JOURNAL OF COMPUTATIONAL BIOLOGY
Volume 7, Numbers 1/2, 2000
Mary Ann Liebert, Inc.
Pp. 171–182

Prediction of RNA Base Pairing Probabilities on Massively Parallel Computers

MARTIN FEKETE,[1] IVO L. HOFACKER,[2] and PETER F. STADLER[1,2]

ABSTRACT

We present an implementation of McCaskill's algorithm for computing the base pair probabilities of an RNA molecule for massively parallel message passing architectures. The program can be used to routinely fold RNA sequences of more than 10,000 nucleotides. Applications to complete viral genomes are discussed.

Key words: RNA secondary structure, RNA folding, RNA partition function, parallel computing, MPI, message passing.

1. INTRODUCTION

RNA molecules serve not only as carriers of information, but also as functionally active units. The three-dimensional shape of RNA molecules plays a crucial role in the process of protein synthesis and may exhibit a large variety of catalytic activities. While the prediction of three-dimensional structures from sequence data is infeasible at present, the prediction of secondary structure is a tractable task even for very large molecules.

RNA secondary structures provide a useful, though coarse grained, description of RNA molecules as exemplified by their frequent use in the literature. A variety of (closely related) dynamic programming algorithms (e.g., Waterman and Smith, 1978; Zuker and Stiegler, 1981; Zuker and Sankoff, 1984; Hofacker *et al.*, 1994) based on graph enumeration are available. These algorithms usually produce only the ground state structure or a limited ensemble of structures close to the ground state (Zuker, 1989).

A more elegant solution was suggested by McCaskill (1990). He proposed an algorithm to compute the partition function of the thermodynamic ensemble and the matrix of base pairing probabilities P_{hl} of an RNA molecule. The large size of, say, HIV genomes ($n \approx 9200$ nucleotides) implies that there is a huge number of low energy states. For example, the frequency of the minimum energy structure in the ensemble at thermodynamic equilibrium is in general smaller than 10^{-23} for RNAs of the size of an HIV viral genome. Hence one would need a huge number of different structures to adequately describe the ensemble. While such an approach is feasible for RNAs with up to some 100 nucleotides (Wuchty *et al.*, 1999), the direct generation and analysis of the necessary amount of structure information for long sequences exceeds by far the capabilities of even the most modern computer systems. The pair probability matrix (P_{hl}) computed by McCaskill's algorithms is a much more suitable representation of such large structure ensembles. Thus McCaskill's approach provides a computationally feasible alternative, which is comparable to the requirements of the simple minimum free energy folding algorithm in terms of the required computational resources.

[1]Institut für Theoretische Chemie, Universität Wien, A-1090 Wien, Austria.
[2]The Santa Fe Institute, Santa Fe, NM 87501.

Secondary structure predictions of large RNA molecules with several thousand nucleotides are usually performed by folding fairly small subsequences. This procedure has two significant disadvantages, however: (i) by definition one cannot detect long-range interactions that span more than the size of the sequence window, and (ii) the results depend crucially on the window's exact location. This is because subsequences fold independently of the rest of the sequence only if they form isolated components by themselves, i.e., if there are no base pairs to the outside of the sequence window. The only way of identifying the component boundaries is, however, to fold the sequence in its entirety.

RNA folding algorithms are quite demanding both in terms of memory and CPU time. For a sequence of length n, CPU time scales as $\mathcal{O}(n^3)$ and memory requirements are $\mathcal{O}(n^2)$. While this is not a problem for small RNA molecules, such as tRNAs, the requirements exceed the resources of most computers for large RNA molecules such as viral genomes. In most cases, memory, rather than computational speed, becomes the fundamental resource bottleneck. The use of modern parallel computers thus becomes unavoidable once the memory requirements exceed, say, 1GByte.

RNA minimum energy folding has been implemented for a variety of parallel computer architectures: We have implemented a version for `Ipsc`-based hardware, such as the `Hypercube` and the `Delta` (Hofacker *et al.*, 1994, 1996a); Zuker's suboptimal folding algorithm (Zuker, 1989) was ported to a `MasPar MP-2` (Shapiro *et al.*, 1995), and an approximate folding procedure that also yields suboptimal structures has been described for a `CM-5` (Nakaya *et al.*, 1995). In this contribution we discuss an implementation of McCaskill's partition function folding algorithm for message passing architectures. Our program, which is available upon request, is written in `C` and uses the `MPI` message passing interface. It should therefore be easily portable to most currently available parallel computers.

In the following section we briefly recall the definition of RNA secondary structures and the standard energy model. A brief description of the partition function algorithm is provided in Section 3. After a detailed description of the message passing logic (Section 4) we review the performance of our implementation. We close with a few examples of applications in Section 6, then a brief discussion.

2. RNA SECONDARY STRUCTURES

Most RNA molecules are single stranded *in vivo*. The molecule folds back onto itself to form double helical regions stabilized by Watson-Crick G-C and A-U base pairs or the slightly less stable G-U pairs. Base stacking and base pairing are hence the major driving forces of structure formation in RNA. Other, usually weaker, intermolecular forces and the interaction with aqueous solvent shape its spatial structure. As opposed to the protein case, the secondary structure of RNA sequences is well defined, provides the major set of distance constraints that guide the formation of tertiary structure, and covers the dominant energy contribution to the 3D structure. Furthermore, secondary structures are conserved in evolutionary phylogeny (Gutell, 1993) and therefore represent a qualitatively important description of the molecules.

Definition 1 (Waterman and Smith, 1978). *A secondary structure consists of a set of vertices* $V = \{1, 2, \ldots, i, \ldots, N\}$ *and a set of edges* $S = \{i \cdot j, 1 \leq i < j \leq N\}$ *fulfilling*

(1) *For* $1 \leq i < n$, $i \cdot (i+1) \in S$.
(2) *For each* i *there is at most one* $h \neq i-1, i+1$ *such that* $i \cdot h \in S$.
(3) *If* $i \cdot j \in S$ *and* $h \cdot l \in S$ *and* $i < h < j$, *then* $i < l < j$.

The first condition simply states that RNA is a linear polymer, the second condition restricts each base to at most a single pairing partner, and the third forbids pseudo-knots[1] and knots. While pseudoknots are important structural elements in many RNA molecules (Westhof and Jaeger, 1992), they are excluded from many studies mostly for a technical reason (Waterman and Smith, 1978): The folding problem for RNA can be solved efficiently by dynamic programming (Waterman and Smith, 1978; Zuker and Sankoff, 1984) in their absence. In many cases pseudoknots can be "added" to a predicted secondary structure graph during a postprocessing step.

[1] Special conformations where a base inside a loop base pairs outside the loop.

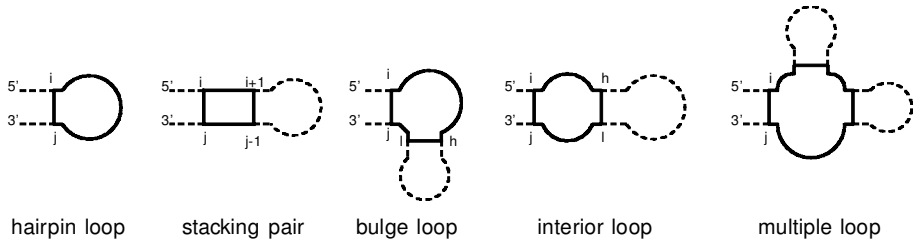

FIG. 1. Secondary structures decompose into five distinct loop types, which form the basis of the additive energy model. One distinguishes three loop energy functions: $\mathcal{H}(i, j)$ for hairpin loops, $\mathcal{I}(i, j, h, l)$ for the three types of loops that are enclosed by base pairs $i \cdot j$ and $h \cdot l$ and the additive model for multi-loops described in the text. Stacked pairs ($h = i + 1, l = j - 1$) and bulges (either $h = i + 1, l \neq j - 1$ or $l = j - 1, h \neq i + 1$) are treated as special cases of interior loops. The energies depend on the types of closing base pairs indicated by $i \cdot j$ and interior base pairs as well as on the size of the loops.

A *base pair* $h \cdot l$ is called *interior* to the base pair $i \cdot j$, if $i < h < l < j$. It is *immediately interior* if there is no base pair $p \cdot q$ such that $i < p < h < l < q < j$. For each base pair $i \cdot j$ the corresponding *loop* is defined as consisting of $i \cdot j$ itself, the base pairs immediately interior to $i \cdot j$ and all unpaired regions connecting these base pairs. In graph theoretical terms, the loops form the unique minimal cycle basis of the secondary structure graph (Leydold and Stadler, 1998).

The standard energy model for RNA contains the following types of parameters: (i) *Base pair stacking* energies depend explicitly on the types of the four nucleotides $i \cdot j$ and $(i + 1) \cdot (j - 1)$ that stack. For the purpose of the recursions in Table 1 it is useful to view stacked base pairs as a special type of interior loop, hence we denote the stacking energies $\mathcal{I}(i, j, i + 1, j - 1)$. (ii) *Loop energies* depend on the type of the loop, its size, the closing pairs and the unpaired bases adjacent to them (see Fig. 1). We write $\mathcal{H}(i, j)$ for hairpin loops and $\mathcal{I}(i, j, h, l)$ for interior loops. Multiloops are assumed to have a linear contribution of the form $\mathcal{M} = \mathcal{M}_C + \mathcal{M}_I \cdot \mathit{degree} + \mathcal{M}_B \cdot \mathit{unpaired}$; in addition, the so-called dangling end energies are taken into account which refer to mismatches next to the base pairs that delimit the loop. The implementation of the folding algorithms used in this contribution assumes the energy parameters summarized by (Walter *et al.*, 1994) except that coaxial stacking of helices is neglected. (Coaxial stacking is, strictly speaking, not part of the secondary structure graph as defined above.) The energy model is thus identical to Zucker's `mfold 2.3` (Zuker, 1996).

3. McCASKILL'S ALGORITHM

McCaskill's partition function algorithm naturally decomposes into two parts, namely the computation of the partition function and the subsequent computation of the pairing probabilities. We will refer to the two parts as *folding* and *backtracking*, respectively. The logic of the folding part is essentially the same as for minimum energy folding (Zuker and Sankoff, 1984) while the backtracking part is much more elaborate. The recursions of McCaskill's algorithm are summarized in Table 1. An efficient implementation for serial machines is part of the `Vienna RNA Package`[2] (Hofacker *et al.*, 1994). In the remainder of this section, we briefly review this algorithm.

The partition function of the complete RNA molecule can be derived from the partition functions of all its subsequences. For the subsequence from i to j, we have to distinguish whether $i \cdot j$ forms a base pair or not. We write Q^B_{ij} for the partition function of the substring subject to the constraint that $i \cdot j$ is paired and Q_{ij} for the unconstrained partition function. Consequently, the partition function of the entire molecule is $Q = Q_{1n}$.

If i to j are paired, this pair can close either a hairpin loop, an interior loop delimited by $i \cdot j$ and $h \cdot l$, or a multicomponent loop. The three terms in Table 1 correspond to these possibilities. Multiloops can be dealt with efficiently due to a linear ansatz for their energies contributions. This allows for a decomposition into three terms: one for unpaired substructures, one for substructures consisting of a single component, and

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

Table 1. Recursion for Computing the Partition Function[1]

Folding	*Backtracking*
$Q^B_{ij} = e^{-\mathcal{H}(ij)/kT} + \sum_{\substack{h=i+1 \\ u\le u_{\max}}}^{j-m-2} \sum_{l=h+m+1}^{j-1} Q^B_{hl}\, e^{-[\mathcal{I}(i,j,h,l)]/kT} + \sum_{h=i+1}^{j-m-2} Q^M_{i+1,h-1} Q^{M1}_{h,j-1}\, e^{-\mathcal{M}_C/kT}$	$P^c_{hl} = \dfrac{Q_{1,h-1} Q^B_{hl} Q_{l+1,n}}{Q_{1n}}$
$Q^{M1}_{ij} = \sum_{l=i+m+1}^{j} Q^B_{il}\, e^{-[\mathcal{M}_{\mathcal{I}}+\mathcal{M}_{\mathcal{B}}(j-l)]/kT}$	$P^i_{hl} = \sum_{\substack{i=1 \\ u<u_{\max}}}^{h-1} \sum_{j=l+1}^{n} P_{ij} \dfrac{Q^B_{hl}}{Q^B_{ij}} e^{-\mathcal{I}(i,j,h,l)/kT}$
$Q^M_{ij} = \sum_{h=i+m+1}^{j-m-1} Q^M_{i,h-1} Q^{M1}_{hj} + \sum_{h=i}^{j-m-1} Q^{M1}_{hj}\, e^{-\mathcal{M}_{\mathcal{B}}(h-i)/kT}$	$P^m_{hl} = Q^B_{hl} e^{-[(\mathcal{M}_C+\mathcal{M}_{\mathcal{I}})/kT]} \times \sum_{i=1}^{h-1} \left(P^{M1}_{il} Q^M_{i+1,h-1} + P^M_{il} Q^M_{i+1,h-1} + P^M_{il}\, e^{-[(h-i-1)\mathcal{M}_{\mathcal{B}}/kT]}\right)$
$Q^A_{ij} = \sum_{l=i+m+1}^{j} Q^B_{il}$	$P^M_{il} = \sum_{j=l+2}^{n} \dfrac{P_{ij}}{Q^B_{ij}} Q^M_{l+1,j-1}$
$Q_{ij} = 1 + Q^A_{ij} + \sum_{h=i+1}^{j-m-1} Q_{i,h-1} Q^A_{hj}$	$P^{M1}_{il} = \sum_{j=l+1}^{n} \dfrac{P_{ij}}{Q^B_{ij}} e^{-[(j-l-1)\mathcal{M}_{\mathcal{B}}/kT]}$
	$P_{hl} = P^c_{hl} + P^i_{hl} + P^m_{hl}$

[1]The parameter m is the minimum size of a hairpin loop, usually $m = 3$.

a multicomponent reminder. The auxiliary variables Q^M and Q^{M1} are necessary for handling multiloop contributions. Introducing Q^A and restricting the size of interior loops to $u \le u_{\max}$ reduces the CPU requirements from $\mathcal{O}(n^4)$ to $O(n^3)$. Most programs set $u_{\max} = 30$. The restriction on the size of interior loops does not have a serious effect in practice, since long interior loops are energetically unfavorable and therefore very rare. For further details we refer to McCaskill's (1990) original paper.

In the backtracking part of the algorithm, the pairing probabilities P_{ij} are obtained by comparing the partition functions Q^B_{ij} and Q_{ij} with and without an enforced pair $i \cdot j$. While the partition function for longer subsequences is computed from shorter ones during the folding part, the backtracking recursion proceeds in the reverse direction. The probability P_{hl} of the pair $h \cdot l$ is the sum of three independent terms: (i) it closes a component with probability P^c_{hl}, (ii) it is an interior base pair of an interior loop, bulge, or stack with probability P^i_{hl}, or (iii) it is immediately interior to a multiloop with probability P^m_{hl}. Again, two auxiliary arrays are needed to handle the multiloop contribution in cubic time. The complete recursion is summarized in Table 1.

For long (sub)sequences the partition functions Q_{ij} become very large since they are the products of a large number of exponential functions. In order to reduce the numerical problems we rescale the partition function of a subsequence of length ℓ by a factor $\tilde{Q}^{\ell/n}$, where $\tilde{Q}$ is an *a priory* estimate for the partition function. A sufficiently accurate estimate can be obtained from the ground state energy $E_{\min}$:

$$\ln \tilde{Q} \approx -1.04 \times E_{\min}/kT \tag{1}$$

We use the message passing implementation of the minimum energy folding algorithm, which is described by (Hofacker *et al.*, 1996a,b) to compute $E_{\min}$.

4. MESSAGE PASSING

Since the folding and the backtracking part are independent of each other, it seems logical to parallelize them independently. The folding part can be parallelized in a way that is very similar to our earlier message-passing implementation of the minimum energy folding algorithm (Hofacker *et al.*, 1994, 1996a). However, some of the intermediate results (partial partition functions Q_{ij} and Q^B_{ij}) are required again during the backtracking stage. Storing these values in such a way that the backtracking recursion can efficiently be distributed among a large number of processors is the main difficulty of our task.

Folding

The crucial observation is that the computation of all those matrix entries that lie on the same subdiagonal $(i, i + d)$ are independent of each other. Furthermore, they depend only on those entries that are located closer to the main diagonal. The computation therefore proceeds from the diagonal of the matrices Q_{ij}, Q^B_{ij}, etc. towards the corner $(1, n)$. In order to compute the entries $(i, i + d)$, all previously computed data from row i of the arrays Q, Q^B, and Q^M and from column $(i + d)$ of Q^{M1} and Q^A are necessary. Furthermore, we need a triangular part of the Q^B array up to depth $u_{\max}$ for the interior loop contributions. For each processor, these triangles add up the trapezoidal area indicated in Figure 2a.

We divide each subdiagonal as evenly as possible between the available N processors. Set $w = \lfloor (n - d)/N \rfloor$ and $r = n - d \bmod N$. Then the first r processors calculate $w + 1$ matrix entries; the remaining $N - r$ processors compute only w entries. After completing a subdiagonal, each processor has to send either the rightmost row or the leftmost column of its memory to its right or left neighbor, respectively; see Figure 3 (upper part). This arrangement, which is the same as for the minimum energy folding (Hofacker *et al.*, 1996b), is quite efficient since each processor sends and receives only n messages with $\mathcal{O}(n)$ bytes during the entire folding computation.

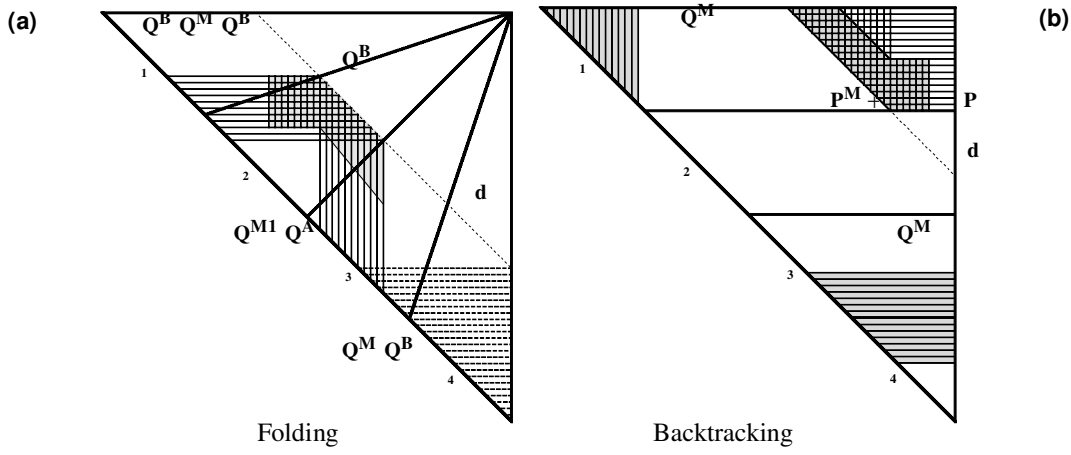

FIG. 2. Logical memory required by a single processor during folding **(a)** and backtracking **(b)**, resp., of the entries of sub-diagonal d. **Folding.** The work is divided among the processors in sectors by evenly dividing each sub-diagonal d. The matrices Q, Q^M, and Q^B are stored in form of rows, the auxiliary arrays of Q^{M1} and Q^A as columns. Each processor calculates the entries of its part of sub-diagonal d (dashed line). The shaded region representing Q^B does not extend to the diagonal, because we have restricted the maximal size of interior loops. After the calculation of one sub-diagonal d the rows of the Q^B and Q^M matrices are stored permanently (dashed lines), the memory allocated to the other arrays is recycled. **Backtracking** proceeds from the longest subsequences to shorter ones. Each processor computes a horizontal slice of the triangle matrices in order to reduce the number of messages. The computation of P^i_{hl} requires entries of P from the shaded region, while newly calculated values of P are then stored in rows (horizontal stripes). The shaded rows and columns of Q^M (shaded, towards upper left and lower right) are needed for multi-loop contribution P^m. The auxiliary arrays P^M and P^{M1} (vertical stripes) are stored as columns; only those columns intersecting the current sub-diagonal are necessary.

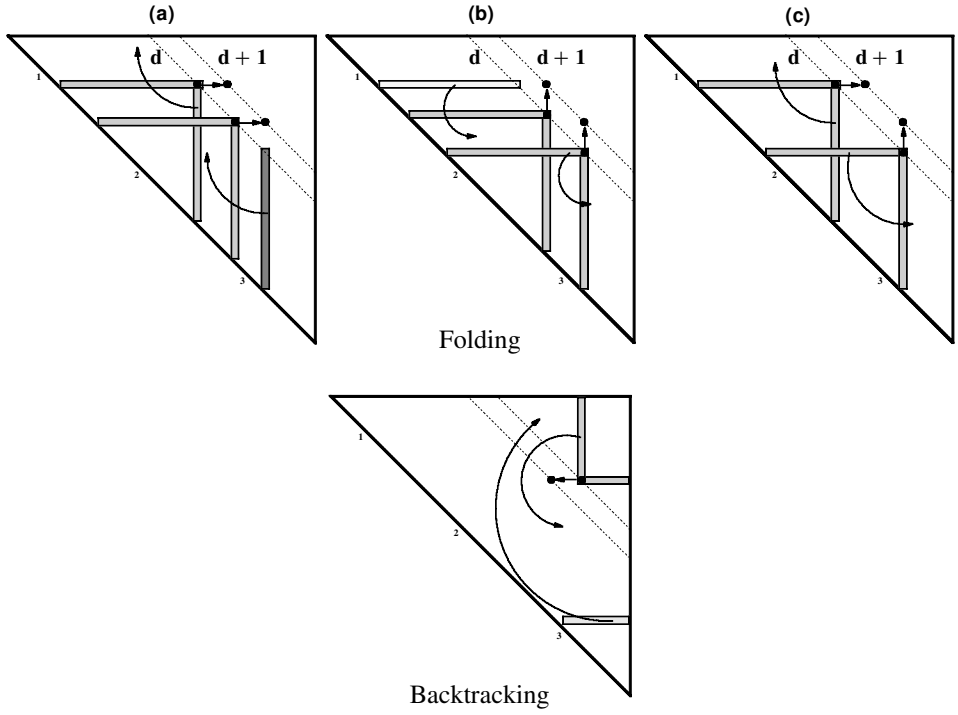

FIG. 3. Message passing requirements. **Top:** Folding. Each processor has to send an/or receive at most rows or columns of data to its neighbors when the calculation proceeds from diagonal d to $d+1$. We have to distinguish three cases: **(a)** The required rows to calculate the sub-diagonal entry for d and $d+1$ are the same, while columns have been shifted. We have to send the left-most column to processor 1 and receive the left-most column of processor 3. **(b)** The required columns stay the same for d and $d+1$. The right-most row is not needed anymore and is sent to processor 3, while processor 2 receives the right-most row of processor 1. **(c)** In this case the left-most row is the same, we have to send the left-most column to processor 1, while the right-most row is not needed anymore and is sent to processor 3. **Below:** backtracking. The required rows to compute a sub-diagonal entry from d to $d+1$ are always the same, while the columns are shifting. We have to send the right-most column of the processor k to processor $k+1$. Additionally we need rows of data, calculated during the folding procedure.

In contrast to minimum free energy folding, we need to store the entire arrays Q^B and Q^M for the backtracking part, where this information will be needed at different processors. Whenever the last entry of a row in Q^B and Q^M has been calculated, the data are stored for backtracking. A row i will be stored on node $\lfloor i \cdot N/n \rfloor$, so that the same number of rows is kept on each processor. This causes an additional n message passing operations during the folding procedure.

Backtracking

The backtracking part starts in the corner $(1, n)$ and proceeds towards the main diagonal. Again, the entries within each subdiagonal are independent from each other. To compute a base pair probability P_{ij}, we need Q^B and Q^M data that were calculated during the folding part as well as P, P^M and P^{M1} data that were calculated earlier during backtracking. A detailed description of data required by each processor is given in Figure 2b.

To simplify memory access, we do not divide the subdiagonals evenly between all processors. Instead, each processor computes a horizontal slice of the triangle matrices as shown in Figure 2b. The first n/N subdiagonals are therefore computed by a single processor at the beginning of the backtracking part. The poor load balancing during the initial steps is not crucial, however, since at the beginning all rows and columns are short and the computational effort is small. Towards the end of the backtracking procedure, when all rows and columns are long, the work is distributed ideally among the available nodes. Although the overall load balancing is somewhat worse than in the folding part, this arrangement minimizes the communication overhead, see Figure 3 (lower part).

5. PERFORMANCE

Memory Requirements

Table 2 summarizes the memory requirements of the message passing implementation. In order to ensure a reasonably efficient computation, it is necessary to store some of the intermediate data more than once. The trapezoidal arrays are necessary for computing interior loop contributions. Their height is determined by the constant $u_{\max}$, i.e., the maximum length of interior loops for which we search rigorously. Their total size is $nu_{\max} + Nu_{max}^2$ and hence is negligible in comparison to the triangular arrays. The matrices P^c, P^i, and P^m need not be stored explicitly. In addition, the matrix Q can be reused to store to the newly computed entries of P since in each subdiagonal we need the Q-values that are located closer to the main diagonal (shorter subsequences) and P-values closer to the upper right corner.

Memory usage is thus dominated by the backtracking part of the algorithm. On each processor we need approximately

$$M = \frac{1}{N}\left(3n^2 + nu_{max}\right) + 7n \tag{2}$$

real numbers. A number of arrays of length n, such as the last column of the matrix Q, are stored on each processor in order to facilitate memory access. In addition, a few integer fields of length n are used to manage memory and message passing.

For sequences longer that some 3000 nucleotides, it is necessary in general to use double-precision reals. Hence, we need some $2.5G$Bytes to fold an HIV sequence with our implementation.

CPU Requirements

The present implementation is suitable for routinely folding large genomic virus RNAs with a chain length of sometimes more than 10,000 nucleotides, see Table 3 for performance data.

The exact number of instructions required for computing the partition function is sequence dependent. We tested the performance of our parallel program on several RNA virus genomes, such as $Q\beta$ bacteriophage, $n = 4220$, polio viruses, $n \approx 7500$, and HIV viruses, $n \approx 10,000$. In the following we will use t to denote the time required to perform the folding in real time on the `Delta`, while $T = tN$ refers to the CPU time consumed on all processors.

The total computational effort is represented quite well by

$$T^* \approx an^3 + bu_{max}^2 n^2 \tag{3}$$

TABLE 2. MEMORY REQUIREMENTS[1]

Matrix	*Row-wise*	*Column-wise*	*Trapezoidal*
Folding			
Q^B	`Qb`		`Qb`
Q^M	`Qm`		
Q^{M1}		`Qmm`	
Q^A		`Qq`	
Q	`Q`		
Backtracking			
	`Qm`	`Qm`	
	`Qb`		
P	`Pr`		`Pr`
P^M		`Prml`	
P^{M1}		`Prmlt`	

[1]The folding part requires 5 triangular matrices, while we need 6 such matrices for the backtracking part because the values of Q^M are required in both row and column form.

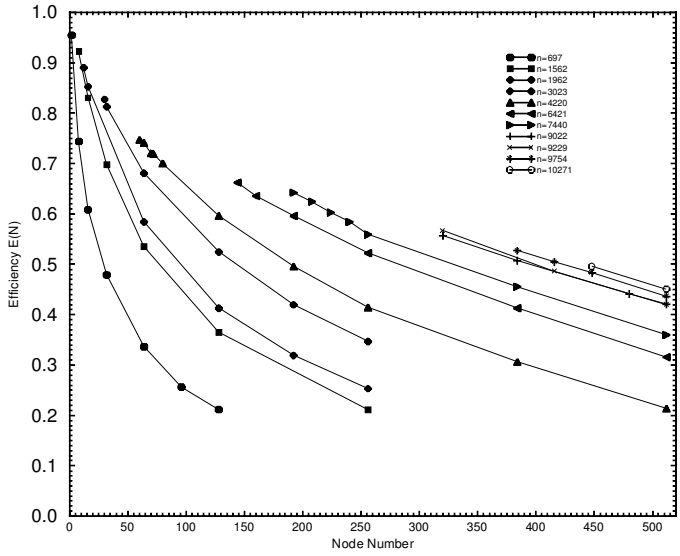

FIG. 4. Efficiency of parallelization versus number N of processors on the `Intel Delta`.

where an^3 comes from the calculation of multiloops and $bu_{max}^2 n^2$ is determined by the calculation of interior loops. From several test runs on the $Q\beta$ sequence with different values of u_{max}, we obtain $a = 900$ns and $b = 1200$ns. The CPU requirements vary very little with the sequence composition. In order to measure the pure CPU requirements of the folding algorithm (as opposed to I/O and message passing overhead) we have extrapolated folding times for different numbers N of processors to a hypothetical single-node CPU requirement T^*. The *efficiency* of the parallelization is then given by

$$\mathcal{E}(N) := T^*/(Nt) \tag{4}$$

The data in Figure 4 show that we achieve efficiencies of more than 50% when the smallest number of nodes satisfying our memory requirements is used. The computation of the minimum energy for estimating $\tilde{Q}$, Eq. (1) take less than 20% of the total execution time. Folding and backtracking each need about 40% of the total time.

Recently cost-effective workstation clusters have become widely available. We use a `Beowulf` architecture consisting of 9 two-processor PCs (Pentium II, 450Mhz) with 512MByte each, connected by 100Mbit `Fast Ethernet`, running `Linux` and `LAM 6.1`. This setup is sufficient for the routine computation of base pairing probability matrices from complete RNA virus genomes. Typical execution times are compiled in Table 3. For comparison, folding the HIV LAI sequence, $n = 9229$, took about 77min using 320 processors on the `Intel Delta` and 2h on 16 Pentium II 450MHz. The serial code took 42h on a `DEC alpha` and 64h on `Cray YMP` for the same sequence.

Despite the relatively slow network connection in the `Beowulf` workstation cluster, we find efficiencies above 50% on 16 nodes already for the chain length $n = 4000$. The efficiencies increase somewhat for larger n. Executing the parallel code on a single CPU shows that the overhead from the parallelization is about 20% to 25%. This is mainly because some parts of the algorithm can be implemented more efficiently in the serial version, where the memory organization is not constrained by requirements of easy message passing.

TABLE 3. WALL CLOCK TIMES

Hardware	*Sequence*	n	N	t (min)
Pentium II 450 Mhz (serial)	$Q\beta$	4220	1	84.0
`Beowulf` Pentium II 450 Mhz	$Q\beta$	4220	16	14.5
	HIV LAI	9229	16	123.6
	Pestivirus	12573	17	315.2
`Intel Delta`	HIV LAI	9229	320	77.0

6. APPLICATIONS

HIV-1 is a highly complex retrovirus with a single-stranded RNA genome that is densely packed with information coding for proteins and for structural elements that regulate the viral life cycle. One of the best known regulatory elements is the Rev response element (RRE) which acts as a binding site for the *Rev* protein. The RRE structure is located within the *env* gene (Figure 5). Binding of the *Rev* protein to the RRE promotes the transport of unspliced HIV transcripts to the cytoplasm (Malim *et al.*, 1989).

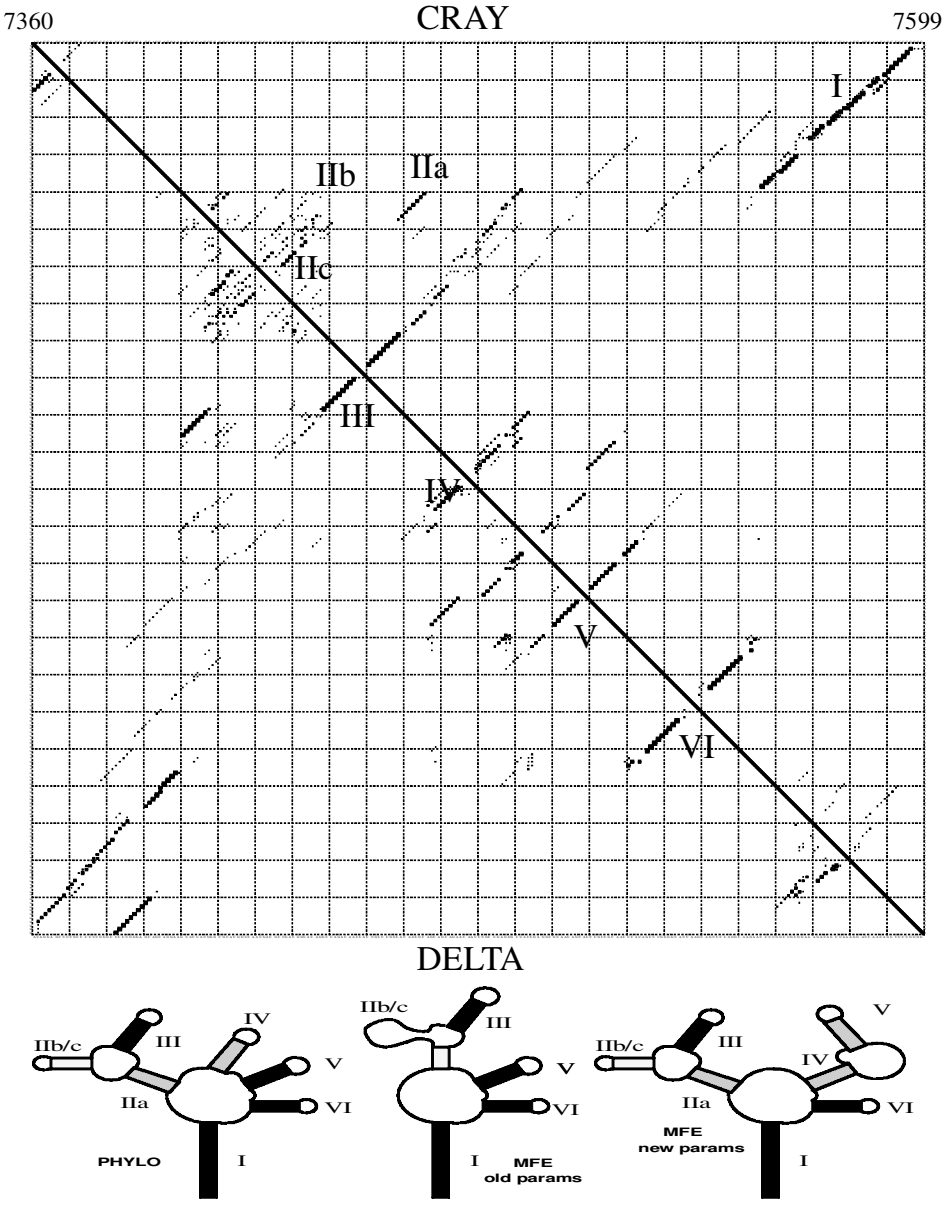

FIG. 5. The RRE region of HIV-1_{LAI}. **Top.** In a *dot plot* a base pair appears as black box at position $i \cdot j$ with an area that is proportional to the pairing probability. The upper right triangle contains the data from an earlier computation (Huynen *et al.*, 1996) on a CRAY YMP using different energy parameters, the lower left part is from the current study. The data sets agree quite well. **Below** we show three possible structures of the RRE region (Hofacker *et al.*, 1996b). The left-most structure has been inferred from phylogenetic comparisons (Konings, 1992), the middle and right structure are the mfe structure obtained with the old and new parameters, respectively. All three structures have very similar energies and all appear in the pairing matrix. The nomenclature of the stems conforms (Dayton *et al.*, 1992).

The base pair probability matrix of the HIV-1 sequence HIVLAI was determined in an earlier study using the serial program RNAfold 1.02 on a CRAY-M90, i.e., a large memory configuration of the CRAY YMP (Huynen *et al.*, 1996). In this work, a simplified energy model neglecting the dangling end energies was used. We have repeated this calculation using our message-passing implementation with an up-to-date energy parameter set. A comparison of the structure prediction for the RRE region with the earlier computation on the CRAY is shown in Figure 5.

The RRE region forms a well-defined structure on the outside of a large bulk of secondary structure. The stem loop structure I, which separates the hairpins of the RRE from the rest of the RNA molecule, consists of 32 base pairs that do not show any significant structural alternatives. The consensus structure for the RRE region consists of five hairpins in a multiple-branched conformation closed by a single stem structure (Konings, 1992). An alternative structure of only four hairpins, in which the hairpins III and IV of the consensus model merge to form one hairpin, has been proposed by Mann *et al.* (1994). Note that this alternative structure matches the minimum energy structure obtained with the old energy parameters.

Extensive computer analysis has shown that the alignment of the RRE at the level of the sequence does not coincide with the alignment at the level of the secondary structure (Konings, 1992). This has two important implications: 1) methods that predict secondary structure of RNA on the basis of covariation of positions within the sequence (Gutell, 1993) cannot provide unambiguous answers for this region, and 2) the RRE has intrinsic structural versatility and hence it is indispensable to consider ensembles of structures rather than only the single minimum energy structure.

The efficiency of our implementation allows us to routinely fold complete RNA virus genomes, thereby providing the data for a recently developed method for elucidating conserved secondary structures in moderate size samples of related RNA sequences (Hofacker *et al.*, 1998; Hofacker and Stadler, 1999). This approach is based on a combination of thermodynamic structure prediction and comparative sequence analysis. The program pfrali uses multiple sequence alignment files and base pairing probability matrices computed with our parallel implementation of McCaskill's algorithm as input and extracts promising structural features without user intervention. We have successfully applied this technique to a variety of large RNA viral genomes, including HIV-1, Hepatitis C virus, Flaviviruses (Mandl *et al.*, 1998). In all cases, we were able to find the structural elements that were previously described in the literature plus a small number of additional promising candidates. An example is shown in Figure 6. The region is located 200 nucleotides upstream of the end of the *gag* gene; to our knowledge no function has yet been assigned to this element.

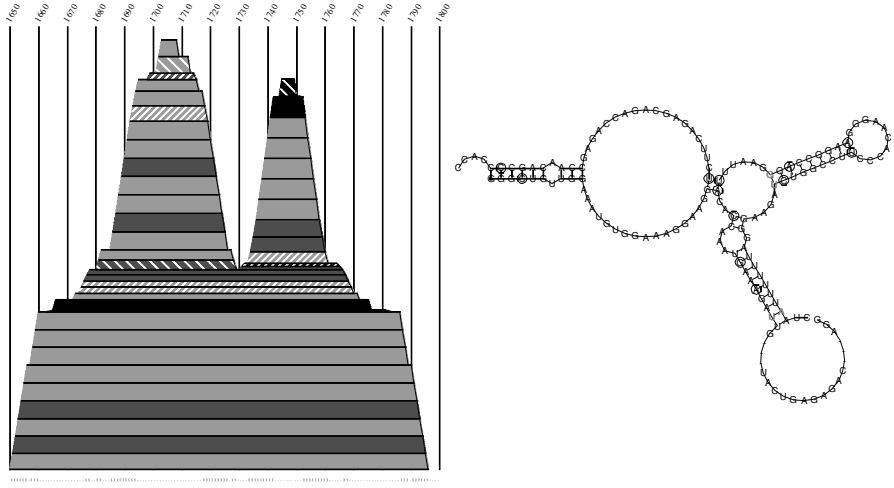

FIG. 6. A conserved structural feature in the HIV-1 genomic RNA detected by the program pfrali. The "mountain plot" on the l.h.s. is a convenient representation of the predicted structure: each base pair $i \cdot j$ is represented by a slab extending from i to j with a height proportional to the mean probability of the pair. Base pairs with compensatory mutations are shown in darker colors, pairs marked by sparse and dense white stripes have one or two noncompatible sequences, respectively. In the secondary structure plot on the r.h.s. base pairs with compensatory mutations are indicated by circles, pairs with noncompatible sequences are shown in gray. Note that only those pairs that are consistently predicted are included, most sequences will form additional pairs in this region.

7. DISCUSSION

We have developed an implementation of McCaskill's partition function algorithm for massively parallel computer architectures that allows the prediction of the base pair probability matrices of complete RNA virus genomes. The current implementation adheres to the `MPI` message passing standard and should therefore be easily portable to most presently available parallel computers. The program was used successfully on an `Intel Delta` supercomputer with 512 nodes, an `IBM SP2`, and workstation clusters, demonstrating that distributed memory architectures are well-suited to the problem of folding the largest RNA sequences available. The optimal partition sizes are those for which the total available memory on each node is utilized.

The necessity for folding long RNA molecules as a single piece instead of composing the fold from a short subsequence arises from the inherent nonlocality of RNA folding. There are long range interactions, as exemplified by the "panhandles" linking the 3' and 5' ends of bunyavirus genome segments (Paradigon *et al.*, 1982). In addition, it is well known that the fold of a subsequence depends strongly on its size and exact location.

While long RNA molecules probably fold sequentially in nature, there are rearrangements between established and new helices during the folding process. Although there have been a number of different approaches to kinetic and/or sequential folding, there is no consensus in the field, and so far these approaches have not proved to be significantly superior to thermodynamic folding, which yields at least a controlled approximation of the real structure. It highlights possible global interactions that may or may not be accessible along kinetic folding pathways. As a consequence, thermodynamic predictions of base pairing probabilities are an ideal starting point for comparative approaches.

ACKNOWLEDGMENTS

This research was in part performed using the CACR parallel computer system operated by Caltech on behalf of the Center for Advanced Computing Research and at the Albuquerque High Performance Computer Center, UNM, New Mexico. Research at the AHPCC is sponsored in part by the Phillips Laboratory, Air Force Materiel Command, USAF, under cooperative agreement number F29601-93-2-0001. "The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Phillips Laboratory or the U.S. Government." Access to the CACR facility was provided by the California Institute of Technology in the context of a collaboration with Paul E. Stolorz. Partial financial support by the Austrian *Fonds zur Förderung der Wissenschaftlichen Forschung*, Proj. No. P 12591-INF, and the Austrian Federal Ministery of Science and Transport is gratefully acknowledged.

REFERENCES

Dayton, E.T., Konings, D. A.M., Powell, D.M., Shapiro, B.A., Butini, L., Maizel, J.V., and Dayton, A.I., 1992. Extensive sequence-specific information throughout the CAR/RRE, the target sequence of the human immunodeficiency virus type 1 Rev protein. *J. Virol.* 66, 1139–1151.

Gutell, R.R., 1993. Evolutionary characteristics of RNA: Inferring higher-order structure from patterns of sequence variation. *Curr. Opin. Struct. Biol* 3, 313–322.

Hofacker, I.L., Fekete, M., Flamm, C., Huynen, M.A., Rauscher, S., Stolorz, P.E., and Stadler, P.F., 1998. Automatic detection of conserved RNA structure elements in complete RNA virus genomes. *Nucl. Acids Res.* 26, 3825–3836.

Hofacker, I.L., Fontana, W., Stadler, P.F., Bonhoeffer, S., Tacker, M., and Schuster, P., 1994. Fast folding and comparison of RNA secondary structures. *Monatsh. Chem.* 125, 167–188.

Hofacker, I.L., Huynen, M.A., Stadler, P.F., and Stolorz, P.E., 1996a. Knowledge discovery in RNA sequence families of HIV using scalable computers. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining, Portland, OR*, 20–25. AAAI Press, Portland, OR.

Hofacker, I.L., Huynen, M.A., Stadler, P.F., and Stolorz, P.E., 1996b. RNA folding and parallel computers: The minimum free energy structures of complete HIV genomes. Tech. rep., SFI, Santa Fe, New Mexico. # 95–10–089.

Hofacker, I.L., and Stadler, P.F., 1999. Automatic detection of conserved base pairing patterns in RNA virus genomes. *Comp. and Chem.* 23, 401–414.

Huynen, M.A., Perelson, A.S., Viera, W.A., and Stadler, P.F., 1996. Base pairing probabilities in a complete HIV-1 RNA. *J. Comp. Biol.* 3, 253–274.

Konings, D. A.M., 1992. Coexistence of multiple codes in messenger RNA molecules. *Comp. & Chem.* 16, 153–163.

Leydold, J., and Stadler, P.F., 1998. Minimal cycle basis, outerplanar graphs. *Elec. J. Comb.* 5, R16. See `http://www.combinatorics.org`.

Malim, M.H., Hauber, J., Le, S.Y., Maizel, J.V., and Cullen, B., 1989. The HIV-1 Rev trans-activator acts through a structured target sequence to activate nuclear export of unspliced viral mRNA. *Nature* 338, 254–257.

Mandl, C.W., Holzmann, H., Meixner, T., Rauscher, S., Stadler, P.F., Allison, S.L., and Heinz, F.X., 1998. Spontaneous and engineered deletions in the 3'-noncoding region of tick-borne encephalitis virus: Construction of highly attenuated mutants of flavivirus. *J. Virology* 72, 2132–2140.

Mann, D., Mikaelian, I., Zemmel, R., Green, S., Lowe, A., Kimura, T., Singh, M., Butler, P., Gait, M., and Karn, J., 1994. A molecular rheostat. Co-operative rev binding to stem I of the rev-response element modulates human immunodeficiency virus type-1 late gene expression. *J. Mol. Biol.* 241, 193–207.

McCaskill, J.S., 1990. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. *Biopolymers* 29, 1105–1119.

Nakaya, A., Yamamoto, K., and Yonezawa, A., 1995. RNA secondary structure prediction using highly parallel computers. *Comput. Appl. Biosci.* 11, 685–692.

Paradigon, N., Vialat, P., Girard, M., and Bouloy, M., 1982. Panhandles and hairpin structures at the termini of germiston virus RNAs (bunyavirus). *Virology* 122, 191–197.

Shapiro, B.A., Chen, J.H., Busse, T., Navetta, J., Kasprzak, W., , and Maizel, J., 1995. Optimization and performance analysis of a massively parallel dynamic programming algorithm for RNA secondary structure prediction. *Int. J. Supercomp. Appl.* 9, 29–39.

Walter, A.E., Turner, D.H., Kim, J., Lyttle, M.H., Müller, P., Mathews, D.H., and Zuker, M., 1994. Co-axial stacking of helixes enhances binding of oligoribonucleotides and improves predicions of RNA folding. *Proc. Natl. Acad. Sci. USA* 91, 9218–9222.

Waterman, M., and Smith, T.F., 1978. RNA secondary structure: A complete mathematical analysis. *Math. Biosc.* 42, 257–266.

Westhof, E., and Jaeger, L., 1992. RNA pseudoknots. *Current Opinion Struct. Biol.* 2, 327–333.

Wuchty, S., Walter Fontana, I. L.H., and Schuster, P., 1999. Complete suboptimal folding of RNA and the stability of secondary structures. *Biopolymers* 49, 145–165.

Zuker, M., 1989. On finding all suboptimal foldings of an RNA molecule. *Science* 244, 48–52.

Zuker, M., 1996. `mfold-2.3`. `http://www.ibc.wustl.edu/~zuker/rna/`.

Zuker, M., and Sankoff, D., 1984. RNA secondary structures and their prediction. *Bull. Math. Biol.* 46, 591–621.

Zuker, M., and Stiegler, P., 1981. Optimal computer folding of larger RNA sequences using thermodynamics and auxiliary information. *Nucl. Acids Res.* 9, 133–148.

Address correspondence to:
Peter F. Stadler
Institut für Theoretische Chemie
Universität Wien
Währingerstrasse 17
A-1090 Wien, Austria

E-mail: studla@tbi.univie.ac.at