11. A. J. Kempner, "Geometry as an avocation," *Amer. Math. Mon.*, v. 40, 1933, p. 455–471.
12. Byron O. Marshall, jr., "The electronic isograph for roots of polynomials," *J. Appl. Phys.*, v. 21, 1950, p. 307–312.
13. J. H. Robertson, "A simple machine capable of Fourier synthesis calculation," *J. Sci. Instr.*, v. 27, 1950, p. 276–278.
14. A. Russell & J. N. Alty, "An electromagnetic method of studying the theory of and solving algebraic equations of any degree," *Phil. Mag.*, s. 6, v. 18, 1909, p. 802–812.
15. T. B. Rymer & C. C. Butler, "An electrical circuit for harmonic analysis and other calculations," *Phil. Mag.*, s. 7, v. 35, 1944, p. 606–616.
16. Walter W. Soroka, *Analog Methods in Computation and Simulation*, McGraw-Hill, New York, 1954.
17. L. Bauer & S. Fifer, "The solution of polynomial equations on the REAC," *Project Cyclone Symposium One*, Reeves Instrument Corp., New York, 1951, p. 31–36.
18. F. W. J. Olver, "The evaluation of zeros of high-degree polynomials," Royal Soc. of London, *Phil. Trans.*, ser. A, v. 244, 1952, p. 385–415.
19. R. A. Brooker, "The solution of algebraic equations on the EDSAC," Cambridge Phil. Soc., *Proc.*, v. 48, 1952, p. 255–270.

Notations for Partitions

The present paper gives a contribution to the automatic calculation of the characters of certain symmetric groups, as described in a paper which recently appeared in *MTAC* [1]. The repeated application of the recursion formula (7) of the cited paper involves the storing of a chain of partitions. The allotted storage space must be sufficient to meet even the case of a chain of as many as about $n/2$ partitions. In other words, provision must be made for storing about $n^2/4$ integers, components of these partitions. It would be a waste of memory space to store these integers separated, numerous and small as they are. Furthermore, as n increases, it will be still more important that the partitions should be available in the rapid access storage of the machine. Therefore, they should be stored by some method of packing, suitable for the operations to be applied, particularly during the calculation of characters.

In the sequel, I shall briefly describe three such methods, which might perhaps be of some interest. The first method [2] was used in an experimental program for character calculation with the Swedish relay computer BARK (August, 1953) and also in the first version of the character program using the Swedish electronic computer BESK (Spring, 1954). The second method is employed in the actual version of this program (tested on BESK, June, 1954). The third method, a modification of the second, has not yet been used in the program, though it is described here for theoretical reasons.

Preliminaries. Theoretically, the minimum of the number, x, of binary digits required to represent all the partitions of an integer n, is the smallest integer, not less than $\log_2 P(n)$, where $P(n)$, the number of the partitions, can be estimated by means of Hardy-Ramanujan's formula [3]. The result obtained,

$$x_{\min} \sim \pi\sqrt{2/3}\,\log_2 e\cdot\sqrt{n} - \log_2 4n\sqrt{3} \sim 3.70\sqrt{n} - \log_2 n - 2.79$$

indicates an ideal towards which the packing methods should lead.

If packing with fixed boundaries between the packed elements, one should store the partition components, each diminished by 1. In this case, n must also be stored. The value of x then lies between n and $3n/2$. For example, in a 40 digit register one can store the partitions of $n \leq 30$ by this method. It will be seen that our first and second methods yield slightly better values of x, namely $x = n$ and $x = n + 1$, respectively, whereas our third method gives approximately $x \leq \sqrt{32n + 4} - 2$, which is, for $n > 28$, in better accord with the Hardy-Ramanujan value.

Our notations for partitions are closely related to the YOUNG diagrams of the partitions [4], which are important in the NAKAYAMA version [5] of the recursive method. This version simplifies the calculation scheme, and it has, therefore, been employed in our program.

It is essential, in the methods which we are going to describe, that the computing machine is equipped with the operation known as "normalization [6]." We assume that the machine represents numbers modulo 2 by N digit words, the fractional part consisting of $N - 1$ binary digits, the remaining binary digit being interpreted as sign digit. Because it is necessary to store the sign associated to each partition appearing in the recursion formula, we shall represent the partition itself entirely in the fractional part of a word.

First method. The partition $(\lambda) = (\lambda_1, \lambda_2, \cdots, \lambda_p)$ is represented by the number

$$L = 2^{-\lambda_1} + 2^{-\lambda_1-\lambda_2} + \cdots + 2^{-\lambda_1-\lambda_2-\cdots-\lambda_p};$$

i.e., starting from the binary point, $(\lambda_1 - 1)$ zeros followed by a 1, $(\lambda_2 - 1)$ zeros followed by a 1, *etc.* The last 1, which means 2^{-n}, is followed by zeros (if $n < N - 1$). The number n need not be stored separately. For example, the partition (5, 2, 2, 1, 1, 1) is represented, the sign digit omitted, by the fraction

$$L = .000010101111 \text{ (and then zeros).}$$

We use shifts to read in the components of (λ), normalizations to read them out [2].

If the partitions of n are enumerated in decreasing lexicographical order, their representing numbers L form an increasing series. Moreover, the transition from one partition to the next is effected by the addition of 2^{-n+1} to L, followed by a slight modification if necessary.

This method is still used in our character program in the routine for passing from one partition to the next one, when several characters are to be calculated in succession.

Second method. This time we write the partition

$$(\lambda) = \lambda_1^{a_1}\lambda_2^{a_2}\cdots\lambda_m^{a_m}$$

where $\lambda_1 > \lambda_2 > \cdots > \lambda_m > 0$ and all $a_i > 0$, and put $\lambda_i - \lambda_{i+1} = b_i$ $(i < m)$, $\lambda_m = b_m$. The positive integers a_i and b_i $(i = 1, 2, \cdots, m)$ define the partition completely. The partition is represented by a number L' constructed in the following manner. Denoting by c an arbitrary, non-negative integer (usually, $c = 0$), yet with the restriction $c + a_1 + b_1 + \cdots + a_m + b_m \leq N - 1$, there are in L', after the binary point, c ones (yet usually, $c = 0$), and then successively a_1 zeros, b_1 ones, a_2 zeros, b_2 ones, *etc.* Eventually, remaining space after the last b_m ones is filled by zeros.

Again, we read in and out with the aid of shifts and normalizations.

Each non-vanishing term in the recursion formula is due to the existence of a so-called "hook [5] of length h" in the Young diagram, h being the actuated component of the class partition. Such hooks are most easily recognized in the number L', namely as a pair of a 0 and a 1, spaced h binary places from each other, the 0 being the most significant digit in that pair. The sign of the corresponding term of the formula is $(-1)^z$, where z is the number of zeros between the 0 and the 1 which were said to form the pair. The removal of the hook is an operation to be performed before the recursion formula is used again. It is equivalent to the interchange of the two digits of the mentioned pair.

These operations, namely the searching for hooks, the determination of signs, and the removal of hooks, form the essential contents of the calculation scheme. We note that the removal of a hook will never cause an increase of the number of digits of L'. Finally, the conjugate partition $(\tilde{\lambda})$ is very easily found, being represented by the "complement on 1" of the number obtained by reading L' in the reverse order. BESK has an instruction for shifting the multiplier register into the accumulator in reversed order.

Third method. The methods just described imply that an integer, t, will be represented by t digits. The fact that each digit is capable of two values is used only to realize the boundaries between adjacent integers. A more efficient notation is obtained as follows. The integer t, which is to be packed together with other integers, is first written in binary notation. The number, τ, of requisite digits is then represented by τ digits, all ones or all zeros. The two parts of the representation thus obtained are most conveniently placed in the different halves of one and the same whole word, τ being placed in the half word containing the binary point. Thus the normalization, being guided by τ, gives us the value of t in binary notation, unpacked.

It should be observed that such symbols as 1, 01, 001, etc., can be used here to represent different values of t. In this way, the integers $t = 1, 2, 3, 4, 5, 6, 7$, etc., will be represented by the symbols 0, 1, 00, 01, 10, 11, 000, etc., respectively, the corresponding values of τ being 1, 1, 2, 2, 2, 2, 3, etc. Therefore, if τ is determined so that

$$2^\tau - 1 \leq t < 2^{\tau+1} - 1,$$

the number t will be written in τ binary digits, constituting a number usually read as $t + 1 - 2^\tau$.

This "half-binary" notation could be used for packing the integers $c, a_1, b_1, \cdots, a_m, b_m$, appearing in the second method. If the numbers of digits in the binary notations of these integers are $\gamma, \alpha_1, \beta_1, \cdots, \alpha_m, \beta_m$, respectively, we write γ ones, α_1 zeros, β_1 ones, *etc.*, in the binary point half word, and the binary notations of c, a_1, b_1, *etc.*, in the remaining half word. This gives us a notation for partitions requiring a number of digits roughly proportional to $\sqrt{n}$, as formerly indicated.

Of course, the method could be iterated (writing τ binarily, *etc.*), but this would not give a very efficient notation for partitions.

Other applications. As an example, we indicate how the methods can be applied to the storing of the factorization of an integer, n, into powers of primes. Let p_i $(i = 1, 2, \cdots, m)$ be the primes dividing n, $p_1 > p_2 > \cdots > p_m \geq 2$, and let a_i be the exponent denoting the highest power of p_i still dividing n, $a_i > 0$. Further, let the function $f(x)$ mean the number of all existing primes $\leq x$. Putting $f(p_i) - f(p_{i+1}) = b_i$ $(i < m)$, $f(p_m) = b_m$, we can use the second or third method above to store the numbers $a_1, b_1, \cdots, a_m, b_m$. These determine the factorization completely.

STIG COMÉT

Matematikmaskinnämnden
(The Swedish Board for Computing Machinery)
Box 6131, Stockholm 6, Sweden

1. R. L. BIVINS, N. METROPOLIS, P. R. STEIN, & M. B. WELLS, "Characters of the symmetric groups of degree 15 and 16," *MTAC*, v. 8, 1954, p. 212.
2. S. COMÉT, "On the machine calculation of characters of the symmetric group," Tolfte Skand. Matematikerkongressen, Lund, 1953, *Comptes rendus*, p. 18–23.
3. G. H. HARDY & S. RAMANUJAN, "Asymptotic formulae in combinatory analysis," London Math. Soc., *Proc.*, s. 2, v. 17, 1918, p. 75.
4. A series of eight papers by A. YOUNG, "On quantitative substitutional analysis," published in London Math. Soc., *Proc.*, beginning in ser. 1, v. 32, 1901, p. 384, and ending in ser. 2, v. 37, 1934, p. 441.
5. T. NAKAYAMA, "On some modular properties of irreducible representations of a symmetric group," *Jap. J. Math.*, v. 17, 1941, p. 165 and p. 411.
6. This involves shifting to the left until the most significant digit is 1, and counting the shifts; see G. KJELLBERG & G. NEOVIUS, "The BARK, a Swedish general purpose relay computer," *MTAC*, v. 5, 1951, p. 29–34.

Determination of Steiner Triple Systems of Order 15

1. Introduction. A Steiner Triple System is a set of subsets of n marks such that: a) Each subset contains three distinct marks, b) each pair of distinct marks appears in one and only one subset. It is easy to see that $n \equiv 1$ or $3 \pmod 6$ [1] and that the number of subsets is $n(n-1)/6$. For $n = 3, 7, 9$ there is one and essentially only one such set, all others being obtained by permutation of the marks. For $n = 13$, there are two distinct systems.

Using SWAC for the computations we have obtained the non-isomorphic systems of order 15. During the course of the calculations, Prof. L. J. PAIGE called our attention to a series of papers [2] which culminate in the listing by F. N. COLE of a set of 80 systems of order 15 with proof that these are distinct and an argument tending to show that the listing is exhaustive. Since following the argument would entail tremendous hand calculation and since the number 80 was below our expectation so that it seemed quite possible that there had been errors or omissions in the determination, we decided to continue. It turns out that these 80 are indeed complete and that Cole's computations were either correct or at any rate did not contain any errors which led to the omission of a valid system. In any case these results testify to Cole's reputation for brilliant computation.

2. The computation—General considerations. There are two stages in the calculation, the formation of the systems and the reduction of isomorphisms. That these must be carried out simultaneously is seen from the fact that if all systems on 15 marks which arise from the 80 distinct systems are counted, the total exceeds 6.10^{13} (actually $(14521/315)15!$).