A CLUSTER-MERGE ALGORITHM FOR SOLVING THE MINIMUM POWER BROADCAST PROBLEM IN LARGE SCALE WIRELESS NETWORKS

Arindam K. Das, Robert J. Marks, Mohamed El-Sharkawi
Department of Electrical Engineering
University of Washington
Box 352500, Seattle, WA 98195.
{arindam,marks,melshark}@ee.washington.edu
and
Payman Arabshahi, Andrew Gray
Jet Propulsion Laboratory
4800 Oak Grove Drive, MS 238-343
Pasadena, CA 91109.
{payman,gray}@arcadia.jpl.nasa.gov

ABSTRACT

In this paper, we address the minimum power broadcast problem in wireless networks. Assuming nodes are equipped with omni-directional antennas, the inherently broadcast nature of wireless networks can be exploited to compute power efficient routing trees. We propose a 2-stage cluster-merge algorithm for computing minimum power broadcast trees. The cluster phase is a look-ahead variant of the Broadcast Incremental Power algorithm [1] and the merge phase is a probabilistic positive reinforcement search procedure, as used in swarm intelligence algorithms. A local tree-improvement procedure is incorporated as an optional step in the merge phase to boost the performance of the algorithm. A key advantage of such a cluster based approach is significant reduction in time complexity. Simulations show that the algorithm is able to generate high quality solutions in relatively little computational time.

I. INTRODUCTION

Broadcasting/multicasting in wireless networks is fundamentally different than wired networks, since multiple nodes can be reached by a single transmission. This, of course, assumes that nodes are equipped with omnidirectional antennas, so that if a transmission is directed from node i to node j, all nodes which are nearer to i than j will also receive the transmission. This is the "wireless multicast advantage" [1] property. For a given network with an identified source node, the minimum power broadcast (MPB) problem in wireless networks is to find a route to all other nodes, such that the overall transmission power is minimized.

Wieselthier *et al* [1] first noted that a "node based" approach is needed for solving the MPB problem in wireless environments. The *Broadcast Incremental Power* (BIP) algorithm discussed in [1] is a node-based minimum-cost tree algorithm. Other techniques for solving this problem are discussed in [2], [3, [5] and [6].

In this paper, we describe a 2-stage $cluster - merge$ (CM) algorithm for solving the MPB problem in large scale wireless networks. A key aspect of the clustering algorithm is that each cluster represents a partial connected subtree, with respect to the overall minimum power broadcast tree. Accordingly, if N nodes are divided into L clusters, a spanning tree can be obtained by adding at most $L-1$ links for connecting the clusterheads. This leads to a significant reduction in complexity and makes the algorithm particularly suitable for use in large scale dense networks, when typically $L << N$. The *merge* phase of the algorithm utilizes positive reinforcement search strategies adopted in *swarm intelligence* paradigms [4].

II. NETWORK MODEL

We assume a fixed N-node network with a specified source node which has to broadcast a message to all other nodes in the network. Any node can be used as a relay node to reach other nodes in the network. All nodes are assumed to have omni-directional antennas, so that if node i transmits to node j, all nodes closer to i than j will also receive the transmission. The power matrix of the network, $\mathbf{P}$, is an $N \times N$ symmetric matrix whose $(i,j)th$ element represents the power required for node i to transmit to node j and is given by:

$$\mathbf{P}_{ij} = \left[(x_i - x_j)^2 + (y_i - y_j)^2\right]^{\alpha/2} = d_{ij}^{\alpha}, \quad i \neq j \qquad (1)$$

0-7803-8140-8/03/$17.00 (C) 2003 IEEE

where $\{(x_i, y_i) : 1 \leq i \leq N\}$ are the coordinates of the nodes in the network, α $(2 \leq \alpha \leq 4)$ is the channel loss exponent and d_{ij} is the Euclidean distance between nodes i and j. We assume that there is no constraint on maximum transmitter power. However, the algorithm we discuss in this paper can be extended straightforwardly to the case where this assumption does not hold, by redefining the power matrix such that its $(i,j)th$ element is ∞ if d_{ij}^{α} is greater than the maximum power limit of node i.

We consider a centralized implementation where construction of the routing tree is done at the source node, which has complete knowledge of the locations of all nodes in the network. Finally, we assume that power expenditures due to signal reception and processing are negligible compared to signal transmission and hence the *cost* of a routing tree is equal to the sum of transmitter powers corresponding to the set of edges chosen in the tree.

III. THE CLUSTER PHASE

We first establish the following notation.

$\mathcal{N}$	= set of all nodes in the network, $N = \lvert\mathcal{N}\rvert$
$\mathcal{D}$	= set of destination nodes, $D = \lvert\mathcal{D}\rvert$
$\mathcal{E}$	= set of all edges in the network $\triangleq \{(i \rightarrow j) : 1 \leq i \neq j \leq N\}$
k	= iteration number
$\mathcal{C}_l^k$	= subtree in cluster l till iteration k
$chd(\mathcal{C}_l^k)$	= clusterhead of subtree in cluster l
Y_i^k	= power level of node i after iteration k
$\mathcal{NR}^k$	= all nodes reached till iteration k
$\mathcal{NNR}^k$	= nodes not reached after iteration k

The clustering mechanism we employ is a *look−ahead* variant of the BIP algorithm, with *backtracking*. For $k = 1$, the edge[1] (source $\rightarrow$ node closest to source) is assigned to the first cluster; *i.e.*, $\mathcal{C}_1^1 = \{(\text{source, node closest to source})\}$. The sets $\mathcal{NR}^k$ and $\mathcal{NNR}^k$ are then updated as:

- $\mathcal{NR}^k = \{\text{source, node closest to source}\}$
- $\mathcal{NR}^k = \mathcal{N} \setminus \mathcal{NR}^k$

For $k \geq 2$, the following sequence of steps is followed:

1. Prepare a set of possible edges to choose from.

$$edge\_list^k = \{(i,j) : i \in \mathcal{NR}^{k-1}, j \in \mathcal{NNR}^{k-1}\} \quad (2)$$

2. Determine the kth edge from $edge\_list^k$ using the minimum incremental cost criterion of the BIP algorithm. Suppose the edge chosen is (p,q), with an associated incremental cost being $\mathbf{P}_{pq} - Y_p^{k-1}$ (see eqn. 1 for definition of the power matrix, $\mathbf{P}$). This is the "provisional incremental cost of maintaining tree-connectivity at node q". Assume that node p is in cluster l_1.

3. Update the parameters:

$$\mathcal{NR}^k = \mathcal{NR}^{k-1} \cup q, \;\; \mathcal{NNR}^k = \mathcal{N} \setminus \mathcal{NR}^k \quad (3)$$

4. Determine the node r, $r \in \mathcal{NNR}^k$, such that the cost of spanning in r from q is the smallest (this step represents the $look - ahead$ variation on the BIP algorithm); *i.e.*, $\mathbf{P}_{qr} < \mathbf{P}_{qs} : \{r,s\} \in \mathcal{NNR}^k, r \neq s$. Note that $\mathbf{P}_{qr}$ is the cost that would be incurred if the connectivity requirement is dropped and the BIP algorithm is restarted from node q on the destination set $\mathcal{NNR}^k$.

5. **If** $\mathcal{NNR}^k \neq \emptyset$ (*i.e.*, if r is not the only unreached node) and $\mathbf{P}_{qr} < \mathbf{P}_{pq} - Y_p^{k-1}$, a new cluster is initialized, node q is assigned to be its clusterhead [2] and node r is spanned in from q. This newly generated cluster is left disconnected, *i.e.*, the edge (p,q) (recall from item 2 above that this edge was determined using the incremental cost criterion) is not included in the routing tree. Essentially, therefore, the BIP algorithm is restarted from node q (on $\mathcal{NNR}^k$) if the provisional cost of maintaining connectivity at that node is greater than the cost that would be incurred by restarting it. Parameters are then further updated as follows:

$$\mathcal{NR}^k = \mathcal{NR}^k \cup r, \;\; \mathcal{NNR}^k = \mathcal{N} \setminus \mathcal{NR}^k \quad (4)$$

$$Y_i^k = \begin{cases} \mathbf{P}_{qr}, & \text{if } i = q \\ Y_i^{k-1}, & \text{otherwise} \end{cases} \quad (5)$$

Else, add the edge (p,q) to cluster l_1 and update the node power vector:

$$Y_i^k = \begin{cases} \mathbf{P}_{pq}, & \text{if } i = p \\ Y_i^{k-1}, & \text{otherwise} \end{cases} \quad (6)$$

6. **If** $\mathcal{NNR}^k = \emptyset$, stop. **Else**, increment $k = k + 1$ and proceed to the next iteration.

To understand how backtracking works, suppose the root of cluster l_1 $(l_1 > 1)$, $chd\left(\mathcal{C}_{l_1}^k\right)$, is disconnected and the current edge chosen is (q,r), where q is in cluster l_2, $l_2 > l_1$. If this transmission also reaches the node $chd(\mathcal{C}_{l_1}^k)$, clusters l_1 and cluster l_2 can be merged, thereby reducing the number of clusters by one. The backtracking phase is not necessary if there are only two clusters since the source, which is always the root of the first cluster, is not required to be reached.

Note that the number of clusters is not preset, it is determined by the algorithm. The maximum number of clusters that can be generated is $\lceil N/2 \rceil$.

[1] In this paper, we use the notation $(i \rightarrow j)$ and (i,j) interchangeably to mean a directed edge from i to j. The notation $\{i,j\}$ will be used to refer to the node pair.

[2] We use the terms clusterhead and root of a cluster interchangeably.

IV. THE MERGE PHASE

Once the network has been clustered into L clusters, a $maximum$ of $L-1$ links, directed to the roots of the clusters $C_2, C_2 \cdots C_L$, need to be chosen to obtain a connected routing tree. Before explaining the edge selection criteria and the tree building algorithm, we establish the following additional notation.

S_{init}	= initial clustered solution.
$cost(S_{init})$	= cost of initial clustered solution.
L	= number of clusters in S_{init}, $L > 1$.
t	= time index.
t^{max}	= maximum time index.
N_A	= number of Type-A ants.
N_B	= number of Type-B ants.
τ^{min}	= minimum pheromone level on any edge.
$\tau_{ij}(t)$	= pheromone level on the edge (i,j) at time t.
β_A	= tunable parameter for Type-A ants, $0 < \beta_A \leq 1$.
β_B	= tunable parameter for Type-B ants, $0 < \beta_B < \beta_A \leq 1$.
ρ	= pheromone decay coefficient, $\rho \in (0,1]$.
q	= uniform random variable over [0,1]
q_0	= tunable parameter, $q_0 \in [0,1]$.
k	= tree-building iteration index.
$C_l^{k,m}$	= subtree in cluster l at iteration k of ant m.

At time $t = 0$, the pheromone level on all edges is initialized to τ^{min}; $i.e.$, $\tau_{ij}(0) = \tau^{min}$.

A. Tree building process

Tree building[3] is an iterative process which starts with an initial clustered solution and continues till there is only one non-empty cluster. The iteration converges in $at\ most$ $L-1$ iterations, $i.e.$, $k \leq L-1$. Because of the wireless advantage property, whereby multiple nodes can be reached by a single transmission, the number of iterations can range from as few as 1 to $L-1$ (this will be the case when exactly one new clusterhead is reached at each iteration).

At each iteration, an edge (i,j) is chosen such that the tail of the edge is in cluster 1 and its head is in the set of clusterheads of clusters 2 to L. Assuming that node j is the root of cluster l, clusters 1 and l are then merged. Further, if the transmission reaches other clusterheads, those clusters are merged with cluster 1 as well (similar to the backtracking step in the cluster algorithm). At the next iteration, an edge is chosen from the modified set of clusters. This process is repeated till the number of non-empty clusters is equal to 1.

[3]In keeping with swarm intelligence terminology, we will refer to the tree-building agents as ants.

For any ant m, a $pseudo-random-proportional$ decision rule is used for choosing an edge at iteration k of its tree building process, as described in Figure 1. The edge is chosen from a set of candidate edges (Step 1 in Figure 1), either deterministically or probabilistically (Step 3 in Figure 1). The extent to which probabilistic decisions are made is controlled by the tunable parameter q_0. In our simulations, we varied q_0 with t so that decision making is predominantly probabilistic during the initial stages of the algorithm and mostly deterministic during the latter stages. This is discussed in Section V.

For deterministic edge selection (eqns. 7, 8), the factors which determine the desirability of choosing an edge (i,j) at iteration k are:

- the transmitter power required to reach node j from i, $\mathbf{P}_{ij}$, scaled exponentially by the parameter β_A or β_B, depending on the type of the ant. Higher the power required, lower the desirability of choosing that edge. The degree of desirability can be varied by properly selecting β_A and β_B, as explained subsequently,
- the pheromone level, $\tau_{ij}(t)$, of the edge at time t. Since edges which are part of better solutions are positively reinforced[4], presence of a high pheromone level on an edge is used to boost the desirability of choosing that edge.

Probabilistic edge selection (9) is used for efficient exploration of the search space. In this mode, Type-B ants rely only on the transmitter powers, unlike Type-A ants which utilize both transmitter powers and pheromone level for choosing an edge. This is done so that a fraction of the ant population (represented by the Type-B ants) can continue to explore the search space throughout the running time of the algorithm, without being biased by high pheromone levels on a select group of edges.

We now explain how the degree of desirability of choosing an edge can be controlled by varying β_A and β_B. Consider an arbitrary 4-node network. Suppose we have one Type-A and one Type-B ant at node 1. Assume that the distances of nodes 2, 3 and 4 from 1 are $d_{12} = 0.5$, $d_{13} = 1.5$ and $d_{14} = 2.0$. If we now set $\beta_A = 1$ and $\beta_B = 0.1$, the probabilities $\{a_{ij} : i = 1, j = 2,3,4\}$ (eqn. 9, or 7, if we assume that all τ_{ij}'s are equal) for the two types of ants are as follows:

- Type-A: $a_{12} = 0.63$, $a_{13} = 0.21$, $a_{14} = 0.16$
- Type-B: $a_{12} = 0.36$, $a_{13} = 0.32$, $a_{14} = 0.32$

Clearly, if both the ants are allowed to choose an edge probabilistically, while the Type-A ant will choose the nearest

[4]At any time t, the pheromone level on the edge (i,j), $\tau_{ij}(t)$, reflects the cumulative knowledge acquired by the ants till time $t-1$ on the desirability of moving to node j from node i.

1. At any iteration k, an ant m can travel *from* any node in cluster 1 *to* the root of any $non-empty$ cluster l, $2 \leq l \leq L$. The set of possible edges to choose from, $edge\_list_m^k$, is given by: $edge\_list_m^k = \{(i,j)\}$, where:

$$i \in C_1^{k-1,m} \; ; \; j \in chd(C_l^{k-1,m}), \; 2 \leq l \leq L, \; C_l^{k-1,m} \neq \emptyset$$

2. Sample q from a uniform distribution over [0,1].
3. if($q < q_0$) /* *Deterministic decision making* */
   - Compute the decision matrix $\mathbf{A}^{k,m} = \{a_{ij} : (i,j) \in edge\_list_m^k\}$ based on which ant m makes its decision for selecting an edge at step k.

$$a_{ij} = \begin{cases} \frac{[\tau_{ij}(t)][\mathbf{P}_{ij}]^{-\beta_A}}{\sum_{x,y}[\tau_{xy}(t)][\mathbf{P}_{xy}]^{-\beta_A}} & \text{Type-}A \text{ ants} \\ \frac{[\tau_{ij}(t)][\mathbf{P}_{ij}]^{-\beta_B}}{\sum_{x,y}[\tau_{xy}(t)][\mathbf{P}_{xy}]^{-\beta_B}} & \text{Type-}B \text{ ants} \end{cases} \quad (7)$$

   where $(x,y) \in edge\_list_m^k$.
   - Choose the strongest edge, (f^k, t^k), from $\mathbf{A}^{k,m}$.

$$(f^k, t^k) = \text{argmax}_{i,j}\{a_{ij}\} \quad (8)$$

else /* *Explore. Probabilistic decision making.* */
   - Compute $\mathbf{A}^{k,m} = \{a_{ij} : (i,j) \in edge\_list_m^k\}$.

$$a_{ij} = \begin{cases} \frac{[\tau_{ij}(t)][\mathbf{P}_{ij}]^{-\beta_A}}{\sum_{x,y}[\tau_{xy}(t)][\mathbf{P}_{xy}]^{-\beta_A}} & \text{Type-}A \text{ ants} \\ \frac{[\mathbf{P}_{ij}]^{-\beta_B}}{\sum_{x,y}[\mathbf{P}_{xy}]^{-\beta_B}} & \text{Type-}B \text{ ants} \end{cases} \quad (9)$$

   where $(x,y) \in edge\_list_m^k$.
   - Choose an edge (f^k, t^k) from $\mathbf{A}^{k,m}$ probabilistically.

end if

Fig. 1. *Pseudo – random – proportional* edge selection criterion at any iteration k of the tree building process by ant m.

node (node 2) 63% of the time, the Type-B ant will choose any of the three nodes with almost even probability. Type-B ants, therefore, can select their edges by looking "deeper" into the network, as opposed to Type-A ants which are "mostly greedy" and tend to choose nearby nodes. Accordingly, we refer to Type-A ants as $narrow-vision$ ants and Type-B ants as $wide-vision$ ants. Note that, wide-vision ants, because of their ability to make decisions by looking deeper into the network, are better suited for exploiting the wireless advantage property than narrow-vision ants, particularly if the pheromone distribution on all edges are almost uniform.

B. Edge reinforcement

In this section, we discuss how edges are selectively reinforced after all ants have completed their tree-building process at time t. A flowchart of the composite $cluster-merge$ algorithm is shown in Figure 2. It may be noted from the figure that the tree building process can be implemented in parallel since there is no interaction between the ants during this phase of the algorithm. Once the ants have built their trees from the initial clustered solution, the best ant is identified and its tree is subjected to a local tree-improvement procedure. Alternately, the improvement heuristic can be moved inside the parallel tree-building phase of the algorithm. That is, a tree built by an ant can first be subjected to the tree-improvement procedure before its cost is computed and the best ant is identified.

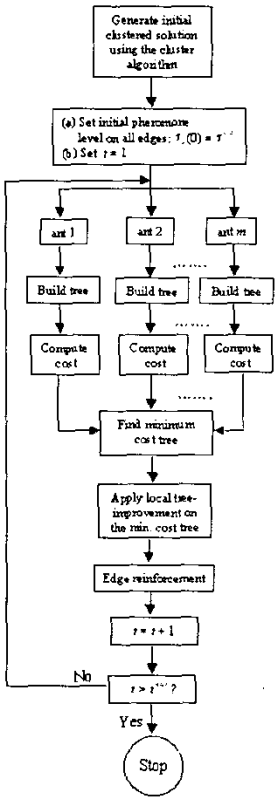

Fig. 2. Flowchart of the overall $cluster-merge$ algorithm. The "Local tree-improvement" block is optional. Edge reinforcement is done as shown in Figure 3.

After the best solution is identified, we reinforce the constituent edges of the current best solution. First, let us define the following parameters.

- $T_{gbest}^{post}(t)$ = global best post-improvement (*i.e.*, after applying the tree improvement heuristic) tree found *till* time t. The final solution is the tree $T_{gbest}^{post}(t^{max})$.
- $Y_{gbest}^{post}(t)$ = cost of $T_{gbest}^{post}(t)$. The cost of the final solution is $Y_{gbest}^{post}(t^{max})$.

For all $(i,j) \in \mathcal{E}$,
if $(i,j) \in \mathcal{E}_c(t)$

$$\tau_{ij}(t+1) = \tau_{ij}(t) + \left(\frac{Y_{gbest}(t)}{Y_{cbest}(t)}\right)\left(\frac{\rho}{Y_{cbest}(t)}\right); \quad (13)$$

elseif $(i,j) \in \mathcal{E}_g(t)$; /* Do nothing */
else

$$\tau_{ij}(t+1) = \max\left[\tau^{min}, (1-\rho)\tau_{ij}(t)\right]; \quad (14)$$

endif

Fig. 3. Edge reinforcement rules.

- $T^{pre}_{gbest}(t)$ = pre-improvement version of $T^{post}_{gbest}(t)$.
- $T^{post}_{cbest}(t)$ = best post-improvement tree found *at* time t.
- $Y^{post}_{cbest}(t)$ = cost of $T^{post}_{cbest}(t)$.

For $t = 1$,

$$T^{post}_{gbest}(1) = T^{post}_{cbest}(1) \text{ and } Y^{post}_{gbest}(1) = Y^{post}_{cbest}(1) \quad (10)$$

For $t > 1$,

$$T^{post}_{gbest}(t) = \begin{cases} T^{post}_{cbest}(t) & \text{if } Y^{post}_{cbest}(t) < Y^{post}_{gbest}(t-1) \\ T^{post}_{gbest}(t-1) & \text{otherwise} \end{cases} \quad (11)$$

$$Y^{post}_{gbest}(t) = \begin{cases} Y^{post}_{cbest}(t) & \text{if } Y^{post}_{cbest}(t) < Y^{post}_{gbest}(t-1) \\ Y^{post}_{gbest}(t-1) & \text{otherwise} \end{cases} \quad (12)$$

Next, define $\mathcal{E}_g(t)$ to be the set of edges which exist in $T^{pre}_{gbest}(t)$, but not in the initial clustered solution, S_{init}.[5] Similarly, let $\mathcal{E}_c(t)$ be the set of edges which exist in $T^{pre}_{cbest}(t)$, but not in S_{init}. Edges are then reinforced as shown in Figure 3.

Let us first assume that the best ant at time t is also the global best ant till time t; *i.e.*, $T^{post}_{cbest}(t) = T^{post}_{gbest}(t) \Rightarrow Y_{gbest}(t) = Y_{cbest}(t)$. In this case, the reinforcement rule is straightforward. All edges which are in $\mathcal{E}_c(t)$ are reinforced by the amount $\rho/Y_{cbest}(t)$ (since $Y_{gbest}(t) = Y_{cbest}(t)$ in (13)). The pheromone levels of the rest of the edges are allowed to decay (14), subject to a minimum threshold τ^{min}, thus making them less attractive for selection at subsequent time indices [6].

[5]It is not necessary to keep a record of $T^{pre}_{gbest}(t)$ since it is required only for determining the set of edges $\mathcal{E}_g(t)$.

[6]Note that the lack of any decay component in (13) can result in some edges having an extremely high level of pheromone, especially if t^{max} is set pretty high, leading to premature stagnation. This situation can be avoided by upper thresholding the pheromone levels or by adding a decay component, $-\rho \cdot \tau_{ij}(t)$, to the rhs of (13). In our simulations, however, we did not observe any stagnation since t^{max} was set to 40 for all N.

If the best ant at time t is not the global best ant till time t, we adopt a conditional reinforcement mechanism. If $(i,j) \in \mathcal{E}_c(t)$, the reinforcement amount $\rho/Y_{cbest}(t)$ is weighted by the ratio of the global best cost to the current best cost (13). This ensures that if the current best solution is almost as good as the global best, its constituent edges receive almost full reinforcement (which is $\rho/Y_{cbest}(t)$). Conversely, if the current best solution is much worse than the global best, its edges receive little reinforcement. If (i,j) is not in $\mathcal{E}_c(t)$ but $(i,j) \in \mathcal{E}_g(t)$, no reinforcement is done. Finally, if (i,j) is neither in $\mathcal{E}_c(t)$ nor in $\mathcal{E}_g(t)$, its pheromone level is allowed to decay (14).

We conclude this section by comparing the complexity[7] of the CM algorithm vis-a-vis the Ant Colony System (ACS) algorithm [5], assuming a sequential implementation. First, note that the cluster phase can be executed in $O(N^3)$ time, similar to the BIP algorithm. Next, note that each tree-building process requires at most $O(NL^2)$ time, if $L << N$. The overall time complexity is therefore $O(N^3) + (t^{max} \times \text{no. of ants}) \times O(NL^2)$. The complexity of the ACS algorithm, on the other hand, is on the order of $(t^{max} \times \text{no. of ants}) \times O(N^3)$.

V. SIMULATION RESULTS

We tested the CM algorithm on 25, 50, 75 and 100-node networks in a 5×5 grid. In each case, 50 networks were randomly generated and the tree powers were averaged to obtain the mean tree power. 'α' was chosen to be equal to 2 for all cases. Values of the parameters used in the simulations are given in Table I. For $N = 25$ and 50, the figures in boldface represent the parameter values used in the ACS algorithm. Note that, for $N = 25$, while $t^{max} \times (N_A + N_B) = 650$ for the ACS algorithm, it is equal to 320 for the CM algorithm. For $N = 50$, $t^{max} \times (N_A + N_B) = 2500$ for ACS and 400 for CM, a reduction of more than a factor of 6.

A key point to note in Table I is the dynamic nature of ϕ and β_B with respect to t. Gradual reduction of ϕ ensures that the bulk of the exploration work (Step 4 in Figure 1) is carried out during the initial stages of the algorithm, when the pheromone distribution is almost even and trail conditions are more suitable for wide-vision ants. Increasing β_B with respect to t has the effect of reducing the visibility of wide-vision ants so that they start behaving more like their narrow-vision counterparts as t increases. In fact, for $\lceil 0.6 * t^{max} \rceil + 1 \leq t \leq t^{max}$, β_B is equal to β_A, which ensures that all ants concentrate on a select group of edges and

[7]We disregard the local tree-improvement step for complexity calculation, which is optional and can be used in either of the two algorithms.

look for better solutions within their neighborhoods during the latter stages of the algorithm.

The mean tree powers for the BIP solutions are shown in column 2 in Table II. The mean tree powers for the BIP solutions followed by the sweep algorithm [1] are shown in column 3. For $N = 25$ and 50, column 4 lists the mean tree powers obtained by applying the ACS algorithm. Finally, column 5 shows the mean tree powers obtained using the CM algorithm with 1-shrink [7] local tree-improvement. The average number of clusters generated after the clustering phase are: (a) $L = 7.32$ for $N = 25$, (b) $L = 15.02$ for $N = 50$, (c) $L = 22.36$ for $N = 75$ and (d) $L = 29.42$ for $N = 100$, approximately 30% of N in each case. Percentage improvement in mean tree power over the BIP solutions are shown in Table III.

It can be seen from Tables II and III that the CM algorithm, with local tree-improvement, is able to find solutions of similar or better quality than the ACS algorithm. It should be noted, however, that no local tree-improvement heuristic was implemented in the latter. We conjecture that the performance of ACS would improve (albeit, at enhanced computation time) if such a step is incorporated. Nevertheless, the improvement in solution quality that can be achieved by implementing the $cluster - merge$ algorithm, compared to BIP, is significant.

TABLE I

Parameter values used in the simulations. For $N = 25$ and 50, the figures in boldface represent the values used in the ACS algorithm [5].

Parameter	$N = 25$	$N = 50$	$N = 75$	$N = 100$
t^{max}	40 (50)	40 (100)	40	40
N_A	4 (7)	6 (13)	8	8
N_B	4 (6)	4 (12)	4	4
ρ	0.2			
τ^{min}	$1/cost(S_{init})$			
β_A	1			
β_B	0.5, if $t \leq \lceil 0.3 * t^{max} \rceil$ 0.75, if $\lceil 0.3 * t^{max} \rceil + 1 \leq t \leq \lceil 0.6 * t^{max} \rceil$ 1, if $\lceil 0.6 * t^{max} \rceil + 1 \leq t \leq t^{max}$			
q_0	0.3, if $t \leq \lceil 0.3 * t^{max} \rceil$ 0.6, if $\lceil 0.3 * t^{max} \rceil + 1 \leq t \leq \lceil 0.6 * t^{max} \rceil$ 0.9, if $\lceil 0.6 * t^{max} \rceil + 1 \leq t \leq t^{max}$			

TABLE II

Mean tree powers for BIP, BIP followed by sweep, ACS and CM.

N	BIP	$BIP(sweep)$	ACS	CM
25	12.46	12.14	10.21	10.23
50	11.67	11.45	10.04	9.90
75	11.63	11.37	-	9.88
100	11.60	11.36	-	9.87

TABLE III

Percentage improvement in mean tree power over BIP.

N	$BIP(sweep)$	ACS	CM
25	-2.57%	-18.06%	-17.90%
50	-1.89%	-13.93%	-15.17%
75	-2.24%	-	-15.05%
100	-2.07%	-	-14.91%

VI. CONCLUSION

In this paper, we have proposed a $cluster - merge$ algorithm for solving the minimum power broadcast problem in wireless networks. While the cluster phase is a look-ahead variant of the BIP algorithm, the merge phase is based on probabilistic positive reinforcement search techniques used in swarm intelligence. Experimental simulations prove that the algorithm is able to find solutions of similar quality as the ACS algorithm, but at significantly reduced computation times. We are currently researching a distributed implementation of the merge phase of the algorithm. Work is also ongoing to extend the ideas described in this paper to a system with directional antennas.

REFERENCES

1) J.E.Wieselthier, G.D. Nguyen and A. Ephremides, "On the construction of energy-efficient broadcast and multicast trees in wireless networks", *IEEE INFOCOM* 2000, pp. 585-594.
2) Ivan Stojmenovic, Mahtab Seddigh and Jovisa Zunic, "Internal Nodes Based Broadcasting in Wireless Networks", *Proc. of the 34th Hawaii International Conference on System Sciences*, 2001.
3) R.J. Marks II, A.K. Das, M.A. El-Sharkawi, P. Arabshahi and A. Gray, "Minimum Power Broadcast Trees for Wireless Networks: Optimizing Using the Viability Lemma", *Proc. of the IEEE International Symposium on Circuits and Systems*, Scottsdale, Arizona, 2002.
4) E. Bonabeau, M. Dorigo, and G. Theraulaz, *"Swarm intelligence: From natural to artificial systems"*, Oxford University Press, 1999.
5) A.K. Das, R.J. Marks II, M.A. El-Sharkawi, P. Arabshahi and A. Gray, "The Minimum Power Broadcast problem in Wireless Networks: An Ant Colony System Approach", *Proc. IEEE CAS Workshop on Wireless Communications and Networking*, Pasadena, CA, Sept. 5-6, 2002.
6) A. K. Das, R.J. Marks II, M.A. El-Sharkawi, P. Arabshahi and A. Gray, "Minimum Power Broadcast Trees for Wireless Networks: Integer Programming Formulations", *Proc. of INFOCOM*, San Francisco, CA, Apr. 1-3, 2003.
7) A. K. Das, R.J. Marks II, M.A. El-Sharkawi, P. Arabshahi and A. Gray, "r-Shrink: A heuristic for improving Minimum Power Broadcast Trees in Wireless Networks", submitted to *GLOBECOM 2003*.