Conclusion

As you will have gathered already, we have had numerous difficulties causing much annoyance and embarrassment—the fact is, however, that these difficulties, though real and big enough at the time, have been superficial—so much froth on the surface. The fundamental skeleton of the system has proved entirely satisfactory; the lesson to be learned here is undoubtedly that when once you have set up an electronic data-processing system and convinced yourself that it is basically sound, stick to it—you will need a lot of tenacity, but do not be put off by criticism. You will find that it is easy for those not connected with electronic computers to lose their sense of proportion; relatively small troubles get magnified. This is not to say, however, that you must not be ready to set up temporary systems to keep things going until the appropriate program can be patched—we have had to do this on several occasions, but that is now something of the past.

Much remains to be done by the way of improving our systems and programs, but even at this early stage in the use of these new techniques it is clear that there are definite possibilities for extending the scope of the new systems and improving our service to the public, coupled with ultimate reductions in cost to ourselves.

This brings me to my conclusion. I have deliberately omitted many points—I fear that I have already spoken for far too long—but the fact is that we *have* made this wholesale switch to electronic data-processing, and it does work.

Acknowledgement

I should like to explain that much of what I have said is not my own. A great deal has been based on the work of my colleagues in Canada, and although it may have appeared that I was speaking from personal experience that has not always been so. I wish to put on record my thanks for the unstinting help afforded me by all my colleagues.

Note on a Test for Repeating Cycles in a Pseudo-random Number Generator

When a computer is generating sampling numbers by a deterministic process for use in a Monte Carlo problem, it is desirable that the numbers should satisfy certain tests of randomness and, in particular, it is undesirable that the sequence should repeat itself during the problem.

Some processes for generating pseudo-random numbers inevitably return to the starting number, while others are capable of returning to an earlier member of the sequence, not necessarily the starting number. In either case a repetitive cycle occurs which may or may not contain enough different random numbers for the process to be useful in a particular problem. To detect the formation of such a repeating cycle, the following empirical test is proposed. For each new member of the sequence compare a certain number, n, of its digits with a fixed pattern of n digits; the probability of a random binary number matching the fixed pattern is 2^{-n}, and n can be chosen so that this probability is of such a size that matches in a random sequence will occur neither too frequently nor too rarely. When a match is found the number of members of the sequence, generated since the previous match, is printed.

If cycles do not occur, the printed numbers are expected to be about 2^n on the average. If a cycle occurs in which no member matches the selected pattern, printing will stop and it will become more and more certain that a cycle has been found as time goes on; if a cycle containing one matching member occurs, all the printed numbers will be equal from then on.

There is a difficulty if a cycle occurs which contains many matching members, and this is that the printed numbers will themselves form a repetition which must be recognized by the programmer. If n is small the trouble involved in doing this is very great since the printed repetition is so long; if n is large, however, it may happen that no printing has occurred by the time that the machine has produced the number of random numbers needed in the particular problem, and, therefore, the occurrence of an unwanted cycle is not disproved.

It is suggested as a compromise, that 100 numbers would not be too long a printed record in which to detect repetitions by inspection, and, therefore, if N random numbers are needed one should choose n accordingly.

$$\text{Setting } 2^{-n} = \frac{100}{N} \text{ gives } n = \log_2\left(\frac{N}{100}\right).$$

Alternative methods are available for matching, and appear to be equally good for the purpose; either the n selected digits, which need not be consecutive ones, are matched against an n-digit pattern, or the n selected digits of the random number are required to be all ones. This latter method is simpler in some machines.

D. G. N. Hunter.