An accelerated algorithm for calculating the secondary structure of single stranded RNAs

Eliahu Comay[1], Ruth Nussinov[2*] and Oded Comay[3]

[1]Physics Department, [2]Computer Science Department, School of Mathematics, and [3]Computer Center, Tel Aviv University, Ramat Aviv, Tel Aviv, Israel

Received 29 July 1983

ABSTRACT

We describe a code designed for secondary structure computation of single stranded RNA molecules. While it incorporates the same principles as the original algorithm of Nussinov et al (1978), its restructuring improves the logic and the approach of the codes based on it. For long sequences the code is at least an order of magnitude faster. For a chain n nucleotides long, references to computer disk memory are reduced from n^3 to less than n^2. For $n>>100$, disk references behave like $n^3/6000$.

1. INTRODUCTION

There are today two fast computer codes for predicting the secondary structures of single stranded RNA sequences. Both the one by Nussinov and Jacobson (1) and the one by Zuker and Stiegler (2) which appeared thereafter, are based on the original mathematical algorithm for planar loop matchings developed by Nussinov et al (3), Griggs (4) and Nussinov (5).

The algorithm for loop matchings (3-5) is dynamic, short and exceedingly simple. For a molecule n nucleotides long, it finds the maximum number of base pairs using n^3 steps and n^2 memory units. All base pairs, i.e. A-U, G-C and G-U are given equal weights and stacking interactions are ignored. Also ignored are the destabilizing effects of single stranded regions, i.e. bulges, internal and hairpin loops. This situation was remedied by Nussinov and Jacobson (1) and by Zuker and Stiegler (2) who introduced the energies of the various loop structures (6-11) into this efficient algorithm. As a result, several studies could have been carried out on long RNA chains. Such studies could not have been contemplated using the previous secondary structure existing codes (12-15). Nussinov and Tinoco (16) have looked into the process of folding the RNA as it is being synthesized or as it refolds after being read by the ribosomes. Nussinov, Tinoco and Jacobson (17) have used the algorithm to study the secondary structure of the whole, 2600 bases long, late SV40 precursor RNA. It was also used by Nussinov, Tinoco and Jacobson (18) to study the effect of small changes in the free energies of single stranded regions on the folding of mRNA sequences. Stiegler et al (19) have used their code (2) to elucidate the 16SrRNA secondary structure.

Nonetheless, the introduction of the free energy assignments considerably complicates the original mathematical algorithm. In order to calculate the energy correctly, we need to know, at every step, the precise loop structure one step back from the newly added base pair. This, as will be explained in detail later, results in a relatively large time requirement especially for long sequences.

This paper proposes a restructuring of the algorithm. The simplification will enable performing routine secondary structure calculations on sequences thousands of nucleotides long. Additional biochemical and chemical data, if present, can be easily incorporated into it and will result in further acceleration. For sequences > 1000 nucleotides in length, the proposed restructured algorithm can accelerate the calculation $\sim$ 10-100 fold.

2. THE ORIGINAL ALGORITHM

The original algorithm (3) has been aimed at maximizing the number of base pairs. The folding rules are simple. All base pairs are equally weighted and only planar, i.e. non-intersecting and non-overlapping bonds are allowed. Rather than check all compatible single and/or consecutive, parallel, bonds as was previously done (13,14) this algorithm suggests a simple, recursive procedure. Basically, the algorithm works inductively as follows: (i) It finds a structure with the maximal number of base pairs for a small section ((i,j), where i<j) and stores it. (ii) The fixed size section is shifted through the molecule (i.e. $i \rightarrow i+1$, $j \rightarrow j+1$; section size, $ip = j-i$, being constant). (iii) Coming back to the beginning (say 5' end) of the molecule, the section

size is increased by one $(i,j) \rightarrow (i,j+1)$; $ip \rightarrow ip+1$) and the search for the optimal structure for the new section is carried out. This new section size is now, repeatedly, shifted throughout the chain as was already described in (ii). Clearly, in calculating the optimal structure for the new section size, we can use the information already computed for each and every smaller subsection contained within it. Thus, (iv) a variable, k, ($i \leq k < j$) is traveling inside the section (i,j) from beginning to end, dividing it into two parts, ((i, k-1) and (k+1,j-1)). The maximal numbers of base pairs possible within both parts, M (i, k-1) and M (k+1, j-1) are read directly. At k = j-1 the search is stopped and the best value attained is inserted in M (i, j):

$$M(i,j) = \mathrm{Max}\begin{cases} M(i,k-1) + M(k+1,\ j-1)+1 & i \leq k < j \\ M(i,\ j-1) & \end{cases} \qquad \text{Eq. (1)}$$

An illustration is given in Figure 1.

For further details of this procedure as well as for an explanation of how a structure which corresponds to the lowest energy value is generated at the end, see refs. 1,3.

3. THE ALGORITHM ADAPTED FOR PREDICTION OF SECONDARY STRUCTURES WITH MINIMAL ENERGIES

The introduction of the free energy assignments for the various loop structures complicates the above procedure. The

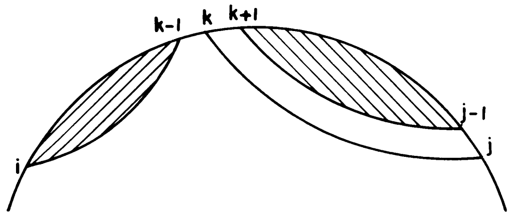


Fig. 1 An illustration of the approach to the problem. By attempting to match base k with base j, two subsections, (i, k-1) and (k+1, j-1) are formed. These subsections are shorter than the present (i,j) section, and thus their optimal matching was already computed and registered.

present goal is to minimize the energy for every section analyzed:

$$E(i,j) = \mathrm{Min}\begin{cases} E(i,k-1) + E(k+1,\, j-1) + E_{kj} \\ \\ E(i,j-1) \qquad i \leq k < j - \ell\mathrm{min} \end{cases} \qquad \text{Eq. (2)}$$

ℓ min is the minimal hairpin loop length allowed.

Correct calculation of the energy of the newly attempted base pair (E_{kj}) requires knowledge of the nature of the base pair(s) preceding it and whether there are unpaired bases on either or both sides of the chains between the preceding base pair and the present one and if so - how many. The number of arms originating from the preceding loop and the knowledge whether we deal with the first base pair closing a hairpin loop is also essential (see Fig. 2a-d). Finding these crucial details necessitates partial backtracking at every single step. Since, a priori, the loop size and the exact location of the base pair(s) closing it, is (are) unknown, the search involves jumping haphazardly through an nxn matrix.

Whereas E(i,j) (with j > i)keeps the minimal energies, E(j,i), the upper right half of the matrix, keeps the k values which, when base paired with j, achieve the minimal energies of the (i,j) sections. Starting at j-1, E(j-1, k+1) = k' is read for this, one step backtracking. If k' = o, E (j-2, k+1) is read etc. If k'$\neq$ o, then j-1 is hydrogen bonded to k' and we next jump to E (k'-1, k+1) checking whether k'-1 is base paired and if so with which base. If E (k'-1, k+1) = o, we repeat as before, trying E (k'-2, k+1). Clearly for large matrices which are kept in the external memory of the computer, this swapping of physical and virtual memories requires a long time.

The energies of each of the traced arms (Fig. 3) are read from the lower left half of the matrix (E (i,j)).

$$E(k+1,j-1) = E(k',j') + E(k'',j'') + E(k''',j''')\ldots + E_z \qquad \text{Eq. (3)}$$

z is the total number of unpaired bases present in the loop (see Fig. 3).

Thus, calculation of the energy E (k+1,j-1) necessitates the search for three values (in this case) in the lower half matrix and numerous values in the upper half matrix. For long sequences, this is a very tedious and time consuming procedure.

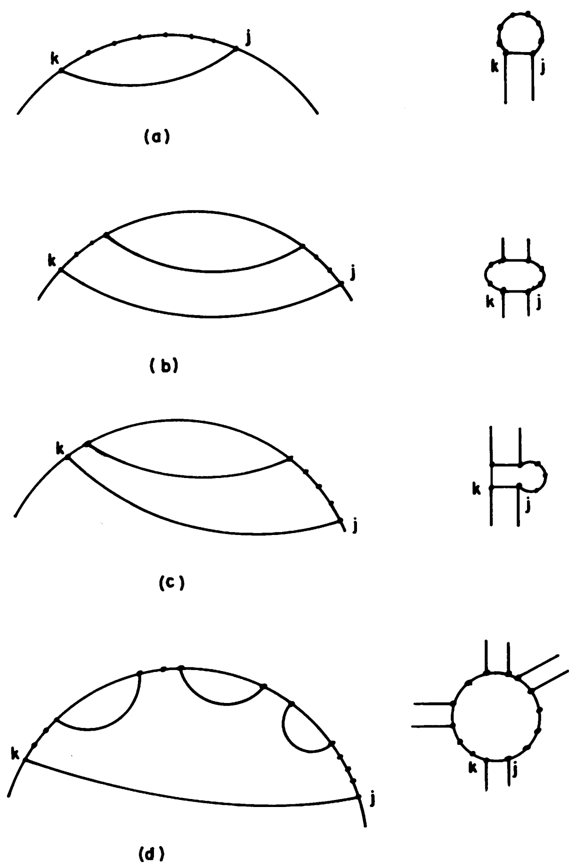


Fig. 2 The configuration adjacent to the kj base pairings. The open arc represents the nucleotide chain. On the left are the representations in the algorithm, on the right the base pairs are shrunk and the structural features are seen more easily. (a) kj is the first pairing in the section, forming a hairpin. (b) the kj base pair forms an internal loop and (c) a bulge. (d) closure of the kj base pair forms a loop from which three arms originate.

For more details see Nussinov and Jacobson (1) and Nussinov et al (17,18).

4. THE ACCELERATED ALGORITHM

(a) Considerations

The complications introduced by the energy calculations are evident. Although recently Nussinov et al (17) have employed

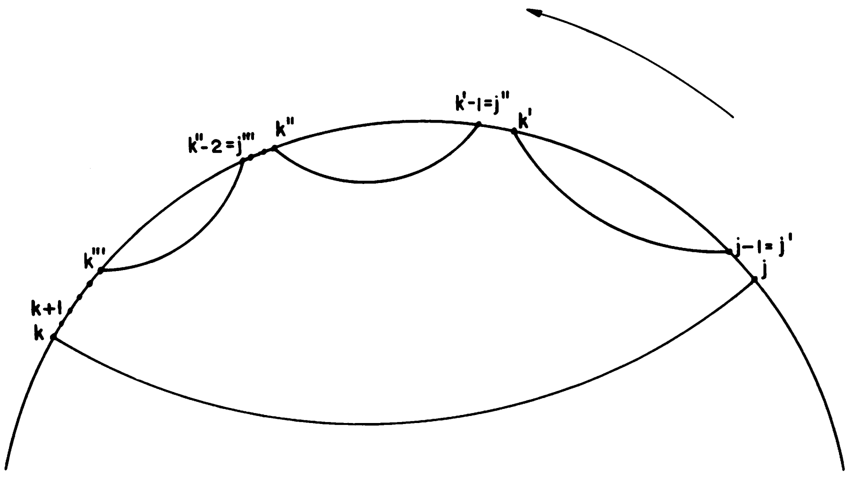


Fig. 3 The arms traced in the partial backtracking. After searching and locating the arms, their corresponding energies are read and added (see Eq. 3). In order to calculate correctly the energy of the kj base pair, z the sum of unpaired bases in the loop (five in this case) is required as well as the nature of the base pairs preceding kj, i.e. k',j'; k",j" and k"',j"'. The arrow indicates the direction of the backtracking.

this code to fold the whole, 2600 nucleotides long, SV40 late precursor RNA, the computer time used by the constant searches in the 2600^2 matrix was too long to allow routine runs of that size. While the actual calculational (CPU) time was 11 days on the VAX/VMS 11 for all section $\leq$ 1800 nucleotides long in the 2600 nucleotide sequence, the total elapsed time was about two months.

Thus, any modification aimed at accelerating the code, should first and foremost reduce the elapsed time, i.e. the random search in the matrix. The modifications suggested here reduce drastically the elapsed time and to a smaller extent also the execution (CPU) time.

(b) Principles

Basically, the approach reverts to the original algorithm. It improves its execution in two distinct ways. Firstly, in the modified algorithm, references to already computed sections are ordered, namely, there are no random searches. Secondly, by keeping more information than previously, (though owing to tight packing the memory requirements do not increase), external references are needed much less often. At present, for all given (i,j) sections, where $j-i \leq 1000$ nucleotides, there is a single disk reference. For a given $j > 1000$ and varying i's, there are less than j disk references. For a chain n nucleotides long, there are then $<n^3/6000$ references.

Previouly, the programmed secondary structure algorithm required $\sim (cn)^3$ references. (c is the sum of the number of tries to locate all converging arms plus the number of disk references in order to read their corresponding energies. Also added is the disk reference for finding the energy of the left handside section, (i,k-1). As shown in equations 1 and 2, energy optimization also required reading E (i, j-1) for each i and j, causing additional n^2 references, which, as will be demonstrated later, are not present in the improved approach.)

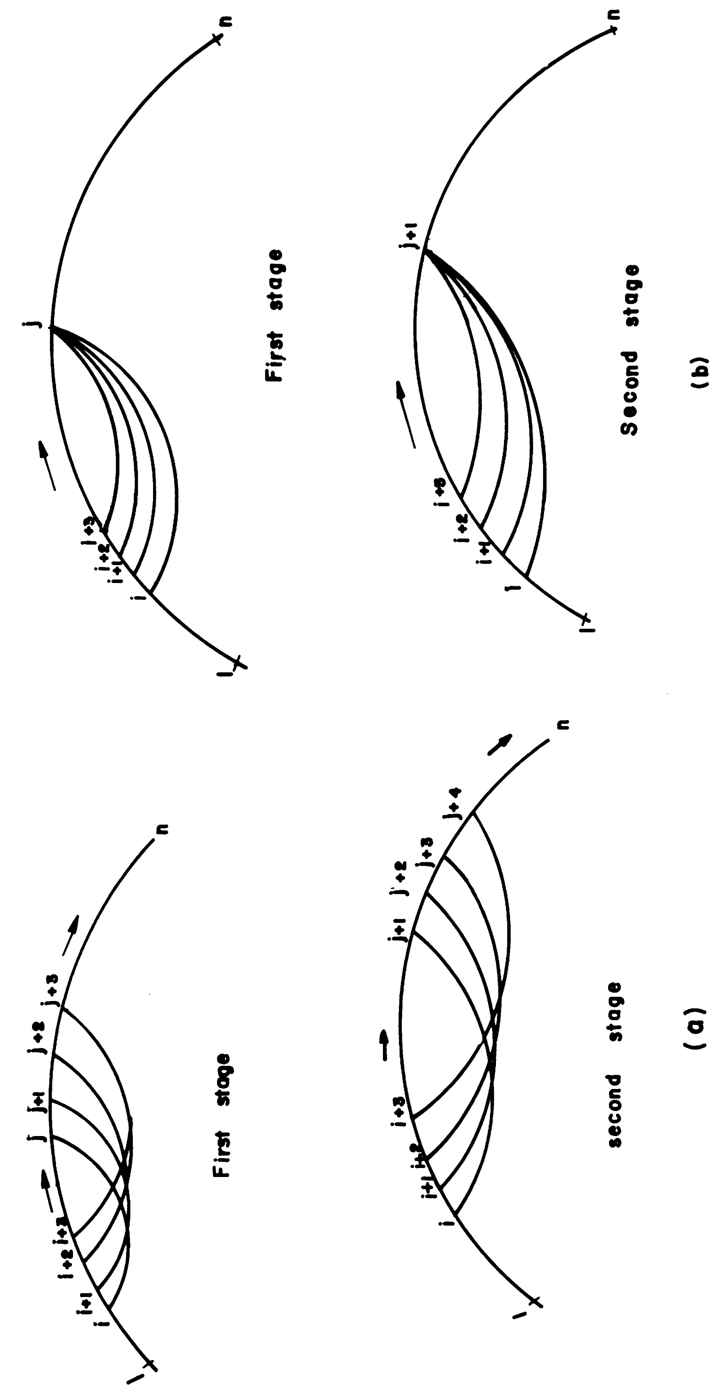
In addition, as pointed above, disk reading is now ordered. All vectors are read consecutively. They and the temporary vectors kept in the physical memory, contain the information needed to avoid the previous, numerous, random disk searches.

(c) Description of the Implemented Changes

The implemented changes are: (i) restructuring of the algorithm, (ii) vectorizing storage of all necessary information, (iii) reduction of the number of operations, (iv) requirement that the sum of the energies of each unpaired region and its closing double stranded stem be $\leq$ o and (v) changes in energy values.

(i) The restructured algorithm.

Previously, the optimal structures were computed for all given sections of the same fixed size which were sequentially shifted throughout the molecule by incrementing simultanuously i (the beginning of the section) and j (its end). Presently, the accelerated algorithm computes the optimal structures for all sections ending with the same j and with sequentially shifted beginnings, i.e. for the sections (i,j), (i+1,j), (i+2,j), (i+3,j).... Next, j is incremented, $j \rightarrow j+1$, and the corresponding calculations are executed for the sections (i,j+1), (i+1, j+1), (i+2,j+1), (i+3,j+1).... This procedure is repeated until j = n, the last nucleotide of the molecule. Within each section,


1
i
i+1
i+2
i+3
j
n
First stage
1
i
i+1
i+2
i+3
j+1
n
Second stage
(b)
1
i
i+1
i+2
i+3
j
j+1
j+2
j+3
n
First stage
1
i
i+1
i+2
i+3
j+1
j+2
j+3
j+4
n
second stage
(a)

the k variable travels from i to j as was already described (section 2). The difference between the previous and the present procedure is shown in Figure 4.

(ii) The stored information

As was pointed out in section 3, correct energy calculations necessitate knowledge of the types of base pairs, the sum of unpaired bases (if any) closed by the newly tried base pair and the knowledge of the distribution of the unpaired bases, i.e. whether they are on one side (a bulge) or both (hairpin or internal loop). This information is stored in compact form in the appropriate vectors.

Instead of the (i,j) matrix used previously (1,3) we now have two series of vectors. The first are those of fixed, given, j's. They contain all the relevant information on all sections ending in the previous base only and are therefore named j-1 vectors. The second series of vectors is more massive. It contains all relevant information pertaining to all i's. The energies of all sections starting with the same given, i and ending with different j's form a single vector. For each given i there are then separate "i" vectors, containing the necessary information.

Let us see how these,very simple, changes bring about a drastic reduction in disk searches and thus in I/O time. As seen in equations 1 and 2, secondary structure calculations demand the optimal energies of sections either starting with i or ending in j-1. The latter is clearly easily accessible, since all sections just computed end, precisely, in

Fig. 4. A comparison of the original algorithm and the presently modified and accelerated one. (a) An illustration of the procedure used in the basic algorithm for maximal number of matchings (3). It was also used in the version adapted for prediction of secondary structures with minimal energies(1). The section size, for which the optimal structure is calculated, is fixed. In the first stage it is j-i. In the second stage, the section size is j+1-i. In the third stage (not drawn) it is j+2-i,etc. In every step, within a given stage, i and j are increased simultanuously. (b) An illustration of the restructured algorithm. For every given j, or stage, the section size decreases. In the first stage, $1 \leq i < j - 4$. In the second stage, $1 \leq i < (j+1) - 4$. In the third stage (not drawn) $1 \leq i < (j+2) - 4$, etc.

j-1. These values need not then be stored on the disk at all, as they are used shortly after their calculation. The energies of the sections starting in i are necessarily stored on the disk, but they are recalled in an ordered fashion.

It would seem that the problem of the haphazard searches required for Eq. 3 is still unsolved. However, since in addition to the energies, the number of unpaired bases and the types of the outermost base pairs in all sections are kept, these searches are eliminated. Note also, that the values needed are, again, those of the sections ending in j-1, which were last registered.

The procedure, then, entails knowing and keeping five values for every section (i,j). We keep the energy, optimal pairing of base j and of base i, the number of unpaired bases and the types of the outermost base pairs in the section. Each of these values is kept twice, in the physical memory of the computer according to its j and packed, ordered according to its i.

(iii) Reduction of Operations was also achieved in the code. Details are given in section (d).

In addition, two physical criteria were implemented: (iv) requirement that the sum of the positive energy of the unpaired region and the negative energy of its closing stem be $\leq$ o and (v) some changes in the free energy assignments. The requirement that the total energy of such a configuration (see Fig. 3) be $\leq$o, is based on the simple physical consideration that a closed structure will not form unless it is more stable than the one which remains open. In practice, this principle eliminates numerous potential base pairs and thus accelerates and simplifies the procedure.

This requirement (iv), in turn, also results in relatively sparsely filled vectors which can be packed further by employing special programming techniques designed for such a purpose. The usage of these efficient techniques causes a further reduction, by about one third, in the number of disk references (and I/o time).

Previously, the energy values were taken from Salser (20) who extended, by computations, the original experimental values (6-11). Nussinov et al. (18) have shown that small changes in free energy assignments do not affect, in general, folding of the RNA molecules. Nonetheless, we have decided to exchange the stair-like energy curves (ref. 20) for the more realistic, calculated, inter- and extrapolated, smooth ones.

(d) The Algorithm

This dynamic algorithm computes the optimal structure for the whole molecule by employing, in essence, three nested "do" loops. Starting with a small j (marking the end of the section) we increment it gradually till j = n. For each j, i (marking the beginning of the section) is shifted, so that $1 \leq i \leq j - 4$. For each given i and j, k travels from i to j-4 (since the minimal number of bases in a hairpin is 3). As in the previous algorithm, we attempt forming a base pair between base k and base j.

Detailed calculations are carried out only if initial analysis indicates that the energy is favorable (see (iv), previous section) and that the double helical section does not begin or end in a GU base pair.

If, by matching base k with base j the above conditions are fulfilled, the secondary structure computations are carried out. The k, which when base paired with base j yields the minimal energy for the section (i, j), is chosen. The present algorithm calculates the energy differently than the previously described one (1). The new algorithm does not contain any matrix and there is no laborious partial backtracking. Instead, it uses the "j-1" vectors (see section (ii)) possessing the needed information. Four vectors are employed for that purpose: the vector with the energy of the section (k+1, j-1), the vector containing the number of unpaired bases between the newly formed base pair and its preceding ones for the (k+1, j-1) section (see Fig. 2), the vector containing the types of the preceding base pairs (AU, GC or GU) and the vector of the positions of the pairings. With this information, the energy of the kj base pair is easily read from the data. The energy of the (i, k-1) section is read from the i vector (see section (ii)).

The chosen k is registered, along with the corresponding energy. In addition, the number of unpaired bases and types of the new, outermost base pairs are updated. This current data and the information to which base i is hydrogen bonded, is registered twice. It is registered in the j vectors and in the i vectors. Whereas the latter vectors are all kept, packed, till the end of the run (and afterwards if we so wish) the j vectors are temporary. Each j vector is written on the previous j-1 vector. This is possible since, in the new algorithmic construction, after each (i, j) computation, the (i, j-1) would not be needed again.

At the end of the procedure (i.e. when j = n), we possess all the information for the whole molecule. This aspect of the algorithm was already described and used by Nussinov and Tinoco (16) to study the sequential folding of an RNA molecule. The actual secondary structure of the molecule is attained by a simple backtracking procedure in a manner similar to that described by Nussinov and Jacobson (1). The difference is that whereas there (1), the backtracking follows the j's (i.e. to which base j is paired etc.) here we follow the i's, since for all i's such information is present and stored.

The new code also contains the useful feature implemented previously (17,18). At fixed intervals results of all computations are written and stored on a tape. Thus, computer or power failure do not cause loss of computation time already used. At the end of the run all information is present on the tape.

5. DISCUSSION

Secondary structure computation is today a recurring theme in molecular biology. With the present experimental techniques, exact structural predictions for long sequences are still impossible. We may obtain information of the overall shape (e.g. by electron microscopy) or knowledge whether a specific nucleotide is paired or unpaired. It thus appears that the best approach at present is merging experimental data into fast computer algorithms.

Lately, the secondary structure predicting codes (1,2) were technically made easier to run (e.g. ref. 21), enabling wide usage of these programs. However, the basic logic of the original dynamic algorithm (3) and the codes which stemmed from it (1,2) were untouched. Pieczenik and Garber (22) have also modified the original algorithm, by treating consecutive potential base pairings as a single unit on the chain.

New algorithms have recently been published by Dumas and Ninio(23) and by Goad and Kanehisa (24). They compute either locally optimal structure (24) or overall lowest energy structure for relatively short sequences (23) up to 150 nucleotides long.

The restructured algorithm presented here greatly improves and facilitates secondary structure computations, especially of large, single stranded polynucleotide chains. For very long molecules, the exact, minimal energy algorithm can theoretically be a hundred times faster, since external memory references (that is, I/o time) are reduced from (random) n^3 to (ordered) $n^3/6000$.

Secondary structure computations of very long sequences can, then, be achieved in a reasonable time. Biochemical data , if present, can easily be implemented into the code, resulting in a biologically more reliable structure.

The program was written and run on the CDC 6600 at Tel Aviv University Computer Center.

*To whom all correspondence should be addressed

[2]Present address: Theoretical Biology, Group T-10, MS M710, Los Alamos National Laboratory, Los Alamos, NM 87545, USA

REFERENCES

1. Nussinov, R. and Jacobson, A.B., (1980) Proc. Natl. Acad. Sci. USA, 77, 6309-6313.
2. Zuker, M. and Stiegler, (1981) Nucl. Acids Res. 9, 133-148.
3. Nussinov, R., Pieczenik, G., Griggs, J.R. and Kleitman, D.J. (1978) SIAM J. Appl. Math 35 (1) 68-82.
4. Griggs, J.R. (1977). Dissertation (M.I.T.).
5. Nussinov, R. (1977). Dissertation (Rutgers Univ.).
6. Delisi, C. and Crothers, D.M. (1971) Proc. Natl. Acad. Sci. USA, 68, 2682-2685.
7. Tinoco, I., Jr., Uhlenbeck, O.C. and Levine, M.D. (1971) Nature (London) 230, 362-367.
8. Gralla, J. and Crothers, D.M. (1973)J. Mol. Biol. 73, 497-511.
9. Tinoco, I.,Jr., Borer, P.N., Dengler, B., Levine, M.D., Uhlenbeck, O.C., Crothers, D.M. and Gralla, J. (1973) Nature (London) New Biol. 246, 40-41.
10. Borer, P.N., Dengler, B., Tinoco, I., Jr. and Uhlenbeck, O.C. (1974), J. Mol. Biol. 86, 843-853.
11. Gralla, J. and Crothers, D.M. (1973) J. Mol. Biol. 78, 301-319.
12. Waterman, M.S. and Smith, T.F. (1978) Math. Biosci. 42, 257-266.

13. Pipas, J.M. and McMahon, J.E. (1975) Proc. Natl. Acad. Sci. USA 72, 2017-2021.
14. Studnicka, G.M., Rahn, G.M., Cummings, I.W. and Salsor, W.A. (1978) Nucleic Acids Res. 5, 3365-3387.
15. Waterman, M.S. (1978) Studies in Foundations and Combinatorics, Advances in Mathematics Supplementary Studies, 1, 67-211.
16. Nussinov, R. and Tinoco, I., Jr. (1981) J. Mol. Biol., 151, 519-533.
17. Nussinov, R., Tinoco, I., Jr. and Jacobson, A.B. (1982) Nucleic Acids Res. 10, 351-364.
18. Nussinov, R., Tinoco, I., Jr. and Jacobson, A.B. (1982) Nucleic Acids Res. 10, 341-350.
19. Stiegler, P., Carbon, P., Zuker, M., Ebel, M.,-P. and Ehresmann, C. (1981) Nucl. Acids Res. 9, 2153-2172.
20. Salser, W. (1977) Cold Spring Harbor Symp. Quant. Biol. 62, 985-1002.
21, Auron, P.E., Rindone, W.P., Vary, C.P.H., Celentano, J.J. and Vournakis, J.N. (1982) Nucl. Acids Res. 10, 403-420.
22. Pieczenik, G. and Garber, G., Unpublished.
23. Dumas, J.,-P. and Ninio, J. (1982) Nucl. Acids Res. 10, 197-206.
24. Goad, W.B. and Kanehisa, M.I. (1982) Nucl. Acids Res. 10, 247-264.