INFORMATICA, 1992, Vol.3, No.2, 198–224

GLOBAL OPTIMIZATION OF A RANDOM WALK FUNCTION

Morgan HERNDON, Cary PERTTUNEN,
Bruce STUCKMAN

Department of Electrical Engineering
University of Louisville
Louisville, Kentucky, USA

Abstract. A random walk can be used to model various types of discrete random processes. It may be of interest at some point to find the peak of this function. A direct method of doing so involves evaluating the function at every point and recording the highest value. However, it may be desirable to find the peak without having to evaluate the function at every point. A search technique was developed to find the peak of a random walk with a minimal number of function evaluations using probabilistic means to guess at where the peak will most likely occur given the parameters of a specific function. A computer program was written to implement the search strategy and a series of random walk functions of varying lengths were generated to test its performance. Data was compiled and the results show that the search is capable of finding the peak with a significant reduction in the number of function evaluations needed for a point by point search, especially for functions of greater walk length.

Key words: random walk function, global optimization.

1. Introduction. The essence of engineering is solving problems. Naturally, engineering strives to find the best solution to a problem. This process is known as optimization. Optimization constitutes a broad and important field of study in all disciplines of engineering. In the field of electrical engineering alone widespread applications include the design of communication systems (Stuckman and Laursen, 1987), nonlinear control systems (Stuckman and Hill, 1986), electronic circuits (Groch, Vidigal and Director, 1985), and optical filters (McKeown and Nag, 1974).

The field of optimization can be categorized into problems of the continuous type and problems that are discrete. Continuous optimization requires that functions be those of continuous variables, meaning that the independent variable may take on any real valued number at any point in time. The functions themselves may discrete or continuous in nature, but the independent variable must always be continuous. Several techniques designed to find the peak of such functions with a minimal number of evaluations have been developed and refined over the past years as computer technology has improved to make implementation of these techniques more feasible. One particular set of methods which has been proven to converge upon the peak more quickly than other methods are the Bayesian sampling techniques (Easom, 1990), (Mockus, 1989). The Bayesian methods typically utilize conditional probability to conduct a search based on known information, in this case evaluated of the function.

However, these methods are not designed for functions of discrete variables. Discrete variable functions are those where the independent variable is allowed to take on only rational, or more commonly, integer numbers. This paper presents the development of an algorithm for finding the global peak of a random walk function. The most direct way to find the peak is to do a point by point comparison test, but this method requires that every point along the function be evaluated before the peak can be determined. The strategy presented here attempts to minimize the number of points which must be evaluated by using probabilistic and successive iteration to guess at where the peak will most likely occur. This work constitutes some of the first steps toward developing an optimization technique for discrete functions by applying Bayesian optimization methods to a specific class of discrete functions known as the random walk. Discrete optimization can be narrowed down to the area of combinatorial optimization which is concerned with the study of arrangements of discrete objects from a finite set (Lawler, 1976). This work may be viewed as an exercise in combinatorial optimization because central to the operation of the algorithm in determining where to make its next guess is the analysis of all fea-

sible walks between each of the evaluated points. This emphasis on arrangement will become more apparent in the section devoted to the derivation of the algorithm. Hopefully, the techniques applied here can later be expanded to a broader class of discrete variable functions until, ultimately, a general application of Bayesian methods for discrete function optimization has been formulated.

The next section discusses the algorithm by defining a random walk, describing the search strategy, defining important equations, and stepping through one example search. The third section presents results of an computer implementation of the algorithm on several sets of random walk functions of various lengths. The fourth and fifth sections present conclusions, and areas of further study.

2. The algorithm

The random walk function. The random walk function, like the one shown in Fig. 1, is a step function with discrete independent and dependent variables. The independent variable typically represents time. It is conventional to have the random walk begin at time zero and progress sequentially, although the function's origin is allowed to be shifted anywhere along the axis if needed. The dependent variable represents the height of the function at any particular time and may take on any integer value. The function is unique in that for each unit of time along the X-axis, the function has an equal probability of one-half of instantaneously jumping up or down one unit the Y-axis. The function cannot remain at the same value for two consecutive time intervals nor can it jump more than a single unit at a time.

The random walk function is a discrete model for a Brownian motion process, which attempts to imitate the random movement of a particle in a medium caused by the molecular motion of that medium. However, the function may best be understood as a model for tossing a coin n successive times, where the function jumps up one unit when one side of the coin shows face up and jumps down when the other side appears. The random walk will serve as the basis for the following algorithm and its results.

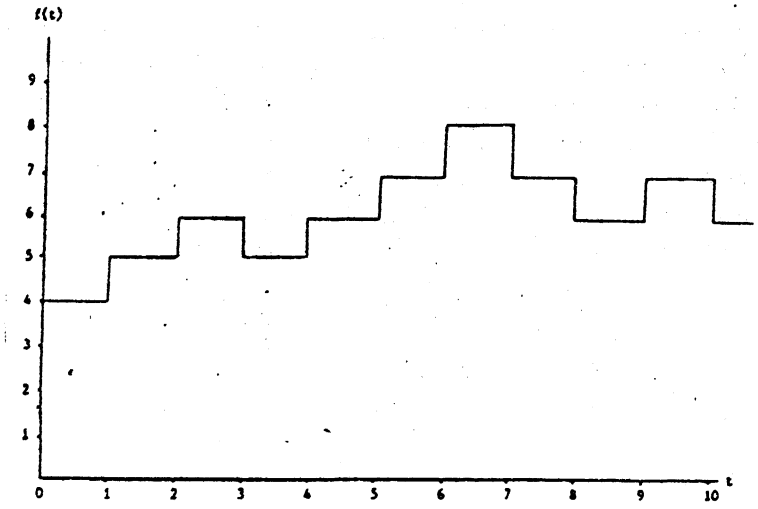

Fig. 1. A random walk function.

Search strategy. The search strategy for finding the peak of a random walk function is based on a probabilistic method of guessing where the peak will most likely occur. At each guess location the function is evaluated and broken up into two segments about that particular point. As more guesses are made, the original function is further subdivided into smaller and smaller segments until the peak value is determined, based on a zero probability of finding a greater maximum.

Knowing only the endpoints of the function and their respective values, the search begins looking for a function maximum which is at least one greater than the current maximum. This will arbitrarily be called C value. At each interval of time, ΔT, the probability of finding the C value is calculated using equations which will be detailed later. The time with the highest probability of containing the C value, and thus a new maximum, is chosen as the best place to look. The function is evaluated at this guess location and then it is divided into two segments about this point.

The procedure is now repeated for each of the two segments. In each of the two segments, the search is looking for the same C value, but each segment will yield its own local guess location with

a corresponding probability indicative of the likelihood of finding a value greater than or equal to the C value. Consequently, for two segments, there are two guess locations and the one with the greatest probability of containing the present C value will be chosen as the next place to look. Again, the function is evaluated at this location and the segment containing the guess location is divided into two segments about this point while the other segment remains intact and its values are undisturbed.

The original function has been divided into three segments and the same procedure is repeated for each of them. Again, each segment yields a local guess location with an associated probability. The guess location having the greatest probability of all three will be chosen as the next location to evaluate the function, and the segment containing this guess location is divided into segments while the other segments and their properties remain unchanged.

The process continues to divide the original function domain into increasingly smaller segments until it converges on the peak. If at any time, however, the value of the function at a particular guess location turns out to be a new maximum, the C value is accordingly adjusted to one more than this new maximum. When this occurs, regardless of how many segments the function has been divided into, all of the segment guess locations and their corresponding probabilities are updated and recalculated based upon the new C value. Sometimes the guess locations change and sometimes they do not, but almost always their respective probabilities decrease as the C value increases. Eventually, as the search progresses, all of the probabilities of finding the current C value in any of the segments decrease to zero. When this happens, the search is terminated because the peak has been found. Now, all of the previously evaluated guess locations need only be compared to determine where the peak of the function occurs and what its value is.

If there are any non-zero segment probabilities after segment values have been recalculated to account for a new maximum, the search continues as before. If the evaluated guess location is not a new maximum, then the C value is not adjusted, no segment

values are recalculated, and the search also continues as before. The peak of the random walk function cannot be determined until all probabilities of finding a function value one greater than the present maximum in any of the segments is equal to zero.

Derivation of search equations. To understand the equation used to calculate the probability of finding the C value at each interval of time within a segment, it is necessary to refer back to the definition of a random walk. First it will be postulated that the random walk begins at the origin of a Cartesian coordinate system. Hence time will begin at zero with a function value of zero,

$$RW(0) = 0, \tag{1}$$

where RW stands for the random walk function. Second, the function at any time t has equal probability of one-half of stepping up one unit, designated $+d$, or stepping down one unit, $-d$, stated as follows

$$P\{RW(t) = RW(t-1) + d\} = \frac{1}{2}, \tag{2}$$

$$P\{RW(t) = RW(t-1) - d\} = \frac{1}{2}. \tag{3}$$

In a random walk to $RW(n) = md$, depicted in Fig. 2, the variable x will represent the number of times the function takes a step up, or the number of $+d$'s. The number of times the function takes a step down is $(n - x)$. The total distance that the function has moved vertically off the X-axis in n steps is md, where m is an integer number. The vertical displacement of the right endpoint is equal to the difference between the number of times the walk takes a step up and the number of times the walk takes a step down

$$m = x - (n - x), \tag{4}$$

$$m = 2x - n. \tag{5}$$

The variable x can then be expressed as

$$x = \frac{n + m}{2}, \tag{6}$$

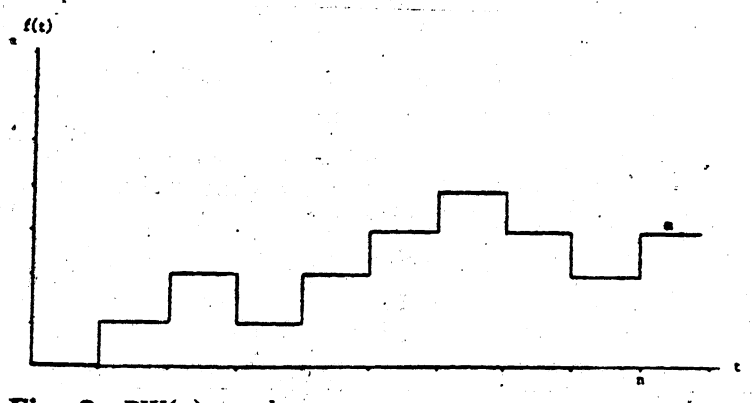

Fig. 2. $RW(n) = md$.

where the quantity $n + m$ must be even in order to represent a valid random walk. The number of possible walks is equal to the number of ways x positive steps can occur in n steps. This can be represented as a combination of n things taken x at a time. The number of walks, N, is equal to

$$N = \binom{n}{x}, \tag{7}$$

which denotes the following mathematical calculation involving the factorial operation

$$\binom{n}{x} = \frac{n!}{x!(n-x)!}. \tag{8}$$

Substituting equation (6) into equation (7) gives

$$N = \begin{cases} \binom{n}{\frac{n+m}{2}} & , \text{ for } (n+m) \text{ even} \\ 0 & , \text{ otherwise.} \end{cases} \tag{9}$$

This result may be extended to find the probability of $RW(n) = td$ given that $RW(i) = md$ where $i > n$, depicted in Fig. 3.

$$P\{RW(n) = td | RW(i) = md\} = \frac{P\{RW(n) = td,\ RW(i) = md\}}{P\{RW(i) = md\}}, \tag{10}$$

$$P\{RW(n) = td | RW(i) = md\} = \frac{N\{RW(n) = td,\ RW(i) = md\}}{N\{RW(i) = md\}}. \tag{11}$$

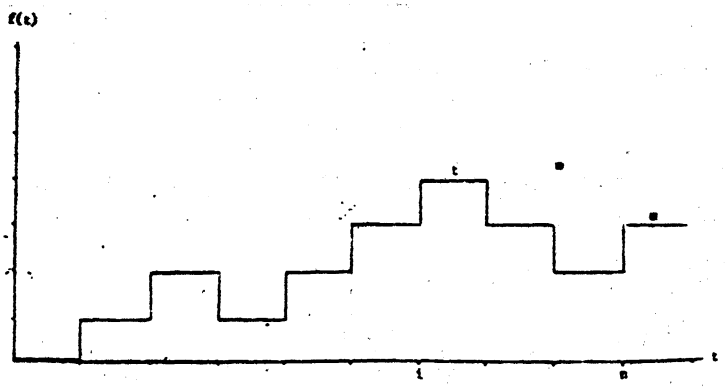

Fig. 3. $\{RW(n) = td\ RW(i) = md\}$.

Making the appropriate variable substitutions into equation (9)

$$N\{RW(i) = md\} = \begin{cases} \binom{i}{\frac{i+m}{2}}, & \text{for } (i+m) \text{ even}, \\ & -i \leqslant m \leqslant i \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

The limits placed on m come from the fact that the height of a random walk can never exceed its length. The numerator of equation (11) is equal to the number of walks to $RW(n) = td$ multiplied by the number of walks from $RW(n) = td$ to $RW(i) = md$, which can be stated as follows

$$N\{RW(n) = td,\ RW(i) = md\} = [N(n,t)][N(i-n,\ m-t)] \tag{13}$$

$$= \begin{cases} \binom{n}{\frac{n+t}{2}} \binom{i-n}{\frac{i-n+m-t}{2}}, & \text{for } (n+t)\ \& \\ & (i-n+m-t) \text{ even} \\ 0, & \text{otherwise,} \end{cases} \tag{14}$$

where the following bounds apply

$$-n \leqslant t \leqslant n, \tag{15}$$

$$-(i-n) \leqslant m-t \leqslant (i-n). \tag{16}$$

Solving for t in equation (16) yields

$$m+n-i \leqslant t \leqslant m+i-n. \tag{17}$$

Combining equations (16) and (17) gives

$$\max\{m+n-i,-n\} \leqslant t \leqslant \min\{m+i-n,n\}. \tag{18}$$

Substituting equations (12) and (14) into equation (11) gives the following result

$$P\{RW(n)=td|RW(i)=md\} = \begin{cases} \dfrac{\binom{n}{\frac{n+t}{2}}\binom{i-n}{\frac{i-n+m-t}{2}}}{\binom{i}{\frac{i+m}{2}}} & \text{for } (n+t),(i+m), \\ & \&(i-n+m-t) \text{ even} \\ 0 & \text{otherwise.} \end{cases} \tag{19}$$

To find the probability of $RW(n)$ greater than sd, perform summation from t equal s to the upper boundary on t as follows

$$P\{RW(n) \geqslant sd|RW(i)=md\} = \sum_{t=s}^{\min\{m+i-n,n\}} P\{RW(n)=td|RW(i)=md\}. \tag{20}$$

To verify that equation (20) works, consider a segment with endpoints zero and five with values of two and three respectively as shown Fig. 4. With only the knowledge of the endpoints, the C value is four, one more than the maximum of three. To find the first guess location of this function, equation (20) will be employed for each interval of time in the function. Remember that equation (20) works on the premise that the random walk begins at the origin. Shifting the function to make this true gives the function depicted in Fig. 5. Now the endpoint values of the function are zero and one and the C value is two. Equation (20) considers all possible walks between the two endpoints in calculating probabilities. All the feasible walks are depicted in Fig. 6, but because they overlap each other, each walk is enumerated in Table 1. The algorithm is looking for the probability of finding a function value of two given the known endpoint parameters. Equation (20) states this as follows

$$P\{RW(n) \geqslant 2|RW(5)=1\}.$$

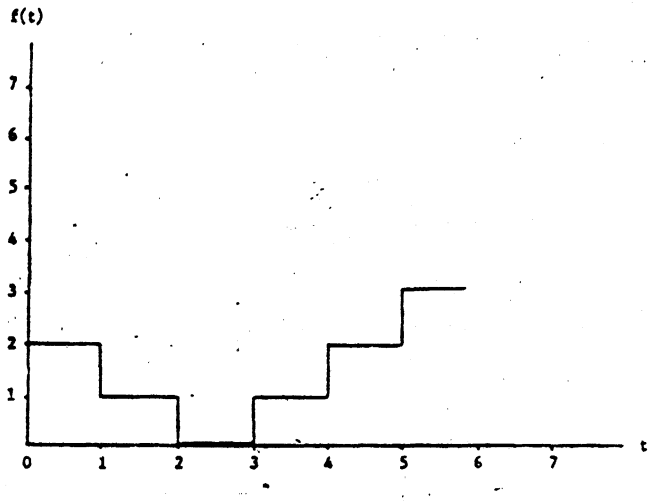

Fig. 4. Example function.

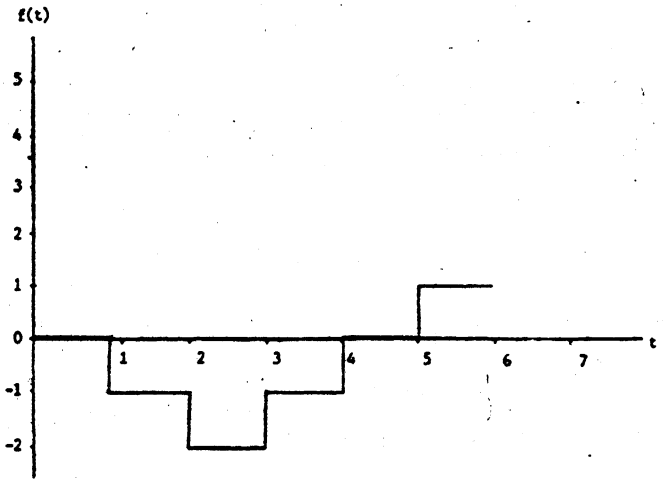

Fig. 5. Example function shifted to origin.

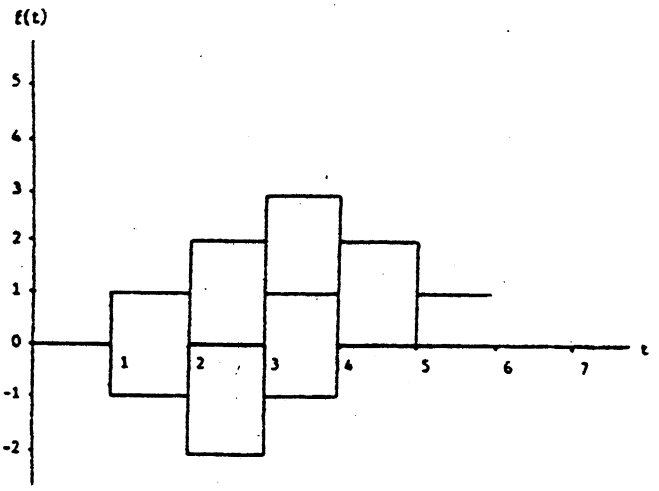

Fig. 6. All possible walks between endpoints of example function.

To calculate the probability of finding the C value at time t equal four, equation (20) yields

$$P\{RW(4) \geqslant 2|RW(5) = 1\} = P\{RW(4) = 2|RW(5) = 1\},$$
$$P\{RW(4) = 2|RW(5) = 1\} = \frac{\binom{4}{3}\binom{1}{0}}{\binom{5}{3}} = \frac{2}{5}.$$

Table 1 verifies this result because it can be seen that four of the ten functions have a value of two at time t equal four. To calculate the probability of finding the C value at time equal three equation (20) yields

$$P\{RW(3) \geqslant 2|RW(5) = 1\} = P\{RW(3) = 3|RW(5) = 1\}.$$

Fig. 6 helps to illustrate why this is true. At time t equal three, there is only one valid walk that has a value greater than or equal to two. It can already be deduced that the probability for this time interval will be one-tenth as equation (20) reveals

$$P\{RW(3) = 3|RW(5) = 1\} = \frac{\binom{3}{3}\binom{2}{0}}{\binom{5}{3}} = \frac{1}{10}.$$

Table 1. Enumeration of all possible walks

$t=0$	$t=1$	$t=2$	$t=3$	$t=4$	$t=5$
0	1	2	3	2	1
0	1	2	1	2	1
0	1	2	1	0	1
0	1	0	1	2	1
0	1	0	1	0	1
0	1	0	-1	0	1
0	-1	0	1	2	1
0	-1	0	1	0	1
0	-1	0	-1	0	1
0	-1	-2	-1	0	1

To find the probability of finding the C value at time t equal two, equation (20) yields

$$P\{RW(2) \geqslant 2 | RW(5) = 1\} = P\{RW(2) = 2 | RW(5) = 1\},$$

$$P\{RW(2) = 2 | RW(5) = 1\} = \frac{\binom{2}{2}\binom{3}{1}}{\binom{5}{2}} = \frac{3}{10}.$$

Table 1 also shows that at time t equal two there are three out of ten walks which have a function value of two. The probability of finding the C value at time t equal one is zero because there are no walks through this point which have a value higher than one. Fig. 6 Table 1 verify this fact. Now that probabilities for all time intervals between the endpoints have been calculated, it can be seen that the highest probability of two-fifths occurs at time t equal four. For this example, time t equal four will become the first guess location of the function. The example function could easily be a segment from another larger function being analyzed and the results would be the same.

Two other equations which are used to calculate some of the final results also need mention. The first one is the equation for the mean value of the percentage ratio. The percentage ratio, denoted as i in the following equations, is a quantity which expresses the

number of evaluated points versus the number of total points and is a measure of how efficient the algorithm is in minimizing the number of function evaluations. The mean of the percentage ratio is calculated as the sample mean. The percentage ratios for each set of functions are totaled and result is then divided by the number of functions in the set.

$$\text{Mean} = \frac{1}{X}\sum_{i=1}^{X} i. \tag{21}$$

Calculating the standard derivation for a set of random walks is slightly more complicated. First, the square of the quantity of the percentage ratio minus the mean is calculated for each function. Then the variance is found by summing up the results and dividing by one less than the total number of functions in the set. The reason division is performed with one less than the total number is to produce unbiased results. Taking the square root of the variance yields the standard deviation.

$$\text{Variance} = \frac{1}{X-1}\sum_{i-1}^{X}(i - \text{Mean})^2, \tag{22}$$

$$\text{Standard Deviation} = \sqrt{\text{Variance}}. \tag{23}$$

Example search. To demonstrate how the algorithm works, it would be helpful at this point to step through one example. A function of length 11, zero through ten steps, was generated as shown in Fig. 7.

The algorithm begins with only the location of the endpoints of the function and their values, and from this knowledge creates the first segment to be used in the search strategy as shown in Fig. 8. The algorithm is searching for a C value of seven, since this is one more than the current maximum of six at the right endpoint. The first run of calculations determines that the best location for finding this C value is at time t equal nine, which becomes the guess location. The probability associated with this guess location is 0.4, or it can be said that there is a 40 percent chance of finding a

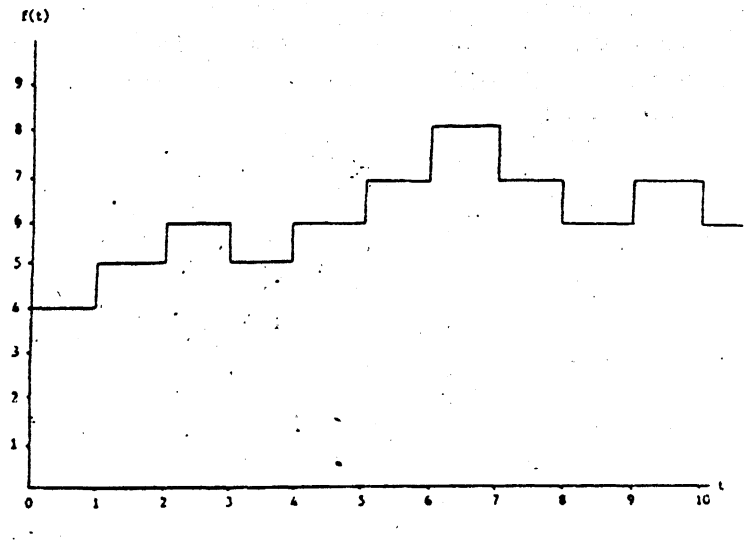

Fig. 7. Example random walk.

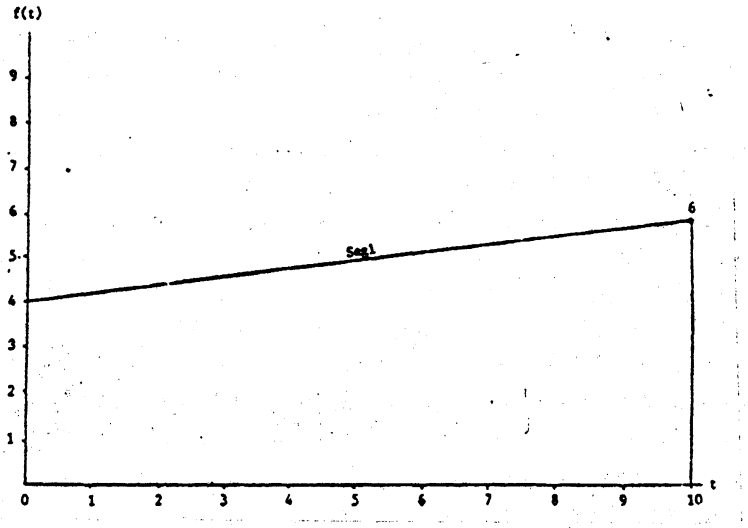

Fig. 8. First segment created from example walk.

value of C or greater at this location. The function is evaluated at this point and found to be seven. Consequently, the first segment depicted in Fig. 8 is divided into two segments about the guess location as shown in Fig. 9.

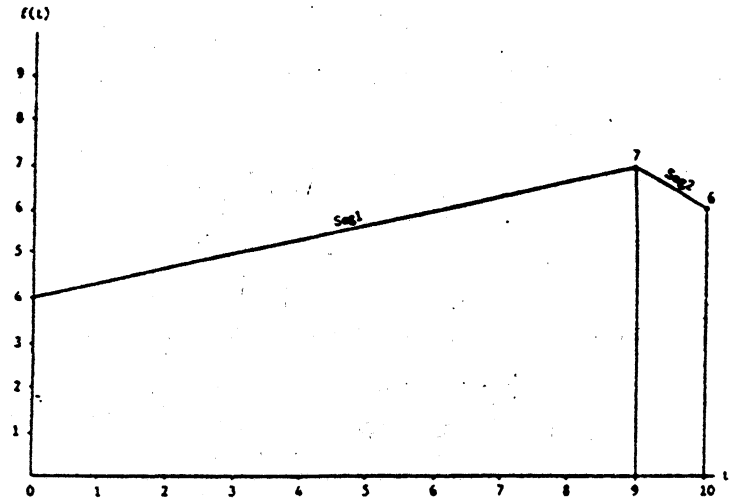

Fig. 9. Example walk divided into two segments.

Now the algorithm is searching for a value of eight or greater, one more than the present maximum. The same calculations are now performed on each segment separately. Segment two, being the shortest possible length, yields a zero probability of finding the C value, but segment one yields a 33 percent chance of finding the C value at time t equal eight. The guess location is evaluated to be six and segment one is divided about this location as shown in Fig. 10. These results depict how segment one is truncated and the newest segment is created from the right portion of the original. This pattern of subdividing segments remains consistent throughout the algorithm and will become more apparent as the search continues. Since the function value of six is not a new maximum, the search continues looking for the highest probability of finding the value of eight or greater in each of the three segments. The three segments are analyzed in the previous manner and segment one yields an 11

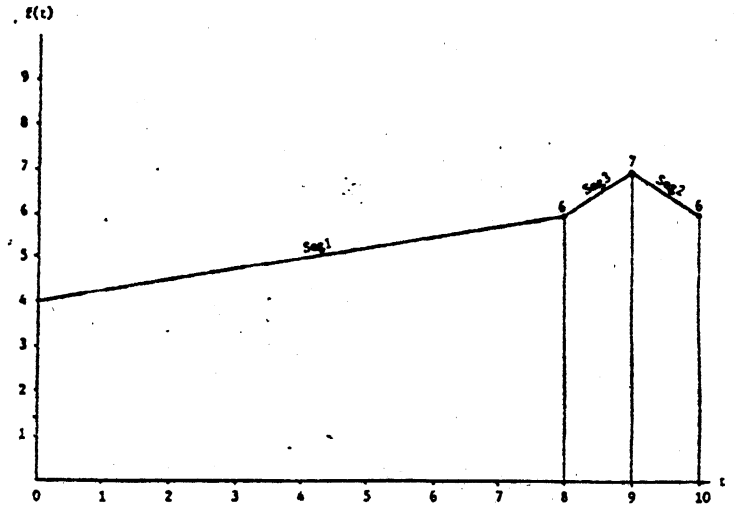

Fig. 10. Example walk divided into three segments.

percent chance of finding the C value at time t equal six, while the other two segments have a zero probability factor. The function is evaluated to be eight and the segment is divided in the usual manner as depicted in Fig. 11. Since the function value of eight constitutes a new maximum, the C value is accordingly increased to nine and all segment values are recalculated based on this change. The recalculation procedure updates the guess locations and associated probabilities of each segment as the function maximum continues to rise. For this particular example, the probabilities recalculated by this process did not change since they had already reached a value of zero. The probabilities for segments one and four were not affected by the recalculation process because they had previously been determined when the segments were first divided. The effects of the recalculation process are more evident for longer walks when more segments of a significant length are involved. As a rule, segment probabilities tend to decrease as the function maximum and C value increase.

After everything has been recalculated, the algorithm continues its search for the peak of the random walk and it is determined that there is a 17 percent chance for finding the a value of nine

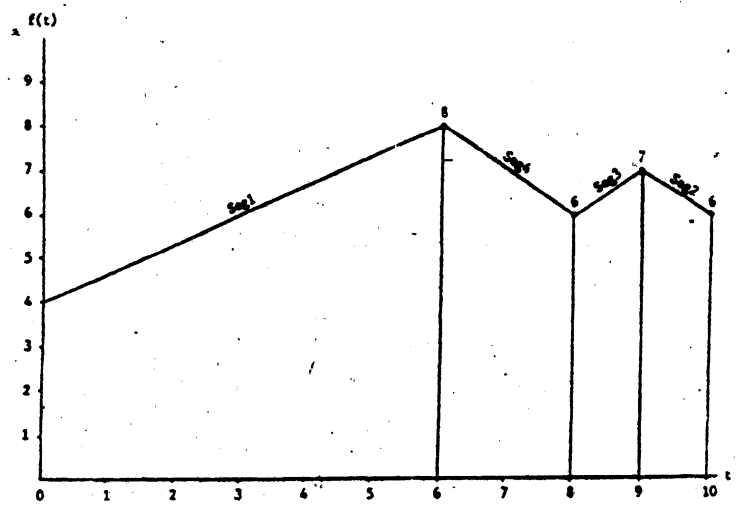

Fig. 11. Example walk divided into four segments.

or greater at time t equal five. All other segment probabilities are zero. The function is evaluated to be seven and segment one is again divided in the familiar manner depicted in Fig. 12. The function value of seven is not a new maximum and so none of the segments need to be re-examined. All five segments are subjected to the same analysis as before and it is determined that all segment probabilities are now zero, meaning that the search is looking for a function maximum in each segment that is unattainable with the given parameters. This zero probability concludes the search and the function peak is revealed to be a value of eight which occurs at time t equal six. Looking at the random walk in Fig. 7 it is evident that the algorithm has indeed successfully converged upon the peak of the function, after only having to evaluate six points in the function to do so. Compared to a point by point search, this algorithm constitutes a 40 percent reduction in the number of evaluated points needed to find the peak.

3. Results. The computer code used to implement the search strategy previously described was written in Turbo Pascal. A pseudocode representation of the program is presented below:

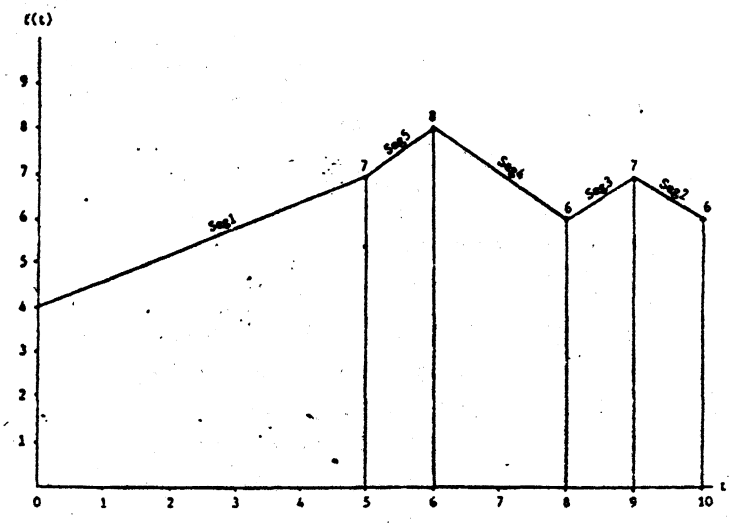

Fig. 12. Example walk divided into five segments.

Major Variables:

Flag	used to signal when a new maximum has occurred
*Flag*1, *Flag*2	used to end search once all segment probabilities have reached zero

Program Flow:

```
Generated random walk
Set Flag to False, Flag1 and Flag2 to True
Read endpoints and values
Determine current maximum and C value
Find guess location and probability
    While (Flag1) do
        Find segment with best guess location
        Evaluate function at guess location
        Set Flag to True if new maximum
        Set Flag1 and Flag2 to False if zero probability
          factor occurs
        If (Flag2) then
            Divide segment about guess location
                If (Flag) then
```

```
                    Update segments with new guess
                      locations and probabilities
                    Set Flag and Flag2 to False if
                      zero probability factor occurs
                Endif
            Endif
            Reset Flag to False
        Endwhile
    Calculate final results
    Display chart of final results
```

Fig. 13 presents a flow chart of the program.

The following results were obtained after running the program five times for five sets of random walks of different lengths. Each separate program run generated and examined ten random walks of equal length. Tables 2 through 6 summarize the results of each run by listing the number of points that had to be evaluated for each function before the peak was found and the ratio of the number of evaluated points versus the total number of points as a percentage. Also, the average number of points evaluated are calculated as are the mean and standard deviation for the percentage ratios.

Compiling the data from all five tables it was discovered that the mean percentage ratio for all fifty functions was 43.19 percent, meaning that on the average the number of points that had to be evaluated using this algorithm was less than half the total number of points involved in a direct comparison method. Also, the corresponding standard deviation for all fifty functions was calculated to be 2.20.

The average percentage ratio and average number of guesses for each set of functions is plotted versus length in Figures 14 and 15, respectively. Both graphs show a nonlinear trend which tends to level off at greater lengths. It is assumed that this trend remains consistent and serves as an accurate predictor for data concerning functions of extensive length.

4. Conclusions. The goal of the algorithm is to converge upon the peak of the random walk function with a minimal number of function evaluations. Although the number of times a par-

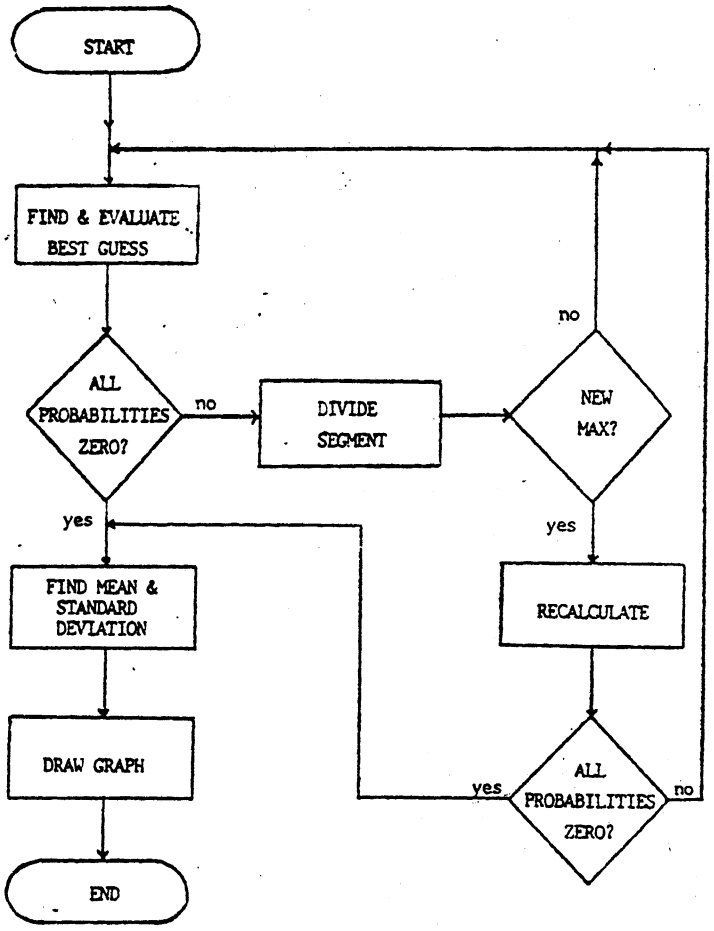

Fig. 13. Program flowchart.

ticular random walk must be evaluated to find its peak depends upon the configuration of that function, the results show that the number of points that had to be evaluated with the algorithm were significantly less than the total number of points in the function. The number of function evaluations depends on both the length and the relative height of the function. Tables 2 through 6 show that as the length of the function increases, so does the number of

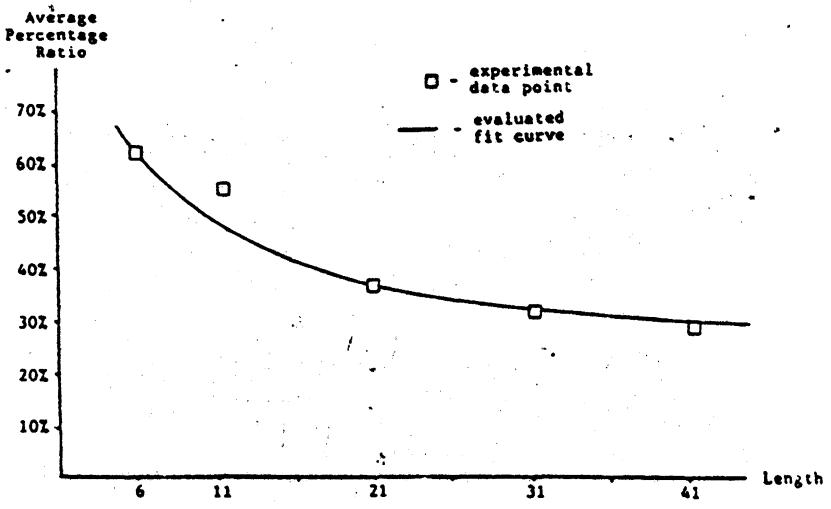

Fig. 14. Average percentage ratio versus length.

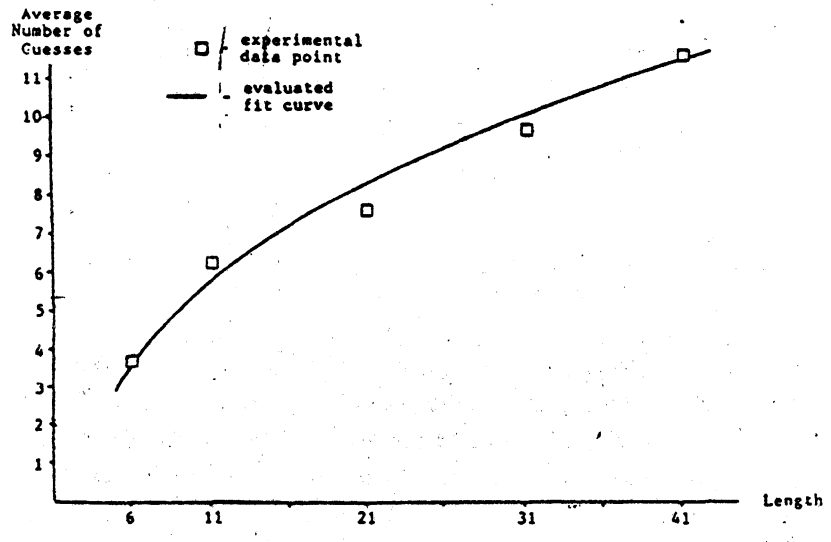

Fig. 15. Average number of guesses versus length.

Table 2. Data for ten random walk functions of length six

Random walk number	Number of guesses to find peak	Percentage ratio (evaluated points / total points)
1	4	66.67%
2	4	66.67%
3	4	66.67%
4	3	50.00%
5	4	66.67%
6	4	66.67%
7	3	50.00%
8	4	66.67%
9	4	66.67%
10	4	66.67%

Average number of guesses is 3.8
Mean is 63.33% Standard deviation is 2.22

Table 3. Data for ten random walk functions of length 11

Random walk number	Number of guesses to find peak	Percentage ratio (evaluated points / Total Points)
1	6	54.55%
2	6	54.55%
3	5	45.45%
4	7	63.64%
5	5	45.45%
6	5	45.45%
7	7	63.64%
8	6	54.55%
9	7	63.64%
10	7	63.64%

Average number of guesses is 6.1
Mean is 55.45% Standard deviation is 2.52

Table 4. Data for ten random walk functions of length 21

Random walk number	Number of guesses to find peak	Percentage ratio (evaluated points / total points)
1	6	28.57%
2	9	42.86%
3	9	42.86%
4	10	47.62%
5	7	33.33%
6	10	47.62%
7	7	33.33%
8	8	38.10%
9	5	23.81%
10	6	28.57%

Average number of guesses is 7.7
Mean is 36.67% Standard deviation is 2.66

Table 5. Data for ten random walk functions of length 31

Random walk number	Number of guesses to find peak	Percentage ratio (evaluated points / total points)
1	11	35.48%
2	8	25.81%
3	10	32.26%
4	9	29.03%
5	8	25.81%
6	11	35.48%
7	14	45.16%
8	8	25.81%
9	11	35.48%
10	9	29.03%

Average number of guesses is 9.9
Mean is 31.94% Standard deviation is 1.9

Table 6. Data for ten random walk functions of length 41

Random walk number	Number of guesses to find peak	Percentage ratio (evaluated points / total points)
1	14	34.15%
2	12	29.27%
3	9	21.95%
4	13	31.71%
5	7	17.07%
6	12	29.27%
7	12	29.27%
8	12	29.27%
9	14	34.14%
10	12	29.27%

Average number of guesses is 11.7
Mean is 28.54% Standard deviation is 1.67

points that have to be evaluated. The number of evaluated points tends to decrease, however, for "steep" functions with a wide swing between maximum and minimum values, and tends to increase for functions with little variation in height. The ratio of the number of evaluated points versus total points decreased as the function lengths increased. This indicates that the algorithm works better with longer functions, remembering that the performance criteria is the minimization of the number of function evaluations.

One result which needs mention is that for functions with multiple peaks of identical values, the algorithm tends to pick one and exclude the others. If one of the peaks happens to be an endpoint, the endpoint will almost always be chosen as the function peak. This does not pose any problem, however, since the location of one function peak is sufficient to meet the goal of the search strategy. The other peaks do not usually figure into the results because the algorithm never searches for values equal to the current maximum, but always a value that exceeds the maximum by one.

5. Areas of further study. The global optimization algorithm for a random walk is a beneficial tool for finding the peak value with minimal function evaluations. However, the random walk function itself is somewhat limited in the ability to model real world experiences. The qualities which define a random walk ultimately restrict its usefulness in practical applications. A step function characterized by a greater degree of versatility would be more adaptable to real world situations. Such an enhanced step function should not be constrained to stepping only one unit at a time but should instead be able to jump any number of units at any instant of time. A logical consequence of this would be to also allow the function to remain constant for an unlimited period of time.

Taking into account the function modifications, areas of further study would parallel the steps taken to produce the results described in this thesis. Before an algorithm could be developed to find the peak of this new function, the derivation of a mathematical equation like that of equation (20) would be necessary to calculate the probabilities associated with finding a specific search value. Then an implementation of this equation in a similar search strategy like the one presented here would be next. Finally, converting the algorithm into computer code to develop a workable program would be the last step. The step function created from the amended random walk would be an improved model for simulating authentic situations and research into this area would undoubtedly be a worthwhile endeavor.

REFERENCES

Easom, E.E. (1990). *A Survey of Global Optimization Techniques*, University of Louisville, MEng. Thesis.

Groch, A., L.M.Vidigal and S.W.Director (1985). A new global optimization method electronic circuit design. *IEEE Transactions on Circuits and Systems*, Vol.CAS–32,No.2.

Lawler, E.L. (1976). *Combinatorial Optimization: Networks and Matroids*, Holt,

Rinehart, and Winston, New York.

McKeown, J.J., and A.Nag (1974). *An application of optimization techniques to the design of an optical filter.* Presented at the IMA Conf. England, Bristol.

Mockus, J. (1989). *Bayesian Approach to Global Optimization,* Kluwer Academic Publichers. Dordrecht.

Stuckman, B.E., and J.C.Hill (1986). Pattern steering in HF phased array antennas. In *Proceedings of the 29th Midwest Symposium on Circuits and Systems.*

Stuckman, B.E., and N.W.Laursen (1987). Modern control design using global search. In *Proceedings of the 8th Meeting Coordinating Group on Modern Control Theory,* U.S. Army.

Received August 1991

M. Herndon received his B.S. degree and his M.Eng. Degree in Electrical Engineering from the University of Louisville in 1989 and 19991 respectively. He is currently a configuration management engineer for Johnson Controls of Cape Canaveral, Florida. U.S.A. Mr. Herndon's research interset include engineering management science and numerical optimization.

C. Perttunen received the B.S. and M.S. degrees in electrical engineering from Oakland University, Rochester, MI, in 1985 and 1987, respectively. He received the M.S. degree in applied statistics from Oakland University in 1989, and the Ph.D. degree in computer science and engineering from the University of Lousville in 1990. Dr. Perttunen received the 1991 John M. Houchens Prize for the outstanding doctoral dissertation over all academic disciplines at the University of Louisville. He is currently a visiting assistant professor in the Department of Engineering Math and Computer Science at the University of Louisville. Dr. Perttunen has authored over 30 papers in the areas of global optimization algorithms, instrumentation and measurements, system modeling, statistical quality control, and robotics.

B. Stuckman received his B.S. Degree in Engineering, his M.S. degree in Electrical and Computer Engineering and his Ph.D. Degree in Systems Engineering from Oakland University in 1981,

1983, and 1987, respectively. He is currently an Assistant Professor of Electrical Engineering at the University of Louisville and Director of Electrical and Systems Engineering for the OASYS Research Group, a private consulting company. Dr. Stuckman is a member of Tau Beta Pi, Eta Kappa Nu, Sigma Xi, ASNE and a Senior Member of IEEE. He has received the University of Louisville President's Young Investigator Award. His research interest include numerical optimization, simulation and modeling, telecommunication and communication systems, instrumentation and measurement error, and technology law and policy.