ALTRAN

Contributed by

W. S. Brown and A. D. Hall
Bell Telephone Laboratories
Murray Hill, N. J.

1. Introduction and Summary

J. A. Campbell [1] has proposed the computation of his Y_{2n} polynomials as a challenging problem for symbolic algebra systems. These polynomials are defined by a recurrence formula involving summations over two and four indices, and a special differentiation rule. Since the number of terms in Y_{2n} grows almost exponentially, large amounts of time and memory are required for large values of n.

In this paper we present an ALTRAN [2,3] program which solves this problem. Using the ALTRAN system installed on a Honeywell 6070 at Bell Laboratories, we have computed these polynomials through n=8 in 25 seconds, through n=10 in 47 seconds, through n=12 in 126 seconds, and through n=16 in 980 seconds. By extrapolation, we estimate that the computing time through n=18 would be about an hour, and we know (see below) that it would require nearly all the memory on our machine.

Using the TRIGMAN system on a CDC 6600, Campbell calculated the Y_{2n} polynomials through n=8 in 11 seconds, and through n=10 in 31 seconds. Since the 6600 is more than twice as fast as the 6070, and since the ratio of our computing time to Campbell's decreases as n increases, it seems fair to say that our ALTRAN program runs faster as well as farther.

As for memory requirements, the ALTRAN system together with our program and a workspace of 5K words (K=1024) occupied a total of 43K words for the case n=10. An additional 11K words of workspace were required for the case n=12. For n=16, we used 162K words of workspace, but we now know that 73K words would have been sufficient. For n=18, an exact calculation shows that 193K words of workspace would be required, so the program would use 231K of the total 256K words of memory.

The TRIGMAN system together with Campbell's program and workspace occupied approximately 29K words for n=8, and required an additional 10K words for the case n=10. Since the word size on the 6600 is 60 bits as compared to 36 bits on the 6070, it appears that the two systems are about the same size, while ALTRAN uses substantially less workspace.

Since ALTRAN is currently in use on a variety of computers, we should be very grateful to receive data on the performance of our program at other installations.

2. Formulation of the Problem

The polynomials $Y_{2n}(\varepsilon_0,\ldots,\varepsilon_{2n-2})$ for $n \geq 0$, which arise in the solution of a certain second order linear differential equation, are defined in [1] by the recurrence

$$Y_0 = 1$$

$$2Y_{2k} = \sum_{\alpha+\beta=k}^{*} Y_{2\alpha}Y_{2\beta} - \sum_{\alpha+\beta+\gamma+\delta=k}^{*} Y_{2\alpha}Y_{2\beta}Y_{2\gamma}Y_{2\delta} + \sum_{\alpha+\beta=k-1} \left[\varepsilon_0 Y_{2\alpha}Y_{2\beta} + \frac{3}{4} Y'_{2\alpha}Y'_{2\beta}\right] - \frac{1}{2} \sum_{\alpha+\beta=k-1} Y_{2\alpha}Y''_{2\beta} , \quad (1)$$

where an asterisk indicates that at least two of the summation indices must be positive (so that none is equal to k), while a prime denotes "differentiation" according to the "chain rule"

$$f' = \sum_{i\geq 0} \frac{\partial f}{\partial \varepsilon_i} \cdot \varepsilon_{i+1} \quad (2)$$

Campbell remarks that this rule can cause difficulties, presumably because each application introduces a new indeterminate. Applying (1) and (2) for k=1 to 3, we find

$$Y_2 = \frac{1}{2}\varepsilon_0$$
$$Y_4 = -\frac{1}{8}\left(\varepsilon_0^2 + \varepsilon_2\right)$$
$$Y_6 = \frac{1}{32}\left(2\varepsilon_0^3 + 6\varepsilon_0\varepsilon_2 + 5\varepsilon_1^2 + \varepsilon_4\right) \quad (3)$$

This suggests the definition

$$Z_n = (-1)^{n+1}2^{2n-1}Y_{2n} \quad (4)$$

which yields the recurrence

$$Z_0 = -\frac{1}{2}$$

$$Z_k = -\sum_{\alpha+\beta=k}^{*} Z_\alpha Z_\beta + 4\sum_{\alpha+\beta+\gamma+\delta=k}^{*} Z_\alpha Z_\beta Z_\gamma Z_\delta + \sum_{\alpha+\beta=k-1}\left[4\varepsilon_0 Z_\alpha Z_\beta + 3Z'_\alpha Z'_\beta\right] - 2\sum_{\alpha+\beta=k-1} Z_\alpha Z''_\beta \quad (5)$$

and the early values

$$Z_1 = \varepsilon_0$$
$$Z_2 = \varepsilon_0^2 + \varepsilon_2 .$$
$$Z_3 = 2\varepsilon_0^3 + 6\varepsilon_0\varepsilon_2 + 5\varepsilon_1^2 + \varepsilon_4 \quad (6)$$

3. The Main Procedure

Let n denote the desired order of computation. Although this is initialized in its declaration, it could just as easily be an input parameter. We declare one-dimensional algebraic arrays z, z1, and z2 for the Z_k, Z'_k, and Z''_k, respectively, and then compute the Z_k according to (5) for k=0,...,n. Differentiation according to (2) is implemented by the function chdiff, while the summations in (5) are implemented by the functions sig1, sig2, sig3, and sig4.

Since we are interested in optimal performance and in large values of n, we use the array-valued invocation maxexp(n) in the layout instead of the scalar n. This permits us to minimize the space that is allocated for exponents in each polynomial term.

The functions maxexp, chdiff, sig1, sig2, sig3, and sig4 are discussed in Sections 4-9, respectively. In invocations of these subprocedures it is an optional matter of style and efficiency to surround an actual argument with parentheses if it is not to be changed and if it would otherwise be eligible to be changed. Now here is the main procedure:

```
    procedure main
# declarations
    integer k, n=10
    long algebraic ( e(0:2*n-2) : maxexp((n)) ) array(0:n) z, z1, z2
    integer array altran maxexp
    long algebraic altran chdiff, sig1, sig2, sig3, sig4
    real delta, t
# initialization
    z(0) = 1/2
# main loop
    do k = 1, n
      #computation of z(k)
        z1(k-1) = chdiff( (z(k-1)), e, 2*k-3 )
        z2(k-1) = chdiff( (z1(k-1)), e, 2*k-2 )
        z(k) = sig1((z),(k)) + sig2((z),(k)) +
               sig3((z),(z1), e(0), k-1) + sig4((z),(z2),k-1)
      #output and timing
        write z(k)
        delta = time(t); write delta, t
    doend
# termination
    end
```

4. Maximum Exponents

It is easy to show by induction that Z_n is "homogeneous" in the sense that each of its terms satisfies the relation

$$\sum_{i=0}^{2n-2} (i+2)\delta_i = 2n \quad (7)$$

where δ_i is the exponent of ε_i. It follows immediately that

$$\delta_i(Z_n) \leq \left[\frac{2n}{i+2}\right] \quad (8)$$

where δ_i denotes the degree in ε_i, and the brackets denote the integer part. The function maxexp is based on this inequality:

```
    procedure maxexp(n)
# declarations
    value n
    integer n, i, a
    array(0:2*n-2) a
# fill in the array
    do i = 0, 2*n-2
        a(i) = iquo(2*n, i+2)
    doend
# if you want to see the results
    write a
# termination
    return(a)
    end
```

5. Differentiation

The function chdiff accepts a polynomial f, an indeterminate array e, and an integer m, and returns the derivative g=f' computed by the chain rule (2). Since f depends (at most) on e(0) through e(m-1), it follows that the derivative will depend (at most) on e(0) through e(m). If m ≤ 0, f must be independent of e, and therefore its derivative is zero.

Since f is a polynomial, the required partial derivatives can be obtained more efficiently with diffpo than with diff. However, since diffpo requires that its argument be in unfactored form, we must expand f before entering the main loop.

```
    procedure chdiff(f,e,m)
# declarations
# diffpo must be renamed in version 1.8 only
    value f, e, m
    integer m, i
    long algebraic f, g=0
    long algebraic array e
    long algebraic altran diffpo = s9difpo
# expand f for diffpo
    f = expand((f))
# chain rule
    do i = 0, m - 1
        g = g + diffpo((f),(e(i)))*e(i+1)
    doend
# termination
    return(g)
    end
```

6. The First Summation

The function sig1 accepts a polynomial array z and an integer k, and returns the negative of the first summation on the right side of (5). Since the summand is symmetric in α and β, we can eliminate the region α > β by simply doubling each term in the region α < β. Hence we sum over the region α ≤ β after multiplying each term by a weight w, which is 1 if α=β and 2 otherwise. Since α+β=k, the inequality α ≤ β is equivalent to α ≤ [k/2]. Furthermore, the requirement that both indices be positive is equivalent to α ≥ 1. Now here is the function:

```
    procedure sig1(z,k)
# declarations
    value z, k
    integer k, a, b, w
    long algebraic z, s=0
    array z
# computation
    do a = 1, iquo((k),2)
        b = k - a
        if (a==b) w=1; else w=2
        s = s - w*z(a)*z(b)
    doend
# termination
    return(s)
    end
```

7. The Second Summation

The function sig2 accepts a polynomial array z and an integer k, and returns 4 times the second summation on the right side of (5). Since the summand is symmetric in the summation indices, we can confine ourselves to the region $\alpha \le \beta \le \gamma \le \delta$ by introducing appropriate weight factors. By moving the 4 inside the sum, we multiply it by an integer (the weight factor) rather than a polynomial (the sum), and furthermore we cancel the denominator of Z_0 instantly whenever it arises.

Consider a term with $\alpha < \beta < \gamma < \delta$. Such a term is the only representative in the region of a set of 24 equal terms corresponding to the 24 permutations of the indices. Similarly, a term with $\alpha = \beta < \gamma < \delta$ is the only representative in the region of a set of 12 equal terms corresponding to the 12 ways of selecting a pair and 2 singles from the set of 4 indices. For a term with $\alpha = \beta = \gamma < \delta$, the weight is 4, since only one index is distinguished. However if $\alpha = \beta < \gamma = \delta$, the weight is 6, since there are 6 ways of choosing a pair. Finally, if $\alpha = \beta = \gamma = \delta$, the weight is 1, since at most one such term occurs in the original unconstrained region.

Having chosen the weights in this manner, our region of summation is defined by

$$\begin{aligned} &\alpha + \beta + \gamma + \delta = n \\ &0 \le \alpha \le \beta \le \gamma \le \delta \\ &1 \le \gamma \end{aligned} \quad , \tag{9}$$

where the last relation is equivalent to the requirement that at least two of the summation indices must be positive. Now, since

$$\begin{aligned} 4\alpha &\le \alpha + \beta + \gamma + \delta = k \\ 3\beta &\le \beta + \gamma + \delta = k-\alpha \\ 2\gamma &\le \gamma + \delta = k-\alpha-\beta \end{aligned} \tag{10}$$

we see that (9) implies

$$\begin{aligned} &0 \le \alpha \le [k/4] \\ &\alpha \le \beta \le [(k-\alpha)/3] \\ &\max(\beta,1) \le \gamma \le [(k-\alpha-\beta)/2] \\ &\delta = k-\alpha-\beta-\gamma \end{aligned} \tag{11}$$

Conversely, it is easy to prove that (11) implies (9). Since (11) is in the appropriate form for a straightforward nested 3-index summation, we are now ready to proceed to the function:

```
    procedure sig2(z,k)
# declarations
    value z, k
    long algebraic z, s=0
    array z
    integer k, a, b, c, d, w, lts
    static array(0:3) w = (1, .null., 12, 24)
# computation
    do a = 0, iquo((k),4)
      do b = a, iquo(k-a,3)
        do c = imax((b),1), iquo(k-a-b,2)
            d = k-a-b-c
          #count the less-than's
            lts = 0
            if(a < b) lts = 1
            if(b < c) lts = lts + 1
            if(c < d) lts = lts + 1
          #set w(1)
            if(b < c) w(1)=6; else w(1)=4
          #now add the term to s
            s = s + 4*w(lts)*z(a)*z(b)*z(c)*z(d)
        doend
      doend
    doend
# termination
    return(s)
    end
```

8. The Third Summation

The function sig3 accepts the polynomial arrays z and z1, an indeterminate $e0=\varepsilon_0$, and an integer k1 = k-1, and returns the third summation on the right side of (5). As in sig1, we introduce a weight w, and sum only over the region α ≤ β. Now here is the function:

```
    procedure sig3(z,z1,e0,k1)
# declarations
    value z, z1, e0, k1
    integer k1, a, b, w
    long algebraic z, z1, e0, s=0
    array z, z1
# computation
    do a=0, iquo((k1),2)
      b = k1-a
      if(a==b) w=3; else w=2
      s = s + 4*w*e0*z(a)*z(b) + 3*w*z1(a)*z1(b)
    doend
# termination
    return(s)
    end
```

Ed. Note: Section 10 and Figure 1 on ALTRAN output have been omitted.

9. The Fourth Summation

The function sig4 accepts the polynomial arrays z and z2 and an integer k1 = k-1, and returns the negative of twice the fourth summation on the right side of (5). Since this summand is not symmetric, we must sum over α all the way from 0 through k1. Now here is the function:

```
    procedure sig4(z,z2,k1)
# declarations
    value z, z2, k1
    integer k1, a
    long algebraic z, z2, s=0
    array z, z2
# computation
    do a = 0, k1
      s = s - 2*z(a)*z2(k1-a)
    doend
# termination
    return(s)
    end
```

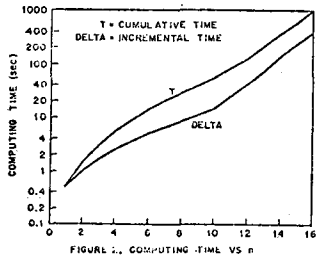

FIGURE 2. COMPUTING TIME VS n

11. Program Behavior

Essentially, two distinct runs of our program were made. The first run, with n initialized to 10 in the main procedure, was designed to produce the shortest execution time for the cases up to n=10. The second run, with n initialized to 16, was an attempt to carry the compu-

tation as far as possible.

For the first 10 cases, the second run took 12% longer than the first. This is due to the fact that computing the Z_n to n=16 on the Honeywell 6070 requires two words of exponents for each polynomial term, while only one word per term is needed to compute the Z_n to n=10.

In Fig. 2, the observed computing times from our second run are plotted against n. The abrupt change in slope at n=10 is due to the fact that the largest coefficient in Z_n for any $n > 10$ cannot be stored in a single word.

In Fig. 3, the number of terms in Z_n is shown plotted against n. For n=16, there are about 1500 terms, which on the Honeywell 6070 require about 6000 words of storage (2 words for each coefficient, and 2 for each exponent set).

Inspection of the output for $n \leq 16$ reveals that all possible terms consistent with (7) occur in Z_n with positive coefficients. Clearly, each term of Z_n corresponds to a partition of 2n into a sum of integers not including 1. Using the theory of unrestricted partitions [4, p. 273], the maximum number of terms in Z_n can easily be shown to be

$$p(2n) - p(2n-1) \tag{12}$$

where p(k) is the well known function giving the number of distinct partitions of k into a sum of integers. The asymptotic behavior of p(k) is [5, p. 825]

$$\frac{1}{4k\sqrt{3}}\, e^{\pi\sqrt{2k/3}} \tag{13}$$

It follows from this that the number of terms in Z_n is asymptotic to

$$\frac{\pi}{48n\sqrt{n}}\, e^{2\pi\sqrt{n/3}} \tag{14}$$

In Fig. 4, the largest coefficient appearing in Z_n is plotted against n. It is interesting to note that the growth rate appears to be faster than exponential.

12. Conclusion

Considering the fact that ALTRAN is written almost entirely in a higher level language (FORTRAN IV), it is natural to seek reasons for its outstanding performance on this problem.

Perhaps the most fundamental reason is that we restricted the domain of data to rational expressions with integer coefficients, in the belief that most operations on more general expressions can be expressed in terms of simple operations on rational expressions. The ease with which the special differentiation rule (2) was incorporated into our program lends substantial weight to this point of view.

Furthermore, restricting the domain of data to rational functions made it possible to devise an extremely compact data representation [6], and fast algorithms for addition, multiplication and division. Also important is the fact that ALTRAN treats all indeterminates in a polynomial in the same way, so that the computing time for operations such as differentiation and coefficient extraction does not depend on which indeterminate is distinguished.

As ALTRAN becomes more widely available and used, we will be able to make a more accurate assessment of the validity of our major design decisions.

Although ALTRAN is both proprietary and experimental, we can make copies available without charge to universities under certain conditions. Further information may be obtained by writing to the authors.

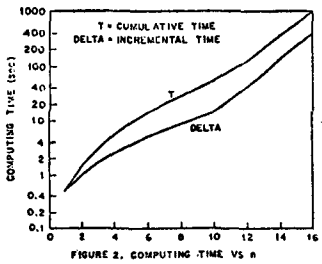

FIGURE 2. COMPUTING TIME VS n

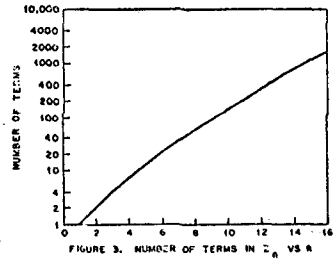

FIGURE 3. NUMBER OF TERMS IN Z_n VS n

REFERENCES

1. J. A. Campbell, Problem 2 - The Y_{2n} Functions, SIGSAM Bulletin, No. 22, pages 8-9 (April 1972).

2. W. S. Brown, ALTRAN User's Manual, Bell Telephone Laboratories, Murray Hill, New Jersey, 1971. Second Edition, 1972.

3. A. D. Hall, ALTRAN Installation and Maintenance, Bell Telephone Laboratories, Murray Hill, New Jersey, 1971. Second Edition, 1972.

4. G. H. Hardy and E. M. Wright, An Introduction to the Theory of Numbers, Oxford University Press, London, 1960.

5. M. Abramowitz and I. A. Stegun (eds.), Handbook of Mathematical Functions, National Bureau of Standards, 1960.

6. A. D. Hall, The ALTRAN System for Rational Function Manipulation - A Survey, Comm. ACM 14, 517-521, (1971).

Input

```
1           PROCEDURE MAIN
2           INTEGER I, N=16
3           REAL DELTA, T
4           LONG ALGEBRAIC(E(0:2**N-2):(4$16,487,713,1691)) ARRAY(0:N)
                                                              Z, Z1, Z2
5           LONG ALGEBRAIC ALTRAN SIG1, SIG2, SIG3, SIG4, CHDIFF
6
7           Z(0) = -1/2
8           DO I=1,N
9                   Z1(I-1) = CHDIFF((Z(I-1)),E,2**I-3)
10                  Z2(I-1) = CHDIFF((Z1(I-1)),E,2**I-2)
11                  Z(I) = SIG1((Z),(I)) + SIG2((Z),(I)) +
12                                 SIG3((Z),(Z1),E(0),I-1) + SIG4((Z),(Z2),I-1)
13                  WRITE Z(I)
14
15                  DELTA = TIME(T)
16                  WRITE DELTA, T
17          DOEND
18          END


1
2           PROCEDURE CHDIFF(F,E,N)
3           VALUE F, E, N
4           LONG ALGEBRAIC F, G=0
5           LONG ALGEBRAIC ALTRAN DIFFPO
6   # E IS AN INDETERMINATE VECTOR.
7   # F DEPENDS ON E(0) THROUGH E(N-1).
8   # G WILL DEPEND ON E(0) THROUGH E(N).
9   # IF N <= 0, CHDIFF RETURNS 0.
10          LONG ALGEBRAIC ARRAY E
11          INTEGER K, I
12  # EXPAND F FOR DIFFPO
13          F = EXPAND((F))
14  # CHAIN RULE
15          DO I=0, N-1
16                  G = G + DIFFPO((F),(E(I)))*E(I+1)
17          DOEND
18  # FINISHED
19          RETURN(G)
20          END


1
2           PROCEDURE SIG1(Z,N)
3           VALUE Z, N
4           LONG ALGEBRAIC ARRAY Z
5           INTEGER N, A, B, W
6           LONG ALGEBRAIC S=0
7
8           DO A=1, IQUO((N),2)
9                   B = N-A
10                  IF(B==A) W=1; ELSE W=2
11                  S = S + W*Z(A)*Z(B)
12          DOEND
13          RETURN(S)
14          END


1
2           PROCEDURE SIG2(Z,N)
3           VALUE Z, N
4           LONG ALGEBRAIC ARRAY Z
5           INTEGER N, A, B, C, D, LTS
6           STATIC INTEGER ARRAY(0:3) W = (1, .NULL., 12, 24)
7           LONG ALGEBRAIC S=0
8
9           DO A=0, IQUO((N),4)
10            DO B=A, IQUO(N-A,3)
11              DO C = IMAX((B),1), IQUO(N-A-B,2)
12                  D = N-A-B-C
13          # COUNT THE LESS-THAN'S
14                  LTS = 0
15                  IF (A<B) LTS = 1
16                  IF (B<C) LTS = LTS + 1
17                  IF (C<D) LTS = LTS + 1
18          # SET W(1)
19                  IF (B<C) W(1)=6; ELSE W(1) = 4
20          # NOW ADD THE TERM TO S
21                  S = S + 4*W(LTS)*Z(A)*Z(B)*Z(C)*Z(D)
22            DOEND
23            DOEND
24          DOEND
25          RETURN(S)
26          END


1
2           PROCEDURE SIG3(Z, Z1, E0, N)
3           VALUE Z, Z1, E0, N
4           LONG ALGEBRAIC ARRAY Z, Z1
5           INTEGER N, A, B, W
6           LONG ALGEBRAIC E0, S=0
7
8           DO A=0, IQUO((N),2)
9                   B = N-A
10                  IF (B==A) W=1; ELSE W=2
11                  S = S + W*4*E0*Z(A)*Z(B) + W*3*Z1(A)*Z1(B)
12          DOEND
13          RETURN(S)
14          END


1
2
3           PROCEDURE SIG4(Z,Z2,N)
4           VALUE Z, Z2, N
5           INTEGER N, A, B
6           LONG ALGEBRAIC ARRAY Z, Z2
7           LONG ALGEBRAIC S=0
8           DO A=0, N
9                   B = N - A
10                  S = S - 2*Z(A)*Z2(B)
11          DOEND
12          RETURN(S)
13          END
```

Output

```
@ Z(1)
      F(0)
@ DELTA
      5.3625E-1
@ T
      5.3625E-1
@ Z(2)
      E(0)**2 + F(2)
@ DELTA
      8.765E-1
@ T
      1.38075
@ Z(3)
      2*E(0)**3 + 6*F(0)*E(2) + 5*E(1)**2 + E(4)
@ DELTA
      1.609734
@ T
      2.990484
@ Z(4)
      5*E(0)**4 + 30*E(0)**2*E(2) + 50*E(0)*E(1)**2 + 10*E(0)*E(4) +
      20*E(1)*E(3) + 15*E(2)**2 + E(6)
@ Z(9)
      1430*E(0)**9 + 51480*E(0)**7*E(2) + 300300*E(0)**6*E(1)**2 +
      60060*E(0)**6*E(4) + 100900A*E(0)**5*E(1)*E(3) + 644644*E(0)**5*
      E(2)**2 + 36036*E(0)**5*E(6) + 5688540*E(0)**4*E(1)**2*E(2) +
      694980*E(0)**4*E(1)*F(5) + 1415700*E(0)**4*E(2)*E(4) + 888030*E(0)**4*
      E(3)**2 + 12470*F(0)**4*E(8) + 3160300*E(0)**3*E(1)**4 +
      4561800*E(0)**3*E(1)**2*E(4) + 15913040*E(0)**3*E(1)*E(2)*E(3) +
      251640*E(0)**3*E(1)*F(7) + 3609320*E(0)**3*E(2)**3 + 680680*E(0)**3*
      E(2)*E(6) + 1195480*E(0)**3*E(3)*E(5) + 717460*E(0)**3*E(4)**2 +
      2860*E(0)**3*E(10) + 13166400*E(0)**2*E(1)**3*E(3) + 26912340*E(0)**2*
      E(1)**2*E(2)**2 + 1445300*E(0)**2*E(1)**2*E(6) + 7946640*E(0)**2*E(1)*
      E(2)*E(5) + 12002640*E(0)**2*E(1)*E(3)*E(4) + 50780*E(0)**2*E(1)*E(9) +
      415950*E(0)**2*E(2)**2*F(4) + 10265540*E(0)**2*E(2)*E(3)**2 +
      170A30*E(0)**2*E(2)*E(8) + 345320*E(0)**2*E(3)*E(7) + 616980*E(0)**2*
      E(4)*E(6) + 359970*E(0)**2*F(5)**2 + 390*E(0)**2*E(12) + 14908500*E(0)*
      E(1)**4*E(2) + 4343400*E(0)*E(1)**3*E(5) + 25811720*E(0)*E(1)**2*E(2)*
      E(4) + 16474220*F(0)*E(1)**2*E(3)**2 + 276300*E(0)*E(1)**2*E(8) +
      46002240*E(0)*E(1)*E(2)**2*E(3) + 1689960*E(0)*E(1)*E(2)*E(7) +
      3403640*E(0)*E(1)*E(3)*E(6) + 4778040*E(0)*E(1)*E(4)*E(5) +
      5400*E(0)*E(1)*E(11) + 5229510*E(0)*E(2)**4 + 2309580*E(0)*E(2)**2*
      E(6) + 916324O*E(0)*F(2)*E(3)*E(5) + 4911660*E(0)*E(2)*E(4)**2 +
      21780*F(0)*E(2)*E(10) + 6184140*E(0)*E(3)**2*E(4) + 60000*E(0)*E(3)*
      E(9) + 120060*E(0)*E(4)*F(8) + 180120*E(0)*E(5)*E(7) +
      102930*E(0)*E(6)**2 + 30*E(0)*E(14) + 828250*E(1)**6 + 3682110*E(1)**4*
      E(4) + 25313A32*E(1)**3*E(2)*E(3) + 457920*E(1)**3*E(7) +
      172A7012*E(1)**2*E(2)**3 + 3762444*E(1)**2*E(2)*E(6) + 6653604*E(1)**2*
      E(3)*E(5) + 4003970*E(1)**2*E(4)**2 + 17490*E(1)**2*E(10) +
      905518A*E(1)*E(2)**2*E(5) + 27473720*E(1)*E(2)*E(3)*E(4) +
      130340*E(1)*E(2)*E(9) + 5771A64*E(1)*E(3)**3 + 32A056*E(1)*E(3)*E(8) +
      592336*E(1)*E(4)*E(7) + 790828*E(1)*E(5)*E(6) + 230*E(1)*E(13) +
      6240260*F(2)**3*E(4) + 11903634*E(2)**2*E(3)**2 + 222186*E(2)**2*E(8) +
      100992**E(2)*F(3)*F(7) + 1623780*E(2)*E(4)*E(6) + 946614*E(2)*E(5)**2 +
      111**F(2)*F(12) + 1022138*E(3)**2*E(6) + 2876A36*E(3)*E(4)*E(5) +
      3634*F(1)*E(11) + 577502*E(4)**3 + 8734*E(4)*E(18) + 18014*E(5)*E(9) +
      22874*E(6)*E(5) + 12469*E(7)**2 + E(16)
@ DELTA
      1.113378E1
@ T
      4.927375E1
@ Z(10)
      4452*E(0)**10 + 214790*E(0)**8*E(2) + 1454600*E(0)**7*E(1)**2 +
      291720*E(0)**7*E(4) + 5717712*E(0)**6*E(1)*E(3) + 3879876*E(0)**6*
      F(2)**2 + 204204*E(0)**6*E(6) + 3A6A2072*E(0)**5*E(1)**2*E(2) +
      4725944*E(0)**5*E(1)*E(5) + 9626760*E(0)**5*E(2)*E(4) + 6038604*E(0)**5*
      E(3)**2 + 47516*F(0)**5*E(8) + 26462550*E(0)**4*E(1)**4 +
      39525300*E(0)**4*E(1)**2*E(4) + 135260A40*E(0)**4*E(1)*E(2)*E(3) +
      2139240*F(0)**4*E(1)*E(7) + 30679220*E(0)**4*E(2)**3 + 5765740*E(0)**4*
      E(2)*E(6) + 10161540*E(0)**4*E(3)*E(5) + 6101810*E(0)**4*E(4)**2 +
      24310*F(0)**4*E(10) + 149219200*E(0)**3*E(1)**3*E(3) +
      305016520*E(0)**3*E(1)**2*E(2)**2 + 18873400*E(0)**3*E(1)**2*E(6) +
      90061920*E(0)**3*E(1)*F(2)*E(5) + 136029920*E(0)**3*E(1)*E(3)*E(4) +
      574630*E(0)**3*E(1)*E(9) + 92479240*E(0)**3*E(2)**2*E(4) +
      115143240*E(0)**3*F(2)*E(3)**2 + 1935960*E(0)**3*E(2)*E(8) +
      4366960*E(0)**3*E(3)*E(7) + 6992440*E(0)**3*E(4)*E(6) + 4079660*E(0)**3*
      E(5)**2 + 4420*E(0)**3*E(12) + 253444500*E(0)**2*E(1)**4*E(2) +
      73837800*E(0)**2*E(1)**3*E(5) + 455799240*E(0)**2*E(1)**2*E(2)*E(4) +
      286861740*E(0)**2*E(1)**2*E(3)**2 + 4697100*E(0)**2*E(1)**2*E(8) +
      78278040*E(0)**2*E(1)*E(2)**2*E(3) + 28727280*E(0)**2*E(1)*E(2)*E(7) +
      57462560*E(0)**2*E(1)*E(3)*E(6) + 81226680*E(0)**2*E(1)*E(4)*E(5) +
      91400*E(0)**2*E(1)*E(11) + 88901670*E(0)**2*E(2)**4 + 39262860*E(0)**2*
      E(2)**2*E(6) + 138775040*E(0)**2*E(2)*E(3)*E(5) + 83498220*E(0)**2*E(2)*
      F(4)**2 + 370260*E(0)**2*E(2)*E(10) + 105130380*E(0)**2*E(3)**2*E(4) +
      1020000*E(0)**2*E(3)*E(9) + 2041020*E(0)**2*E(4)*E(8) + 3062040*E(0)**2*
      E(5)*E(7) + 1749810*E(0)**2*E(6)**2 + 510*E(0)**2*E(14) + 28160500*E(0)*
      E(1)**6 + 125191740*E(0)*E(1)**4*E(4) + 86067028*E(0)*E(1)**3*E(2)*
      E(3) + 15569280*E(0)*E(1)**3*E(7) + 58775840*E(0)*E(1)**2*E(2)**3 +
      127923096*E(0)*E(1)**2*E(2)*E(6) + 226222536*E(0)*E(1)**2*E(3)*E(5) +
      136134980*E(0)*E(1)**2*E(4)**2 + 594660*E(0)*E(1)**2*E(10) +
      307876392*E(0)*E(1)*E(2)**2*E(5) + 93410648O*E(0)*E(1)*E(2)*E(3)*E(4) +
      4432920*E(0)*E(1)*E(2)*E(9) + 196243376*E(0)*E(1)*E(3)**3 +
      11153904*E(0)*E(1)*E(3)*E(8) + 20139424*E(0)*E(1)*E(4)*E(7) +
      26888152*E(0)*E(1)*E(5)*E(6) + 8092*E(0)*E(13) + 212168840*E(0)*
      E(2)**3*E(4) + 401323556*E(0)*E(2)**2*E(3)**2 + 7554324*E(0)*E(2)**2*
      E(4) + 34337552*E(0)*E(2)*E(3)*E(7) + 55208520*E(0)*E(2)*E(4)*E(6) +
      32252876*E(0)*E(2)*E(5)**2 + 38012*E(0)*E(2)*E(12) + 34752692*E(0)*
      E(3)**2*E(6) + 97812424*E(0)*E(3)*E(4)*E(5) + 123692*E(0)*E(3)*E(11) +
      19635068*E(0)*E(4)**3 + 296956*E(0)*E(4)*E(10) + 544476*E(0)*E(5)*E(9) +
      777852*E(0)*E(6)*E(8) + 437546*E(0)*E(7)**2 + 34*E(0)*E(16) +
      71247480*E(1)**5*E(3) + 243564834*E(1)**4*E(2)**2 + 17419710*E(1)**4*
      E(6) + 164013536*E(1)**3*E(2)*E(5) + 255034560*E(1)**3*E(3)*E(4) +
      1192500*E(1)**3*E(9) + 521971140*E(1)**2*E(2)**2*E(4) +
      65450874O*E(1)**2*E(2)*E(3)**2 + 12214884*E(1)**2*E(2)*E(8) +
      27776804*E(1)**2*E(3)*E(7) + 44667220*E(1)**2*E(4)*E(6) +
      26095426*E(1)**2*E(5)**2 + 30342*E(1)**2*E(12) + 599658104*E(1)*E(2)**3*
      F(3) + 37731072*E(1)*E(2)**2*E(7) + 152957768*E(1)*E(2)*E(3)*E(6) +
      215356248*E(1)*E(2)*E(4)*E(5) + 266724*E(1)*E(2)*E(11) + 135746860*E(1)*
      E(3)**2*E(5) + 163583604*E(1)*E(3)*E(4)**2 + 804804*E(1)*E(3)*E(10) +
      17759A0*F(1)*E(4)*E(9) + 2964924*E(1)*E(5)*E(8) + 3814924*E(1)*E(6)*
      E(7) + 304*E(1)*E(15) + 409A3566*E(2)**5 + 34680068*E(2)**3*E(6) +
      144792292*E(2)**2*E(3)*E(5) + 111364926*E(2)**2*E(4)**2 +
      544170*E(2)**2*E(10) + 281004100*E(2)*E(3)**2*E(4) + 3021740*E(2)*E(3)*
      F(9) + 6876140*E(2)*F(4)*E(8) + 9140140*E(2)*E(5)*E(7) +
      5227602*E(2)*E(6)**2 + 1630*E(2)*E(14) + 29551761*E(3)**4 +
      3423398*F(3)**2*E(8) + 13457916*E(3)*E(4)*E(7) + 18534652*E(3)*E(5)*
      E(6) + 6118*E(3)*E(13) + 11145226*E(4)**2*E(6) + 13055198*E(4)*E(5)**2 +
      17134*E(4)*E(12) + 37126*E(5)*E(11) + 63646*E(6)*E(10) +
      A7514*F(7)*E(9) + 48619*E(8)**2 + E(15)
@ DELTA
      1.484331E1
```