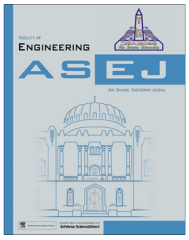
Ain Shams Engineering Journal (2014) 5, 113–120

Ain Shams University

Ain Shams Engineering Journal

www.elsevier.com/locate/asej
www.sciencedirect.com



ELECTRICAL ENGINEERING

A hybrid ACO/PSO based algorithm for QoS multicast routing problem

Manoj Kumar Patel *, Manas Ranjan Kabat, Chita Ranjan Tripathy

Dept. of Comp. Sc. & Engg., Veer Surendra Sai University of Technology, Burla, Sambalpur, Orissa, India

Received 1 January 2013; revised 22 June 2013; accepted 18 July 2013
Available online 7 October 2013

KEYWORDS

Swarm agent;
ACO;
PSO;
Multicast;
QoS routing

Abstract Many Internet multicast applications such as videoconferencing, distance education, and online simulation require to send information from a source to some selected destinations. These applications have stringent Quality-of-Service (QoS) requirements that include delay, loss rate, bandwidth, and delay jitter. This leads to the problem of routing multicast traffic satisfying QoS requirements. The above mentioned problem is known as the QoS constrained multicast routing problem and is NP Complete. In this paper, we present a swarming agent based intelligent algorithm using a hybrid Ant Colony Optimization (ACO)/Particle Swarm Optimization (PSO) technique to optimize the multicast tree. The algorithm starts with generating a large amount of mobile agents in the search space. The ACO algorithm guides the agents' movement by pheromones in the shared environment locally, and the global maximum of the attribute values are obtained through the random interaction between the agents using PSO algorithm. The performance of the proposed algorithm is evaluated through simulation. The simulation results reveal that our algorithm performs better than the existing algorithms.

1. Introduction

The rapid development in network multimedia technology has enabled more and more real-time multimedia services such as videoconferencing, online games, and distance education to become the mainstream Internet activities. These services often require the underlying network to provide multicast capabilities. The multicast refers to the delivery of packets from a single source to multiple destinations. These real-time applications have a stringent requirement of QoS parameters like bandwidth, delay, jitter, and so on to ensure smooth, consistent, and fair transmission to the receivers. The central problem of QoS routing is to set up a multicast tree that can satisfy certain QoS parameters. However, the problem of constructing a multicast tree under multiple constraints is NP Complete [1]. Hence, the problem is usually solved by heuristics or intelligent optimization.

In recent years, some meta-heuristic algorithms such as the ant colony algorithm [2–9], genetic algorithm [10–12], particle swarm optimization [13–17], and Tabu search [18,19] have been adopted by the researchers to solve the multi-constrained QoS routing problems.

* Corresponding author. Tel.: +91 9861461357.
E-mail addresses: patel.mkp@gmail.com (M.K. Patel), manas_kabat@yahoo.com (M.R. Kabat), crt.vssut@yahoo.com (C.R. Tripathy).

Peer review under responsibility of Ain Shams University.

Production and hosting by Elsevier

2090-4479 © 2013 Production and hosting by Elsevier B.V. on behalf of Ain Shams University.
http://dx.doi.org/10.1016/j.asej.2013.07.005

In [2], an intelligent routing algorithm ANTNET based on ant colony algorithm was proposed. Their algorithm has some attractive distribution features and it can find a near-optimal path from the source node to each destination node. It also provides the required results in simulation. Although the ANTNET is a unicast routing algorithm, it can be easily applied to multicast routing with some modifications. In spite of the said merits of the ANTNET, it suffers from a serious drawback i.e. the slow convergence rate. Another ant intelligence algorithm was introduced in [3] for the computation of the QoS multicast tree. A tree growth based ACO algorithm (TGBACA) has been proposed in [8]. It generates a multicast tree in the way of tree growth and optimizes the ant colony parameters through their most efficient combinations. The major weakness of the ant colony algorithm is that it converges slowly at the initial step and takes more time to converge. This happens due to improper selection of the initial feasible parameter [3]. The overhead also increases due to merging and pruning of the trees.

Subsequently, the genetic algorithm (GA) was used to find a multicast tree satisfying the constraints of bandwidth and delay with least cost [10–12]. The GA has three operators: selection, crossover, and mutation. The individuals are stored in connective matrices by adopting the binary coding system. The initial colony is generated randomly without considering QoS constraints. The selection operation adopts Roulette wheel algorithm to select the best individuals from the parent generation to pass onto the child generation. Then, the crossover operation is used to find out the fittest among the best. A penalty function is adopted to solve QoS constraints in the multicast trees, which do not satisfy QoS constraints. Although sometimes the algorithm's performance is observed to be satisfactory, still it encounters some faults, such as the local search ability, premature convergence, and slow convergence speed. Further, the genetic algorithm does not assure to find a global optimum. It happens very often when the populations have a lot of subjects.

In [13–17], researchers have proposed some PSO algorithms to solve QoS constraint routing problem. The PSO algorithm proposed in [14] solves the QoS multicast routing problem and can obtain a feasible multicast tree by exchanging the paths. This algorithm can converge to the optimal or near-optimal solution with lower computational cost. Another algorithm based on the concept of quantum mechanics named as Quantum-Behaved Particle Swarm Optimization (QPSO) was proposed in [15]. Here, the proposed method converts the QoS multicast routing problem into integer programming problem and then finally solves using the QPSO. The QPSO finds the path from the source node to each destination node and constructs the tree by merging the paths. A tree based PSO has been proposed in [17] for optimizing the multicast tree directly. However, its performance depends on the number of particles generated. Another drawback of the said algorithm is the merging of multicast trees. The elimination of directed circles and nesting of directed circles are also very complex and are considered as some of the limitations of the PSO [17].

In recent days, many researchers have solved the QoS constrained multicast tree using Tabu search [18,19]. A Tabu search method was proposed in [18] to search for the multicast tree with the least cost that satisfies the constraints of bandwidth and delay. This algorithm obtains a complete graph of all group members at the initial step and obtains the initial Steiner tree via the generated tree of the complete graph. In this way, the k-shortest paths replace the edges to find the chances of getting better results. The method mentioned above is similar to the method of path combination. However, it does not operate directly on the multicast tree. This weakness makes it impossible to eliminate the constraints of conventional multicast routing algorithms. Hence, there arises a need to proceed further and do more amount of work in searching paths and integrating the multicast trees.

In this paper, we propose a hybrid ACO/PSO algorithm based on the swarming agent architecture for QoS multicast routing. Our work is inspired by the swarming agent algorithm proposed by Brueckner and Parunak [20] for distributed data pattern and Meng [21] for Proteomic Pattern detection of ovarian cancer. In our work, a large amount of mobile agents are generated in the search space. Two collective and coordination process for the mobile agents are proposed. One is based on the ACO [8] algorithm for guiding the agents' movements by pheromones in the shared environment locally and the another is based on the PSO algorithm [17] for obtaining the global maximum of the attribute values through the random interaction between the agents.

The rest of the paper is organized as follows: A mathematical model is proposed to model a computer network in Section 2. The proposed algorithm and its working principles are presented in Section 3. The results of the simulation are presented in Section 4. Finally, the Section 5 concludes the paper.

2. Mathematical model

This section is devoted toward development of a mathematical model and problem statement to be used in the next section.

A network is modeled as a directed, connected graph $G = (V, E)$, where V is a finite set of vertices (network nodes) and E is the set of edges (network links), representing connection of these vertices. Let $n = |V|$ be the number of network nodes and $l = |E|$ be the number of network links. The *link* $e = (u,)$ from node $u \in V$ to node $v \in V$ implies the existence of a link $e' = (,u)$ from node v to node u. Four non-negative real value functions are associated with each link $e(e \in E)$: cost $C(e)$: $E \to R^+$, delay $D(e)$: $E \to R^+$, loss rate $L(e)$: $E \to R^+$, and available bandwidth $B(e)$: $E \to R^+$. The link cost function, $C(e)$, may be either monetary cost or any measure of resource utilization that must be optimized. The link delay, $D(e)$, is considered to be the sum of switching, queuing, transmission, and propagation delays. The link loss rate, $L(e)$, is the packet loss rate on the receiving end on link e. The link bandwidth, $B(e)$, is the residual bandwidth functions. $D(e)$, $L(e)$, and $B(e)$ define the criteria that must be constrained (bounded). Because of the asymmetric nature of the communication networks, it is often the case that $C(e) \neq C(e')$, $D(e) \neq D(e')$, $L(e) \neq L(e')$, *and* $B(e) \neq B(e')$.

A multicast tree $T(s, M)$ is a subgraph of G spanning the source node $s \in V$ and the set of destination nodes $M \subseteq V - \{s\}$. Let $m = |M|$ be the number of multicast destination nodes. We refer to M as the destination group and $\{\{s\} \cup M\}$ the multicast group. In addition, $T(s, M)$ may contain relay nodes (Steiner nodes), the nodes in the multicast tree but not in the multicast group. Let $P_T(s, d)$ be a unique path in the tree T from the source node s to a destination node $d \in M$.

We now introduce the parameters that characterize the quality of the multicast tree below.

The total cost of the tree $T(s, M)$ is defined as sum of the cost of all links in that tree and can be given by

$$C(T(s, M) = \sum_{e \in T(s,M)} C(e) \tag{2.1}$$

The total delay of the path $P_T(s, d)$ is simply the sum of the delay of all links along $P_T(s, d)$:

$$D(P_T(s, d)) = \sum_{e \in P_T(s,d)} D(e) \tag{2.2}$$

Hence, the total loss rate of the path:

$$L(P_T(s, d)) = 1 - \prod_{e \in P_T(s,d)} (1 - L(e)) \tag{2.3}$$

The bottleneck bandwidth of the path $P_T(s, d)$ is defined as minimum available residual bandwidth at any link along the path:

$$B(P_T(s, d)) = \min\{B(e), e \in P_T(s, d)\} \tag{2.4}$$

The delay jitter of the tree $T(s, M)$ is defined as the average difference of delay on the path from the source to the destination node:

$$DJ(T(s, M)) = \sqrt{\sum_{d \in M} (D(P_T(s, d) - delay\_avg)^2)} \tag{2.5}$$

where *delay_avg* denotes the average value of delay on the path from the source to the destination nodes.

Let Δ be the delay constraint, ζ be the loss rate constraint, β be the bandwidth constraint of the path from source to the destination node d, and δ be the delay jitter constraint. The multi-constrained least-cost multicast problem is defined as follows:

Minimize $C(T(s, M))$ subjects to:

$$\begin{aligned} &D(P_T(s, d)) \leqslant \Delta \quad \forall \quad d \in M \\ &L(P_T(s, d)) \leqslant \zeta \quad \forall \quad d \in M \\ &B(P_T(s, d)) \geqslant \beta \quad \forall \quad d \in M \\ &DJ(T(s, M)) \leqslant \delta \end{aligned} \tag{2.6}$$

3. The proposed algorithm (hybrid ACO/PSO algorithm for multicast routing optimization)

In this section, first, we discuss the relevant aspects of ACO and PSO. We proceed to create a hybrid ACO/PSO algorithm for multicast routing using swarming agents.

An agent is an independent processing entity that interacts with the external environment and the other agents to pursue its particular set of goals. By using the ACO algorithm, the agents in the systems coordinate their behaviors and communicate their results through the pheromone, which is a shared dynamic environment. The self-organizing and self-adapting system-level behaviors can be obtained through these pheromone interactions. Each agent can only observe the limited environment within its local sensors. In other words, the agent can only sense the pheromone that is close to its current position, while it has no idea of the existence of other pheromone far away. Without a central host, it is possible that most agents may easily get locked in a local maximum.

On the other hand, the agents using PSO algorithm coordinate their behaviors through the random interaction with other agents. By following certain rules of interaction, the agents in the population adapt their scheme of belief to the ones that are most successful among their social network. Over the time, a global optimization can be obtained. In this section, a hybrid ACO/PSO algorithm is proposed for the optimization of swarming agent based multicast routing problems. Before introducing the proposed algorithm, we make brief discussion on ACO, PSO, and swarming architecture.

In the following section, we discuss the two most important optimization algorithms: ACO and PSO followed by discussion on swarming architecture.

3.1. ACO algorithm

An ACO algorithm is essentially a system that simulates the natural behavior of ants, including their mechanisms of cooperation and adaptation. The ACO algorithms are based on the following ideas. First, each path followed by an ant is associated with a candidate solution for a given problem. Second, when an ant follows a path, the amount of pheromone deposited on that path is proportional to the quality of the corresponding candidate solution for the target problem. Third, when an ant has to choose between two or more paths, the path(s) with a larger amount of pheromone are more attractive to the ant. After some iteration, eventually, the ants will converge to the path, which is expected to be the optimum or a near-optimum solution for the target problem.

3.2. PSO algorithm

The PSO algorithm is population-based, where a set of potential solutions evolves to approach a convenient solution (or set of solutions) for a problem. The aim of this optimization method is finding the global optimum of a real-valued function (fitness function) defined in a given space (search space). Rather than simply a social simulation, the PSO can be treated as a powerful search algorithm, capable of optimizing a wide range of N-dimensional problems.

In the PSO algorithm, each individual is called a "particle," and is subject to a movement in a multidimensional space that represents the belief space. Particles have memory, and thus, they retain part of their previous states. There is no restriction for the particles to share the same point in belief space, but in any case, their individuality is preserved. Each particle's movement is the composition of an initial random velocity and two randomly weighted influences: individuality (i.e., the tendency to return to the particle's best previous position) and sociality (i.e., the tendency to move toward the neighborhood's best previous position).

The velocity and position of the particle at any iteration are updated based on the following equations:

$$v_{id}^{t+1} = w.v_{id}^{t} + c_1.rand()(p_{id}^{t} - x_{id}^{t}) + c_2.Rand()(p_{gd}^{t} - x_{id}^{t}) \tag{3.1}$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \tag{3.2}$$

where v_{id}^{t} is the component in dimension d of the ith particle velocity in iteration t; x_{id}^{t} is the component in dimension d of the ith particle position in iteration t; c_1 and c_2 are constant weight factors; p_i is the best position achieved by ith particle; p_{gd} is the best position found by the neighbors of ith particle; *rand( )* and *Rand( )* are random factors in the [0, 1] interval; and w is the inertia weight.

The PSO algorithm requires tuning of some parameters: the individual weight (c_1), sociality weight (c_2), and the inertia factor (w).

3.3. Swarming agent architecture

Initially, n multicast tree patterns are generated randomly, and m key values as attributes are calculated for m destinations of each multicast tree pattern. Therefore, the structure of the pattern is defined by the following equation:

$$T_i = \{a_{i1}, a_{i2}, \ldots\ldots, a_{im}\} \text{ for } i = 1..n \tag{3.3}$$

where T_i denotes the multicast tree pattern i, a_{ij} represents the jth attribute of pattern i in each pattern, and n is the number of the generated patterns.

For each pattern, an associated *pattern agent* is created which is fixed to the pattern. Then, n numbers of mobile agents are generated to detect the fit patterns from the randomly generated patterns and recombine some of the selected patterns together to build a pattern with more fitness value. These mobile agents are referred as *particle agents,* which can move from one pattern to another in the search space and interact with other particle agents dynamically. Initially, the particle agents are uniformly distributed in the search space. The particle agents converge to the fittest pattern eventually after some iterations of the algorithm.

The particle agents are allowed to deposit pheromones and sense local attributes in each pattern. The pattern agent is responsible for executing the dynamics of pheromone aggregation, dispersion, and evaporation. There are two levels of pheromones: one is for pattern pheromone and the other is the attribute pheromone inside the pattern. The pattern with high pheromone has either high probability to become the best fit pattern or some of the attributes inside this pattern have the higher potential to be included in the best fit pattern. This attracts more particle agents to move to the pattern. The pheromone gets evaporated over time if no agent deposits more pheromone on it. Once an agent enters into a pattern, it deposits its pheromone on the selected attributes and deposits the pheromones on the pattern. So, both pattern and the selected attributes are proportional to their fitness values.

3.4. Proposed hybrid ACO/PSO algorithm (HACOPSO)

Initially, n discriminating candidate patterns are generated randomly. These patterns are filled into the grids in the search space of dimension $k \times l$, where each pattern corresponds to one grid based on the order of their generation. Next, n particle agents are generated and uniformly distributed to the search space, where each particle agent occupies one grid. Now, the iteration starts, and for each iteration, first each particle agent evaluates the fitness value of the tree pattern from the cost of the tree, where it is currently located. Using the ACO algorithm, pheromones are deposited on each attribute based on their attribute value by the particle agents.

Fig. 1 shows the grid of dimension 5×6 in which thirty multicast tree patterns are filled. The pattern agent T_i is associated with ith multicast tree pattern and filled sequentially in row major fashion. Thirty mobile particle agents are randomly distributed in search space where each particle agent is attached to one pattern agent.

Due to the fixed topology of the particle agents in the search grid, the maximum number of neighbors is 8 as shown in Fig. 1. Hence, the particle agent can interact with the maximum 8 neighboring particle agents only. If the fitness of the best neighbor is higher than the current pattern, then it is set as the agent's best previous position. Then, the PSO algorithm is used to update the pattern pheromone and attribute pheromone comparing the fitness of the trees and their corresponding attributes. This process is repeated for a fixed number of iterations, and the pattern with the maximum fitness is returned as the solution that satisfies all the constraints. The pseudo-code of the proposed hybrid ACO/PSO algorithm is outlined below.

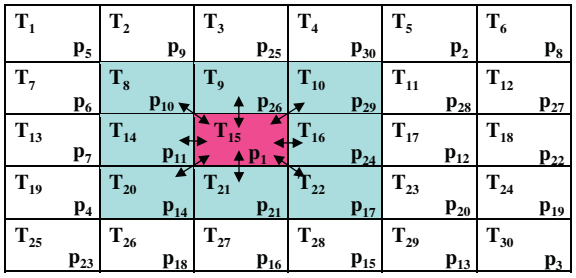


Figure 1 Arrangement of particle and pattern agents.

3.4.1. Pseudo-code for hybrid ACO/PSO algorithm

```
Algorithm HACOPSO (s, n, m, Count_max)
/* n is the maximum number of pattern generated, s is the
source node, m is the number of receivers and Count_max is the
maximum number of iterations the HACOPSO runs */
Generate randomly n discriminating candidate patterns T_i
Fill the patterns into the grid of dimension kxl based on
the order of their generation.
Generate n particle agents randomly and distribute them
uniformly over the search space so that each particle agent
occupies one grid.
while (itr < Count_max)
/* Use ACO algorithm to accept pheromone to the pattern and
attributes */
for i = 1 to n
  Update the pheromone of the pattern selected by the
particle agents i,
  Otherwise, decrease the pheromone in the pattern.
    for j = 1 to m
      Update the pheromone of all attributes trail, by
increasing pheromone in the trails followed by agent i.
    end for
    end for
/*Use the modified PSO algorithm to adjust the movement of
the particle agent in this iteration */
for i = 1 to n
  Update the pattern and attribute pheromone
    If pattern fitness of current position < their best
neighbors' pattern fitness value,
    Then move to the neighbor's pattern.
    If the fitness value of the attribute(s) in the last
position is higher than the new position,
    Then replace those lower-fitness-value attribute with
the higher-fitness-value attribute from last position.
    If the newly generated pattern has more fitness than the
earlier fitness, Then update the attribute with higher
fitness.
end for
end while
```

The multicast tree patterns are generated randomly, and these patterns reside in a search space which is defined as a $k \times l$

rectangular grid. Given source node s, the group member d_1, d_2, $d_3, \ldots, d_m$ where m is the number of group members. The pseudo-code for multicast tree generation is given below.

3.4.2. Pseudo-code for multicast tree pattern generation

```
procedure PatternGeneration (Ti)
1.  Begin
2.  initialize VT = {s}
3.  delaysofar(s) = 1, losssofar(s) = 1, costsofar(s) = 0;
4.  cur_node = s
5.  repeat
6.    Ncur_node = ϕ
7.    for each neighbour node v of the cur_node
            such that v ∉ VT
8.      if(B(cur_node,v) ⩾ β and delaysofar(cur_node)+
          D(cur_node) ⩽ Δ and 1-(losssofar(cur_node)*
            (1-L(cur_node,v)) ⩽ ζ)
9.        Ncur_node = Ncur_node ∪ {v}
10.     end if
11.   end for
12. j = SelectRandom(Ncur_node)
13. cost = cost + C(cur_node, j)
14. costsofar(j) = costsofar(cur_node) + C(cur_node,j)
15. pheromone(j) = 1/costsofar(j)
16. delaysofar(j) = delaysofar(cur_node) + D(curnode,j)
17. losssofar(j) = losssofar(cur_node) *(1−L(curnode,j))
18. if(j ∉ M){
19.   cur_node = j
20. else
21.   cur_node = SelectRandom(VT)
22.     for all (u ∈ M and u ∈ VT)
23.       pheromone(u) = pheromone(u)+pheromone(cur_node)
24.     end for all
25.   end if
26. until (VT contains all nodes of the multicast group)
27. if Ti satisfies the delay jitter that is DJ(Ti(s,M)) ⩽ δ
28.   then calculate pheromone(Ti)from the cost of the tree
29. return Ti
30. end PatternGeneration
```

Initially, the multicast tree V_T is set to the source node s, which is also set as the current node. Accumulated delay and loss up to source node s, *delaysofar*(s) and *losssofar*(s), are set to 0 and 1, respectively. Then, a node j is selected randomly from the neighbor nodes of the current node that satisfies the QoS constraints. The node j is added to the multicast V_T, and the *delaysofar*(j) and *losssofar*(j) up to node j are calculated as shown in Steps 14 and 15 of the algorithm, respectively. If the node j is not the destination node, then the current node is set to j; otherwise, the current node is selected randomly from V_T. The pheromones of the multicast group member that are already added to the multicast tree are updated as shown in Step 23. This process is repeated till all the group members are added to the multicast tree.

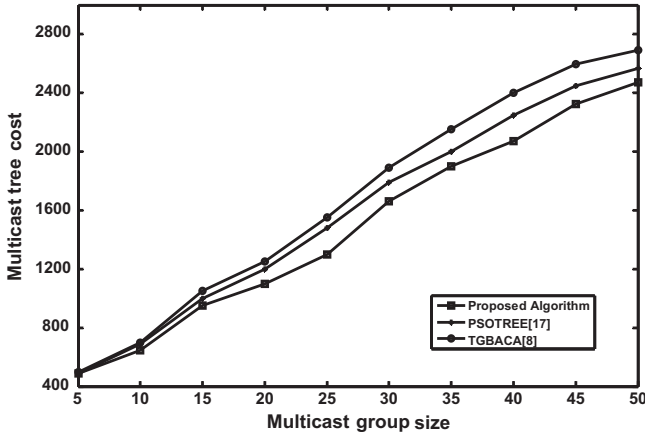


Figure 2 Multicast tree cost vs. group size (No of nodes = 100).

The particle moves to the new position and brings the last position's attribute and its associated fitness values along with its movement to the new position. The new position is updated with better quality attributes discarding the low quality attribute of the new position. The outcome is a new combined pattern with higher quality than both original ones. During the next iteration, the newly built pattern will be executed by the agents and deposit the pheromone as appropriate. After much iteration, eventually, the most strong pheromone trail will be the fittest discriminating pattern.

3.4.3. Time complexity

In this subsection, the time complexity of the proposed algorithm is carried out. The proposed algorithm uses procedure *PatternGeneration* to generate a QoS aware multicast tree pattern randomly. In the *PatternGeneration* procedure, the lines 02–04 initialize the variables in $O(1)$. The maximum number of neighboring nodes of a node can be n. Therefore, the worst case time complexity of line 07 is $O(n)$. The lines 08 and 09 are computed in $O(1)$. Thus, the lines 07–11 can be computed in $O(n)$. The lines 12–21 are computed in $O(1)$. The lines 22–24 are computed in $O(m)$, where m is the number of destinations. The worst case time complexity of computation performed in lines 6–25 is $O(n)$. Since the lines 6–25 are executed for m number of iterations, the worst case time complexity of the procedure *PatternGeneration* is $O(mn)$.

The proposed HACOPSO algorithm starts with generating *partnum* number of tree patterns and runs for a maximum $Count_{max}$ number of iterations. In each iteration, the n particle agents interact with maximum 8 numbers of neighbors, and the pheromone of maximum m number of attributes is updated. The worst case time complexity of our proposed algorithm is $O(partnum \times Count_{max} \times n \times m \times 8)$ i.e. $O(m \times n \times partnum \times Count_{max})$. The time complexity of TGBACA [8] is $O(Count_{max} \times antnum \times n \times |E|)$, and the time complexity of PSOTREE [17] is $O(Count_{max} \times partnum \times n^2)$ where $\text{Count}_{\max}$ is the maximum number of iterations, antnum and partnum are the number of ants and number of particles, respectively, n is the number of nodes, $|E|$ is the number of edges, and m is the number of receivers. Thus, it is clear that the complexity of the proposed algorithm is comparable to the existing algorithms [17,8].

4. Simulation results

We have implemented our proposed algorithm in Visual C^{++}. The experiments are performed on an Intel Core i3 @ 2.27 G.Hz. and 2 GB RAM based platform running Windows 7.0.The positions of the nodes are fixed randomly in a rectangle of size 4000 km × 2400 km. The Euclidean metric is then

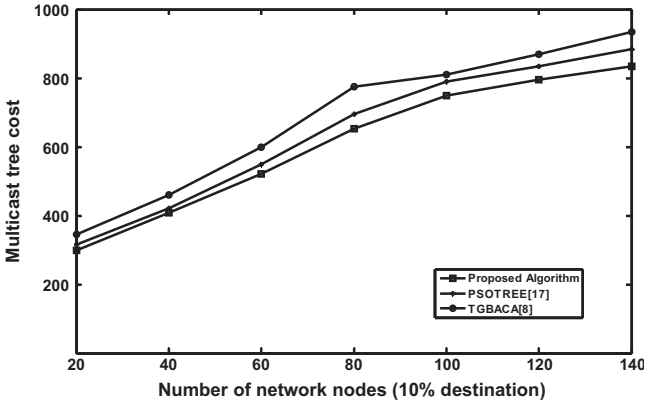


Figure 3 Multicast tree cost vs. network size with 10% nodes as destinations.

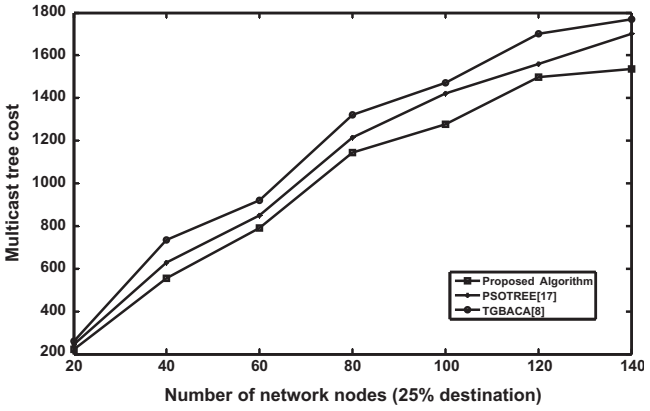


Figure 4 Multicast tree cost vs. network size with 25% nodes as destinations.

used to determine the distance between each pair of nodes. The network topology used in our simulation was generated randomly using Waxman's topology [22]. The edges are introduced between the pairs of nodes u and v with a probability that depends on the distance between them. The edge probability is given by $P(u,) = \beta \exp(-l(u,v)/\alpha L)$, where $0 < \alpha$, $\beta <= 1$ $l(u,v)$ is the Euler distance from node u to v and L is

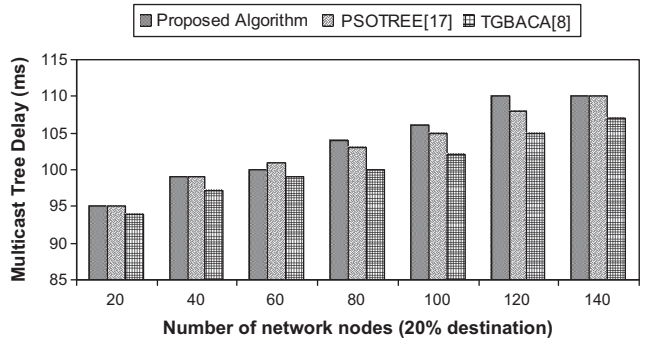


Figure 5 Multicast tree delay vs. network size with 20% nodes as destinations.

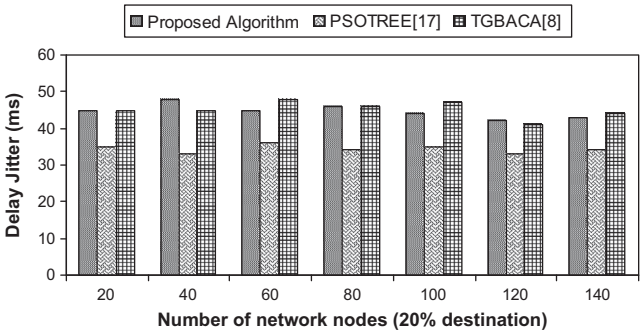


Figure 6 Multicast tree delay jitter vs. network size with 20% nodes as destinations.

the maximum distance between any two points in the network. The value of α controls the number of short links in the randomly generated network topology. The smaller the value of α, the higher is the number of shorter links. Similarly, β controls the number of links in the randomly generated network topology. The lower the value of β, the larger is the number of links. The value of α and β is set to 0.8 and 0.7, respectively, in our simulation setup. The delay, loss rate, and band width of the links are set randomly from 1 to 30, 0.0001 to 0.01, and 2 to 10 Mbps, respectively. Similarly, the cost parameter is set to be 1–100. The maximum number of iterations $Count_{max}$ is considered to be 30.

The source node is selected randomly and destination nodes are picked up uniformly from the set of nodes chosen in the network topology. The delay bound Δ, the delay jitter bound, and the loss bound are set 120 ms, 60 ms, and 0.05, respectively. The bandwidth requested by a multicast application is generated randomly. We also implemented PSOTREE [17] and TGBACA [8] algorithms in the same environment to study and compare the performance of our proposed algorithm with the existing algorithms. We generated 30 multicast trees randomly to study the performances of PSOTREE and TGBACA, and next 30 multicast trees are generated for our proposed algorithm. These 30 multicast tree patterns are arranged in a rectangular grid of size 5×6 in the order of their generation. The simulation is run for 100 times for each case, and the average of the multicast tree cost is taken as the output.

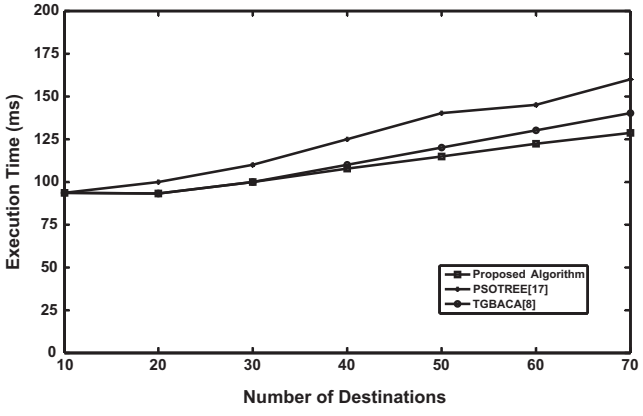


Figure 7 Comparison of execution time (ms) in different topology scales (No of nodes = 100).

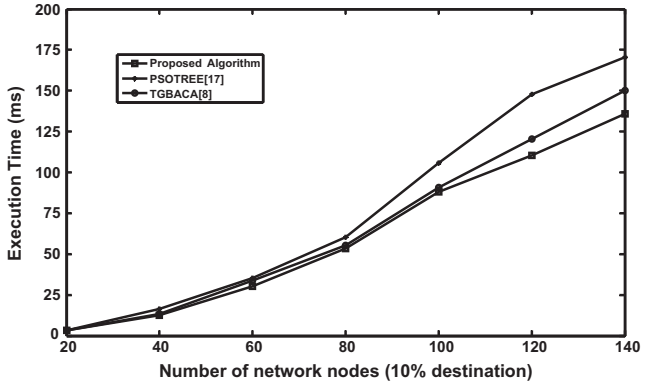


Figure 8 Comparison of execution time (ms) in different topology scales for 10% nodes as destinations.

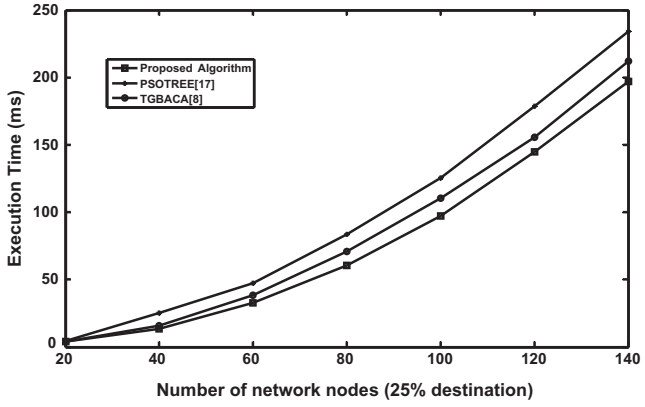


Figure 9 Comparison of execution time (ms) in different topology scales for 25% nodes as destinations.

The multicast tree cost versus multicast group size for a network of 100 nodes is shown in Fig. 2. Figs. 3 and 4 show the multicast tree cost versus the network size with 10% and 25% of the nodes as the group size, respectively. The multicast trees generated by PSOTREE, TGBACA, and our proposed algorithm satisfy the delay, delay jitter, loss rate, and bandwidth constraints. However, the results clearly reveal that the cost of the multicast tree generated by our proposed algorithm is less than the multicast trees generated by the PSOTREE and TGBACA [8]. The PSOTREE [17] algorithm constructs the multicast tree by combining the multicast trees and removing directed cycles. This algorithm in [17] removes the links that are in any of the trees, but not in both, and have minimum fitness. However, the approach [17] fails to generate a better tree, because the links deleted from the cycle may be better than the links not in the directed cycles. The TGBACA algorithm [8] follows a pheromone updating strategy to construct the best multicast tree. The algorithm updates pheromones on the links used by the global best tree and the best tree generated after each generation. Though this strategy accelerates the convergence process, the solution falls into local optimization.

However, our algorithm combines two multicast tree patterns by bringing the better attributes of one pattern to another pattern. It generates a new tree pattern after each iteration, which is better than both the patterns. Therefore, the proposed algorithm converges to a multicast tree after a few iterations. Considering the above aspects, our algorithm (HACOPSO) is found to perform better in comparison with the previous works PSOTREE [17] and TGBACA [8].

Figs. 5 and 6 show the multicast tree delay and delay jitter for a network of 100 nodes with 20% nodes as destinations, respectively. It is observed that the proposed algorithm along with PSOTREE [17] and TGBACA [8] satisfies the delay and the delay jitter constraints. The multicast tree generated by our algorithm experiences a delay and delay jitter comparable with PSOTREE [17] and TGBACA [8]. However, the multicast tree generated by our algorithm performs significantly better in terms of multicast tree cost.

The Comparison of execution time in milliseconds is shown in Figs. 7–9. Figs. 8 and 9 show the comparison of execution time for a network of nodes with 10% nodes as destinations and for 25% nodes as destinations, respectively. It can be observed that the execution time of our proposed algorithm is less than that of the existing algorithms. This is because in our algorithm when two particle agents interact, the path of one tree pattern is replaced by the better path of another tree pattern. In PSOTREE, the two trees are merged and the loops are deleted for which it takes more time. In TGBACA, the trees are constructed, and then, the final tree is constructed after tree pruning. Thus, the execution time is less than the PSOTREE and more than our proposed algorithm as shown in the Figs. 7–9.

5. Conclusions

This paper presented a swarming agent based intelligent hybrid algorithm (HACOPSO) algorithm using the concept of ACO and PSO. The performance of the proposed algorithm was evaluated through extensive simulation. The results of simulation are compared with two existing algorithms PSOTREE [17] and TGBACA [8]. Our algorithm is found to construct the multicast tree patterns more sensibly such that the tree patterns not only satisfy the QoS constraints, but also tries to minimize the tree cost. The proposed algorithm also uses the collective and coordination process for the mobile agents attached to each pattern. The final multicast trees generated by our algorithm are found better compared to the multicast trees generated by PSOTREE [17] and TGBACA [8]. The time complexity of our algorithm is also found to be comparable to the existing ones.

References

[1] Hakimi SL. Steiner problem in graphs and its implementation. Networks 1971;1:113–33.
[2] Di Caro G, Dorigo M. AntNet: a mobile agents for adaptive routing. In: Proceedings of the 31st Hawaii international conference on systems; 1998. p. 74–83.
[3] Di Caro G, Dorigo M. AntNet: distributed stigmergetic control for communications networks. J Artific Intell Res 1998;9:317–65.
[4] Dorigo M, Di Caro G. The ant colony optimization meta-heuristic, new ideas in optimization. McGraw-Hill; 1999.
[5] Sim KM, Sun WH. Ant colony optimization for routing and load balancing: survey and new directions. IEEE Trans Syst, Man, Cybernet 2003;33:560–72.
[6] Cheng H, Cao J, Wang X. A heuristic multicast algorithm to support QoS group communications in heterogeneous network. IEEE Trans Vehicul Technol 2006;55(3):831–8.

[7] Mullen R, Monekosso D, Barman S, Remagnino P. A review of ant algorithms. Exp Syst Appl 2009;36(6):9608–17.
[8] Wang H, Xu H, Yi S, Shi Z. A tree-growth based ant colony algorithm for QoS multicast routing problem. Exp Syst Appl 2011;38:11787–95.
[9] Gong B, Li L, Wang X. Multicast routing based on ant algorithm with multiple constraints; 2007.
[10] Zheng YX, Tian J, Dou WH. Vector constraint multicast routing based on GA. Chin J Comput 2003;26:746–52.
[11] Haghighat A, Faez K, Dehghan M, Mowlaei A, Ghahremani Y. GA based heuristic algorithms for bandwidth-delay-constrained least-cost multicast routing. Comput Commun 2004;27(1):111–27.
[12] Zhou J, Cao Q, Li C, Huang R. A genetic algorithm based on extended sequence and topology encoding for the multicast protocol in two-tiered WSN. Exp Syst Appl 2010;37(2):1684–95.
[13] Kennedy J, Eberhart RC. Particle swarm optimization. In: IEEE International Conference on Neural Network, Perth, Australia; 1995. p. 1942–8.
[14] Liu J, Sun J, Xu W. QoS multicast routing based on particle swarm optimization. Lect Notes Comput Sci 2006;4224:936–43.
[15] Sun J, Liu J, Xu W. QPSO-based QoS multicast routing algorithm. Lect Notes Comput Sci 2006;4247:261–8.
[16] Li C, Cao C, Li Y, Yu Y. Hybrid of genetic algorithm and particle swarm optimization for multicast QoS routing. In: IEEE international conference controls automation; 2007. p. 2355–9.
[17] Wang H, Meng X, Li S, Xu H. A tree-based particle swarm optimization for multicast routing. Comput Netw 2010;54:2775–86.
[18] Ghaboosi N, Haghighat A. Tabu search based algorithms for bandwidth delay-constrained least-cost multicast routing. Telecommun Syst 2007;34(3):147–66.
[19] Wang H, Meng X, Zhang M, Li Y. Tabu search algorithm for RP selection in PIM-SM multicast routing. Elsevier Comput Commun 2009;33:35–42.
[20] Brueckner SA, Parunak HVD. Swarming agents for distributed pattern detection and classification. Lect Notes Artific Intell 2005;3374:232–45.
[21] Meng Y. A swarm intelligence based algorithm for proteomic pattern detection of ovarian cancer. In: IEEE symposium on computational intelligence in bioinformatics and computational biology; 2006. p. 1–7.
[22] Waxman BM. Routing of multipoint connections. IEEE J Select Areas Commun 1988;6(9):1617–22.

Manoj Kumar Patel has received his M.Sc in Mathematics and MCA from Sambalpur University and IGNOU, India. He is currently pursuing Ph.D under Sambalpur University, India. He is also working as a Technical Asst. Gr-I(Computer) in the Department of Computer Science & Engineering at Veer Surendra Sai University of Technology, India. His research area includes QoS Routing, Reliable Multicast and Soft Computing Techniques. He has published about 07 research papers in various International Journals and Conferences.

Manas Ranjan Kabat has received his M.E. degree in Information Technology and Computer Engineering from Bengal Engineering College, India, and the Ph.D degree in Computer Science and Engineering from Sambalpur University, India. He is currently working as Reader and Head in the Department of Computer Science and Engineering at Veer Surendra Sai University of Technology, Odisha, India. His research involves Multicast Routing, Reliable Multicast, High Speed Computer Networks and e-Governance. He has published about 12 research papers in various International Journals and Conferences.

Chita Ranjan Tripathy is a Professor, Department of Computer Science and Engineering, Veer Surendra Sai University of Technology, Odisha, India. He received his Master Degree and Ph.D in Computer Science and Engineering from Indian Institute of Technology, Kharagpur, India. He has published more than 80 research papers in different International journals and conferences. His research area includes Computer Networks, Parallel Processing and Reliability. He has received his "Sir Thomas Ward Memorial" gold medal for researches in Parallel Processing. He is a Fellow of Institution of Engineers (India) and life member of Indian Society for Technical Education (ISTE), Instrument Society of India and Orissa Information Technology Society (OITS).