PREDICTION OF RNA BASE PAIRING PROBABILITIES USING MASSIVELY PARALLEL COMPUTERS

M. FEKETE[1], I.L. HOFACKER[1], P.F. STADLER[1,2]
[1] *Institut für Theoretische Chemie, Universität Wien Währingerstrasse 17, A-1090 Wien, Austria*
[2] *Santa Fe Institute, 1399 Hyde Park Rd., Santa Fe NM 87501, USA*

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 10000 nucleotides. Applications to complete viral genomes are discussed.

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 tractable even for large molecules. RNA secondary structures provide a useful, though coarse grained, description of RNA molecules that can be computed efficiently by dynamic programming algorithms based on graph enumeration.[1,2] These algorithms usually produce only the ground state structure or a limited ensemble of structures close to the ground state.[3]

A more elegant solution was suggested by McCaskill,[4] who proposed an algorithm to compute the partition function of the thermodynamic ensemble and the matrix P_{ij} of base pairing probabilities 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 a viral genome. Hence one needs more than 10^{23} of different structures to adequately describe the ensemble, and the direct generation and analysis of this amount of structure information is way beyond the capabilities of even the most modern computer systems. McCaskill's approach provides a computationally feasible alternative, which in fact is comparable to the requirements of the simple minimum free energy folding algorithm.

Secondary structure predictions of large RNA molecules with several thousand nucleotides are usually performed by folding fairly small subsequences.

This has two 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 a component by themselves, i.e., if there are no base pairs to the out-side of the sequence window. The only way, however, of identifying the component boundaries is 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 is $\mathcal{O}(n^3)$ 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.

2 McCaskill's Algorithm

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 $(i.j)$ and $(i+1, j-1)$ 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, the size of loop, the closing pairs and the unpaired bases adjacent to them, see figure. 1. We write $\mathcal{H}(i, j)$ for hairpin loops and $\mathcal{I}(i, j, k, l)$ for interior loops. Multi-loops 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. Our implementation uses the parameters summarized in Walter *et al.*,[5] except that we neglect co-axial stacking of helices.

McCaskill's algorithm decomposes naturally into two parts, the computation of the partition functions (*folding*) and the computation of the pairing probabilities (*backtracking*). The logic of the folding part is essentially the same as for minimum energy folding,[6] the backtracking part, however, is more elaborate. The recursions of McCaskill's algorithm are compiled in table 1. An efficient implementation for serial machines is part of the `Vienna RNA Package`.[7,a]

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

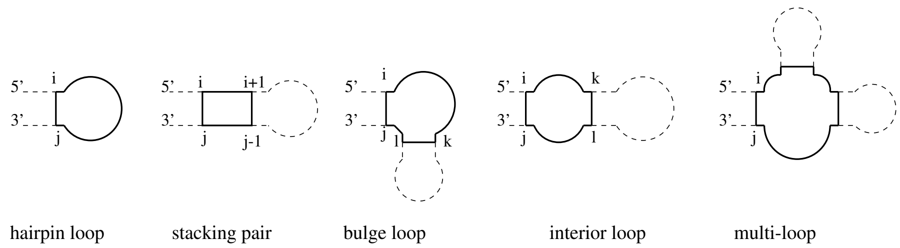

Figure 1: Secondary structures decompose into five distinct loop types, which form the basis of the additive energy model. One distinguishes three loop energy functions: $\mathcal{H}(i,j)$ for hairpin loops, $\mathcal{I}(i,j,k,l)$ for the three types of loops that are enclosed by base pairs $(i.j)$ and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

The partition functions Q^B_{ij} and Q_{ij} of substructures on the substring $[ij]$ with and without the constraint that i and j form a base pair can be determined from smaller sequence fragments. If $(i.j)$ is a base pair, then it may be the closing pair of a hairpin, it may close an interior loop (or extend a stack), or it might close a multi-loop. The auxiliary variables Q^M and Q^{M1} are necessary for handling the multi-loops. Introducing Q^A and restricting the size of interior loops to $u \leq u_{\max} = 30$ reduces the CPU requirements to $O(n^3)$. The restriction of u does not have a serious effect in practice, since very long interior loops do not occur in "real" sequences. There are three distinct contributions to the unconstrained partition function Q_{ij}: The first term accounts for the unpaired structure. The second term collects all structures that consist of a single component, possibly with an unpaired "tail" at the 3' end. The final term arises from the formal construction of multi-component structures from a 1-component part at the 3' side and an arbitrary structure at the 5' side. The quantities Q^B_{ij}, $Q^M_{ij}, Q_{1,j}$ and $Q_{i,n}$ must be stored for the backtracking procedure during the folding calculation.

The pairing probabilities are obtained by comparing the partition functions Q^B_{ij} and Q_{ij} with and without an enforced pair $(i.j)$. While the folding procedure computes the partition functions for longer subsequences from shorter ones, the backtracking recursion runs in the reverse direction. The base pairing probability P_{ij} of pair $(i.j)$ can be composed from three additive contributions: P^c_{kl}, P^i_{kl}, and P^m_{kl} describe the probability that the base pair $(k.l)$ closes a component, an interior loop, or a multi-loop, respectively. Again we need two auxiliary arrays to reduce the execution time to $\mathcal{O}(n^3)$.

Table 1: Recursion for Computing the Partition Function.

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

Being products of a large number of terms the partition functions Q_{ij} become very large if n is large. In order to reduce the numerical problems arising for long sequences we rescale the partition function of a subsequence of length ℓ by a factor $\tilde{Q}^{\ell/n}$, where

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

is a semi-empirical estimate for the partition function based on the ground state energy $E_{\min}$, which has to be computed beforehand.[7] Long sequences require very accurate estimates of the internal scaling factor $\tilde{Q}^{1/n}$ to avoid overflows and/or underflows of the floating point arithmetic.

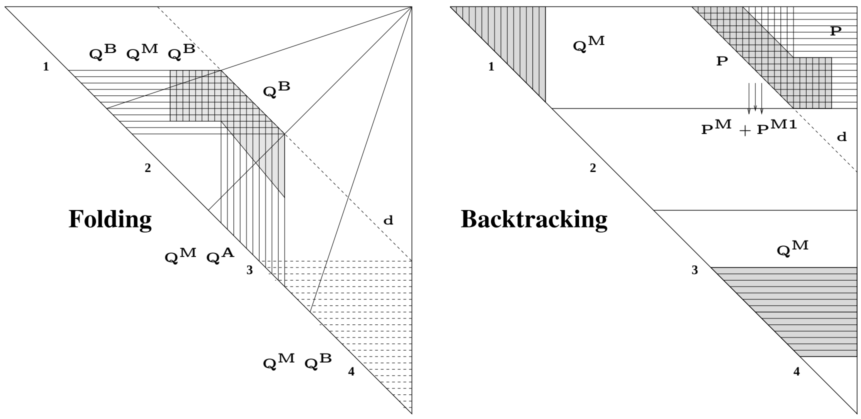

Figure 2: Logical memory required by a single processor during folding (l.h.s) and backtracking (r.h.s.), resp., of the entries of sub-diagonal d.
Folding. The work is divided among the processors in sectors by evenly dividing each sub-diagonal d. The matrices Q, Q^M, and Q^B are in form of rows, the 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 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_{kl} 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 $P^M + P^{M1}$ (vertical stripes) are stored as columns; only those columns intersecting the current diagonal are necessary.

3 Message Passing Implementation

RNA minimum energy folding has been implemented for different parallel computer architectures. A version for `Ipsc` based hardware (hypercube, Delta) was written by our group,[7,8] Zuker's[3] suboptimal folding algorithm was ported to a `MasPar MP-2` by Shapiro *et al.*,[9] and an approximate folding procedure for a `CM-5` yielding also suboptimal structures was described by Nakaya *et al.*[10]

Our program is written in `C` and uses the `MPI` message passing interface. It should therefore be easily portable to most currently available parallel computers. The implementation of the folding part is very similar to our earlier message passing implementation of minimum energy folding algorithm.[7,8] In fact, we use the latter to explicitly compute $E_{\min}$.

The computation proceeds from the diagonal of the matrices Q_{ij}, Q^B_{ij}, etc. towards the corner $(1, n)$. The crucial observation is that the computation of all entries $(i, i+d)$ are independent of each other and depend only on entries that are located closer to the diagonal. Memory rearrangements (message passing) is therefore necessary (at worst) when a sub-diagonal has been completed. Hence we divide each sub-diagonal evenly between the available N processors.

While the message passing for the folding part is rather straight forward[11] the situation is complicated here by the fact that we need to store the entire arrays Q^B and Q^M for the backtracking part, and that this information will be needed at different processors. A scheme of the information required by a single processor during folding and backtracking, resp., is given in figure 2.

The backtracking step is parallelized in a different way in order to simplify the communication and to reduce the amount of message passing. Backtracking proceeds from the longest subsequences to shorter ones, i.e., in reverse order than calculation of the partition function. Each processor computes a horizontal slice of the triangle matrices as shown in figure 2. As a consequence, the first n/N sub-diagonals at the beginning of the backtracking are computed by a single processor. The poor load balancing in these initial steps is not crucial, however, because 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 on the nodes. Although the overall load balancing is somewhat worse than in the folding part, this arrangement minimizes the communication overhead.

The actual implementation of the message passing steps is rather intricate since the newly computed entries of the partition function matrices (in the folding part) and the pairing probabilities (in the backtracking part) are required by different processors in different processing stages. Details can be found in the M. Fekete's MSc thesis.[12]

4 Performance

Our implementation is designed for routinely folding genomic virus RNAs which have a chain length of sometimes more than 10 000 nucleotides. On the `Intel Delta` the folding of $\mathrm{HIV1}_{LAI}$, $n = 9229$, for instance, takes about 77min using 320 processors.

The exact number of instructions required for computing a minimum free energy structure 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 ($n \approx 10000$). 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 total time used for the computation.

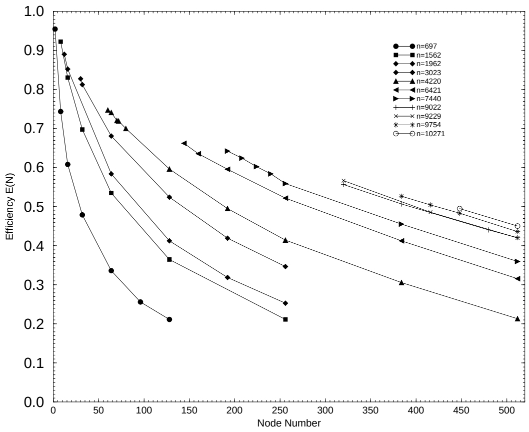

Figure 3: Efficiency of parallelization.

The total computational effort is represented quite well by

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

where an^3 comes from the calculation of multi-loops 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. 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

$$\mathcal{E}(N) := T^*/(Nt) \tag{3}$$

The data in figure 3 show that we achieve efficiencies of more than 50% when the smallest number of nodes satisfying our memory requirements is used.

5 Applications

As a first application we calculated the base pair probability of a full length HIV1$_{LAI}$ genome using the parallel partition function algorithm. The same sequence was folded in an earlier study [13] on a `CRAY-M90` (a large memory configuration of the `CRAY YMP`) using the `RNAfold 1.02`, which did not take into account dangling end energies and used a slightly different set of energy parameters.

HIV1 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 (fig. 4). Binding of the *Rev* protein to the RRE promotes the transport of unspliced HIV transcripts to the cytoplasm.[16] A comparison of the structure prediction for the RRE region with the earlier computation on the `CRAY` is shown in figure 4.

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 5 hairpins in a multiple branched conformation closed by a single stem structure.[14] An alternative structure of only 4 hairpins, in which the hairpins III and IV of the consensus model merge to form one hairpin, has however been proposed.[17] Note that this alternative structure matches the minimum energy structure obtained with our 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.[14] This has two important implications: 1) methods that predict secondary structure of RNA on the basis of co-variation of positions within the sequence [18] cannot provide unambiguous answers for this region, and 2) the RRE has intrinsic structural versatility and hence one should consider ensembles of structures rather than a 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.[19,20] 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.[21] 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.

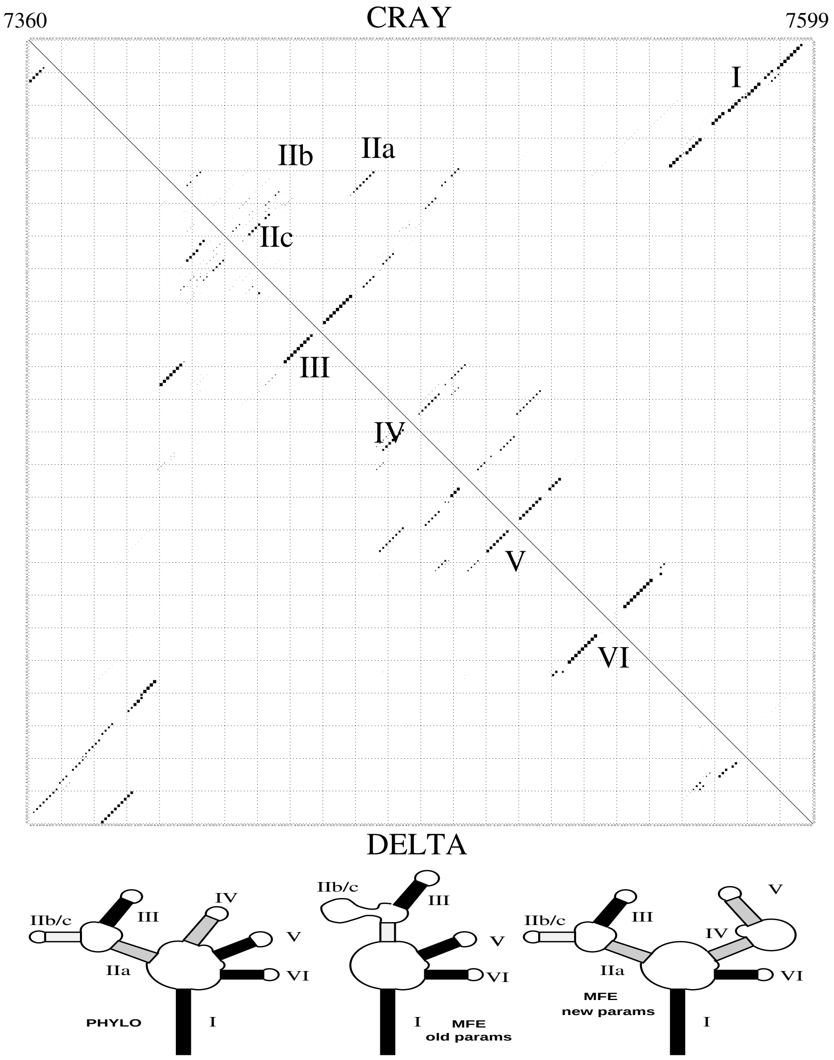

Figure 4: The RRE region of $HIV1_{LAI}$. **Top.** In a *dot plot* a base pair appears as black box at position $(i.j)$ with an area that is proportional to the pairing probability. The upper right triangle contains the data from an earlier computation[13] on a `CRAY YMP` using different energy parameters, the lower left part is from the current study. The data sets agree quite well. **Below** we show three possible minimum energy structures of the RRE region[11]. The left-most structure has been inferred from phylogenetic comparisons,[14] the middle structure was obtained in the `CRAY` computation, while the right-most structure is the minimum energy structure for the current parameter set. All three structures have very similar energies and all appear in the pairing matrix. The nomenclature of the stems conforms Dayton *et al.*[15]

6 Discussion

We have developed an implementation of McCaskill's partition function algorithm [4] for massively parallel computer architectures, which 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. Our implementation of dynamic programming RNA folding algorithms on up to 512 nodes of the `Delta` supercomputer demonstrates 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 non-locality of RNA folding. There are long range interactions, as exemplified by the "panhandles" linking the 3' and 5' ends of bunyavirus genome segments,[22] and the fold of a subsequence depends strongly on its size and exact location.

While long RNA molecules are probably folded 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 Ministery of Sciences is gratefully acknowledged.

References

1. M.S. Waterman and T. F. Smith. RNA secondary structure: A complete mathematical analysis. Math. Biosc. **42**, 257–266 (1978).
2. M. Zuker and P. Stiegler. Optimal computer folding of larger RNA sequences using thermodynamics and auxiliary information. Nucl. Acids Res. **9**, 133–148 (1981).
3. M. Zuker. On finding all suboptimal foldings of an RNA molecule. Science **244**, 48–52 (1989).
4. John S. McCaskill. The equilibrium partition function and base pair binding probabilities for RNA secondary structure. Biopolymers **29**, 1105–1119 (1990).
5. Amy E. Walter, Douglas H. Turner, James Kim, Matthew H. Lyttle, Peter Müller, David H. Mathews, and Michael Zuker. Co-axial stacking of helixes enhances binding of oligoribonucleotides and improves predicions of RNA folding. Proc. Natl. Acad. Sci. USA **91**, 9218–9222 (1994).
6. M. Zuker and D. Sankoff. RNA secondary structures and their prediction. Bull. Math. Biol. **46**, 591–621 (1984).
7. Ivo L. Hofacker, Walter Fontana, Peter F. Stadler, Sebastian Bonhoeffer, Manfred Tacker, and Peter Schuster. Fast folding and comparison of RNA secondary structures. Monatsh. Chem. **125**, 167–188 (1994).
8. Ivo L. Hofacker, Martijn A. Huynen, Peter F. Stadler, and Paul E. Stolorz. 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*, pp. 20–25, (AAAI Press, Portland, OR, 1996).
9. B. A. Shapiro, J. H. Chen, T. Busse, J. Navetta, W. Kasprzak, , and J. Maizel. Optimization and performance analysis of a massively parallel dynamic programming algorithm for RNA secondary structure prediction. Int. J. Supercomp. Appl **9**, 29–39 (1995).
10. Akihiro Nakaya, Kenji Yamamoto, and Akinori Yonezawa. RNA secondary structure prediction using highly parallel computers. Comput. Appl. Biosci. **11**, 685–692 (1995).
11. Ivo L. Hofacker, Martijn A. Huynen, Peter F. Stadler, and Paul E. Stolorz. RNA folding and parallel computers: The minimum free energy structures of complete HIV genomes. (Technical report 95-10-089, SFI, Santa Fe, New Mexico, 1996).

12. Martin Fekete. RNA secondary structure prediction using parallel computers. Master's thesis, Faculty of Sciences, University of Vienna, Austria, 1997.
13. M. A. Huynen, A. S. Perelson, W. A. Viera, and P. F. Stadler. Base pairing probabilities in a complete HIV-1RNA. J. Comp. Biol. **3**, 253–274 (1996).
14. D. A. M. Konings. Coexistence of multiple codes in messenger RNA molecules. Comp. & Chem. **16**, 153–163 (1992).
15. E. T. Dayton, D. A. M. Konings, D. M. Powell, B. A. Shapiro, L. Butini, J. V. Maizel, and A. I. Dayton. 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 (1992).
16. M. H. Malim, J. Hauber, S. Y. Le, J. V Maizel, and B.R. Cullen. The HIV-1Rev trans-activator acts through a structured target sequence to activate nuclear export of unspliced viral mRNA. Nature **338**, 254–257 (1989).
17. D.A. Mann, I. Mikaelian, R.W. Zemmel, S.M. Green, A.D. Lowe, T. Kimura, M. Singh, P.J. Butler, M.J. Gait, and J. Karn. 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 (1994).
18. Robin R. Gutell. Evolutionary characteristics of RNA: Inferring higher-order structure from patterns of sequence variation. Curr. Opin. Struct. Biol **3**, 313–322 (1993).
19. Ivo L. Hofacker, Martin Fekete, Christoph Flamm, Martijn A. Huynen, Susanne Rauscher, Paul E. Stolorz, and Peter F. Stadler. Automatic detection of conserved RNA structure elements in complete RNA virus genomes. *Nucl. Acids Res.* (1998), submitted, Santa Fe Institute Preprint 98-03-020.
20. Ivo L. Hofacker and Peter F. Stadler. Automatic detection of conserved base pairing patterns in RNA virus genomes. *Comp. & Chem.* (1998), submitted.
21. Christian W. Mandl, Heidemarie Holzmann, Tamara Meixner, Susanne Rauscher, Peter F. Stadler, Steven L. Allison, and Franz X. Heinz. 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 (1998).
22. P. Vialat N. Paradigon, M. Girard, and M. Bouloy. Panhandles and hairpin structures at the termini of germiston virus RNAs (bunyavirus). Virology **122**, 191–197 (1982).