Proceedings of the 1996 ACM/SIGAPP Symposium on Applied Computing, February 18-20, 1996, Philadelphia, PA., Pages 258-262, ACM Press.

A GENETIC ALGORITHM FOR THE MINIMUM BROADCAST TIME PROBLEM USING A GLOBAL PRECEDENCE VECTOR *

Cory J. Hoelting, Dale A. Schoenefeld and **Roger L. Wainwright**

Department of Mathematical and Computer Sciences
The University of Tulsa
600 South College Avenue
Tulsa, OK 74104-3189
http://www.utulsa.edu
[hoeltc,dschoen,rogerw]@euler.mcs.utulsa.edu

Key Words: Genetic Algorithms, Heuristic Techniques, Local Broadcasting, Minimum Broadcast Time

ABSTRACT

The problem of broadcasting a message through a network is considered. The objective is to minimize the number of time steps necessary to complete the broadcast. This problem is known as the Minimum Broadcast Time Problem or the Local Broadcasting Problem. Finding an optimal broadcast using a local broadcasting scheme is known to be NP-Complete. A genetic algorithm (GA) is used as a heuristic technique to find near optimal solutions to this problem. The GA is compared to a variant of a recent heuristic technique presented in the literature.

INTRODUCTION

The problem considered in this research is the Minimum Broadcast Time problem (MBT). This problem is formally presented by Garey and Johnson in [3, page 219]. An equivalent statement of the problem follows: Given an undirected graph, without self-loops or multiple edges, $G = (V,E)$, a subset $V_0 \subseteq V$, and $k \in Z^+$, the objective is to "broadcast" a message throughout the graph in k time steps. Broadcasting a message is carried out as follows:

1. The message is present at all of the vertices in the set V_0 at time zero.
2. We must find a sequence $V_0, E_1, V_1, E_2, ..., E_k, V_k$, such that $V_i \subseteq V, E_i \subseteq E$, and $V_k = V$, for $1 \leq i \leq k$, subject to the conditions:
   (a) Every edge in E_i has one endpoint in V_{i-1} and the other in $V - V_i$.
   (b) No two edges in E_i have an endpoint in common, and
   (c) $V_i = V_{i-1} \cup \{v | \{u, v\} \in E_i\}$.

Therefore, the sequence given in (2) above represents the broadcast where V_{i-1} contains the vertices with the message at time *i-1*, E_i represents the transmission of the message during the time step between time *i-1* and time i, and V_i contains the vertices that have a copy of the message at time i. Note that during each time step a vertex that has the message can send it to only one other adjacent vertex that does not already have the message. However, during the next time step all of the vertices that have a copy of the message can re-transmit the message.

A thorough discussion of broadcasting and related topics is given by Hedetniemi *et al.* [4]. They present many applications of broadcasting, some are specific to computer networks and others deal with general communication in groups. The use of broadcasting in the context of distributed file systems and distributed database systems is found in Scheuermann *et al.* [7].

Several interesting variations of the MBT are contained in the literature. One such variation is the multi-destination broadcast, also known as a multicast. In this

*Research partially supported by OCAST Grant AR2-004 and Sun Microsystems, Inc.

variant of the MBT problem the objective is to broadcast the message to a subset of V instead of broadcasting to all of V. Note that if the base set consists of just one vertex, this is equivalent to the point-to-multipoint routing problem. The multi-destination broadcasting problem was proposed by Dalal and Metcalfe [2]. In the MBT problem a message can only be sent to an adjacent vertex. This type of scheme is called local broadcasting. Another variant is the line broadcasting problem where a message is sent during one time step along a path in the network, instead of just along a single edge. Two references describing this variation are [5] and [6].

DETERMINISTIC STRATEGIES

Scheuermann and Wu developed an exact algorithm for the Minimum Broadcast Time problem [7]. This algorithm is based on dynamic programming and uses several facts concerning the problem to prune the state space tree. However, this algorithm becomes inefficient for large networks.

Scheuermann and Wu [7] also developed several heuristic algorithms for the MBT problem. One algorithm is based on a series of Least-Weight Maximum Matchings in a series of induced bipartite graphs. A bipartite graph is induced on a network at every time step in the broadcast. One set of vertices, denoted by S, has a copy of the message, and the complement set, denoted by R, does not have a copy of the message. The edges in the bipartite graph are edges from the original graph that connect elements of S and R, that is $E' \subseteq E$ where $E' = \{(u,v) | u \in S \wedge v \in R \wedge (u,v) \in E\}$. This bipartite graph is denoted as $G_S = [(S,R), E']$. The weights on the edges in E' can be determined in several ways. The technique that performed the best for Scheuermann and Wu assigns each edge $(u,v) \in E'$ a weight equal to $-d_R(v)$, where $v \in R$. The function $d_R(\bullet)$ is called the *d*-value of a vertex. For each vertex without a copy of the message, say v, $d_R(v)$ is the degree of v in the subgraph induced by R. Another scheme employed by Scheuermann and Wu is to replace the Least-Weight Maximum Matching algorithm (LWMM) with an Approximate Matching (AM) algorithm.

The tests conducted by Scheuermann and Wu [7] were on networks where $|V|$ was 15 and 20. In both cases the Approximate Matching algorithm runs in less than half the time of the Least-Weight Maximum Matching algorithm for the edge weighting technique described above. From now on this edge weighting technique will be referred to as the degree weighting technique. With 15 vertices the LWMM algorithm significantly outperforms the AM algorithm. However, in the case with 20 vertices, the AM performance is very near that of the LWMM. This seems to suggest that for the degree weighting technique the running time of the AM algorithm will be significantly less than that of the LWMM algorithm, and the performance of the AM will approach that of the LWMM as the size of the network increases.

We present the Approximate Matching heuristic algorithm found in [7]. The testing in this research used a variant of this algorithm that sorted R based on the degree of the vertices rather than on the d-values of the vertices. We found this easier to implement.

AM HEURISTIC

S – set of vertices that have the message at the beginning of the time step. (equivalent to V_{i-1} at time step i)

R – set of vertices that do not have the message at the beginning of the time step. (equivalent to $V - V_{i-1}$ at time step i)

Apply the following procedure at each time step:

1. Sort R in descending order of d-value.
2. Sort S in ascending order of vertex degree.
3. Find the first r in R that has not been used in the time step.
4. If every element of R has been used, then stop.
5. Find the first s in S that has not been used in the time step and is adjacent to r.
6. If no such s is found, then mark r as used in the time step and goto Step 3.
7. Otherwise, s transmits message to r.
8. Mark s and r as used in the time step and goto Step 3.

In Step 1, R is sorted in descending order because sending messages to higher degree vertices first increases the chance that more vertices will be able to transmit a copy of the message in the next time step. In Step 2, S is sorted in ascending order because sending from low degree vertices first increases the chance that more vertices will be able to transmit a copy of the message during the current time step.

GA ENCODING

The encoding used for the Minimum Broadcast Time problem is a permutation of the integers from 1 to $|V|$. The decoding of a chromosome splits the chromosome into two lists V_i and $V - V_i$ at each time step. Next, the order of the list of vertices in $V - V_i$ is reversed. This reversal is done for the same reason R is sorted in descending order in the AM heuristic. That is, otherwise vertices with larger degrees or d-values will tend toward the end of the list of vertices. After reversing the order, we tend to send the message to a vertex with a large degree or d-value, thus increasing the chance that this new vertex will be able to re-transmit the message in the next time step. Vertices with larger degree or d-value tend toward the end of the list in the GA because of the crossover that is used, and they are forced to the end of the list by sorting in the AM.

Next, the first vertex in the list of vertices in V_i will send the message to the first vertex in the reversed list of vertices from $V - V_i$ that it is adjacent to. At this point, these two vertices will be removed from consideration for the remainder of the time step. This process continues until no more copies of the message can be sent during the present time step. At this point the chromosome will be split again into the two lists, V_{i+1} and $V - V_{i+1}$, and the transmission of the message for the next time step will be determined by the same procedure. This process will continue until all the vertices have a copy of the message. The fitness of the chromosome is the number of time steps required to complete the broadcast.

EXAMPLE DECODINGS

Given the undirected graph $G = (V, E)$ where $E = \{(1,2),(2,4),(2,6),(3,4),(3,5),(4,5),(5,6)\}$, and $V = \{1,2,3,4,5,6\}$ in Figure 1, the decoding of two example chromosomes that represent broadcasts in this graph are illustrated below.

Consider chromosome 1 as [1 5 3 6 4 2] with $V_0 = [3]$. Thus, it follows that $V - V_0 = [1, 5, 6, 4, 2]$. First, the order of $V - V_0$ is reversed to obtain [2, 4, 6, 5, 1]. The reason for reversing the order of $V - V_0$ was explained in the previous section. We intentionally refrained from representing $V - V_i$ and V_i as sets because the order of the vertices is important. At each step in the decoding process, the vertices in V_i are scanned left to right. For each vertex in V_i, the vertices in $V - V_i$ are scanned left to right searching for the first vertex that a message can be broadcast to. Each vertex in $V - V_i$ that is able to receive a broadcast during the time step is removed from $V - V_i$ and placed into V_i. Note that vertices are placed in V_i in the same relative order, left to right that they appear in the original chromosome. For example, the decoding of chromosome 1 is as follows: during time step 1 the message is sent from vertex 3 to vertex 4, because vertex 4 is the first vertex in the reversed list that is adjacent to vertex 3. So V_1 becomes [3, 4] and $V - V_1$ becomes [2, 6, 5, 1]. Following this process, during time step 2 the message is sent from vertex 3 to vertex 5 and from vertex 4 to vertex 2. This makes $V_2 = [5, 3, 4, 2]$

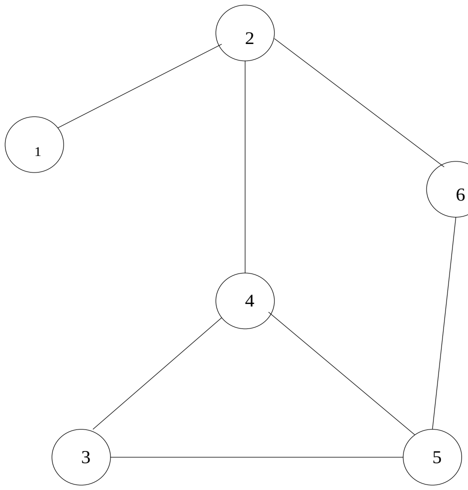

Figure 1: An Example Undirected Graph

and $V - V_2 = [6, 1]$. In time step 3 the message is sent from vertex 5 to vertex 6 and from vertex 2 to vertex 1. At this point $V_3 = [3, 4, 5, 2, 6, 1]$ and $V - V_3 = \phi$. The fitness of this chromosome is 3, the number of required time steps.

Consider [4 6 5 3 1 2] as a second example chromosome with $V_0 = [3]$. The reader is encouraged to verify that at time step 1: vertex 3 sends a message to vertex 5, at time step 2: vertex 5 sends to vertex 6 and vertex 3 sends to vertex 4, at time step 3: vertex 4 sends to vertex 2, and during time step 4: vertex 2 sends to vertex 1. This gives chromosome 2 a fitness of 4.

GENETIC OPERATORS

The crossover operator used in this research is MX1, a member of the MX family of crossover operators introduced by Blanton and Wainwright [1]. Associated with a problem instance is a global precedence vector (GPV) which is used during the application of the MX1 operator. In our case, the global precedence vector consists of the vertices in increasing order of degree. From Figure 1 it is clear that deg(1) = 1, deg(2) = 3, deg(3) = 2, deg(4) = 3, deg(5) = 3, and deg(6) = 2, where deg(y) is the degree of vertex y. Using this information the global precedence vector (GPV) for this problem is GPV = (1, 3, 6, 2, 4, 5).

To illustrate the MX1 crossover operator applied to the MBT problem, consider the two chromosomes in the previous section as the selected parents. The "*" indicates the allele currently under consideration.

```
                        *
Step 1.     Parent 1:   1 5 3 6 4 2
            Parent 2:   4 6 5 3 1 2
```

```
    Consider the first allele in each parent.
    1 comes before 4 in the GPV so 1 is
    selected.  Parent 2 is fixed by exchanging
    alleles 1 and 4.  Now consider the second
    allele.

                                   *
 Step 2.      Parent 1:  1 5 3 6 4 2
              Parent 2:  1 6 5 3 4 2
              Child:     1 x x x x x

    6 comes before 5 in the GPV so 6 is
    selected.  Parent 1 is fixed by
    exchanging alleles 6 and 5.
    Now consider the third allele.

                                     *
 Step 3.      Parent 1:  1 6 3 5 4 2
              Parent 2:  1 6 5 3 4 2
              Child:     1 6 x x x x

    3 is before 5 in the GPV so 3 is
    selected.  Parent 2 is fixed by
    exchanging alleles 3 and 5.
    Consider the next allele.

                                       *
 Step 4:      Parent 1:  1 6 3 5 4 2
              Parent 2:  1 6 3 5 4 2
              Child:     1 6 3 x x x

Now the parents are identical, so the
child is uniquely determined:

              Child:     1 6 3 5 4 2.
```

The mutation operator is swap (i.e. two allele values are exchanged). Note that the global precedence vector favors vertices with lower degree. Hence, there is a tendency to force the lower degree vertices toward the front of the chromosome and the higher degree vertices toward the end of the chromosome. During the decoding evaluation, it is clearly desirable to transmit to higher degree vertices first. As stated before, this is the reason why the list of vertices without a copy of the message is reversed at the beginning of each step.

RESULTS

Assuming that $|V_0| = 1$ and that $|V| = n$, several interesting facts about the broadcast time can be derived. The objective function maps a broadcast time to a value in set $A = \{\lceil log_2 n \rceil, ..., n-1\}$. Note that $n-1$ is an upper bound on the broadcast time. That is, for any graph and any initial vertex with the message, there is at least one broadcast that takes $n-1$ time steps. However, unless the edges of the graph are sparse this is probably not the optimal value. If $n-1$ is not the optimal value, it is unlikely that either the AM or the GA will produce a broadcast with fitness of $n-1$. This is due to the greedy nature of both the AM heuristic and the chromosome decoding scheme in the GA.

The smallest element of A, $\lceil log_2 n \rceil$, is a theoretical lower bound. It is a lower bound, but not always an attainable value. Thus the optimal value will be greater than or equal to $\lceil log_2 n \rceil$, and the broadcast time given by the AM heuristic and the GA will be greater than or equal to the optimal and less than or equal to $n-1$. These bounds are very useful in the interpretation of the test results.

A first set of tests was run on randomly generated graphs containing 10 to 500 vertices with low edge connectivity probabilities. The results of these tests are summarized in Table 1.

Network Size	Edge Probability	Broadcast Time	
		AM	GA
10	0.275	5	4*
10	0.25	5	5
20	0.2	5	5
20	0.17	9	7*
30	0.15	6	6
40	0.15	6	6
100	0.1	7	7
100	0.2	7	7
100	0.3	7	7
250	0.1	9	9
250	0.2	8	8
250	0.3	9	8*
500	0.1	10	9*
500	0.2	9	9
500	0.3	9	9

Table 1: AM and GA Results for Random Networks with Low Edge Probabilities

In Table 1, we see that the GA gave the same or better results in every case tested. The GA outperformed the AM in four cases indicated by an "*" in Table 1. The maximum fitness value of the initial pool of chromosomes was never more than the minimum value plus 3. Most of the time the maximum fitness was 1 or 2 greater than the minimum value. Due to the fact that several permutations have the same fitness value and the range of the fitness values present at any generation was small, the GA converged rapidly, even with mutation rates as high as 0.1.

In our second set of tests, we contrived a set of networks of size 40, 80 and 160. The characteristic of these networks is that there are a large number of highly connected clusters of nodes that are in turn connected

Network Size	Broadcast Time		
	AM	GA	Optimal
40	14	8	8
80	15	9	9
160	23	13	11

Table 2: AM and GA Results for a Contrived Set of Networks

by only one or two edges to each of the other clusters. This is similar to a network layout for a company that may have several business centers around the world connected by a low connectivity backbone. One side effect of this type of network is that the AM heuristic gives very poor solutions.

In Table 2, we see that indeed the AM heuristic yields extremely poor results, while the GA produces excellent solutions. The networks were contrived, hence the optimal broadcast time is known and presented in the table for comparison.

CONCLUSION AND FUTURE WORK

From our results, it seems that the AM performs fairly well in most situations. On occasion, however, the AM gives poor results, such as in our contrived networks presented in Table 2. The rapid evolutionary behavior of the GA encoding presented in this paper is most likely attributable to the many-to-one behavior of the mapping from permutations to fitness values and the limited range of this mapping. However, this did not deter the GA from producing excellent results for randomly generated networks, as well as our contrived networks. The obvious precedence of the nodes (ordered by degree) allowed us to construct a global precedence vector. This in turn allowed us to use the MX crossover operators (which require a precedence vector) to drive the GA to optimal or near optimal solutions. To the authors' knowledge, this is the first implementation of the MBT problem using genetic algorithms, and the results are excellent.

One variant of the MBT problem that could be considered for future work is the introduction of more than one message into the problem. That is, start with two or more messages and broadcast until every vertex has a copy of all the messages. A second variation of the MBT that may be investigated in the future is the multicast or point-to-multipoint problem. In this formulation the fitness function could be either the number of time steps or the sum of the weights of the edges used in the multicast. The multicast problem with edge weight fitness should not be hampered by the problems encountered in this research that were caused by the limited range of the fitness values and the many-to-one nature of the objective function. Although we have not yet implemented a GA solution to these variations of the MBT problem, we suspect the GA will perform very well.

Acknowledgements

This research has been supported by OCAST Grant AR2-004. The authors also wish to acknowledge the support of Sun Microsystems, Inc.

References

[1] J. L. Blanton Jr., R. L. Wainwright. "Multiple Vehicle Routing with Time and Capacity Constraints using Genetic Algorithms", *Proceedings of the Fifth International Conference on Genetic Algorithms ICGA'93*, 1993, pp. 452-459.

[2] Y. K. Dalal and R. M. Metcalfe. "Reverse Path Forwarding of Broadcast Packets", *Communications of the ACM*, Vol. 21, No. 12, p.1040-1048.

[3] M. Garey and D. Johnson. "Computers and Intractability: a Guide to the Theory of NP-Completeness", W. H. Freeman, San Francisco, 1979.

[4] S. M. Hedetniemi, S. T. Hedetniemi, and A. L. Liestman. "A Survey of Gossiping and Broadcasting in Communication Networks", *Networks*, Vol. 18, p.319-349, 1988.

[5] J. O. Kane and J. G. Peters. "Line Broadcasting in Cycles", Technical Report, Simon Fraser University.

[6] D. B. Peters and J. G. Peters, "Bounded Depth Broadcasting", Technical Report, Simon Fraser University.

[7] P. Scheuermann and G. Wu, "Heuristic Algorithms for Broadcasting in Point-to-Point Computer Networks", *IEEE Trans. on Computers*, Vol. C-33, No. 9, p.804-811.