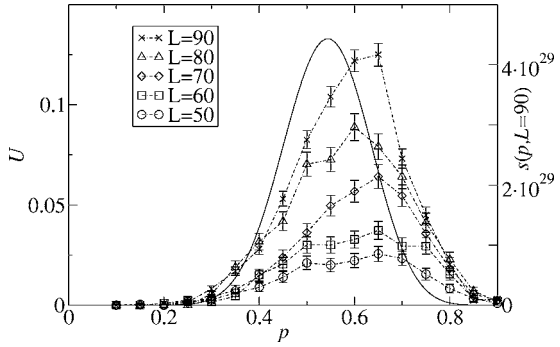

FIG. 6. The undesignability U of random structures for an underlying three-letter alphabet is shown as a function of the probability p that a base is paired. In comparison to the two-letter case (Fig. 5) many more structures are designable, but still a reasonable fraction of structures is undesignable. In light gray the average number $s(p,L=90)$ of structures of length 90 is shown [see Eq. (12)]: The maximum of this curve is at smaller p value than the maximum of $U(p,L=90)$. (Parameter used, $E_p=-2$, $E_s=-1$, $h_{\min}=2$, 1000 samples.)

IV. NUMERICAL RESULTS

For an ensemble of randomly chosen structures of given sequence length L, we examined whether these structures are designable or not. We used different alphabets with two, three, and four letters. All calculations for the results presented below were performed with the parameters $E_p=-2$, $E_s=-1$, and $h_{\min}=2$. Note that increasing the stacking energy E_s in comparison to the pair energy makes the design problem more difficult: in the limit $E_s\to-\infty$ it would be favorable to remove all nonstacked pairs from the structure, if this allows only one additional stacked pair. Considering the minimum distance $h_{\min}$ between two paired bases of natural RNA, it seems to be more appropriate to use a larger value for $h_{\min}$—e.g., $h_{\min}=5$ would be more appropriate—but this increases the computational effort without changing the qualitative results: only $h_{\min}=1$ has a different qualitative behavior (see Fig. 2).

A. Two-letter alphabet

The alphabet consists of two complementary letters, e.g., A and U, only. In Fig. 5 the fraction U of the undesignable structures is shown as a function of the probability p that a base is paired. For small p the fraction U for all lengths L increases quickly with growing p from small values to its maximum possible value close to one. Thus, in particular for moderate RNA lengths $L\approx100$, almost no structure is designable. For structures where many bases are paired, only a quite restricted class of structures is possible—i.e., structures with many nested base pairs, which obviously have a high probability to be designable. For this reason the undesignability U decreases again for larger p.

For fixed p values the value of U increases with the sequence length L, which seems to be plausible because, if a structure of small length is undesignable, a larger structure containing this structure are also undesignable.

We conclude that two letters do not suffice to provide a large variety of secondary structures needed in nature to perform the large number of required RNA functions.

B. Three-letter alphabet

The alphabet consists of two complementary letters—e.g., A and U—and one additional letter—e.g., C—not complementary to any other letter. As one can see from Fig. 6 compared to the two-letter case a larger amount of structures is designable, but with larger sequence lengths still a larger fraction becomes undesignable.

We also looked at the "time" T required to find a solution—if any exists for the fully deterministic branch-and-bound algorithm. By "time" we mean here, how often either of the two functions GROUND-STATE ENERGY ($\mathcal{R}^*_\alpha$) and GROUND-STATE BOUND ($\mathcal{R}^*_\alpha$) (see Fig. 3) is called. Since these two functions are called at least L-times, T is at least $O(L)$. In Fig. 7 the average of $\ln(T/L)$ is shown as a function of p. Because $T\geqslant L$ a value close to zero indicates that a solution is found (on the average) almost immediately.

The maxima of these curves are almost at the positions as in Fig. 6, meaning that for values of p, where a large fraction of the structures is undesignable, it is difficult—i.e., it requires many steps—to find a solution for the designable structures. The structures which are not designable behave a bit differently, cf. Fig. 8. There the time needed to prove that no design is possible increases monotonously with p, and is much larger than the time needed to find a solution in the designable cases. Nevertheless, the total running time of the branch-and-bound algorithm is mostly determined by the designable case, hence we observe a peak close to $p=0.6$ as well, see lower curve in Fig. 8. This behavior of the running time is similar to the behavior found for suitable random ensembles of classical combinatorial optimization problems [29,30], as observed for the satisfactorily problem [31] or the vertex-cover problem [32]. Also in these and other cases, the

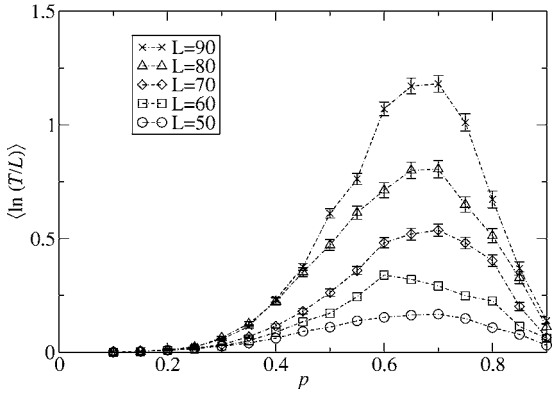

FIG. 7. For the three-letter alphabet the design time T for designable structures is shown as a function of the pairing probability p. The positions of the maxima are at similar positions as the corresponding maxima in Fig. 6. Missing error bars are of the size of the symbols or smaller, and omitted for legibility. (Parameter used, $E_p=-2$, $E_s=-1$, $h_{\min}=2$, 1000 samples.)

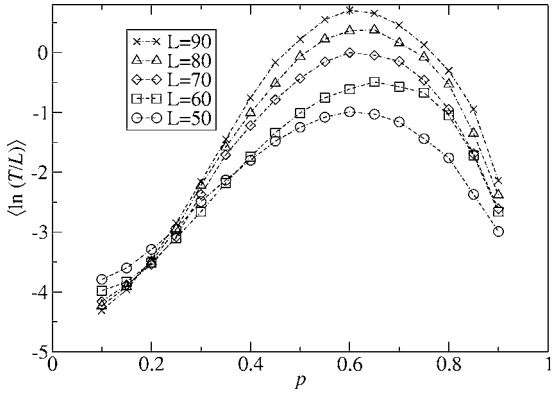

FIG. 9. For the four-letter alphabet the design time T for designable structures is shown as a function of the pairing probability p. The positions of the maxima are at similar positions as the corresponding maxima in Fig. 7. Missing error bars are of the size of the symbols or smaller, and omitted for legibility. (Parameter used, $E_p=-2$, $E_s=-1$, $h_{\min}=2$, 1000 samples.)

running time of exact algorithms similar to branch-and-bound increases strongly when the average number of unsolvable random instances increases. The only difference to the present case is that for these classical optimization problems in the limit of diverging system sizes, phase transitions between solvable and unsolvable phases can be observed. In the case of RNA secondary structures, we are interested only in finite lengths, because in nature finite (rather short) RNA sequences dominate anyway.

Finally, we also want to mention that the maximum of the average number of structures $s(p,L)$—as shown in Fig. 6—is at a slightly smaller $p \approx 0.54$ than the maxima of $U(p)$ and $\langle \ln T/L \rangle$. Hence, in contrast to the two-letter case, there is at least a window of p values, where a large number of designable structures exist. On the other hand, in the range $p \in [0.6, \ldots, 0.8]$, where most of the wild-type RNA can be found, the number of designable structures is still small. Especially for sequence lengths $L \gtrsim 1000$ we expect that again most structures are undesignable. Hence, three letters seem also not to be sufficient.

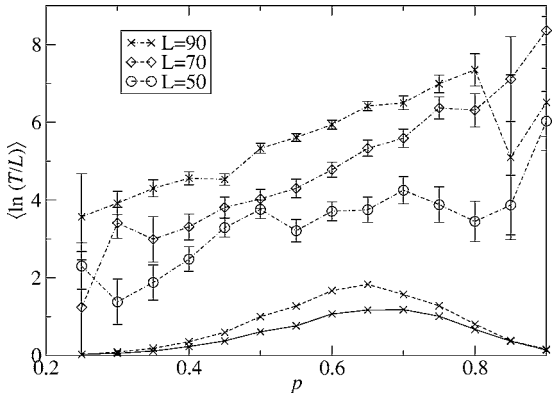

FIG. 8. The average time for the undesignable structures required to verify the undesignability is shown for three different lengths (symbols with error bars) for the three-letter case. For comparison in the lower part of the figure the average time ($L=90$) for designable structures (solid curve; cf. Fig. 7) and the average time to prove either designability or undesignability (dashed curve) are shown. In general the higher the pair probability p is the more difficult it becomes to prove the undesignability. Furthermore one can see that it is much more difficult to prove undesignability than to find a solution in the designable case. For probabilities p below 0.3 or above 0.8 only a few structures are undesignable and the corresponding error bars become large—i.e., more samples are required to get better results in this regime. (Parameter used, $E_p=-2$, $E_s=-1$, $h_{\min}=2$, 1000 samples.)

C. Four-letter alphabet

The alphabet consists of two pairs of complementary letters—e.g., A, U and C, G. In this case we observe that for all lengths up to $L=90$ the undesignability U is essentially zero—i.e., so far we have not found any random structure that is undesignable. This means that four letters are sufficient, at least for moderate system lengths, to design all possible structures that may be needed in cell processes. Nevertheless, as shown in Sec. IV D, structures exist that are undesignable even in the four-letter case, but such structures must be quite rare for lengths up to $L=90$. This means that in the limit of infinite RNA lengths, which is only of abstract academic interest, almost all *random* structures become undesignable, because the probability that somewhere in the infinite sequence there is an undesignable subsequence of finite length, is exactly one, as explained in the next section. Since, as already pointed out above, naturally occurring RNA must be only of rather restricted lengths to perform their functions, this effect has no influence and a four-letter alphabet seems to be sufficient.

In Fig. 9 we show the average "time" T to find a solution as a function of p, but here we used a combined deterministic-randomized algorithm, which is quite fast for low pairing probabilities—i.e., $p<0.4$—where on the average less than L ground-state calculations are necessary to find a solution. On the other hand, for values $p \approx 0.6$ the design time T seems to grow faster than exponentially in the sequence length L. This strong increase of the running time

FIG. 10. Principle of a nondesignable structure. Structures consisting of a repeated pattern of simple paired bases become undesignable, if this pattern is repeated often enough. For results see Table I.

is *not* accompanied by an increase of the undesignability U, at least not on the length scales we can access with the algorithm, since we do not find any undesignable structures in this range. This is different from the three-letter case and from the classical optimization problems cited above. Nevertheless, it is striking that the structures which are hardest to design are close to the region $p \in [0.6, \ldots, 0.8]$, where the naturally occurring RNA secondary structures can be found. Furthermore, this strong increase of the running times means that one *cannot* use the randomized algorithm to look quickly for probably undesignable structures in the four-letter case: one cannot just stop searching after a search time which only increases polynomially with the sequence lengths, because in this case one would even miss the designable structures. Hence, longer RNA—i.e., random RNA which are not designable—currently seem out of reach.

D. Discussion

While in the two-letter case a large fraction of random structures is not designable, only a small fraction of them is undesignable when using a three-letter alphabet. In the four-letter case designability seems to dominate the structure space by far: in fact, so far we have not found any random structure which is undesignable for the given parameter ($E_s=-1$, $E_p=-2$, $h_{\min}=2$). This leads to the question as to whether there are any undesignable structures at all.

Indeed, there are such structures (see Fig. 10): for a given length L build a non-nested structure by the pairs $[(h_{\min}+1)n+1, (h_{\min}+1)(n+1)]$ with $n=0,1,2,\ldots$ and $(h_{\min}+1)\times(n+1)\leqslant L$. Such structures are examples for chains: a *chain* $\mathcal{C}$ of length l is a set of pairs $\mathcal{C}=\{(i_1,j_1),(i_2,j_2),\ldots,(i_l,j_l)\}$ with the property $j_n+1=i_{n+1}$ for $n=1,\ldots,l-1$. A chain $\mathcal{C}$ which is a subset of a structure $\mathcal{S}$, i.e., $\mathcal{C}\subset\mathcal{S}$, is called a *subchain* of $\mathcal{S}$. Chains of large enough lengths—e.g., the structure sketched in Fig. 10—are undesignable for a similar reason which makes the structure shown in Fig. 2 undesignable (with $h_{\min}=1$): there are only many finite possible combinations of bases being paired, such that after a while a repetition occurs. Nevertheless, the argument is more complex here and we do not go into details. We only show in Table I the minimum length of structures sketched in Fig. 10 for which these become undesignable for different $h_{\min}$ and the corresponding running times of the branch-and-bound algorithm.

This implies that structures $\mathcal{S}$ which contain a subchain $\mathcal{C}$ of length $l\geqslant 16$ are also undesignable. In the limit $L\to\infty$ with pair probability $p>0$ we expect that almost all random structures contain a subchain of size $l\geqslant 16$, thus making these structures undesignable. However, for native RNA this limit is not relevant: For an ensemble of 10.000 random structures of length $L=1024$ and pair probability $p=0.7$ we looked for each structure for the subchain of the longest length l and found none longer than 11. Assuming that all undesignable structures in the four-letter case are undesignable because they contain a subchain longer than $l=15$, such structures are very rare even for biological lengths.

TABLE I. The minimum length of structures according to Fig. 10 that are undesignable. In the last column the time T required to prove the undesignability with the branch-and-bound algorithm is shown.

$h_{\min}$	L	pairs	T
2	48	16	6×10^7
3	60	15	5×10^8
4	75	15	2×10^9

Finally, we want to mention briefly the five-letter case: two pairs of two complementary bases (A-U, C-G) and an unpairable fifth letter (e.g., X). In this case it is easy to see that even structures as explained in Fig. 10 are designable: Start with a sequence of type ACUGACUGACUGACUG…, replace the bases at positions 2,5,8,… with $h_{\min}-1$ letters of type X, e.g., yielding in the case $h_{\min}=2$: AXUGXCUXACXGAXUG…·. First, in this sequence stacked pairs are impossible, because for non-pair $r_i r_{i+1}$ there is a required complementary pair $\bar{r}_{i+1}\bar{r}_i$. Further, this sequence is compatible with the structure and there are exactly as many complementary bases paired as there are pairs in the structure. Of course, this does not prove that with five letters all structures are designable, but undesignable structures are at least expected to be even much less frequent than in the four-letter case.

V. SUMMARY

We numerically investigated the RNA secondary-structure design problem for different alphabet sizes. We used a deterministic branch-and-bound algorithm to get exact answers as to whether a given structure is designable or not. Due to efficiency reasons in the designable cases, we combined this algorithm with a probabilistic one, gaining significant performance improvements in the four-letter case.

We examined the designability for an ensemble of random structures as a function of the probability that a base of a sequence is paired. Our findings for the two-letter case are that it is almost impossible to design most of the structures. In the three-letter case already for small sequence sizes ($L\approx 90$) about 10% of the structures are undesignable for biological relevant pairing probabilities, leading to the conclusion that for biological sequence sizes ($L\approx 1000$) again most structures are undesignable.

Interestingly, this changes when going to the (natural) four-letter alphabet: within our study we have not found a single random structure that we could prove to be undesignable. Although there are structures that are undesignable, they occur with very low frequencies.

We further studied the computational time required to design a structure. Although this surely depends strongly on the algorithm, we found in the three-letter case that the required time is maximal in the regime where the undesignability is largest. In the four-letter case the design times look similar to those of the three-letter case: again we observed a maximum of the design times for $p \approx 0.6$, close to the region where naturally occurring RNA can be found. Although (almost) all structures are designable, it is sometimes difficult to design them.

ACKNOWLEDGMENTS

The authors thank M. Jungsbluth for helpful remarks and I. A. Campbell for critically reading the paper. The authors have obtained financial support from the *Volkswagenstiftung* (Germany) within the program "Nachwuchsgruppen an Universitäten." The simulations were performed at the Paderborn Center for Parallel Computing in Germany and on a workstation cluster at the Institut für Theoretische Physik, Universität Göttingen, Germany.

[1] *The RNA World*, 2nd ed., edited by R. F. Gesteland, T. R. Cech, and J. F. Atkins (Cold Spring Harbor Laboratory, New York, 1999).
[2] P. G. Higgs, Q. Rev. Biophys. **33**, 199 (2000).
[3] H. F. Noller, Annu. Rev. Biochem. **53**, 119 (1984).
[4] R. Green and H. F. Noller, Annu. Rev. Biochem. **66**, 679 (1997).
[5] S. H. Kim, F. L. Suddath, G. J. Quigley, A. McPherson, J. L. Susman, A. M. J. Wang, N. C. Seeman, and A. Rich, Science **185**, 435 (1974).
[6] K. Kruger, P. J. Grabowski, A. J. Zaug, J. Sands, D. E. Gottschling, and T. R. Cech, Cell **31**, 147 (1982).
[7] J. Schmidt-Brauns, Acta Virol. **47**, 65 (2003).
[8] R. Bundschuh and T. Hwa, Phys. Rev. Lett. **83**, 1479 (1999).
[9] M. Zuker, Science **244**, 48 (1989).
[10] J. S. McCaskill, Biopolymers **29**, 1105 (1990).
[11] I. L. Hofacker, W. Fontana, P. F. Stadler, L. S. Bonhoeffer, M. Tacker, and P. Schuster, Monatsh. Chem. **125**, 167 (1994).
[12] R. Lyngsø, M. Zuker, and C. N. S. Pedersen, Bioinformatics **15**, 440 (1999).
[13] T. Liu and R. Bundschuh, Phys. Rev. E **69**, 061912 (2004).
[14] P. G. Higgs, Phys. Rev. Lett. **76**, 704 (1996).
[15] R. Bundschuh and T. Hwa, Phys. Rev. E **65**, 031903 (2002).
[16] E. Marinari, A. Pagnani, and F. Ricci-Tersenghi, Phys. Rev. E **65**, 041919 (2002).
[17] A. Pagnani, G. Parisi, and F. Ricci-Tersenghi, Phys. Rev. Lett. **84**, 2026 (2000).
[18] R. Mukhopadhyay, E. Emberly, C. Tang, and N. S. Wingreen, Phys. Rev. E **68**, 041904 (2003).
[19] M. Andronescu, A. P. Fejes, F. Hutter, H. H. Hoos, and A. Condon, J. Mol. Biol. **336**, 607 (2004).
[20] I. Tinoco, Jr. and C. Bustamante, J. Mol. Biol. **293**, 271 (1999).
[21] M. E. Burkard, D. H. Turner, and J. Tinoco Ignacio, *The RNA World*, 2nd ed. (Cold Spring Harbor Laboratory, New York, 1999), Chap. The interactions that shape RNA structure, pp. 233–264.
[22] P. G. Higgs, J. Phys. I **3**, 43 (1993).
[23] B. Burghardt and A. K. Hartmann, Phys. Rev. E **71**, 021913 (2005).
[24] P. Schuster, W. Fontana, P. F. Stadler, and I. L. Hofacker, Proc. R. Soc. London, Ser. B **255**, 279 (1994).
[25] B. Korte and J. Vygen, *Algorithms and Combinatorics*, 2nd. ed. (Springer, Berlin, 2002), Vol. 21.
[26] A. K. Hartmann and H. Rieger, *Optimization Algorithms in Physics* (Wiley-VCH, Weinheim, 2001).
[27] J. J. Gillespie, M. J. Yoder, and R. A. Wharton, J. Mol. Evol. **61**, 114 (2005).
[28] I. L. Hofacker, P. Schuster, and P. F. Stadler, Discrete Appl. Math. **88**, 207 (1998).
[29] A. K. Hartmann and M. Weigt, *Phase Transitions in Combinatorial Optimization Problems* (Wiley-VCH, Berlin, 2005).
[30] M. R. Garey and D. S. Johnson, *Computers and Intractability* (W. H. Freeman and Company, San Francisco, 1979).
[31] D. Mitchell, B. Selman, and H. Levesque, *Proceedings of the 10th National Conference on Artificial Intelligence (AAAI'92)* (AAAI Press/MIT Press, Menlo Park, CA, 1992), pp. 440–446.
[32] M. Weigt and A. K. Hartmann, Phys. Rev. Lett. **84**, 6118 (2000).
[33] Surprisingly, we found that the insertion in plain order from 1 to L is better than many other complicated strategies.