CONCURRENCY AND COMPUTATION: PRACTICE AND EXPERIENCE
Concurrency Computat.: Pract. Exper. 2015; **27**:3483–3496
Published online 17 March 2015 in Wiley Online Library (wileyonlinelibrary.com). DOI: 10.1002/cpe.3466

SPECIAL ISSUE PAPER

An improved EDA for solving Steiner tree problem

Lei Liu, Hua Wang*,† and Guohong Kong

School of Computer Science and Technology, Shandong University, Jinan, Shandong Province 250100, PR China

SUMMARY

Steiner tree problem and its derivations are widely employed to optimize the design of transportation, communication networks, biological engineering and the QoS multicast routing problem. It is one of well-defined open issues which have attracted many research efforts. Different from the existing works, this paper develops a new method of solving Steiner tree problem by using the improved estimation of distribution algorithms (EDA). Further, the performance of developed method is validated by applying on multicast routing optimization. The developed method randomly initializes *n* trees which contain the source node and the destination nodes. And some individuals select the crossover operation randomly to add the population diversity and avoid the algorithm premature convergence. The algorithm constructs a probabilistic model according to the selected elites, which is capable of estimating the probability distribution of the solution. The probabilistic model is updated according to the new population. New trees are generated based on the probabilistic model. This process iterated until designated termination criteria are met. The improved EDA algorithm gradually evolves trees to obtain a better solution. Simulation validations suggest that the developed method leads to better performance. In particular, the complexity in terms of the converging speed improves significantly compared to other algorithms. Copyright © 2015 John Wiley & Sons, Ltd.

Received 18 March 2014; Revised 1 December 2014; Accepted 10 December 2014

KEY WORDS: Steiner tree; estimation of distribution algorithm; approximation algorithm; intelligence optimization; probabilistic model

1. INTRODUCTION

In contemporary, how to discover a low cost path among the huge number of network nodes has attracted extensive attentions. It is an enabler to minimizing the transmission cost, such as energy, retransmission efficiency and throughputs, and hence improving network performance. In the open literature, researchers have dedicated to achieve the goal of finding the optimized solutions. It can be generally described as finding a minimum cost spanning tree (MST) with an edge weighted graph in polynomial time, which is a classical Steiner tree problem [1]. Steiner tree problem aims at determining subtree spanning all terminal vertices and minimizing the total cost of the obtained tree. Unfortunately, it has been proven to be an NP-Complete problem [2].

Various existing studies have been contributed to the Steiner tree problem, for the reason that, the solution for Steiner tree problem can be applied to solve many noted problems in computer science, such as the optimization of multi-cast routing. Researchers have made a great effort on solving very large scale integrated circuits design, optical and wireless communication networks [3, 4]. For instance, the Quality of Service (QoS) multicast is important for the applications subjected to end-

*Correspondence to: Hua Wang, School of Computer Science and Technology Shandong University Shunhua Road, Jinan, Shandong Province 250100, PR China.
†E-mail: Wanghua@sdu.edu.cn

to-end delay constraints, cost and throughput. Further, the provisioning of multi-constraint multicast application optimization has become an impressing demand. These problems can be eventually generalized into the Steiner tree problem [5–7]. Hence, researchers seek to optimize the performance of aforementioned cases by solving variety of Steiner tree problems.

The pioneer work by Melzak on the Steiner tree problem has been given in 1961 [8]. Later, Winter showed an improved result based on Melzak's work. So far, there are several mechanisms have been proposed for solving Steiner tree problem [9, 10]. To the best of our knowledge, the aforementioned works are not suitable for solving large-scale problems, for the reason that they took immense computing time when the number of network nodes scales. Instead, heuristic algorithms are developed to improve the time efficiency. Unfortunately, heuristic algorithms cannot obtain the best Steiner tree in most cases, although it is able to find quasi-Steiner tree in a short time close to optimum [11]. Most recently, intelligent algorithms, such as tabu search, simulated annealing, genetic algorithm, artificial neural network, colony algorithm and PSOTREE, have been employed to solve Steiner tree problem [12, 13]. For the ease of computation, researches add the specific constraints, which are not descripted in the history of Steiner tree problem [14], such as the delay and hop constraints [15, 16]. Similar assumptions can be found in other studies [17, 18].

The efficiency and accuracy using the above methods highly depend on parameter settings and coding design style. But yet, inappropriate parameter settings may lead to unexpected outcomes and hence make the algorithm fall into local optimization, which may result in unprecedented low efficiency. To this end, novel algorithms for solving Steiner tree problem, which take both accuracy and efficiency into account, have been becoming more and more urgent.

Estimation of Distribution Algorithm (EDA) is a class of stochastic optimization algorithms, which was first introduced in the year of 1996 [19]. EDA combines statistical approach and generic concepts. It not only enriched but also enhanced the class of evolutionary computation family. Furthermore, EDA method is widely adopted for its capability of optimizing large class instances in two-dimensional and three-dimensional lattices [20], multi-objective knapsack [21], and ground water remediation design [22] etc. But to the best of our knowledge, EDA has not been applied to solve the Steiner tree problem.

To fill the gap, an improved algorithm based on the Estimation of Distribution Algorithms (EDA) for solving Steiner tree problem will be introduced in this paper. At the very beginning, the algorithm randomly initializes n trees. Next, all the leaves in the tree are the destination nodes. Then special trees, namely elites, are chosen by the selection operation. A probabilistic model is constructed based on the selected elites, which are more likely featuring the probability distribution of solutions. New trees are generated by this model. Each one randomly selects the crossover operation. The probability model is updated according to the new population. The process is repeated until it meets termination criteria. The new algorithm gradually evolves trees to obtain a better solution tree. Experiments in terms of simulation topologies of undirected and directed graphs are used to validate the performance of the developed algorithm. The results show that the developed algorithm is feasible and effective. Consequently, the developed algorithm is applied on solving a practical issue, i.e. the optimization of multi-cast routing problem. The comparisons show the reduction of the convergence time without lowering the accuracy.

The rest of this paper is organized as follows. Section II defines the research history of the Steiner tree, its definition and the background of the EDA. Section III briefs the new EDA for the Steiner tree and reports in detail the steps of solving Steiner tree problem. Section IV describes the experiments result. Section V gives the summary of the research.

2. PRELIMINARIES

The research of Steiner tree problem can be dated back to the past decade and was proved to be a nondeterministic polynomial time complete (NPC) problem by Garey and the cooperate authors [2]. In general, Steiner tree problem can be described as finding the minimum cost of a number of interconnected nodes with weighted edges. Winter has introduced how the Steiner tree problem can be generalized into network issues [10]. Later, Hwang and Richards have summarized the optimal

DOI: 10.1002/cpe

algorithms and heuristic algorithms to this problem in their work [23]. In the close-form derivation algorithms, the increment of the number of nodes in the topology results in the increase of computation complexity, which is apparently not suitable for large-scale applications. As a result, heuristic algorithms are proposed in order to shrink the convergence time. However, such algorithms fail to achieve the optimal Steiner tree. In recent years, with the development and explosion of the intelligent algorithm such as the simulated annealing, artificial neural network and ant colony algorithm, researchers are trying to apply such algorithms for solving the Steiner tree problem [24–27].

For decades, researchers keep seeking new or improved algorithms to solve Steiner tree problem. This is because, with the rapid development of the information technologies, more and more realistic issues can be eventually generalized into Steiner tree problem. For instance, the optimization of performance, cost, energy consumption and path finding can be approached by solving Steiner tree problem [28, 29]. In multicast routing enabled networks, nodes and routing path can be generalized into a classic Steiner tree problem. The solution helps optimize energy consumption, utilization of the network resource and quality-of-service provided by the application service provider. Hence, in this paper, the improved EDA algorithm is finally applied on the multicast routing in order to validate the accuracy and efficiency of our work. In existing studies, researchers have discussed the importance and capability of solving Steiner tree problem in wired network and wireless network [30–32]. From the aforementioned works, it is convincible that optimization issues in networks can be generalized into Steiner tree problem. Pioneer studies impose some constraints on the edge selection in construction of the tree. To overcome the aforementioned weaknesses, there come heuristic algorithms, but the convergence speed is very slow, especially when the size of topology increases. Then, intelligent algorithms have been brought to our sight, such as the genetic algorithm [33], ant colony algorithm [30] and tabu algorithm [34]. These algorithms have been applied to different topologies with a variety of constraints. The solutions are better than those of using heuristic algorithms in terms of accuracy and feasibility. However, the complexity has not yet been lowered down. In order to improve the convergence speed of the such algorithm, our work tries to reduce the convergence time without losing the performance of solving the Steiner tree in light of the estimation of distribution algorithms (EDA). In the following, Steiner tree and EDA are formulized for the ease of understanding.

The definition of the Steiner tree is formulized as follows [35]. Given a graph $G=(V,E)$, V is the node set of graph G, E is the edge set of graph G, $|V|$ is defined as the node number of graph G and $|E|$ denotes the number of edges or links of graph G. The cost function of edge is denoted by $\cos t: E \rightarrow R$, the destination node set is defined as $D \subseteq V$, $m=|D|$. The minimal Steiner tree is defined as finding the minimal tree covering all the nodes in set D. It may contain Steiner nodes, that is, the nodes in the Steiner tree but not in the set D. If the tree has the minimum cost, then it is called a Steiner tree. Then, the tree is described as $T_s(V_T, E_T)$, where $D \subseteq V_T \subseteq V, E_T \subseteq E$.

The Steiner tree can be formalized as

$$\min \sum_{e \in E_T} \text{cost}(e) \tag{1}$$

After the formulation of Steiner tree, we will briefly introduce Estimation of distribution algorithms. EDAs are stochastic optimization algorithms that explore the space of potential solutions by building and updating probabilistic models according to the promising candidate solutions. This model-based method has allowed EDA to solve complex combinatorial optimization problems [36, 37].

EDA is originated from and inspired by the genetic algorithm (GA). It is not flawless because of its disadvantage of premature convergence similar to the genetic algorithms despite its capability. On the other hand, the evolutionary mechanism of EDA is different from GA. In general, GA simulates the biological evolution on the view of the micro-level, but EDA simulates the biological evolution at the macro level to build and the sample according to the probability model. This novel evolutionary mechanism makes EDA have unique properties. EDA has been successfully applied to optimization of many instances. This new optimization method can usually make it find the best solution and has higher efficiency.

According to the complexity of the probabilistic model and the different sampling methods, EDA has developed many different specific implementation methods, which can be summarized as the

DOI: 10.1002/cpe

following three main steps. Start with a random population of vector *N*, and implement an iter*ation consisting of*

(1) Select *M* vectors using a selection method.
(2) Build a probabilistic model describing the solution space. According to the fitness function to assessment of the population, a promising collection of elites are selected, and then a probabilistic model is constructed for describing the current solution by means of statistical learning method.
(3) Generate new populations by the probabilistic model. A new vector *N* is generated according to the probability model.

There are different ways for sampling, but truncation selection is of most popular. There are also improved probabilistic models such as UMDA [38] and PBIL (Population Based Incremental Learning) [19], which treat each variable independently. In this paper, the probability model is based on the PBIL.

3. THE NEW IMPROVED ESTIMATION OF DISTRIBUTION ALGORITHM FOR THE STEINER TREE

EDA is widely known as the probabilistic model. It combines the statistical learning theory with the principle of genetic algorithm. Through the construction of the probability model, sampling and update the probability model to evolve the population. A probability distribution describes the trend of the population. In genetic algorithm, the combination and mutation operations are used to generate the new solution. However, the new candidates are generated by sampling the probability distribution in EDA, and EDA is easy to solve the nonlinear variable problems. So the convergence speed is faster than the other intelligent algorithms without degrading the performance. This is the reason why EDA has attracted many research efforts [39–41]. Despite EDA's short convergence time, it has some shortcomings, such as, during the evolutionary process, the individuals are easy to move to the same solution, so the algorithm stagnates in local optimum and decreases the population diversity. These shortcomings affect the performance of the EDA seriously. In order to add the diversity of the population, we developed a new improved EDA algorithm for Steiner tree problem.

In order to avoid the EDA falling into local optimum, each individual selects the random crossover operation after finding the solution in the new algorithm, which is good for the algorithm avoiding premature convergence and adds the diversity of the population. Each one finds the solution starts at the root node, selects one link and adds it to the current tree comprehensive considerate the edge used frequency and the cost value. Each step of the adding edge operation expands the tree rather than path merger, which is similar to that of ant colony intelligent algorithm [6]. The difference between the developed algorithm and ant colony intelligent algorithm is the method of selection operation of roulette. The former uses the frequency and the cost value of the edge; the latter relies on the pheromones. When the current solution covers all the nodes in the set D, it stops growing. The basic idea is that a population of *n* individuals is initialized and each one has a solution tree. Prune operation is executed for every solution to remove the leaf nodes which are not the group members and add the cost value of the solution. The trees are then scored according to the estimation function. This function gives a numerical ranking for each tree, with the higher the number the better the tree. In the first iteration, we select the current global optimization solution. In the following iterations, each individual selects randomly the crossover operation with the current global optimization or the current local optimization solution. This operation is good for maintaining population diversity and can avoid the population premature convergence to some extent. According to the estimation value of the trees, elite trees whose values are lower than the others are selected. The algorithm then calculates the probability of each edge according to the elites and then constructs a probabilistic model which attempts to estimate the probability distribution of the selected trees. The probabilistic calculation is according to the frequency of the edge used which means that the edge used more often has the higher probability of being chosen for constructing the tree. While we select the edges to construct the tree, we not only consider the frequency of the edge but also the weight of the edge. This is good for finding the best minimum spanning tree. At last the solution would locate the global optimum or

DOI: 10.1002/cpe

its accurate approximation. Once the model is constructed, new solutions are generated by this model, and the probabilistic model is updated by the new elites. The method mentioned above is repeated until the algorithm convergence or there is not a big change of the solution.

In summary, the implementation of improved new EDA algorithm consists of five steps: (1) generate the initial population, (2) random select the crossover operation, (3) *M*(*elite*) individuals are selected from the population, (4) probabilistic model is introduced according to the elites and (5) find the new population N according to the probabilistic model.

New EDA pseudo code of tree optimization is given as follows.

```
procedure new_EDA_STEINER_TREE
      input c //maximize iteration number
       input elite //selection elite number
      converge = FALSE
  Initialization:
        for i = 0 to n do
         Population_Init (root, edge)
        end_for
         p_g = Calculate_Tree_Fitness(stree)
         h[elite] = Find_Elite(solution)
         for i = 0 to elite-1 do
           Initial_P(p)
         end_for
     for i = 0 to c do
  Sampling:
   for j =0 to n do
          Population_Init (root, edge)
          totalcost = Steiner_TreeCost(tmptree)
       if totalcost < Steiner_TreeCost(p_lg)
            p_lg = tmptree
       end_if
   end_for

   for j = 0 to n do
     r = random()
      Initial(tmptree) &b_k;/&e_k;/tmptree is the result of the crossover operation
     switch(r)
     case 1:totalcost = crossover(solution[i], p_lg)
     case 2:totalcost = crossover(solution[i], p_g)
     case 3:skip the crossover operation
       if totalcost < Steiner_TreeCost(p_g)
          p_g = tmptree
       end-if
     end_for
  Selection:
     h[elite] = Find_Elite(solution)
  Modeling:
     Update_P(p)
     if converge = TRUE
        break
     end_if
   end_for
  print besttree
end_procedure
```


DOI: 10.1002/cpe

In the above algorithm, for the first step, we use the *Population_Init* function to initialize the population of n individuals. The variable of *root* is the source node, and *edge* is used to construct the Steiner tree. The function of *Calculate_Tree_Fitness* is used to calculate the cost value of the Steiner tree and get the current global optimal solution saved in the variable of p_g. The number of *elite* individuals is selected by the *Find_Elite* function from the *solution* which is the n individuals. *Initial_P* function is used to construct the probability model according to the elites. In the iterations, we use the *Steiner_TreeCost* function to calculate the cost value of the current tree *temptree* saved in the variable of *totalcost*. The current local optimal solution is updated in the iterations according to the *totalcost*. For the n individuals, each one selects the crossover operation or not according to the random number r. And the probability is updated by the *Update_P* function. At the end of the algorithm, the *besttree* of the solution is returned.

Space complexity analysis of the process of initializing the random Steiner tree: Suppose n individuals are initialized. There are $|V|$ nodes and $|E|$ edges. Each one holds an adjacent matrix to store the solution. So this space needs $O(n|V|^2)$. The maximum limit of solution construction needs to traverse all the nodes, hence the time complexity of n individuals to construct the solution is $O(n|V|\times|E|)$. To construct the probabilistic model it is needed to count the edge used frequency by traveling all the individuals, so the time complexity is $O(elite|V|^2)$. *elite* is smaller than the n. Above all, the time complexity of the iteration process cannot be over $O(cn|V|^2)$. c represents the maximum of the iteration number. So the time complexity is $O(n|V|^2)$.

In this algorithm, the solution trees are initialized randomly at the beginning, and the one with minimum cost is selected as the current optimal tree. The operation of removing circles, pruning useless node and crossover operation randomly makes the result trees lower cost. In construction of each tree, the current subtree selects an edge by means of roulette-wheel function considering the probability of the edge. The edge with higher probability has greater possibility to be selected and kept in the tree. Once this construction operation is completed, we could sample this distribution according to the selected elites. Ideally, the repeated refinement of the probabilistic model, which is based on representative samples of high quality solutions, would keep increasing the probability of generating the global optimum. After a reasonable number of iterations, the procedure will locate the global optimum or its accurate approximation.

The algorithm has good performance through multiple iterations and random guarantee of initialized tree. In the following sections, the steps of algorithm operation, including initializing the trees, crossover operation, finding the elites and calculating the probability, will be introduced.

3.1. Initializing EDA trees

Each individual keeps a solution tree. If the number of network nodes is $|V|$, then the individual keeps a $|V|\times|V|$ matrix x, where $x[i][j]=1$ means that the edge from node i to j is selected to the current subtree; otherwise the edge is not selected.

The algorithm initializes the tree randomly. Set the node set in the source root tree $S=\{s\}$, where s is the source root node of solution tree. Then randomly select node i which is in the neighboring domain of nodes in the set S. Then add node i to the set S until all destination nodes in the set D belong to the set S. The selection operation is according to the probability of the edge which is in the vertexes of the set S neighboring edges but not in the current tree by roulette. And the selection weight of every edge we set as the following equation:

$$w_{ij} = \alpha p_{ij} + \beta \frac{1}{\mathrm{cost}(\mathrm{e}_{ij})} \tag{2}$$

where p_{ij} is the used frequency probability of edge e_{ij}. α and β control the edge used time and cost value $\mathrm{cost}(\mathrm{e}_{ij})$ which influence the roulette selection. Assume that the current node is s and the next step of selection one node by roulette selection which means that all the nodes adjacent to the node s but the edge not in the set S are chosen by the weight probability. The edge with more weight has the higher probability of being chosen. The selection probability is as follows:

$$S_{si} = \frac{w_{si}}{\sum_{e\in Ne} w_e} \tag{3}$$

DOI: 10.1002/cpe

where w_{si} is the weight of the nodes s and i, and N_e is the candidate edges set which contains all the adjacent edges with vertexes in the set S but not include the edges in the current tree. w_e is the weight of the edge e from the N_e.

The Steiner tree is constructed following the steps shown in the Figure 1. The node 2 is the source node, and the set D has four members 0,3,4,6 nodes.

Figure 1 is an illustration of the edge selection strategy and the weight on the edge is randomly set. Figure 1(a) shows the tree starting at the root node 2. Figure 1(b) is the tree growth after selecting

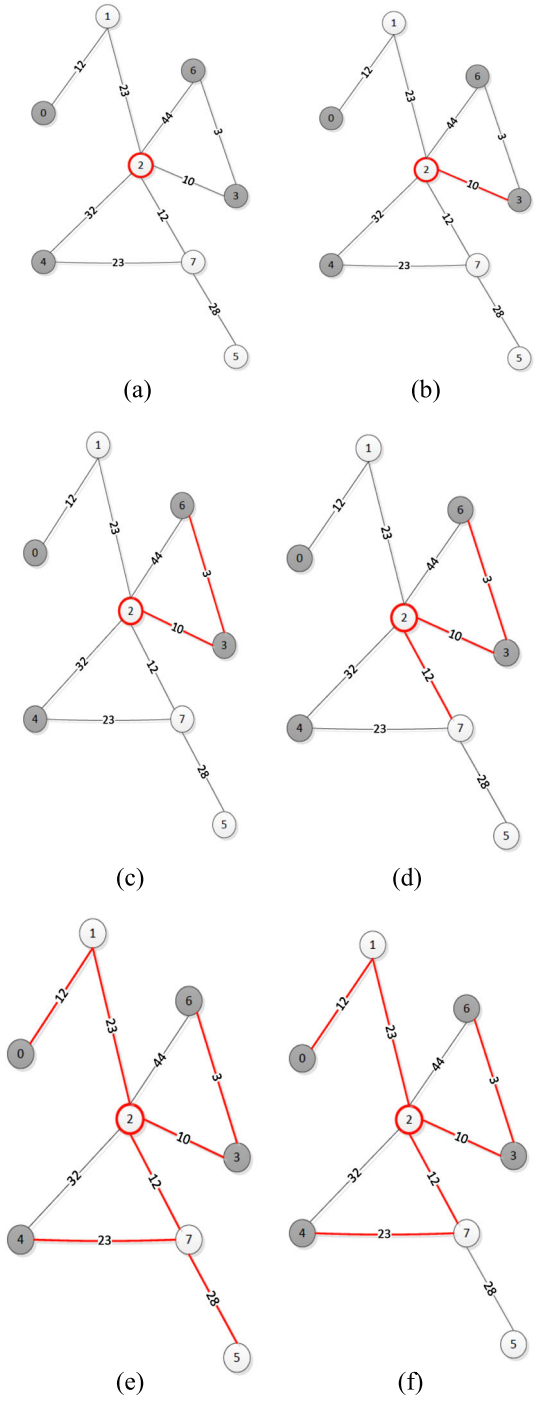

Figure 1. Illustration of the new EDA Initialization.

DOI: 10.1002/cpe

randomly the node adjacent to set S and meanwhile not in set S according to the weight of the edges by roulette operation the same as the Figure 1(c). In the next selection operation, the edge between 2 and 6 vertexes is not included in the candidate edges, because the edge will make a circle, the result just like in Figure 1(d).

The result is shown in Figure 1(e), but it contains the node not in the set D. So it needs to prune the leaf nodes which are not concluded in the destination node set, the result like in Figure 1(f). If the edge added to the current tree makes the subtree have a cycle, we will cancel the add operation. And each leaf node is examined during the prune operation, if it is not a destination node, and there is an edge e_{mi}, where the node m is the parent of node i, then remove edge e_{mi} from the tree. Do the same operation to the parent node m of i until the node m is either the destination node or its out degree is larger than 0. Through the mentioned steps a tree is generated.

3.2. Crossover operation

Our crossover operation is based on the tree. We directly optimize the tree rather than the path. There are three steps: merging the multicast tree, eliminating circles and pruning non-destination member leaf nodes.

We employ that T_c is the current solution, and T_g and T_l are the global optimum tree and local optimum tree, respectively. Each individual keeps a $|V|\times|V|$ bi-dimensional array. T_c selects the merging object randomly. T_{temp} is the result of the merging operation. It needs elimination and prune operation [42]. In the elimination operation, we select a simple method eliminating the maximum cost value in the circle. And the prune operation is an iteration starting at one leaf node until the node is set D member or its out degree is not zero. The time complexity of the merging operation is as follows. There are n individuals, and we assume that each one should select the merging operation. So n iterations are needed, and the time complexity is $O(n|V|^2)$. The next step, a DFS, is used to eliminate the circle, so the time complexity is less than the $O(n|V|^2)$. The maximum time complexity of the last prune operation is $O(n|V|^2)$. Then, the time complexity of the crossover process can be limited to $O(n|V|^2)$.

3.3. Finding the elites and probabilistic model

According to improved EDA algorithm, Elite individuals will be selected for building the probabilistic model. In this work, M elites are selected as sample individuals which have the lower estimation value as in the N initial trees. PBIL [19] probabilistic model was applied in this paper. The probabilistic matrix $p=(p_{ij})$ $(0<i,\ j<V)$ represents the probability model, where p_{ij} denotes the probability of selecting the edge e_{ij} or e_{ji}, and it is initialized as $p_{ij}=\frac{1}{|E|}$; every edge has the same probabilistic to be chosen for constructing the solution expanding the search scope to some extent. In the $t+1$ iteration, the matrix is updated as the following equation:

$$p_{ij}(t+1)=\rho * p_{ij}(t)+(1-\rho)*\left(1+\frac{1}{N}\sum_{l=1}^{N}x_{ij}^{l}\right), 1\le i,j\le V \tag{4}$$

where

$$x_{ij}^{l}=\begin{cases}1 & ,\forall e_{ij} or e_{ji}\in l\\ 0, & \text{otherwise}\end{cases} \tag{5}$$

ρ controls old probabilistic on how to affect the new selection. The larger of the ρ value indicates more contribution of the information learned from the history. Through extensive experiments we set $\rho=0.8$.

DOI: 10.1002/cpe

4. EXPERIMENTAL RESULTS

Owing to the development of network technology, multimedia applications have flooded the communication networks and expanded in all science and engineering domains, such as the broadcasting, video conference and distance education. These applications have high demands on the bandwidth, latency and energy consumption. To this end, multicast is considered to be an effective solution. In this part, the improved EDA is applied on an application, i.e. multicast routing optimization. The results validated the efficiency and accuracy of the developed algorithm.

The improved EDA was programmed with VC6.0 and run on the machine Intel Pentium® 4 3.00 GHz, 4G memory in the operating system of Windows 7.

The improved EDA algorithm is validated through four scenarios in terms of four aspects in the simulation process, which include the influence of initial population size on search effect, the effect of destination node scale on convergence result, the parameter setting and the comparison among algorithm performance. Two network topologies are employed in order to test the feasibility of the work in this paper. The first network topology used follows certain power law rule. The other one is the random topology. Inet 3.0 [37] adopts PLRG (Power-Law Random Graphs) algorithm and priority attachment. In this paper, B kinds of topologies in SteinLab Repository [39] are used. Figure 2 is the topology graph with 50 nodes generated via Inet 3.0.

4.1. The influence of the initial population size

The size of initial population is critical for the statistical analysis of the algorithm. The bigger the size, the more accurate the probability distribution; however, it needs more time to converge. It is a trade-off. To this end, a proper size of the initial population may enable the algorithm to obtain optimized cost with little expense and suitable for finding the probability model to describe the probability distribution. A lot of simulation experiments have been carried out in order to get proper size.

In the first scenario, the network has 50 nodes following the power-law topology. Two groups of results have been given. Figure 3 shows how the Steiner tree cost has been affected by the initial population size increment. On the other hand, Figure 4 depicts significant convergence time increment with the growth of the initial population size. The truncation selection operation is used with threshold $\tau = 50\%$, which means 50% best solutions as the elites individuals are selected. From the test of this scenario, it is readily seen that the Steiner tree cost indeed lowered down with large population size. As aforementioned, the convergence time increases when the population size grows. The burst in Figure 3 is caused by the random selection of the elites. For the scenario of 50 node network, we can compare the tree cost and the corresponding convergence time by using the developed the method. It is not hard to see that the initial population of 160 is the optimized results.

Figures 5 and 6 are based on the topologies generated via Inet 3.0. The improved EDA algorithm is marked as new-EDA. The results by using PSOTREE algorithm are obtained from Zhou's work [35], which is widely recognized compared to other algorithms in solving multicast routing optimization. It is worth noting that the developed algorithm is applied on solving multicast routing optimization in

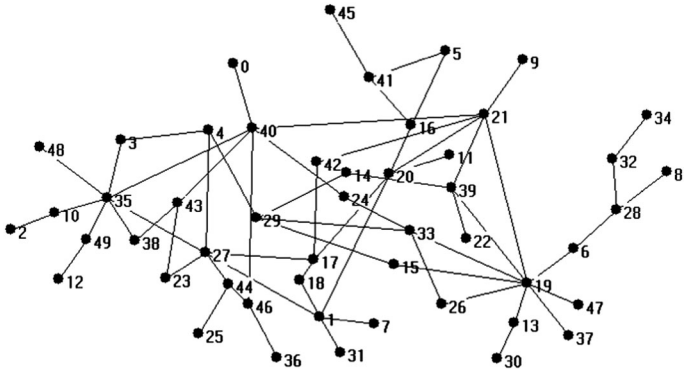

Figure 2. Network topology with 50 nodes generated via Inet 3.0.

DOI: 10.1002/cpe

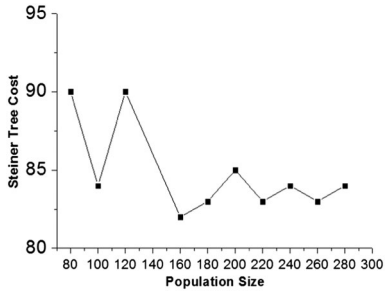

Figure 3. Steiner tree cost with different initial population sizes.

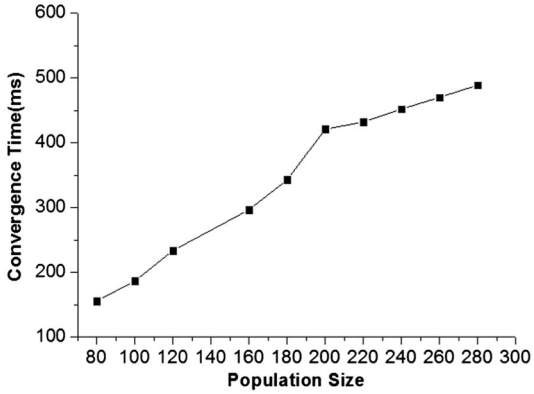

Figure 4. Convergence time with different initial population sizes.

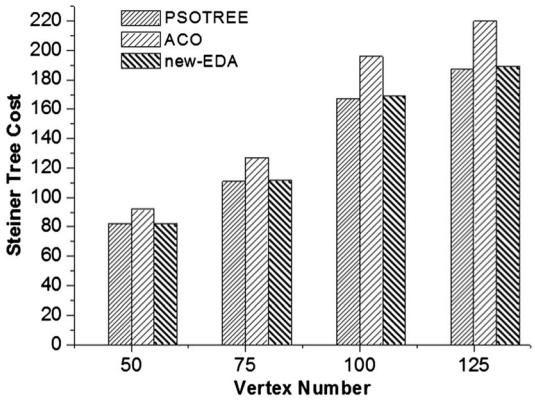

Figure 5. Comparison of Steiner tree cost in different scales.

order to compare the results with Zhou's work, and the following ant colony optimization (ACO) algorithm. The topology nodes are from 50 to 100, and the number of the group member is 15% of the topology nodes. As shown in Figure 5, ACO, which is an intelligence random search algorithm proposed by Dorigo [43]. According to the ants colony optimization algorithm, a large number of parameters are required to retrieve positive feedback of pheromone. Inappropriate parameter settings may lead to the unexpected way. Figure 5 suggests that the Steiner tree cost by employing new-EDA is close to those of PSOTREE algorithm [42] on all the test topologies, and it is much better than those of ACO algorithm. On the other hand, the convergence time of adopting new-EDA is faster than the other two algorithms shown in Figure 6 especially with the increment of network size. With the increase of the iterations, the probability distribution is stabilized, and the solution

DOI: 10.1002/cpe

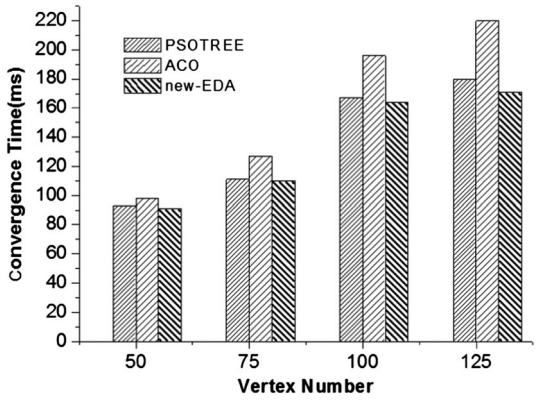

Figure 6. Comparison of convergence time in different scales.

approaches to the optimal. This is because the new EDA algorithm enables the individual to select the crossover operation randomly, which make the algorithm get the solution with less convergence time.

4.2. *Optimizing the parameters via orthogonal experiment*

How to select a new link and add it to the current solution plays an important role in the developed algorithm. In Equation 2, when α goes large, mean edge used more frequency probability which dominated the link selection. Otherwise, the link cost value has a greater impact when β goes large enough. The orthogonal experimental design has already been used in many fields. This method makes use of the mathematical statistics which selects representative experimental points from quantities of experiments.

The topology of 50 nodes was used in this experiment. The group member was set to take up to 15% of the total number of topology nodes, and all the group members are selected randomly. We selected part of the parameter values out of α and β to conduct the experiment. In Table I, taking the cost value and time into account, we set $\alpha = 0.9$ and $\beta = 0.1$.

4.3. *The effect of the destination node scale on convergence speed*

The topology generated every two nodes i and j which have the probability $p(i,j) = \alpha 1 \times \exp\left[-\frac{l(i,j)}{\beta 1 * \max l}\right]$ to be connected. $l(i,j)$ represents the Euler distance between the node i and j, and $\max l$ is the maximum value of $l(i,j)$. $\alpha 1$ controls the number of the total links, and $\beta 1$ controls the number of short links. We use the two kinds of topologies which is good for testing the algorithm in many aspects.

Figures 7 and 8 illustrate the solution cost and iteration time with different number of destination node in the topology of 100 nodes. It is obvious that the more the destination nodes the higher the cost of the Steiner tree as shown in Figure 7. With the increase of the number of destination nodes, iteration of the algorithm does not rise according to a linear way. Owing to the more destination nodes, the fewer the candidate Steiner trees cover all the destination nodes, and the easier the algorithm converges. However, the time increases significantly to find a candidate Steiner tree covering all the destination nodes. The iteration does not increase continuously. Figure 8 shows that

Table I. orthogonal experiments of the algorithm parameters.

Cost/time	$\beta = 0.9$	0.8	0.5	0.1
$\alpha = 0.1$	90/297	83/296	84/297	86/297
0.2	84/281	84/281	89/296	83/297
0.5	83/297	83/281	83/280	84/280
0.9	84/296	88/281	83/312	82/297

DOI: 10.1002/cpe

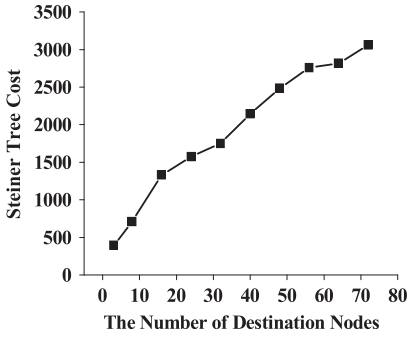

Figure 7. Comparison of tree cost with different node scales.

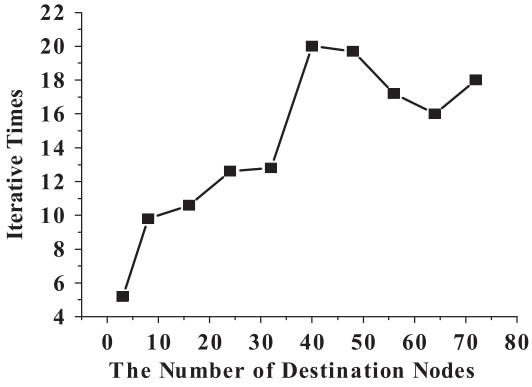

Figure 8. Comparison of iteration with different node scales.

the number of iterations reaches its maximum value of 20 when the number of destination is 50, i.e. 50% of the total node number.

4.4. The comparison with other algorithms on the new topologies

We still used the multicast routing to test the algorithm. The destination nodes are set to be 15% of the total nodes. We compared the developed algorithm with PSOTREE and ACO. Figure 9 shows that with the increment of the topology, the Steiner tree cost also increased. That is because there are more nodes in the destination which should be included in the solution. And the results of the other two intelligent algorithms are better than those of ACO. Both the PSOTREE and the new EDA use the algorithm based on the tree operation which is suitable for finding the optimum solution and

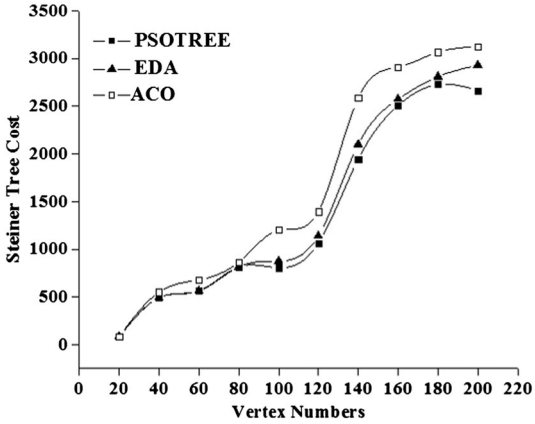

Figure 9. Comparison of cost value in different scales.

DOI: 10.1002/cpe

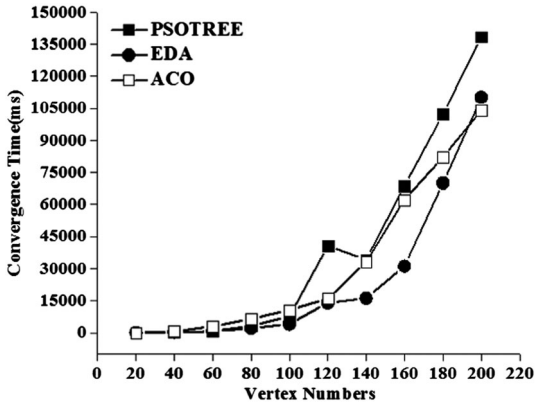

Figure 10. Comparison of convergence time in directed graphs in different scales.

have the advantage of global searching. Since ACO needs more iteration to find the best solution, the convergence time increases. On the other hand, it is readily seen in Figure 10 that the new EDA needs less time than other algorithms. The reason is that new EDA uses the mathematical statistics to find the edges which used most frequency to construct the solution. In our new algorithm, the edge selection operation, we consider not only the edge used frequency, but also the cost value, which is more suitable of finding the best solution, thus shrinking the convergence time.

5. CONCLUSION

This paper has developed an improved EDA algorithm for Steiner tree problem. The algorithm is inspired by the tree shape change via learning from the probabilistic model to solve Steiner tree of multicast routing problem. The developed algorithm is applied on an application of Steiner tree problem, namely, multicast routing. We compared with PSOTREE and ACO algorithms which are used to solve the multicast problem. Simulation experiments show that the results are very much close to the expectation especially the PSOTREE which performs better than the other algorithms to solve the multicast problem. In addition, the developed algorithm exhibits better performance in searching and converging speed. Most importantly, the computing complexity did not increase significantly with the growth of the network scale. Hence, it is more applicable to large-scale topology and provides an efficient method for the multicast routing problem.

ACKNOWLEDGEMENT

This work was partially supported by NSFC61402262 and ZR2013FQ013. We would like to show our appreciation to the reviewers for their insights and the editor-in-chief for his prompt reply and efficient work.

REFERENCES

1. Kuhn HW. Steiner's problem revisited. *Studies in Optimization* 1974; **10**:52–70.
2. Garey MR, Graham RL, Johnson DS. The complexity of computing Steiner minimal trees. *SIAM Journal on Applied Mathematics* 1977; **32**(4):835–859.
3. Al-Karaki JN, Kamal AE. Routing techniques in wireless sensor networks: a survey. *Wireless Communications, IEEE* 2004; **11**(6):6–28.
4. Rajagopalan R, Varshney PK. Data aggregation techniques in sensor networks: a survey. 2006.
5. WangC-F JRH. A factoring approach for the Steiner tree problem in undirected networks. *Information Sciences* 2007; **177**(12):2418–2435.
6. Wang H, Shi Z, Ge A, Yu C. An optimized ant colony algorithm based on the gradual changing orientation factor for multi-constraint QoS routing. *Computer Communications* 2009; **32**(4):586–593.
7. Wang H, Shi Z, Li S. Multicast routing for delay variation bound using a modified ant colony algorithm. *Journal of Network and Computer Applications* 2009; **32**(1):258–272.
8. Edmonds J. Optimum branchings. *Journal Of Research Of The National Bureau Of Standards Section B-Mathematical Sciences* 1967; **4**:233.

DOI: 10.1002/cpe

9. Du D-Z, Lu B, Ngo H, Pardalos PM. Steiner tree problems. *Encyclopedia of Optimization*, Springer US, 2001; **5**:227–290.
10. Winter P. Steiner problem in networks: a survey. *Networks* 1987; **17**(2):129–167.
11. Voß S. Steiner's problem in graphs: heuristic methods. *Discrete Applied Mathematics* 1992; **40**(1):45–72.
12. Tiande YWG. An ant colony optimization algorithms for the minimum steiner tree problem and its convergence proof [J]. *Acta Mathematicae Applicatae Sinica* 2006; **2**:016.
13. Chakraverty S, Batra A, Rathi A. Directed convergence heuristic: a fast & novel approach to steiner tree construction. In proc. 2006 IFIP International Conference on Very Large Scale Integration, 2006; 255–260.
14. Brazil M, Graham RL, Thomas DA, Zachariasen M. On the history of the Euclidean Steiner tree problem. *Archive for History of Exact Sciences* 2014; **68**(3):327–354.
15. Fu Z-H, Hao J-K. Breakout local search for the Steiner tree problem with revenue, budget and hop constraints. *European Journal of Operational Research* 2014; **232**(1):209–220.
16. Yang C, Zhao X. A new delay-constrained multicast routing algorithm based on shared edges. *Communications & Network* 2014; **6**(1):43–47.
17. Li Z, Xiao W. Nearly optimal solution for restricted Euclidean bottleneck steiner tree problem. *Journal of Networks* 2014; **9**(4):1000–1004.
18. Li Z, Xiao W. Fast approximation algorithm for restricted euclidean bottleneck steiner tree problem. *Journal of Multimedia* 2014; **9**(4):522–526.
19. Baluja S. Population-based incremental learning. a method for integrating genetic search based function optimization and competitive learning, DTIC Document, 1994.
20. Pelikan M, Hartmann AK. Searching for ground states of Ising spin glasses with hierarchical BOA and cluster exact approximation. In *Scalable Optimization via Probabilistic Modeling*. Springer: Berlin Heidelberg, 2006; 333–349.
21. Shah R, Reed P. Comparative analysis of multiobjective evolutionary algorithms for random and correlated instances of multiobjective d-dimensional knapsack problems. *European Journal of Operational Research* 2011; **211**(3):466–479.
22. Hayes MS, Minsker BS. *Evaluation of advanced genetic algorithms applied to groundwater remediation design*. University of Illinois at Urbana-Champaign: Champaign, Illinois, 2005.
23. Hwang F, Richards DS. Steiner tree problem. *Networks* 1992; **22**(1):55–89.
24. Gendreau M, Larochelle JF, Sansò B. A tabu search heuristic for the Steiner tree problem. *Networks* 1999; **34**(2):162–172.
25. Dowsland KA. Hill-climbing, simulated annealing and the Steiner problem in graphs. *Engineering Optimization* 1991; **17**(1–2):91–107.
26. Hesser J, Männer R, Stucky O. On Steiner trees and genetic algorithms. In *Parallelism, Learning, Evolution*. Springer: Berlin Heidelberg, 1991; 509–525.
27. Biniaz A, Maheshwari A, Smid M. An optimal algorithm for the Euclidean bottleneck full Steiner tree problem. *Computational Geometry* 2014; **47**(3):377–380.
28. Althaus E, Polzin T, Daneshmand SV. *Improving linear programming approaches for the Steiner tree problem*. Springer: Berlin Heidelberg, 2003.
29. Koch T, Martin A. Solving Steiner tree problems in graphs to optimality. *Networks* 1998; **32**(3):207–232.
30. Cheng H, Cao J, Wang X. A heuristic multicast algorithm to support QoS group communications in heterogeneous network. *Vehicular Technology, IEEE Transactions on* 2006; **55**(3):831–838.
31. Chiang T-C, Tai C-F, Hou T-W. Adaptive two-way uniform partition for multicast routing problem with separate paths in ad hoc networks. *Expert Systems with Applications* 2009; **36**(1):959–969.
32. Papageorgiou C, Christodoulopoulos K, Doulamis N, Varvarigos E. On least expected transmissions multicasting in wireless networks. In proc. International Conference on Computing, Networking and Communications, 2014; 836–841.
33. Haghighat AT, Faez K, Dehghan M, Mowlaei A, Ghahremani Y. GA-based heuristic algorithms for bandwidth-delay-constrained least-cost multicast routing. *Computer Communications* 2004; **27**(1):111–127.
34. Ghaboosi N, Haghighat AT. Tabu search based algorithms for bandwidth-delay-constrained least-cost multicast routing. *Telecommunication Systems* 2007; **34**(3–4):147–166.
35. Zhou BD. *Steiner Tree Optimization in Multicast Routing*. M.S. Thesis, University of Guelph(Canada), 2002.
36. Pelikan M, Sastry K, Cantú-Paz E. *Scalable optimization via probabilistic modeling*. Springer: Berlin Heidelberg, 2006.
37. Gao S, Qiu L, Cao C. Estimation of distribution algorithms for knapsack problem. *Journal of Software* 2014; **9**(1):104–110.
38. Mühlenbein H. The equation for response to selection and its use for prediction. *Evolutionary Computation* 1997; **5**(3):303–346.
39. Jin C, Jin S-W. Software reliability prediction model based on support vector regression with improved estimation of distribution algorithms. *Applied Soft Computing* 2014; **15**:113–120.
40. Wan Z, Mao L, Wang G. Estimation of distribution algorithm for a class of nonlinear bilevel programming problems. *Information Sciences* 2014; **256**:184–196.
41. Zhang Q, Muhlenbein H. On the convergence of a class of estimation of distribution algorithms. *Evolutionary Computation, IEEE Transactions on* 2004; **8**(2):127–136.
42. Wang H, Meng X, Li S, Xu H. A tree-based particle swarm optimization for multicast routing. *Computer Networks* 2010; **54**(15):2775–2786.
43. Colorni A, Dorigo M, Maniezzo V. Distributed optimization by ant colonies. Toward a Practice of Autonomous Systems, MIT Press/Bradford Books, 1991; 134–142.

DOI: 10.1002/cpe