SIAM J. APPL. MATH.
Vol. 39, No. 1, August 1980

COMPUTATION OF GENERATING FUNCTIONS FOR BIOLOGICAL MOLECULES*

J. A. HOWELL,† T. F. SMITH‡ AND M. S. WATERMAN¶

Abstract. The object of this paper is to give algorithms and techniques for computing generating functions of certain RNA configurations. Combinatorics and symbolic computation are utilized to calculate the generating functions for small RNA molecules. From these generating functions, it is possible to obtain information about the bonding and structure of the molecules. Specific examples of interest to biology are given and discussed.

1. Introduction. Our interest in this paper is in describing the intra-molecular bonding that takes place within a single-stranded RNA in solution. This problem is of significant importance in biology and has been the subject of several previous studies (see [9] for references). The approach we take is unique and a description of the approach is given after a discussion of the bonding of interest.

The linear sequence of bases of an RNA is said to be its primary structure. These bases are adenine (A), guanine (G), cytosine (C), and uracil (U). The primary structure of an RNA, then, is a word over an alphabet of four letters. When the only bonds of an RNA are those of the primary structure, the RNA is said to be a random coil.

An RNA does not usually remain a random coil. Instead, it folds back on itself and forms new bonds, creating helical regions. The usual pairing (or bonding) rules to create these helical regions are: An A can bond with a U and a G with a C forming what are known as Watson–Crick base pairs. This bonded folding of the RNA primary structure is referred to by biologists as secondary structure. Other less restrictive, non-Watson–Crick, bonding interactions result in what are referred to as tertiary foldings. Our concern here is to describe and explore the secondary structure folding only. We now give a mathematical definition of secondary structure. Let $\mathbf{s} = s_1 s_2 \cdots s_n$ be the RNA sequence. In addition to the requirement that adjacent bases are bonded (primary structure), secondary structure requires that (i) each base can be bonded to at most one other non-adjacent base and (ii) s_i and s_j form a bond only if they satisfy the Watson–Crick pairing rules and (iii) if s_i and s_j are bonded, then any bonding of s_k, $i < k < j$, must be with s_l, $i < l < j$. This last condition prohibits certain unobserved knot structures [9].

For an example, consider $\mathbf{s} = AACGGGCGGGACCCUUCAACCCUU$. A secondary structure for this word is given in Fig. 1. The unpaired regions II and I are, together, called an interior loop, and the unpaired A near III is called a hairpin loop.

Our approach to the problem of secondary structure is to utilize generating functions. Our generating functions correspond to the partition functions of statistical mechanics. The partition or generating functions for simpler situations, such as considered by Poland and Scheraga [7], have required much effort to extract the infromation of interest. Here, we utilize combinatorics and symbolic computation to obtain the generating functions of small molecules. Then the desired information can be read directly from the generating function. Even for the simplest molecules of interest, this is the first time such calculations have been performed. As will become evident, a computer is essential in performing the analysis.

* Received by the editors August 14, 1978.

† Los Alamos Scientific Laboratory, Los Alamos, New Mexico 87545.
‡ Department of Mathematics, Northern Michigan University, Marquette, Michigan 49855.
¶ Los Alamos Scientific Laboratory, Los Alamos, New Mexico 87545.

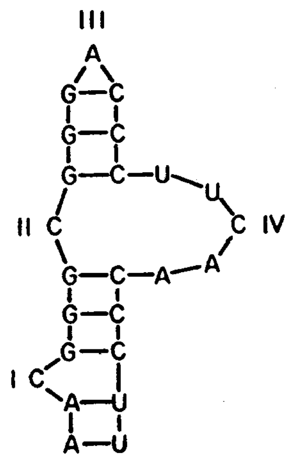


FIG. 1. *A secondary structure for AACGGGCGGGACCCUUCAACCCUU.*

2. Combinatorics. In this section, we consider the number of secondary structures of various types. Our aim is to introduce concepts and techniques useful for the generating function and to indicate the large number of possible structures. For simplicity, we will not handle a specific RNA with associated pairing rules. Instead, all bondings will be considered possible. (In the next section this assumption will be dropped.) Algorithms based on the same ideas as this section have been independently developed for secondary structure prediction [10].

THEOREM 1. *Let $R(n)$ be the number of secondary structures for a molecule of length n. Then*

$$R(n+1)=R(n)+\sum_{j=1}^{n-1} R(j-1)R(n-j) \tag{1}$$

where $n \geqq 2$ and $R(0)=R(1)=R(2)=1$.

Proof. The only secondary structure for $n = 1$ or 2 is the random coil. Assume $R(k)$ is known for $1 \leqq k \leqq n$. For a molecule of length $n+1$, either $n+1$ is not paired or it is paired with j where $1 \leqq j \leqq n-1$. This implies

$$R(n+1)=R(n)+\sum_{j=1}^{n-1} R(j-1)R(n-j),$$

since the strings $1-2-\cdots-(j-1)$ and $(j+1)-(j+2)-\cdots-n$ are free to form any secondary structure. (See Fig. 2.)

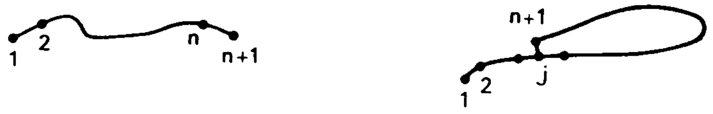


FIG. 2. *The configurations of* $1-2-\cdots-(n+1)$.

For our next result, we examine $R(n)$ in more detail. Our interest is in how many members of $R(n)$ have i bonds (or pairs). Let

$$S^n = (s_0^n, s_1^n, \cdots)$$

where

$$s_i^n = \begin{cases} 1, & \text{if } i = 0. \\ \text{the number of secondary structures with } i \text{ bonds} & \\ \quad \text{for a molecule of length } n, & \text{if } i > 0. \end{cases}$$

Also define a shift operator ζ by $\zeta(a_0, a_1, a_2, \cdots) = (0, a_0, a_1, \cdots)$ and the Cauchy product of $\mathbf{a}$ and $\mathbf{b}$ by $\mathbf{c} = \mathbf{a} * \mathbf{b}$ where

$$c_m = \sum_{i=0}^{m} a_i b_{m-i}.$$

THEOREM 2. *We have*

$$\mathbf{S}^0 = \mathbf{S}^1 = \mathbf{S}^2 = (1, 0, 0, \cdots)$$

and for $n \geqq 2$,

(2) $$\mathbf{S}^{n+1} = \mathbf{S}^n + \sum_{j=1}^{n-1} \zeta(\mathbf{S}^{j-1} * \mathbf{S}^{n-j}).$$

The form of Theorem 2 is similar to Theorem 1. This result can be considered a vector generalization of the Catalan numbers.

Proof. The only structure for $n = 1, 2$ is the unpaired string, so $\mathbf{S}^1 = \mathbf{S}^2 = (1, 0, \cdots)$.

In general, $n+1$ is unpaired or $n+1$ is paired with j, $1 \leqq j \leqq n-1$. If $n+1$ is unpaired, s_{i+1}^n structures exist with $i+1$ bonds. Otherwise, $n+1$ is paired with j, and i additional bonds are needed. If there are k bonds in the structure for $1-2-\cdots-(j-1)$, then there must be $i-k$ bonds in the structure for $(j+1)-(j+2)-\cdots-n$. Thus

$$s_{i+1}^{n+1} = s_{i+1}^n + \sum_{j=1}^{n-1} \left(\sum_{k=0}^{i} s_k^{j-1} s_{i-k}^{n-j} \right).$$

The term in parentheses is the ith coordinate of $\mathbf{S}^{j-1} * \mathbf{S}^{n-j}$ and the result follows.

As illustration, we calculate $\mathbf{S}^0, \mathbf{S}^1, \cdots, \mathbf{S}^6$.

$$\mathbf{S}^0 = \mathbf{S}^1 = \mathbf{S}^2 = (1, 0, 0, 0, \cdots),$$
$$\mathbf{S}^3 = (1, 1, 0, 0, \cdots),$$
$$\mathbf{S}^4 = (1, 3, 0, 0, \cdots),$$
$$\mathbf{S}^5 = (1, 6, 1, 0, \cdots),$$

and

$$\mathbf{S}^6 = (1, 10, 6, 0, \cdots).$$

$\mathbf{S}^6$, for example, indicates 1 unpaired structure, 10 structures with one bond, and 6 structures with two bonds. Also,

$$R(6) = \sum_{i \geqq 0} s_i^6 = 1 + 10 + 6 = 17.$$

The corresponding structures for $\mathbf{S}^6$ are shown in Fig. 3.

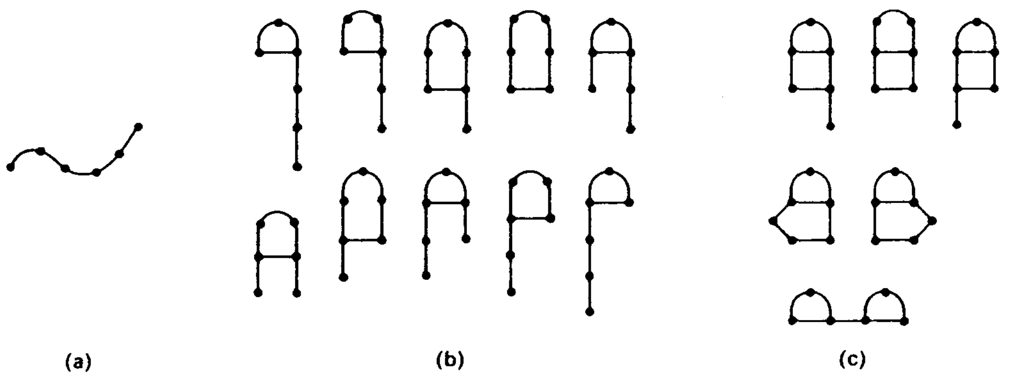


FIG. 3. *Secondary structures on* 1-2-3-4-5-6. (a) *Random coil,* (b) *the* 10 *single-bond structures,* (c) *the* 6 *double-bond structures.*

3. Generating functions. In this section we wish to consider the computation of generating functions of a given molecule of small or moderate size. We treat configurations possible for a given molecule by imposing specific pairing rules. The generating function is a polynomial whose terms indicate the number of possible structures with certain configurations. As mentioned above, the concept of generating function in mathematics corresponds to the concept of partition function in statistical mechanics.

To begin with, we consider AU and GC pairs but other structural components (such as adjacent bonds, bulges and loops) are ignored. No loops of length less than m are allowed. (This is the only structural constraint imposed.) Let $s = s_1 s_2 \cdots s_N$ be the given RNA of length N and define

$$P_{ij} = \begin{cases} \alpha, & \text{if } \{s_i, s_j\} = \{A, U\}, \\ \beta, & \text{if } \{s_i, s_j\} = \{C, G\}, \\ 0, & \text{otherwise.} \end{cases}$$

This definition reflects the pairing rules given in the introduction. Sometimes GU bonds are included but we have omitted them for speed of computation. Conceptually, they are easily included.

Next we define the generating function, $Z_{1,N}$, for $s = s_1 s_2 \cdots s_N$ by

$$Z_{1,N} = \sum_{j,k \geq 0} a_{jk} \alpha^j \beta^k,$$

where a_{jk} is the number of secondary structures for s with j $AU(\alpha)$ bonds and k $GC(\beta)$ bonds.

In order to illustrate the concept of the generating function, consider $s = AUAUAUA$. The nine possible secondary structures are shown in Fig. 4. There is one structure with no bonds, six structures with one α-bond, and two structures with two α-bonds. Consequently, the generating function, $Z_{1,7}$, is

$$Z_{1,7} = 1 + 6\alpha + 2\alpha^2,$$

where $m = 1$.

For calculation purposes, it is useful to define $Z^i_{l,n}$ to be the generating function for $s_l s_{l+1} \cdots s_n$ with i bonds $(1 \leq l < n \leq N)$. Of course,

$$Z_{l,n} = \sum_{i \geq 0} Z^i_{l,n}.$$

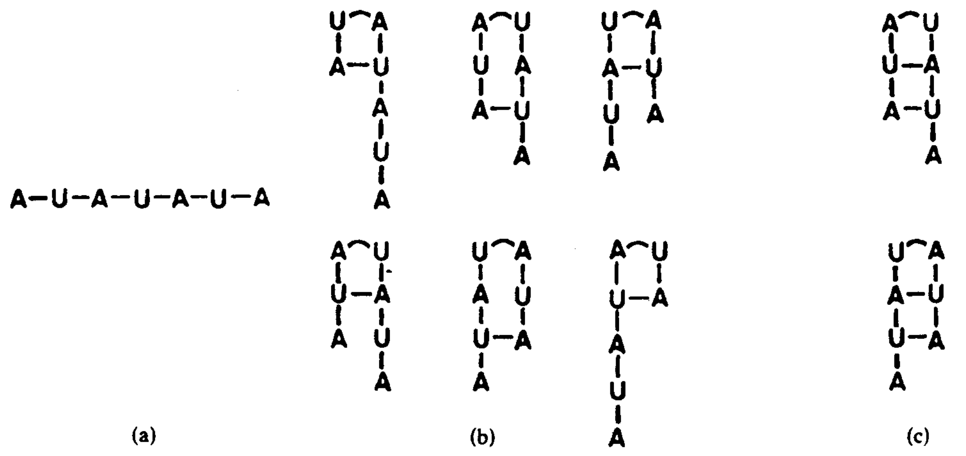


FIG. 4. *Secondary structures on AUAUAUA.* (a) *Random coil,* (b) *the 6 single-bond structures,* (c) *the 2 double-bond structures.*

THEOREM 3. *The function $Z^{i}_{l,n}$ satisfies*

$$Z^{i+1}_{l,n+1} = Z^{i+1}_{l,n} + \sum_{j=l}^{n-m} \sum_{k=0}^{i} Z^{k}_{l,j-1} Z^{i-k}_{j+1,n} P_{j,n+1}, \tag{3}$$

where m is the minimum loop size.

Proof. To establish Equation (3), note that s_{n+1} is either bonded or unbonded. (See Fig. 2.) If s_{n+1} is unbonded, the contribution to the generating function is $Z^{i+1}_{l,n}$. Otherwise, s_{n+1} is bonded with $s_j (l \leqq j \leqq n-m)$. If k bonds are formed in $s_l \cdots s_{j-1}$, then, in order to form a total of $i+1$ bonds, $i-k$ bonds must be formed in $s_{j+1} \cdots s_n$. The relevant term of $Z^{i+1}_{l,n+1}$ is then

$$Z^{k}_{l,j-1} Z^{i-k}_{j+1,n} P_{j,n+1}.$$

The last term of the product, $P_{j,n+1}$, is to include the contribution from an $s_j s_{n+1}$ bond. Equation (3) then follows.

Next, we generalize $Z^{i}_{l,n}$ to include nearest neighbor effects. To do this we introduce a variable η which is to be included any time adjacent bonds occur. The variable η is to be interpreted as the statistical weight associated with the so called stacking energy resulting from nearest neighbor pairs of Watson–Crick bonds. In general, there are 16 possible stacked pairs of bonds and the stacking energies are not all equivalent. The implied differences are small compared with the differences in α and β (resulting from the differences in AU and GC bond formation). In addition, the inclusion of 16 different ηs is not difficult in principle but makes these calculations impractical. Thus a single statistical weight η is included here as a first approximation. No work other than [9] rigorously allows such an approximation. For example, the generating function for $AUAUAUA$ becomes

$$Z_{1,7} = 1 + 6\alpha + 2\alpha^2\eta.$$

(See Fig. 4.)

To generalize Equation (3), $Z^{i+1}_{l,n+1}$ must be decomposed into two components:

$$Z^{i+1}_{l,n+1} = X^{i+1}_{l,n+1} + Y^{i+1}_{l,n+1}, \tag{4}$$

where $X^{i+1}_{l,n+1}$ is the generating function for $i+1$ bonds on $a_l \cdots a_{n+1}$ and a_l *does not* bond with a_{n+1}, and where $Y^{i+1}_{l,n+1}$ is the generating function for the same string but a_l *does* bond a_{n+1}.

The formula for $Y^{i+1}_{l,n+1}$ is easily obtained:

$$Y^{i+1}_{l,n+1} = P_{l,n+1}(\eta Y^{i}_{l+1,n} + X^{i}_{l+1,n}). \tag{5}$$

However, $X^{i+1}_{l,n+1}$ is more complicated and we include the result as a theorem. (Equation (5) is also established in the proof of Theorem 4.)

THEOREM 4. *The term $X^{i}_{l,n}$ satisfies*

$$\begin{aligned} X^{i+1}_{l,n+1} &= (X^{i+1}_{l+1,n} + Y^{i+1}_{l+1,n}) \\ &\quad + \sum_{k=0}^{i} \sum_{j=l+1}^{n-m} P_{j,n+1}(X^{k}_{l+1,j-1} + Y^{k}_{l+1,j-1})(X^{i-k}_{j+1,n} + \eta Y^{i-k}_{j+1,n}) \\ &\quad + \sum_{k=0}^{i} \sum_{j=l+m+1}^{n} P_{l,j}(X^{k}_{j+1,n} + Y^{k}_{j+1,n})(X^{i-k}_{l+1,j-1} + \eta Y^{i-k}_{l+1,j-1}) \\ &\quad + \sum_{p=1}^{i} \sum_{q=0}^{i-p} \sum_{j=m+l+1}^{n-m-1} Y^{p}_{l,j} \left\{ \sum_{k=j+1}^{n-m} Y^{i+1-p-q}_{k,n+1}(X^{q}_{j+1,k+1} + Y^{q}_{j+1,k+1}) \right\}. \end{aligned} \tag{6}$$

Proof. To justify Equation (5) for $Y^{i+1}_{l,n+1}$, note that $Y^{i+1}_{l,n+1} = 0$ unless s_l bonds s_{n+1}. In any case $P_{l,n+1}$ is multiplied by an appropriate term for $s_{l+1} \cdots s_n$. If s_{l+1} does not bond s_n, then multiply by $X^{i}_{l+1,n}$. If s_{l+1} bonds s_n, then multiply by $\eta \cdot Y^{i}_{l+1,n}$ to account for the formation of a nearest neighbor.

In order to establish Equation (6) for $X^{i+1}_{l,n+1}$, four cases must be considered. Each case corresponds to a term of Equation (6).

Case (i). Both s_l and s_{n+1} are not bonded. In this case, the generating function is the generating function for $s_{l+1} \cdots s_n$, $Z^{i+1}_{l+1,n} = X^{i+1}_{l+1,n} + Y^{i+1}_{l+1,n}$.

Case (ii). s_l is unbonded and s_{n+1} is bonded. For this case, s_{n+1} must be bonded to s_j $(l+1 \leqq j \leqq n-m)$. If k bonds are formed on $s_{l+1} \cdots s_{j-1}$, then $i-k$ bonds must be formed on $s_{j+1} \cdots s_n$. If s_{j+1} bonds s_n, a nearest neighbor is formed, so that the term of $X^{i+1}_{l,n+1}$ is

$$P_{j,n+1}(X^{k}_{l+1,j-1} + Y^{k}_{l+1,j-1})(X^{i-k}_{j+1,n} + \eta Y^{i-k}_{j+1,n}).$$

Case (iii). s_l is bonded and s_{n+1} is not bonded. This case is similar to case (ii) and we omit these details.

Case (iv). s_l is bonded, s_{n+1} is bonded, and s_l is *not* bonded to s_{n+1}. To have this situation, s_l must bond s_j $(m+l+1 \leqq j \leqq n-m)$. s_{n+1} must bond s_k $(j+1 \leqq k \leqq n-m-1)$. The remaining bonds must be for the center structure $s_{j+1} \cdots s_{k-1}$. Therefore a typical term has the form

$$Y^{p}_{l,j} \cdot Y^{i+1-p-q}_{k,n+1}(X^{q}_{j+1,k-1} + Y^{q}_{j+1,k+1}).$$

We remark that, if $\eta \equiv 1$, then the sum of $X^{i+1}_{l,n+1}$ and $Y^{i+1}_{l,n+1}$ is equal to $Z^{i+1}_{l,n+1}$ of the earlier discussion (Equation (3)).

The complexity of the recursive functions X and Y makes obvious the need of a computer for their evaluation. Since the function values are symbolic expressions rather than numerical values, we use a symbolic manipulation language (MACSYMA) rather than a numeric language, such as FORTRAN. The next section describes the computer program.

4. A program implementation. This section describes a computer program written to compute the functions $X^i_{k,j}$ and $Y^i_{k,j}$. These generating functions are computed using the algebraic manipulation system MACSYMA [6]. MACSYMA runs on a DEC KL-10 at MIT. The configuration at the time of these runs was 0.5 million words of high-speed memory and a CPU speed of 1.5 million instructions per second (about $\frac{1}{10}$ the speed of a CDC 7600). Since MACSYMA is not presently portable, we used the MIT computer via dialup lines. As with most algebraic systems, all expressions are computed exactly. One of the features we exploit is the Algol-60-like syntax.

There are five recursive functions defined using this syntax: X, Y, $F1$, $F2$, and $F3$. In addition to these we use three functions which are built into MACSYMA: ENTIER, EXPAND, and SUM. The function ENTIER (R/S) computes the greatest integer less than or equal to the rational number R/S. EXPAND (exp) returns an expression in which all terms in the algebraic expression exp have been multipled out, that is, expanded. SUM (exp, I, L, U) returns an expression equivalent to

$$\sum_{I=L}^{U} \text{exp}.$$

In the following, N is the length of the molecule, M is the minimum loop size, and P is the matrix in § 3.

We define the functions X and Y as follows:

```
$X[I, K, J] :=$
    **if** $I = 0$ **then** 1 **else**
        **if** $J - K \leqq M + 1$ **then** 0 **else**
            **if** $I >$ ENTIER $((J - K - M)/2)$ **then** 0 **else**
                EXPAND $(X[I, K+1, J-1] + Y[I, K+1, J-1] + F1[I, K, J]$
                        $+ F2[I, K, J] + F3[I, K, J])$;

$Y[I, K, J] :=$
    **if** $I = 0$ **then** 0 **else**
        **if** $J - K \leqq M$ **then** 0 **else**
            **if** $P_{K,J} = 0$ **then** 0 **else**
                **if** $I >$ ENTIER $((J - K + 1 - M)/2)$ **then** 0 **else**
                    EXPAND $(P_{K,J} \cdot ($ETA $\cdot\ Y[I-1, K+1, J-1]$
                                        $+ X[I-1, K+1, J-1]))$;
```

We assume the initial conditions $X^0_{K,J} = 1$ and $Y^0_{K,J} = 0$. The second condition tested in both X and Y is related to the amount of bending that can occur in bonding together the Kth and Jth elements of the molecule. Condition three in X (four in Y) is related to the number of bonds possible with loop size M. Functions $F1$, $F2$, and $F3$ correspond to the second, third, and fourth terms of Equation (6). Thus, we have

```
$F1[I, K, J] :=$
    **if** $J - M - K < 2$ **then** 0 **else**
        SUM(SUM(**if** $P_{T,J} = 0$ **then** 0 **else**
            **if** $X[L, K+1, T-1] + Y[L, K+1, T-1] = 0$ **then** 0 **else**
                $P_{T,J} \cdot (X[L, K+1, T-1]$
                                $+ Y[L, K+1, T-1]) \cdot (X[I-1-L, T+1, J-1]$
                    $+$ETA $\cdot\ Y[I-1-L, T+1, J-1])$,
                                        $T, K+1, J-1-M), L, 0, I-1)$;
```

```
$F2[I, K, J] :=$
    **if** $J - M - K < 2$ **then** 0 **else**
        SUM(SUM(**if** $P_{K,T} = 0$ **then** 0 **else**
            **if** $X[L, T+1, J-1] + Y[L, T+1, J-1] = 0$ **then** 0 **else**
                $P_{K,T} \cdot (X[L, T+1, J-1]$
                                        $+ Y[L, T+1, J-1]) \cdot (X[I-1-L, K+1, T-1]$
                        $+ \mathrm{ETA} \cdot Y[I-1-L, K+1, T-1]),$
                                                    $T, K+M+1, J-1), L, 0, I-1);$

$F3[I, K, J] :=$
    **if** $J - 2 \cdot M - K < 3$ **then** 0 **else**
        SUM(SUM(SUM(**if** $P_{K,L} = 0$ **then** 0 **else** $Y[S, K, L] \cdot$
            SUM(**if** $P_{T,J} = 0$ **then** 0 **else**
                $Y[I-S-Q, T, J] \cdot (X[Q, L+1, T-1] + Y[Q, L+1, T-1]),$
                    $T, L+1, J-M-1), L, M+K+1, J-M-2),$
                                                    $Q, 0, I-1-S), S, 1, I-1);$
```

It is frequently the case that certain function values must be computed more than once in the solution of a problem. Therefore, we have chosen the array function which is a form of the function in MACSYMA that stores function values in an array once they have been computed. Thus, we write $X[I, K, J]$ instead of $X(I, K, J)$. For the examples discussed here this appears to produce a savings of approximately 20% in both time and memory.

Examples produced using this program are discussed in § 5. The primary limitation of this program is the size of the problem we can solve. The problems discussed here are among the largest we feel we can successfully handle. A larger memory would allow the solution of larger problems. In all the examples shown here, all of memory was used.

5. Examples.

Calculation of generating functions is performed for various reasons. As we discuss below, our main interest is in coefficients of $Z^i_{1,N}$ for larger i such that $Z^i_{1,N} \neq 0$. Then we present three segments of RNA molecules that are of interest in molecular biology. The leading terms of the generating function $Z_{1,N}$ are given, and, in two cases, the most favorable structures are drawn. We have developed an algorithm to produce the structures of interest which is presented in the next section. In addition, the form of the generating function gives valuable clues as to the structures.

The structure of an RNA or of an RNA segment has important biological functions. For example, transfer RNA (tRNA) functions in the assembly of amino acids into a linear sequence (word) called a protein. The role of tRNA is to move (transfer) amino acids from their free state in the cell to the assembly point (ribosome) for the protein. Its ability to perform this function requires a unique three-dimensional structure which arises from its secondary structure. Other important biological functions, such as the rate of protein synthesis, may be controlled by the secondary structure of the leading end of messenger RNA (mRNA) [3].

The next problem is to determine secondary structure from the primary structure (or word). Thermodynamic considerations suggest that those structures with the lowest free-energy will be the most probable structures. Roughly speaking, the low free-energy structures are those with the most bonds and nearest neighbors. If the terms of $Z_{1,N}$ are of the form $k\,\alpha^l \eta^n \beta^m$, the terms with the larger values of $l+m$ and n represent the more probable or more stable structures. That is, we wish the largest i such that $Z^i_{1,N} \neq 0$.

As our first example, consider *UACCAUAAGCCUAAUGGAGCGAAUU*, which is the leader of the Gal operon mRNA in the bacterium *E.coli* [8]. For our calculation, $N = 25$ and we let $M = 4$. We obtained

$$Z_{1,25} = \sum_{i=0}^{7} Z^i_{1,25} = \sum_{i=0}^{7} X^i_{1,25},$$

where

$$\begin{aligned}
X^0_{1,25} &= 1;\\
X^1_{1,25} &= 19\beta + 38\alpha;\\
X^2_{1,25} &= 3\eta\beta^2 + 61\beta^2 + 2\alpha\eta\beta + 293\alpha\beta + 7\alpha^2\eta + 241\alpha^2;\\
X^3_{1,25} &= 11\eta\beta^3 + 44\beta^3 + \alpha\eta^2\beta^2 + 54\alpha\eta\beta^2 + 449\alpha\beta^2\\
&\quad + \alpha^2\eta^2\beta + 62\alpha^2\eta\beta + 726\alpha^2\beta + \alpha^3\eta^2\\
&\quad + 59\alpha^3\eta + 309\alpha^3;\\
X^4_{1,25} &= 9\eta\beta^4 + 2\beta^4 + 91\alpha\eta\beta^3 + 187\alpha\beta^3\\
&\quad + \alpha^2\eta^3\beta^2 + 16\alpha^2\eta^2\beta^2 + 186\alpha^2\eta\beta^2 + 430\alpha^2\beta^2\\
&\quad + 13\alpha^3\eta^2\beta + 164\alpha^3\eta\beta + 316\alpha^3\beta + 7\alpha^4\eta^2\\
&\quad + 67\alpha^4\eta + 64\alpha^4;\\
X^5_{1,25} &= 37\alpha\eta\beta^4 + 8\alpha\beta^4 + 6\alpha^2\eta^2\beta^3 + 91\alpha^2\eta\beta^3\\
&\quad + 22\alpha^2\beta^3 + 9\alpha^3\eta^3\beta^2 + 42\alpha^3\eta^2\beta^2 + 126\alpha^3\eta\beta^2\\
&\quad + 66\alpha^3\beta^2 + 20\alpha^4\eta^2\beta + 98\alpha^4\eta\beta + 14\alpha^4\beta\\
&\quad + 2\alpha^5\eta^2 + 7\alpha^5\eta + 2\alpha^5;\\
X^6_{1,25} &= 4\alpha^2\eta\beta^4 + 3\alpha^3\eta^2\beta^3 + 3\alpha^3\eta\beta^3 + 15\alpha^4\eta^3\beta^2\\
&\quad + 22\alpha^4\eta^2\beta^2 + 28\alpha^4\eta\beta^2 + 2\alpha^5\eta^2\beta + 6\alpha^5\eta\beta;\\
X^7_{1,25} &= 2\alpha^5\eta^3\beta^2;\\
X^i_{1,25} &= 0, \qquad i \geqq 8;\\
Y^i_{1,25} &= 0, \qquad i \geqq 0.
\end{aligned}$$

In the table which follows, C_i denotes the sum of coefficients of $X^i_{1,25}$. These sums indicate the number of possible secondary structures that can be assumed by the molecule.

i	1	2	3	4	5	6	7
C_i	57	607	1717	1553	550	83	2

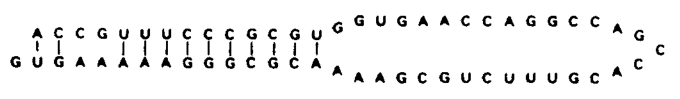
As can be seen from the polynomials $X^i_{1,25}$, there are a large number of possible structures. Those of interest are $X^6_{1,25}$ and $X^7_{1,25}$. As the largest number of bonds and the highest power of η both occur in $X^7_{1,25}$, those two structures are very likely to occur in solution. The example $X^7_{1,25}$ required about 5 minutes of CPU time.

The second sequence considered is the leader of the Lac *i*-gene mRNA from the bacterium *E. coli* [2]:

ACCGUUUCCCGCGUGGUGAACCAGGCCAGC

CACGUUUCUGCGAAAACGCGGGAAAAAGUG.

A proposed structure [2] resulting in a large loop containing the actual initiation site *ACG* is shown:


A C C G U U U C C C G C G U G G U G A A C C A G G C C A G C
G U G A A A A A G G G C G C A A A A G C G U C U U U G C A C

The length ($N = 60$) of this molecule presents us with a new problem. For a loop size of reasonable length ($M = 5$, say), the problem is too large to handle. Since the structure proposed in the literature has a large loop size, for comparison we used $M = 30$. The largest i such that $Z^i_{1,60} \neq 0$ was $i = 12$ and

$$\begin{aligned} Z^{12}_{1,60} = X^{12}_{1,60} &= \alpha^5\eta^{10}\beta^7 + 8\alpha^5\eta^9\beta^7 + 16\alpha^5\eta^8\beta^7 \\ &+ 12\alpha^5\eta^7\beta^7 + 3\alpha^5\eta^6\beta^7 + \alpha^4\eta^9\beta^8 + 7\alpha^4\eta^8\beta^8 \\ &+ 9\alpha^4\eta^7\beta^8 + 3\alpha^4\eta^6\beta^8. \end{aligned}$$

While 60 structures occur in $X^{12}_{1,60}$, the "best" structure is that corresponding to $\alpha^5\eta^{10}\beta^7$, and this structure in fact corresponds to the proposed structure shown above. This example required 3.2 minutes of CPU time.

Finally, we consider the promoter site segment for the start of the *E. coli* tRNA$_{tyr}$ precursor RNA [2].

GCGGGGCGCAUCAUAUCAAAUGACGCGCCGC.

For this calculation, $N = 31$ and $M = 5$. Since the ends bond, we have some $Y^i_{1,31} \neq 0$. In fact, the largest i such that $Z^i_{1,31} \neq 0$ is $i = 11$ and

$$Z^{11}_{1,31} = Y^{11}_{1,31} = 4\alpha^3\eta^8\beta^8 + 6\alpha^3\eta^7\beta^8.$$

This calculation required 7.1 minutes of CPU time. As these ten structures are very close in terms of bonds and nearest neighbors, we present them in Figure 5. It would be difficult to chose a "best structure" from these ten, and we feel this illustrates the need for a rigorous examination of secondary structure. That is, simply exhibiting a structure that looks "good" is not adequate for a determination of optimal structures.

6. Structure determination. In the discussion which follows we outline an algorithm for deducing the structures of interest. While this algorithm is not rigorous in a familiar sense [5, p. 357], we feel that the concepts are intuitive enough to be understood in this form. A complete ALGOL-like program takes several pages to list and would surely convey less information.

Our algorithm produces a tree from which we can directly read the secondary structures of an RNA associated with a term of the generating function. In order to describe the algorithm, we need to define certain terms. These terms are familiar in the theory of languages [4].

First there is a syntactic category from which the structures of our RNA molecule are derived. This category consists of "nonterminals" or "variables". These are the X's, Y's, $F1$'s, $F2$'s, and $F3$'s. Secondly, there are the symbols which carry the structural

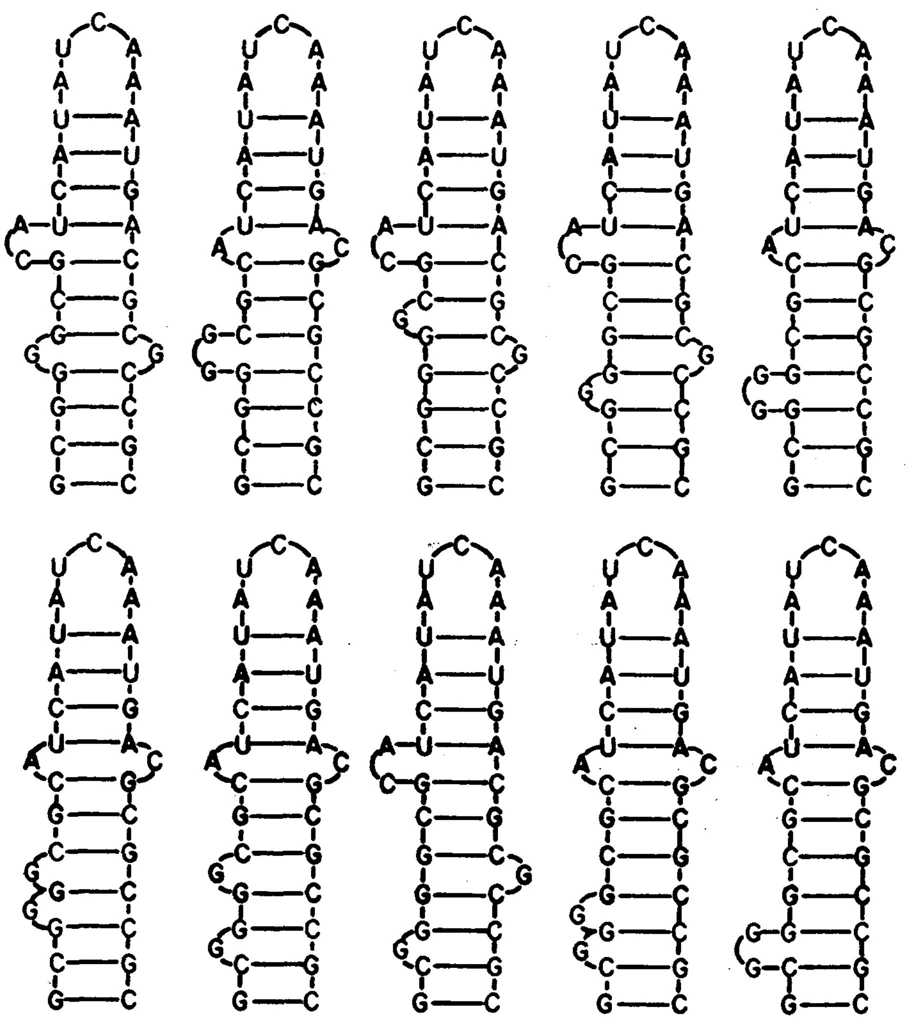


FIG. 5. *The* 10 *structures corresponding to* $Z^{11}_{1,31}$.

information themselves. These are called "terminals" and are the P_{ij}. (Recall that P_{ij} tells us if base i bonds with base j.)

The third concept is the relation that exists between the variables and terminals. These relationships are called "rewrite rules." These may be read almost directly from the equations (5) and (6). We write them as follows:

$$X^i_{k,j} \rightarrow X^i_{k+1,j-1} + Y^i_{k+1,j-1} + F1^i_{k,j} + F2^i_{k,j} + F3^i_{k,j},$$

$$Y^i_{k,j} \rightarrow P_{k,j} \cdot (\eta \cdot Y^{i-1}_{k+1,j-1} + X^{i-1}_{k+1,j-1}),$$

$$F1^i_{k,j} \rightarrow \sum_{l=0}^{i-1} \sum_{t=k+1}^{j-1-m} P_{k,j} \cdot (X^l_{j+1,t-1} + Y^l_{k+1,t-1}) \cdot (X^{i-1-l}_{t+1,j-1} + \eta \cdot Y^{i-1-l}_{t+1,j-1}),$$

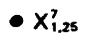
$$F2^i_{k,j} \rightarrow \sum_{l=0}^{i-1} \sum_{t=k+m+1}^{j-1} P_{k,j} \cdot (X^l_{t+1,j-1} + Y^l_{t+1,j-1}) \cdot (X^{i-1-l}_{k+1,t-1} + \eta \cdot Y^{i-1-l}_{k+1,t-1}),$$

$$F3^i_{k,j} \rightarrow \sum_{s=1}^{i-1} \sum_{q=0}^{i-1-s} \sum_{l=m+k+1}^{j-m-2} Y^s_{k,l} \cdot \sum_{t=l+1}^{j-m-1} Y^{i-s-q}_{t,j} (X^q_{l+1,t-1} + Y^q_{l+1,t-1}).$$

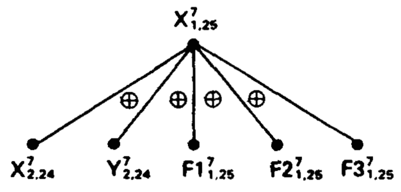
The fourth concept is that of a "start symbol". This is just the variable whose structures we wish to determine. In our example which follows it is $X^7_{1,25}$ from the first problem in § 5.

Thus, what we have defined here is a "grammar", (V_n, V_t, R, S), where V_n is the set of nonterminals, V_t is the set of terminals, R is the set of rewrite rules, and S is the start symbol. In accordance with the grammar, our algorithm will construct a tree from which we can directly read the structural information. The relationship between trees and certain types of grammars is well known [1] and will be made clear here by our example. We present a single example here and follow it with the algorithm itself. Most of the tree structure nomenclature used here can be found in [5].

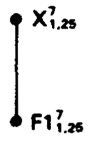
We begin constructing the tree by placing $X^7_{1,25}$ at the root of the tree.



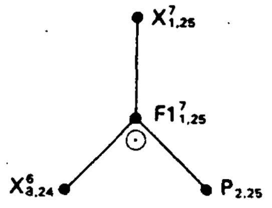
Next, we append to this node a subtree associated with the rewrite rule for this variable.



Because the values of all of these variables are stored in memory after computing the polynomial generating function $X^7_{1,25}$ we know that some of these variables have the value 0. In order to construct a tree which is a manageable size we omit these nodes, having the following new tree:



We repeat the above process of appending a subtree, omitting the trivial nodes. At this stage we have:



Note that we are including in the tree the operators $\oplus$ and $\odot$ wherever appropriate. The last subtree which was appended came from the rewrite rule

$$F1^7_{1,25} \to \sum_{l=0}^{6} \sum_{t=2}^{20} P_{t,25} \cdot (X^l_{2,t-1} + Y^l_{2,t-1}) \cdot (X^{6-l}_{t+1,24} + \eta \cdot Y^{6-l}_{t+1,24}).$$

The only nontrivial terms in this double summation were $X^6_{3,24}$ and $P_{2,25}$.

Since $P_{2,25}$ is a terminal symbol we are finished with that part of the tree. We shall continue appending subtrees to all variables on the tree until we have a tree whose leaves are all of the form P_{ij}. At this stage in this example we have the tree in Fig. 6.

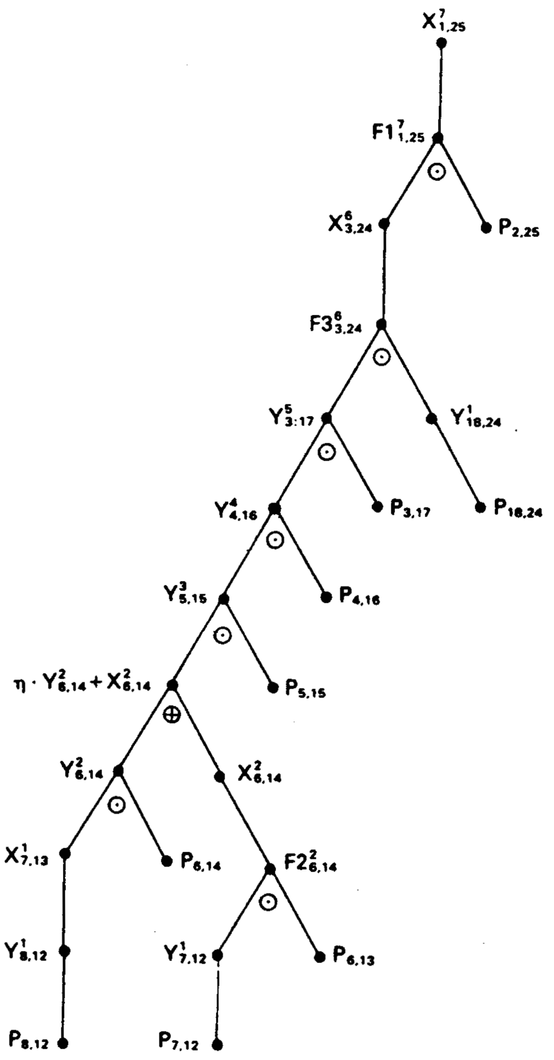


FIG. 6. *Tree structure representing $X^7_{1,25}$ from the first example in* § 5.

To read the molecular structures from this tree we read the leaves as if they were part of an arithmetic expression, assuming the usual precedence of $\odot$ over $\oplus$. In this example we read:

$$P_{8,12} \cdot P_{6,14} \cdot P_{5,15} \cdot P_{4,16} \cdot P_{3,17} \cdot P_{18,24} \cdot P_{2,25}$$
$$+ P_{7,12} \cdot P_{6,13} \cdot P_{5,15} \cdot P_{4,16} \cdot P_{3,17} \cdot P_{18,24} \cdot P_{2,25}.$$

Thus we have two structures having the following bonding:

(1) 8–12, 6–14, 5–15, 4–16, 3–17, 18–24, 2–25

(2) 7–12, 6–13, 5–15, 4–16, 3–17, 18–24, 2–25

These are the two secondary structures associated with $X^7_{1,25} = 2\alpha^5\eta^3\beta^2$ in our example.

The following rules define the construction of a subtree associated with a given rewrite rule:

(i) the root of the subtree corresponds to the left-hand side of a rewrite rule;
(ii) the remaining nodes correspond to the right-hand side of a rewrite rule;
(iii) the subtree structure corresponds to the arithmetic interpretation of the rewrite rule; each operation $\cdot$ or $+$ corresponds to a nonterminal node of the tree, and arcs originating from the node lead to the operands.
(iv) for simplicity omit the symbol η and all trivial nodes (those that evaluate to 1 in the case of a multiply and those that evaluate to 0 in the case of an addition.)

We use the following notation to mean "attach to node A a subtree corresponding to its rewrite rule in accordance with rules (i) through (iv)":

$$\text{ATTACH}(A, R(A)).$$

Our algorithm for generating a tree from which the structures of the RNA molecule can be read is summarized as follows.

```
Step 1. Make the term of interest the root of the tree.
Step 2. While there are terminal nodes on the tree
                which are not of the form $P_{ij}$ do
                $N$ is leftmost terminal node not of the form $P_{ij}$;
                ATTACH ($N$, $R(N)$)
        end.
Step 3. Write the algebraic expression corresponding to the tree and its terminal
        nodes.
```

An alternate and simpler method of computing the structures would be to change the program described in § 4 so that the matrix P is defined as follows:

$$P_{ij} = \begin{cases} 0, & \text{if } \{S_i, S_j\} = \{A, U\} \text{ or } \{G, C\}, \\ P_{ij}, & \text{otherwise.} \end{cases}$$

That is, there are no α's or β's in the expression being computed, just the terminal symbols P_{ij}. We can then read the structures directly from the generating function. The advantage of this method is that the programming is simple, requiring only a modification of the program in § 4. The disadvantage is that more memory is required due to the storage of many zero valued array elements.

Acknowledgment. The authors wish to thank the Laboratory for Computer Science, Mathlab Group, at the Massachusetts Institute of Technology for the use of their computer system MACSYMA, which is supported in part by the United States Energy Research and Development Administration under Contract No. E(11-1)-3070 and by the National Aeronautics and Space Administration under Grant NSG 1323.

REFERENCES

[1] A. T. Berztiss, *Data Structures, Theory and Practices*, Academic Press, New York, 1971.
[2] R. Dickson, J. W. Abelson and W. Reznikoff, *Genetic regulation: The lac control regions*, Science, 187, (1975) 4171, pp. 27–35.
[3] W. Gilbert, *RNA Polymerase*, Cold Spring Harbor Laboratory Press, 1976.
[4] J. E. Hopcroft and J. D. Ullman, *Formal Languages and their Relation to Automata*, Addison-Wesley, Reading, MA, 1969.

[5] D. E. KNUTH, *The Art of Computer Programming*, vol. 1, Addison-Wesley, Reading, MA, 1969.
[6] MATHLAB GROUP, *MACSYMA Reference Manual*, Laboratory for Computer Science, Massachusetts Institute of Technology, Cambridge, MA.
[7] D. POLAND AND H. A. SCHERAGA, *Theory of Helix-Coil Transitions in Biopolymers*. Academic Press, New York, 1970.
[8] J. SKLAR, P. YOT AND S. WEISSMAN, *Determination of genes, restriction sites, and DNA sequences surrounding the 6S RNA template of bacteriophage lambda*, Pro. Nat. Acad. Sci., 72 (1972), pp. 1817–1821.
[9] M. S. WATERMAN AND T. F. SMITH, *RNA secondary structure: A complete mathematical analysis*, Math. Biosci., 42 (1978), pp. 257–266.
[10] R. P. NUSSINOV, G. PIECZENIK, J. R. GRIGGS AND D. KLEIFMAN, *Algorithms for loop matchings*, this Journal, 35 (1978), pp. 68–82.