# MONTE CARLO AND STOPPING RULES FOR SOME COMBINATORIAL PROBLEMS

Paul H. Randolph
Room 4530—Federal Building
1200 Pennsylvania Avenue
Washington, D.C. 20461

Oren N. Dalton
National Range Operations Directorate
White Sands Missile Range, N.M. 88002

## ABSTRACT

One major problem in combinatorial theory is to find a combination that will optimize a given objective function. However, for many real problems even with moderate requirements, no algorithm exists which can find an optimum in a livable span of time. This paper proposes that if the N! combinations are viewed as points in a sample space, then a Monte Carlo sampling procedure will provide a "good" solution. Statistical stopping rules are used to determine when the solution is "good enough," as well as provide an estimate of the closeness of the solution to the true optimum.

## INTRODUCTION

Many problems of practical significance can be considered as problems of combinatorial mathematics. In particular, the problem of finding a combination that optimizes a given objective function is one that has plagued many people. A classical example is that of job shop scheduling. In essence, the problem is to schedule J jobs on M machines in such a way that the total make time is minimized and also that a technological ordering of the machine operations on each job is maintained.

There are few formal algorithms for finding optimal combinations for such problems, except by complete enumeration, and there are virtually none for, what might be viewed as, the "general" combinatorial problem. Among the few exceptions we might mention the solution to the two-machine J-job problem given by Selmer Johnson [7], and his solution to a special case of the three-machine, J-job problem. Also, Ford and Fulkerson [4] have a rather slick way to find the maximum flow in a network.

Another classical example is known as the traveling salesman problem in which it is desired to find the shortest route by which a salesman can visit a given set of cities. Some algorithms have been developed for solving this problem for a fairly respectable number of cities [8]. However, if more "realism" is injected, such as by restricting the visit time of the cities to certain days and imposing priorities, by considering transportation facilities, etc., there is no general answer for this problem either.

Although an optimal combination cannot be found for many problems, such problems still exist, they require solutions, and various empirical methods are used to find them. For example, in the job shop scheduling problem, Gantt charts have been used for years for finding schedules. The Gantt chart aids the scheduler to visualize a subset of all possible schedules; he then chooses the best of these. It is possible, of course, for this to be an optimal schedule, but it is more likely to be no more than a feasible schedule; it could be far from optimal.

Since a computer can generate combinations very rapidly, it seems reasonable to extend this concept to computer technology. A computer generated set would, perforce, be far greater than any that could be generated by hand methods. On the other hand, even a computer will not be able to examine more than a fraction of the total number of possible combinations for most combinatorial problems. For example, in the traveling salesman problem, the total number of possible routes for an N city problem is N!. To get a feel for the size of N!, consider a 23-city tour. At one microsecond per tour, the total time for a complete enumeration would exceed the estimated age of the universe! Even the fastest computer cannot generate a miniscule fraction of all the possible combinations.

In this paper, we will be concerned with combinatorial problems, not amenable to standard or extant algorithms, which yield large numbers of possible combinations; the kind, incidentally, most often encountered in the real world. A Monte Carlo technique, combined with a set of statistical stopping rules, is a major contender (indeed, it may be the only contender) for finding "good" solutions to general combinatorial problems. Reiter and Rice [13], for example, have suggested using Monte Carlo for linear and nonlinear programming problems. Randolph, et al [12] and Swinson, et al [15] have applied this technique to dynamic programming problems. Monte Carlo is the method of choice in this paper for finding solutions to large scale recalcitrant combinatorial problems.

In order to form a structure to take advantage of the Monte Carlo technique, each combination is defined as a sample point, and the total number of possible combinations defines a sample space. The sample space may be enormous. Concomitant with this is the necessity for describing the elements of a sample as optimal or near-optimal, and the degree of confidence of such a description. This is provided by treating the sample space in a probabilistic manner, and defining a superstructure by which a sample point can be given a numerical description. This may then be compared with some sort of numerical expression representing a goal or ultimate optimum.

Two questions remain:

1. How large should the sample be?

2. How should the combinations be chosen?

Consideration of the first question forms the major emphasis of this paper. Essentially, we suggest that combinations be generated randomly one-at-a-time and evaluated—that is, by Monte Carlo sampling. This random generation continues until the "cost" of sampling exceeds the expected improvement that can be anticipated from additional samples. At that time, sampling is stopped. The place at which sampling is stopped is determined through statistical stopping rules.

As to the second question, a number of proposals have been advanced. We will mention some of them later and indicate our own answers.

## STATISTICAL STOPPING RULES

Let $X_1$, $X_2$, ... denote the random variables of the payoffs associated with generating successive combinations by Monte Carlo sampling methods. For the present, assume that each combination payoff is an integer and that the objective of the combinatorial problem is to find a combination for which the payoff is maximized. Furthermore, without loss of generality, assume that all payoffs are positive and bounded above by the known integer $l$. Also, let $y_n$ denote the maximum of the observed payoffs $x_1,\dots,x_n$ obtained from the first n combinations, that is, $y_n = \max(x_1,\dots,x_n)$.

The probability function for each combination payoff is the multinomial characterized by $P(X = k) = p(k)$, $k = 1,\dots,l$. When the values of $p(k)$ are known, then from [2], it is evident that the stopping rule is obtained by calculating the expected increase in gross payoff associated with generating another combination,

$$T(y_n) = \sum_{k=y_n}^{k=l} (k-y_n)p(k),$$

and comparing this with the relative cost c of generating a single combination on the computer; that is, if $T(y_n) > c$, continue to another combination; if $T(y_n) \leq c$ stop and use the best combination already generated. The function $T(y_n)$ is sometimes called the stopping rule function.

For combinatorial problems, however, the values of $p(k)$, $k-1,\dots,l$, are not known, and thus this rule is not appropriate. Instead, a Bayesian stopping rule [10,11] can be used. To find such a rule, it is necessary to define the vector $\theta = (\theta_1,\dots,\theta_l)$ such that, for the nth observation $X_n$, the probability function is given by $P(X_n=k|\theta) = \theta_k$, $k = 1,\dots,l$, where $\theta$ is an element of the simplex

$$S = \{\theta: \sum_{k=1}^{k=l} \theta_k = 1,\ \theta_k \geq 0, k=1,\dots,l\}$$

Since the conjugate prior density [10] for the multinomial is the Dirichlet function, the initial prior density of $\theta$ can be written as

$$f_0(\theta) = \Gamma(m)\prod_{k=l}^{k=1} [\theta_k^{m_k-1}/\Gamma(m_k)],$$

where $m_1,\dots,m_l$ and $m = \sum_{i=1}^{i=l} m_i$ are strictly positive parameters of the distribution. If n payoffs have been observed and if $n_k$ is the number of the payoffs having value k (i.e., $n_k=\#\{i:x_i=k, i=1,\dots,n\}$, where $\sum_{k=1}^{k=l} n_k=n$), then the posterior density of $\theta$, given the n payoff values, is

$$f_n(\theta) = \Gamma(m+n) \prod_{k=1}^{k=l} [\theta_k^{m_k+n_k-1}/\Gamma(m_k+n_k)].$$

This is the Bayesian prior density of $\theta$ for observation $X_{n+1}$. Furthermore, since the joint density function for $X_{n+1}$ and $\theta_k$ is $\theta_k f_n(\theta)$, then the marginal distribution

$$p_n(k) = (m_k + n_k)/(m + n), \quad (k=1,\ldots,l)$$

is the probability that $X_{n+1}$ will take on the value k. Thus, the conditional expected gross improvement in the payoff for the (n+1)st combination is seen to be

$$T_{n+1}(y_n) = \sum_{k=y_n}^{k=l} (k-y_n)p_n(k)$$

$$= (m+n)^{-1} \sum_{k=y_n}^{k=l} m_k(k-y_n).$$

This is the stopping-rule function for an unknown multinomial distribution of combination payoffs. Comparing the value of this function with the value of c will determine the stopping point; that is, if $T_{n+1}(y_n) \leq c$, the sampling of combination payoffs should be stopped. Since $y_n$ is a monotonically nondecreasing function of n, then $T_{n+1}(y_n)$ is a decreasing function of n, which approaches zero as n increases. Thus, sampling will always stop eventually.

The stopping-rule function depends on specifying a set of parameters associated with the prior Dirichlet density function. If these parameters $m_1,\ldots,m_l$ are examined, it will be noted that they can be written in terms of the initial probabilities as $m_k = mp_0(k)$, $k=1,\ldots,l$. Since the $p_0(k)$ are essentially normalized values of the $m_k$, it may be preferable to specify the $l-1$ independent initial probabilities and the parameter m, rather than to estimate the $m_k$ directly.

The parameter m has some interesting characteristics. A lower bound for m is zero, and this can be a greatest lower bound only when $p_0(k) = 1/l$, $k=1,\ldots,l$. As $m\to 0$, then $T_2(y_1)\to 0$, and the Monte Carlo process stops with the first observation, implying no confidence in the initial probabilities and complete confidence in any data value. On the other hand, as $m\to\infty$, then

$$T_{n+1}(y_n) = m(m+n)^{-1} \sum_{k=y_n}^{k=l}(k-y_n)p_0(k) \to$$

$$\sum_{k=y_n}^{k=l} (k-y_n)p_0(k) = T(y_n),$$

which is the expected improvement for a known multinomial distribution, indicating a complete confidence in the initial probabilities and no amount of data will shake the experimenter's confidence in this initial probabilities. Thus, the parameter m can be interpreted as a coefficient of confidence in the initial probabilities. In fact, it can be considered as being analogous to the sample size that would be needed to obtain through a random sample the same quality of estimate of $p_0(k)$ as those given by the specified prior probabilities.

In order to determine the prior distribution of $\theta$ for discrete payoffs, values of $m_k$ may be obtained through specifying the initial probabilities $p_0(k)$. To find $p_0(k)$ for continuous payoffs, suppose that the sequence payoffs can assume arbitrary values in the interval $[0,l]$, and let $A_1,\ldots,A_v$ be any partition of this interval, where $A_k$ is defined as $A_k = [x_{k-1},x_k]$, $k=2,3,\ldots,v; A_1 = [0,x_1]$. Suppose $H(x)$ is a distribution function of $[0,l]$ such that for a continuous probability function for a sample space having arbitrary maxima in the interval $(0,l)$ could be achieved by a limiting process of a partition $A\triangleq\{A_1,\ldots,A_p\}$ of this interval. If $A_k \epsilon A$ is defined by two points: $(A_k \triangleq (x_{k1}, x_{k2})$, and if $H(x)$ is the distribution function over $(0,l)$, then

$$P_0(A_k) = \int_{A_k} dH(x) = H(x_{k2})-H(x_{k1})$$

is the prior intuition of the initial probabilities regardless of the method of partitioning $(0,l)$. If $x_k$ is any point in $A_k$, then $T_B(y_n)$ is the integral:

$$x(x+n)^{-1} \int_{y_n}^{l} (x-y_n)dH(x) = \lim_{\nu\to\infty} m(m+n)^{-1}$$

$$\sum_{k=1}^{\nu} (x_k-y_n)[H(x_{k2})-H(x_{k1})]I(x_k \geq y_n).$$

A number of possible prior distributions have been examined. The uniform seemed likely because it exemplified our ignorance. Since our ignorance was not absymal, we tested this hypothesis; several thousand combinations were generated randomly. Like Heller [12], we found that the distribution of the combinations tended to be clustered about that distribution of impeccable breeding, the normal.

Assume that a normal distribution reflects the experimentor's faith in the initial probabilities. Then if $\Phi$ is the standardized normal distribution function and $\phi$ is the corresponding density function, we have

$$T_{n+1}(y_n) = m(m+n)^{-1} \{\sigma[\phi(z_y)-\phi(z_l)]$$

$$+ (\mu-y_n)[\Phi(z_l)-\Phi(z_y)]\}$$

where $\mu$ and $\sigma$ are the mean and variance of the prior distribution, $z_y$ is $(y_n-\mu)/\sigma$, and $z_l$ is $(l-\mu)/\sigma$. In any problem, initial observations can be used to estimate these variables; as the problem continues they are continuously updated.

## A GENERALIZATION OF THE TRAVELING SALESMAN PROBLEM ANA A DISCUSSION OF SOME EXPERIMENTAL RESULTS

In this paper, emphasis has been to find "good" solutions, or combinations, for problems unsolvable by any extant methods. In particular, the authors have had experience with a type of problem which might be designated as a Generalized Traveling Salesman Problem. Various methods have been devised for finding solutions to the classic form of this problem. Bell [1], for example, reviewing many procedures and presenting some accelerated algorithms, examines it in some depth. However, the classic form rarely reflects the anticipated activity of a real salesman, nor the anticipated activity of cognate problems. In fact, when it is generalized, it scarcely resembles its classic prototype. On the other hand, it seems to epitomize a far broader class of problems which might be referred to as scheduling problems.

Assume we begin with the classic problem: a salesman is to visit N cities in the most facile manner possible. Now the first hooker: this is to be accomplished over a time-span, T. In fact, it may not even be possible to visit all N cities during T. Furthermore, to each city $C_i$, is assigned a priority, $P_i$. Such an assignment is reasonable and, in fact, is likely for any real situation. Clearly, for sufficiently large T, if all the $P_i$ are equal, the classic problem emerges. Although, in the classic prototype, consideration of problems of transportation are ignored; a generalization would consider them. Thus for each pair of cities, $C_i$ and $C_j$, let

$$T_{xij} = \{T^1_{xij}, T^2_{xij}, \ldots\}$$

be a set of "time windows" which describe the times (in T) in which transportation is available from $C_i$ to $C_j$. Note that in general, $T_{xij} \neq T_{xji}$ for $i \neq j$. It is immediately obvious that each city may also be tagged with respect to T. That is, let

$$T_{ci} = \{T^1_{ci}, T^2_{ci}, \ldots\}$$

be the time windows in T in which city, $C_i$, may be visited. For example, it may be that our salesman can visit city, $C_i$, only on Tuesday or Friday and, perhaps, be restricted as to the time-of-day in which the visit can be made. This, of course, suggests that the priorities may also be tagged relative to T. Thus, let

$$T_{pi} = \{T^1_{pi}, T^2_{pi}, \ldots\}$$

be the relative priorities assigned to different time windows for city, $C_i$. The reasoning behind this might be that on Tuesday an excellent contact can be made, but on Friday the contact would be less advantageous.

At this point, the problem is formidable, but it may be necessary to include a further restriction which might be characterized as a "dynamic" priority. It could be positive or negative. Suppose we obtained a schedule with the sequence: $C_i \rightarrow C_j \rightarrow C_k \rightarrow C_m$, in which the salesman is to visit the indicated city during the day and travel at night. At cities $C_i$ and $C_j$, presumably, he would be fresh and could do his best. But at city $C_k$ he may be tired so that his performance suffers, a condition intensified at $C_m$. A kind of negative dynamic priority could be assigned here, negligible at $C_j$, significant at $C_k$, and large at $C_m$. This "fatigue factor" can, obviously, refer to men or machines. (Another common form of negative dynamic priority is that of overtime. That is, a balance between the urgency for extra production versus the increased attendant cost.) Any number of contingencies can be imagined which could induce a positive dynamic priority. That is to say that the priority of a visit to one city is enhanced by a visit to another. All the priorities listed are in addition to the common ones of least time, least cost, or least distance.

This, then, is the kind of problem to which the technique of Monte Carlo and stopping rules is addressed. Even without dynamic priority, it is clear that this generalization is not amenable to any extant algorithm. Moreover, dynamic priority can rarely be evaluated before a schedule has been formed.

We should mention one further type of restriction which is common to many problems of this kind. Let C be the set of cities, J be a set of salesmen, and for $J_i \epsilon J$ associate a subset of C, $\Gamma_i \epsilon C$. For any $J_i$, $J_j \epsilon J$, $i \neq j$ it may be that $\Gamma_i \cap \Gamma_j = \Lambda ij \neq \phi$. If $T_{\Lambda i}$, $T_{\Lambda j}$ are sets of the times of visits by $J_i$, $J_j$, to $\Gamma_i$, $\Gamma_j$, respectively, then we have an exclusion (or conflict-free) requirement that $T_{\Lambda i} \cap T_{\Lambda j} = \phi$. In any real problem, it may be that this exclusion requirement is necessary in some cases but not in others.

It is obvious that the above generalization will apply to a broad class of problems. (We called the parameter, T, time, but it needn't be.) For example, the transportation problem, for real trucks on réal roads, will be hedged in by restrictions. If it is generalized, its appearance will be similar to the above. The scheduling of ships for duty or dry-dock or deployment of material are other examples.

A problem, similar to that discussed above, without dynamic priority, was investigated by the authors. In our context, "cities" were machines and "salesmen" were jobs. We called it, with commendable originality, the "scheduling algorithm."

Each possible schedule was considered as a sample point and our sample space was defined for problems, typically, involving 20 to 40 jobs, each job requiring 5 to 20 machines. Considerable conflict was built into the testing programs and the machines were hedged in by restrictions. Samples were produced from random permutations of the jobs to be scheduled. We evaluated problems for which the optimum schedules were known, others for which it could be estimated, and some in which no a priori information was available. We were able to produce optimal or near optimal schedules, and a numerical value, the variance, which could be flourished as a measure of our confidence in the degree of optimality.

The question remains as to how the combinations should be chosen. Early versions of the scheduling algorithm spent an inordinate amount of time slogging through the lowlands rather than scaling the peaks. Neighborhood search techniques discussed by Peterson [9] and Reiter and Sherman [14] increase the efficiency of the Monte Carlo procedure by exhaustively searching all combinations in a "neighborhood" of a given solution. Green and Randolph [5] describe such a neighborhood search technique wherein, for each random permutation of N jobs, a total cycle of them is examined. That is, the first job to be scheduled in a particular permutation is moved to the bottom and the resultant schedule is again evaluated. This cyclic shift is continued for all N jobs, and the neighborhood maximum is chosen as the sample point. This search was applied to the scheduling algorithm, and a number of savings resulted:
(1) Evaluating a succeeding schedule in a cycle requires considerably less effort than evaluating a schedule from a new random permutation; (2) The average of local maxima is significantly greater; and especially important; (3) The value of the variance shrinks to a fraction of the values of individual schedules with the result that far fewer schedules are required for a given estimate of the level of optimality. Typically, in our investigations, about 15 local maxima were needed before the variance had shrunk one-hundredth to one-thousandth of its initial value.

Several authors have proposed a variety of more complex neighborhood search techniques [9, 14], but our experience showed that computer time soared and the mean and variance were not improved significantly. Similarly, dispatch rules [9] gave poorer results than random sampling with a cyclic neighborhood research. Reiter and Sherman [14] suggested that perhaps future search should be near a "good" result, if found, but this seemed to prevent the finding of better schedules as well as debasing the values of the mean and variance.

It appears that if search rules are imposed on the algorithm, a concomitant structure is imposed on the sample space. Reiter and Sherman's suggestion, in particular, would indicate that good combinations or schedules would be found clustered together; that some sort of functional relationship, a kind of local modality, exists on the sample space. This was found to be not true. A mediocre or poor schedule was as likely to be found near a good one as another good schedule. Indeed, if the majority of the combinations were mediocre (as was true in our experiments) a "good" schedule was more likely to be surrounded by mediocre ones than otherwise. Thus, using a cyclic neighborhood search method, provided the best solutions with the least amount of computer time in scheduling problems.

## CONCLUSIONS

Monte Carlo sampling offers a method for finding combinations for the generalized combinatorial problem, and stopping rules provide a logical procedure for terminating the sampling process. The combination selected will not necessarily be the best combination, and, in fact, for large-size problems, the best solution will almost never be obtained. However, the marriage of Monte Carlo sampling and Bayes stopping rules in general selects a "good" solution and also provides a statistical measure of how close to the optimum the selected solution might be through the expected improvement value.

Finally, the ease of applying Monte Carlo sampling with stopping rules in sequencing indicates that this process could probably be applied to a wide variety of problems in combinatories.

## BIBLIOGRAHPY

1. Bell, D.E., "The Resolution of Duality Gaps in Discrete Optimization," Tech Rpt. 81, Methods of Operations

Research, Operations Research Center, MIT Cambridge, Mass., Aug. 1973.
2. Chow, Y.S. and Robbins, H., "On Optimal Stopping Rules," Zeit. Warscheinlichkeitstheorie, 3:33-49 (1963).
3. M. DeGroot, "Some Problems of Optimal Stopping," J. Roy. Stat. Soc. 30, 108-122 (1968).
4. Ford, L.R., and Fulkerson, D.R., "A Suggested Computation for Maximal Multi-Commodity Network Flows," Management Science, 5:97-101 (1958).
5. Green, R., and Randolph, P.H., "How to Beat Monte Carlo: An Improved Scheduling Technique," Transactions of the Eighteenth Conference of Army Mathematicians, ARO-D Report 73-1, Office of the Chief of Research and Development, Army Research Office, Durham, NC, 1973.
6. Heller, Jack, "Some Numerical Experiments for an M×J Flow Shop and Its Decision Theoretical Aspects," Operations Research, 8:178-184 (1960).
7. Johnson, S.M., "Optimal Two-and-Three-Stage Production Schedules with Setup Times Included," Naval Research Logistics Quarterly, 1:61-8 (1954).
8. Little, J.D.C., et al, "An Algorithm for the Traveling Salesman Problem," Operations Research, 11:979-989 (1969).
9. Peterson, C.C., "Solving Sequencing Problems Through Reordering Operations," AIIE Transactions, 5:68-73 (1973).
10. H. Raiffa and R. Schlaifer, Applied Statistical Decision Theory, Harvard Business School, Boston, Massachusetts, 1961.
11. P.H. Randolph, "Optimal Stopping Rules for Multinomial Observations," Metrika, 14: 48-61 (1968)
12. Randolph, P.H., Swinson, G.E., and Walker, M.E., "A Non-linear Programming Warehouse Allocation Problem," in E.M.L. Beale, Applications of Mathematical Programming Techniques, The English Universities Press Ltd., 235-249 (1970).
13. Reiter, S. and Rice, D.B., "Discrete Optimizing Solution Procedures for Linear and Nonlinear Integer Programming Problems," Management Science, 12:829-850 (1966).
14. Reiter, S., and Sherman, G., "Discrete Optimizing," Jour. Soc. for Ind. and Appl. Math, 13: 864-889 (1965).
15. Swinson, G.E., Randolph, P.H., Dunn, B.S., Walker, M.E., and Williams, R.D., "A Model for Allocating Interceptors from Overlapping Batteries: A Method of Dynamic Programming," Operations Research, 19:183-193 (1971).