UNIVERSITY OF NORTH CAROLINA
Department of Statistics
Chapel Hill, N. C.

ON MONTE CARLO METHODS IN CONGESTION PROBLEMS
I. SEARCHING FOR AN OPTIMUM IN DISCRETE SITUATIONS

by

E. S. Page

February 1963

This research was supported by the Office of Naval Research under contract No. Nonr-855(09) for research in probability and statistics at the University of North Carolina, Chapel Hill, N. C. Reproduction in whole or in part is permitted for any purpose of the United States Government.

Institute of Statistics
Mimeo Series No. 354

ON MONTE CARLO METHODS IN CONGESTION PROBLEMS.[1]

I: SEARCHING FOR AN OPTIMUM IN DISCRETE SITUATIONS

by

E. S. Page

University of North Carolina, Chapel Hill, N. C.

and

University Computing Laboratory, Newcastle upon Tyne, England

= =

Introduction.

Monte Carlo methods have been far more often used to estimate means than the extreme points of distributions of finite or semi-infinite extent. When estimating a mean by r independent repetitions of some sampling process it is known that the standard error of the average of the observations is $\sigma/\sqrt{n}$ where σ is the standard deviation of a single observation. Usually it is assumed that conditions for the applicability of the Central Limit Theorem are satisfied* and the tables of the normal distribution are employed to set confidence limits for the unknown population mean. Consequently when all feasible devices to reduce σ have been used it is known that a ten-fold increase in accuracy will require a hundred-fold increase in computational work. This is certainly little comfort when results are required to an accuracy of a small fraction of σ (when Monte Carlo methods are likely to be impractical) but is a help in deciding the size of simulation studies necessary for applicable results. Even this degree of comfort is not present in simulation for extreme values. The probability that the smallest observation in

---

[1]This research was supported by the Office of Naval Research under contract No. Nonr-855(09) for research in probability and statistics at the University of North Carolina, Chapel Hill, N. C. Reproduction in whole or in part is permitted for any purpose of the United States Government.

*Examples are nevertheless met where contrary conditions hold e.g. Walsh (1956).

a sample of r is less than X is

$$1 - \{1-F(X)\}^r$$

where $F(X)$ is the distribution function for a single observation. For continuous distributions this tends to a limiting form (Fisher and Tippet 1928) which depends, of course, on the terminal point of the original distribution and its behaviour near this point. For some distributions it is known that the limiting distribution is approached very slowly.. For discrete distributions, let the probability that a single observation equal r be p_r; then the probability that the smallest member of a sample of r be less than r is

$$1 - \sum_r^\infty p_i^r \sim 1 - e^{-r \sum_r^{r-1} p_i} .$$

In the example we shall consider the individual p_i near the extremes of the distribution will be very small - the ratio of an integer at most a few hundreds to the factorial of an integer equal to 20 or more. Consequently the chances are not high that the most extreme classes will be represented in any random samples of feasible size. Further, if only the least observation is of interest, as a sample of r is taken one by one there will only be about $\log r - 0.4229$ occasions on the average when an observation less than all those previously observed will be obtained (Foster and Stuart, 1954). If one knew sufficient about the distribution to be able to approximate the probability of an observation less than a specified value not far from the extreme (in particular the least observed value in some sample of moderate size) then the distribution of the number of further observations necessary to improve on this value would be known. Naturally, the nearer to the extreme we are, the more work we have to do on the average to get nearer still. Large numbers of observations will be taken that are useless before an isolated one of interest is found.

These considerations imply that the rate at which useful information is obtained by independent random sampling is much less when estimating extremes than when estimating means. The suggestion is therefore to examine ways in which the previous observations can be used to modify the subsequent sampling and so to follow the line of development of Monte Carlo solution of deterministic problems. (Halton, 1962)

When the problem to be simulated has a number of well defined factors upon which the response is believed to depend the response surface may be explored and the optimum found by techniques such as those of Box and Wilson which have been so widely employed in industrial plant observations. The appropriate experimental design of the simulation runs can be decided and performed much more advantageously than haphazard searching over the possible experimental region.

However many problems of interest do not readily lend themselves to this treatment. For example, it may be required to maximise some function $f(x_1,\ldots x_n)$ over all possible permutations of the arguments; it is not then clear how factors or a response surface may usefully be defined.

Permutation Problems.

Suppose that the possible configurations in some problem are the $n!$ permutations of n objects $A_1 \ldots A_n$, and that we need to find without too much computation a permutation, π, such that the response $f(\pi)$ is to be optimum, or near to the optimum. In these problems the condition on limiting the amount of computation is clearly essential; the number of possibilities is finite and so in theory could be enumerated but the size of $n!$ prohibits this approach for all but the smallest values of n (i.e. n less than about 8 or 9).

A Crude Monte Carlo approach would be to sample permutations at random and to select the best permutation from a large sample. Such sampling would probably be performed without additional calculations to check for repetition. Such a method can undoubtedly get good results but at a high cost of computing; it makes no use of permutations with good responses that have been found in the search for better ones. Intuitively one would hope that the set of permutations "near" in some sense to a good one would be more likely to contain a better one than those far away. An assessment of what measures of nearness for two permutations will vary from problem to problem.

In (i) and (ii) two items are interchanged from the natural order. If the numbers are keys on pieces of paper

(i)	1	2	3	4	5	7	6	8
(ii)	1	2	7	4	5	6	3	8
(iii)	5	6	7	8	1	2	3	4
(iv)	8	7	6	5	4	3	2	1
(v)	1	3	5	7	8	6	4	2
(vi)	7	8	4	5	6	1	2	3

arranged in a pile (i) is a little easier to improve by hand than (ii); if the papers are being put into pigeonholes all within reach there is nothing to choose between (i) and (ii) but if the pigeon holes are far apart (e.g. a mailman delivering letters) (ii) gives appreciably more trouble than (i). Again, a pile of papers in order (iii) need only to be cut like a pack of cards and in (iv) to be dealt off the bottom of the pack; if these operations are not permissible ones the orders may be some way from the natural one. Order (v) is a "back to back" arrangement favored by postmen in Britain where odd numbered houses are on one side of the road and even numbered on the other; thus this order for this purpose is near the natural. Order (vi) like (iii) has sections of the natural order re-

arranged and a shuffling of these sections bodily can restore the natural sequence; accordingly, in this case, a measure of nearness might be the number of breaks in the order (thus, one for (iii) and two for (vi)).

A Scheduling Problem

In order to illustrate the modification of the Monte Carlo method suggested we consider a particular problem of practical interest. Given J jobs each requiring processing on M machines, each job passing from machine 1 to machine 2 and so on until completion. We require to decide on the order in which the jobs shall be presented to the machines so that some cost function shall be optimised or nearly so. For a cost function like the total time of completion of all the jobs it is reasonable to suppose that orders containing the same subsequences of jobs will tend to have cost functions near to one another. Thus (i), (iii) and (vi) would be near to one another, (ii) a little farther away and (iv) and (v) far apart. If the order 1 2 3 4 5 6 7 8 had been found to give a small cost it would be more reasonable to look at small perturbations of this order like (i), (iii) and (vi) than to grope around among orders far away.

The suggested method which we call Chain Monte Carlo is to select permutations at random from those within a certain distance according to some measure of the base permutation until a better one is found; this then becomes the base permutation and sampling is continued until no improvement has occurred for some time. The base permutation thus far obtained is clearly quite good and so the random search is continued in a closer region about it.

For the scheduling problem and cost functions like the total time of completion or the total idle time on the machines we have adopted as a measure of the distance between a given permutation and a base the number of items which are not followed by the same item as in the base permutation. Thus if the base is the natural order the distances of (i) to (vi) from it are respectively

3, 4, 1, 7, 6, 2 while the distance between (i) and (ii) is 4. It is known (Page, 1962) that permutations with good costs are nearer to an optimum permutation than chance ones in problems with small numbers of jobs when this measure is used.

Chain Monte Carlo Scheduling

Given J jobs and their machine times we first select permutations which are at most a distance $k < J$ from the starting one; whenever a lower cost is found the permutation giving it is taken as a new base. If N trials are made without improving the cost, samples are taken at most a distance $k/2$ from the base and so on until permutations distance 2 are to be sampled.

First, (k-1) independent uniform variates x_i in the continuous interval (0, J + 1) are formed by taking $(J + 1) \times u_i$ where u_i are the uniform variates in (0,1) provided directly by the pseudorandom number generator. If $0 = x_0 < x_1 < \ldots < x_{k-1} < x_k = J + 1$, the x_i divide the permutation 1, 2, ..., J into k sections consisting of the integers between (x_i, x_{i+1}) or the integers in the corresponding positions for any other base permutation. Some of these sections may of course be null. A random permutation of the k sections is then formed to give a permutation of 1, 2, ... J which is within a distance k of the base permutation. For example, if the base permutation is 2 5 6 8 1 7 3 4 $k = 4$ and $x_1 = 2.1$, $x_2 = 2.5$, $x_3 = 6.2$ the sections are (2,5)(null), (6,8,1,7), (3,4). If the permutation of (1,2,3,4) that is obtained is 4 2 3 1 the job permutation is 3,4,6,8,1,7,2,5 which is at distance 2 from the base.

It is important to note that although this process seems more complicated than taking a random permutation of 1, 2, ... J it is in fact at least as quick to perform. The time for the selection of a random permutation increases approximately linearly with the number of items (Page, 1962) and so for Crude Monte Carlo

is proportional to J; for Chain Monte Carlo the main work (involving the random number generator) is picking the divisions and forming the permutation of 1,2,...k - both of which are proportional to k. Since k wil usually be a small fraction of J and the constants of proportionality are comparable each sample in Chain Monte Carlo requires comparatively little computing.

Since the process is an exploration it is not vital that each attainable permutation should have an equal chance of selection so that changes for convenience in the method of selection are permissible. The number of permutations $V_{J,k}$ attainable from a given base increases rapidly with J, k. If, for example, the base permutation is 1, 2, ... J then each such permutation can be found by dividing 1, 2, ... J into $r \leq k$, non-empty sections (i.e. by selecting r-1 division points from J-1 possible ones) and re-arranging these r sections (in $u_{r,o}$ ways) so that none follows its predecessor in the base. Hence the number of attainable permutations a distance (r-1) from the base is

$$\binom{J-1}{r-1} u_{r,o} \quad .$$

Therefore

$$V_{J,k} = \sum_{r=1}^{k} \binom{J-1}{r-1} u_{r,o} \quad .$$

It is shown that (Page, 1962)

$$u_{r,o} = (r-1)\, u_{r-1,o} + (r-2) u_{r-2,o}$$

and hence

$$u_{r,o} = \frac{(r+1)!}{r} \left[\frac{1}{2!} - \frac{1}{3!} + \cdots + \frac{(-1)^{r+1}}{(r+1)!} \right].$$

The first few values of $u_{r,o}$ are given below. As r increases,

$$u_{r,o} \sim \frac{(r+1)!}{r\,e} \sim \sqrt{2\pi}\; e^{-r-1}\,(r^{r+\frac{1}{2}} + r^{r-\frac{1}{2}}).$$

Even for r as small as 4, the last formula gives $u_{r,o} \doteq 10.8$.

r	1	2	3	4	5	6	7	8
$u_{r,o}$	1	1	3	11	53	309	2119	16687

It is clear that any permutation can be reached from any base in a finite number of steps. The first k-1 items at least can be made to coincide at the first transition and so on at succeeding transitions. Many other sequences of the same length or longer can achieve the same final permutation.

Results of Simulations

The Chain method has been tried for six runs each on several examples constructed by Heller (1959) and compared with his Crude Monte Carlo results. The first two problems have 10 machines and 20 jobs, one having a small range of machine times (0-9) and the other a larger range (1-99). The third example has 5 machines and 30 jobs with machine times ranging from 3-997. In each case the base permutations were divided into $k=16$ pieces which were then rearranged at random and this number of divisions was continued until 100 such trials had resulted in no improvement in the cost. Then the number of divisions was halved until $k=2$. It appears that there is little point in performing the divisions with $k=2$; in a few test runs of 100 trials no improvements were noticed and even if some had been suspected it would be more efficient to try these J-1 permutations systematically.

Example 1: 20 jobs, 10 machines: small range

	Crude Monte Carlo	Chain Monte Carlo					
		Run 1	Run 2	Run 3	Run 4	Run 5	Run 6
Completion time	150	151	151	148	147	146	150
Number of trials	3,000	514	368	455	532	459	643

Example 2: 20 jobs, 10 machines: large range

	Crude Monte Carlo	Chain Monte Carlo					
		Run 1	Run 2	Run 3	Run 4	Run 5	Run 6
Completion time	1598	1569	1592	1535	1596	1651	1570
Number of trials	10,000	432	549	610	739	462	431

Example 3: 30 jobs, 5 machines

	Crude Monte Carlo	Chain Monte Carlo					
		Run 1	Run 2	Run 3	Run 4	Run 5	Run 6
Completion time	16350	16699	16635	16802	16714	16401	16947
Number of trials	15,250	522	410	703	432	573	462

The orders achieved by the Chain method in the first two examples are slightly better than the Crude sampling and a little worse in the third example. However, while the Chain method took about 500 trials for its results those from the Crude samples are the best observations in samples from six to thirty times as large.

Naturally lower completion times often can be found at more computing cost by increasing the number, N, of the unsuccessful trials permitted before searching in a smaller region (i.e. halving k). The average completion times for the 30 job problem for six runs with different N's are shown in Table 3 below. The

Effect of Increasing N

N	50	100	150	200
Average Completion Time	16812	16700	16666	16537
Average Number of Trials	252	510	640	960
Time after exchanging	16566	16644	16611	16536
Average Total Number of Trials	353	549	683	994

costs of delays in completion and of computing will determine just how much computing can be undertaken.

Further Improvement of the Order.

Some improvement of the best order obtained by Chain Monte Carlo can sometimes be gained by conducting a systematic search among some of the close permutations. One of the simplest methods is Exchanging (Page 1961) in which adjacent jobs are interchanged and retained in that order if an improvement results. Cycles of these J-1 interchanges are continued until no improvement occurs during a whole cycle. We are thus looking at some of the permutations within a distance 3 of the base. It is clear from the lower rows of Table 3 that a little effort so invested is worthwhile especially on the orders produced in the shorter Monte Carlo runs.

Both the Chain method and exchanging seek improvements in an existing order. It is uneconomic to exchange starting with a poor order since the successive improvements obtained are likely to be only small; chain scheduling does not suffer from the same disadvantage.

It is a feature of the Chain and exchanging methods that when one good order is found several others with equal or slightly greater costs appear too. For example, one run with N=150 and a few cycles of exchanging gave several orders with completion times 16153, 16170, 16298, 16318, 16339 for the 30 job problem, all better than the Crude Monte Carlo order after over 15,000 trials.

Efficiency of Chain Monte Carlo

The efficiency of Chain relative to Crude Monte Carlo will clearly vary from problem to problem and will certainly depend upon the measure of distance that is adopted. In the scheduling problem considered above there is ample evidence from the runs performed that the completion times for orders near to a good order are lower than the mean for all orders; the lower the base order and the nearer the derived orders the lower is the mean of the derived orders. The corresponding distributions differ little and accordingly the Chain selection method tends to discover better orders more quickly than Crude Monte Carlo sampling especially once a fairly good order has been obtained. If the initial base order were a very poor one there might be several Chain trials before a good order were discovered; however if k is initially taken large enough so that a substantial region of the space of permutations may be explored the hindrance is likely to be slight.

It seems that the gain of the Chain method in the scheduling problems is about a factor of 5 or 6 over Crude sampling. However this gain is not sufficient to make the Monte Carlo method preferable to some deterministic methods when they are applicable. For example, the merging method following by exchanging

(Page, 1961) applied to the 30 job, 5 machine problem gave an order better than that found in the Crude Monte Carlo trials with about one two hundredth of the computing (although the order was not as good as some of those listed above from Chain Scheduling). Similarly the Chain method on examples of the Travelling Salesman problem gives better results than Crude Monte Carlo but not as good as the approximation technique based on Dynamic Programming (Held and Karp, 1961). Accordingly neither Crude nor Chain Monte Carlo would be recommended in problems for which satisfactory deterministic methods are available. However for other problems Monte Carlo may be a last resort and then techniques such as the Chain method for increasing the power of Crude Monte Carlo are welcome.

REFERENCES

Fisher, R. A. and Tippett, L. H. C. (1928). Limiting forms of the frequency distribution of the largest or smallest member of a sample. Proc. Camb. Phil. Soc. 24, 180-190.

Foster, F. G. and Stuart, A. (1945). Distribution free tests in time series based on the breaking of records. J. Roy. Statist. Soc. B 16, 1-13.

Halton, J. H. (1962). Sequential Monte Carlo. Proc. Camb. Phil. Soc. 58, 57-78.

Held, M. and Karp, R. M. (1962). A dynamic programming approach to sequencing problems. J. Soc. Indust. Appl. Math. 10, 196-210.

Page, E. S. (1961). An approach to the scheduling of jobs on a computer. J. Roy. Statist. Soc. B, 23, 484-492.

Page, E. S. (1962). On the scheduling of jobs by computer. Comp. J. 5, 214-221.

Walsh, J. E. (1956). Symposium on Monte Carlo Methods. Wiley, New York. p. 141-144.