Degree-Constrained Minimum Spanning Tree Problem Using Genetic Algorithm

Keke Liu, Zhenxiang Chen, Ajith Abraham*, Wenjie Cao and Shan Jing
Shandong Provincial Key Laboratory of Network Based Intelligent Computing
University of Jinan, Jinan, P. R. China
*Machine Intelligence Research Labs (MIR Labs), WA, USA
*IT For Innovations, VSB - Technical University of Ostrava, Czech Republic
E-mail: dp_liukk@ujn.edu.cn, czx@ujn.edu.cn, ajith.abraham@ieee.org, 492282716@qq.com, jingshan@ujn.edu.cn

***Abstract*—Computer network technology has been growing explosively and the multicast technology has become a hot Internet research topic. The main goal of multicast routing algorithm is seeking a minimum cost multicast tree in a given network, also known as the Steiner tree problem, which is a classical NP-Complete problem. We measure the multicast capability of each node through the degree-constraint for each node and discuss the problem of multicast in the case of degree-constraint, which has an important significance in the communication network. Limiting the capacity of each node during the replication process of information transmission can improve the speed of the network, which has an important significance in real-time service. In this paper, we solve constrained multicast routing algorithm based on genetic algorithm. The idea is to simulate the Darwinian theory of biological evolution. At the same time, we improve the generating random tree and replace the variation by the combination of the two variations. On one hand, we improve the efficiency of generating random tree and on the other hand, we can control the mutation of different variations in a more flexible manner.**

I. INTRODUCTION

Multicast is an effective mechanism supporting for multi-point communication that a communication form a sender to multiple recipients. In multicast, you can reduce network traffic which multiple receivers at the same time received through send one single flow of data to multiple receivers. Multicast routing algorithm is to discuss how to find a multicast tree an in a relatively short period of time and spending a relatively small resources and bandwidth which connected to the source node and destination node. Data flows along the tree path and copied just at the bifurcation of the tree and the bandwidth can be shared in the common part of the path, thus saving network resources. Traditional multicast routing problem assumes that each node has a multicast function, then to seek a minimal cost multicast tree that contains the multicast nodes. While in the actual network, the capability of multicast node is not the case. Many nodes do not support multicast and the ability of nodes to copy the information is limited. Some nodes only store and forward functions without copy function. Considering generality, we define the multicast capability of the network node as degree-constraint. We measure the multicast capability of each node through the degree-constraint for each node and discuss the problem of multicast in the case of degree-constraint, which has an important significance in the communication network. Limiting the capacity of each node during the replication process of information transmission can improve the speed of the network, which has an important significance in real-time service. Degree-constrained [1] multicast problem can be described as a degree-constraint Steiner tree problem, which is also a NP-Complete problem.

When compared with the unconstrained Steiner tree problem, little research has done on the degree-constrained Steiner tree problem. Several heuristic algorithms were proposed for this problem: the degree-constrained shortest path heuristic SPH [2] and the degree-constrained kruskal shortest path heuristic algorithm K.SPH [3]. The first, we study genetic algorithm [4-6] in-depth and grasp the basic idea of genetic algorithms and problem-solving steps. Finally, we design and realize degree-constrained multicast routing algorithm based on genetic algorithm. The algorithm is based on the genetic evolution of thinking, simulating Darwin's biological theory of evolution. The algorithm uses the node matrix encoding a multicast routing tree. The encoding is simple and easy to implement using two-dimensional array. In this paper, we initialize the population of individuals through random depth-first algorithm. Using crossover and mutation, parent individuals produce offspring, and then the excellent individuals are selected to form offspring's according to the survival of the fittest (for the next generation). In order to ensure convergence of the algorithm, we not only adopted proportional selection operator but also retain the best individual strategy. At the same time, we have improved the random tree generation algorithm and mutation operation. When we execute the random tree generation algorithm, we often encounter invalid node, for which the degree is invalid or constitute a loop. We record the node to avoid search invalid node again when executing random tree generation next time. In the mutation operation, we have abandoned the original variation, instead of using a combination of local variations and global variations. In this way, we can adjust the variation of parameters to obtain the desired variation effects easily. The coding of the algorithm is simple and solved the difficulty of encoding and decoding of multicast tree problem based on genetic algorithm effective and can obtain suboptimal solutions quickly.

The network model can be regarded as a connected undirected graph $G(V, E)$. Where V is the set of network nodes, E is the set of network links. Let size=$|V|$ be the number of network nodes, $|E|$ the number of the network links. Edge between node u and v is defined as Edge (u, v).

978-1-4673-4768-6

We define the cost parameters: C (e), refers to all the costs of the edges in the tree (here, the cost is a generalized definition, it can be measured by distance, channel bandwidth, average traffic, communication the overhead, the average queue length, delay and other factors). We define the node degree constraint: Degcon (v). We define Deg (v) as the degree of a node in the current multicast tree. Multicast tree T(s,M) is a generated sub-tree of the graph $G(V, E)$. This subtree covers the source node s $\in$ V and all the destination nodes M. We define M as the set of all destination nodes: M = $\{d_1, d_2, \cdots, d_{num}\}$, num = $|M|$ is the number of multicast group. In addition, $T(s, M)$ include the intermediate nodes in the multicast tree which not belong to the set M and are not source node. $P_T(s, d_i)$ is a path from the source node s to destination node $d_i \in M$ in the multicast tree T. The $RanPath_T(u, v)$ means that a random path from the tree T to the node v in the network topology, node $u \in T$. Crossover probability: P_C mutation probability: P_M

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

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

The cost of the degree-constraint multicast routing problem may be defined as:

$$\begin{cases} MinC(T(s, M)) \\ Deg(v) \le Degcon(v), \forall v \in T(s, M) \end{cases}$$

II. GENETIC ALGORITHM

The genetic algorithm uses the overall search strategy and optimization of the search [7-9], so the calculation does not depend on the gradient or other auxiliary knowledge and only need affect the search direction of the objective function and the corresponding fitness function. The genetic algorithm provides a generic framework for solving complex problems and it does not depend on the specific areas of the problem and it has a strong robustness. So, genetic algorithm is widely used in many subjects.

A. Coding Schemes

The individual in the multicast routing [10] problem can be as a spanning tree generated by the graph $G(V, E)$. In fact, we can identify a tree or individual uniquely if we determine each edge of the multicast tree. So, we encode individual by node matrix and realize it by a two-dimensional array. In Table 1, we define the connection matrix:

$$Y_{n \times n} = \{ T(v_i, v_j), E(v_i, v_j) \in \{0, 1\} | v_i \in V, v_j \in V \}$$

as an individual and the Chromosome coding is illustrated.

TABLE 1: CHROMOSOME CODING

	$v1$	$v2$	$v3$	$v4$	$v5$	$v6$
$v1$	-	0	0	1	1	0
$v2$	-	–	1	0	1	0
$v3$	-	–	–	0	0	1
$v4$	-	–	–	–	0	1
$v5$	-	–	–	–	–	1
$v6$	-	–	–	–	–	–

B. Generation of initial population

Generated initialization of the population, we use the random depth-first algorithm to get the degree-constrained spanning tree T (M) and the number of the initial trees or individuals is *pop_size*. We get each tree according to the following steps. First, we initialize a multicast tree contains only isolated the source node. Then we select a destination node $d_i \in$ from the set M. Then we generated a random path from the source node s to the destination node d_i. Then we merge the path into the multicast tree T until the entire destination node has been merged into the multicast tree T. Among them, we use the random depth-first search algorithm to generate a random path. Moreover, we improved the random depth-first search algorithm. Due to the degree-constraint conditions in the path, we often meet invalid node, which does not satisfy the degree-constrained condition. This moment, we will records the invalid node down in order to avoid search invalid node again in the next path generation. In a way, we improved the efficiency of the random path generation. Random tree generation algorithm is described as follows:

Procedure： RandomTreeGenerater()

Begin：

$T(s, M) = \{s\}$； //initial a multicast tree which only contains a source node s

For each d_i in M do

{

/*Generate a random path which from the tree T to the destination node d_i by calling the random path generation function RandomPathGenerater $(d, T(s, M))$ */

Generate a random path which from the tree T to the destination node d_i as $RanPath_T(u, d_i)$； //$u \in T$

Merge the path $RanPath_T(u,d_i)$ to the tree $T(s,M)$;

}

End

Random path generation algorithm is described as follows:

Procedure: RandomPathGenerater $(d, T(s,M))$

Begain:

Visited [size] ; //Record the node in $RanPath_T(u,d)$, 0: not in the tree; 1: in the tree.

Current $\leftarrow$ d;

Visited [d] $\leftarrow$ 1; //add the destination node into path.

While Current not in $T(s,M)$ do

{

Get a node from {V-{Visited}} as Current random;

Path [current][d] $\leftarrow$ 1 ; //merge the $Edge(Current,d)$ into the path;

}

// Degrees constraint judgment to fork node

If $Deg(Current) > Degcon(current)$ then

Add *current* into {Visited} ; //recode the invalid node;

Calls RandomPathGenerater($T(s,M)$) recursive;

Compute the degree of node in the path $P_T(s,d)$ that is the degree of destination node and fork node and 1, the degree of intermediate node and 2;

Return $RanPath_T(current,d)$;

End

C. Fitness Function

Fitness function is the standard for genetic algorithm to determine the individual good or bad and smaller the cost is, higher its fitness be. It is a minimum optimization problem. This problem can be converted into the problem that seeking the maximum value of the optimization of objective function by gets the reciprocal of objective function. So for each multicast tree $T(s,M)$, fitness function can be defined as the reciprocal of the total cost of the tree $T(s,M)$:

$$F(T(s,M)) = \frac{1}{C(T(s,M))} = \frac{1}{\sum_{T(s,M)} C(e)}$$

D. Selection

The genetic algorithm is based on Darwin's natural selection process mainly, so the selection plays a key role in evolution. The sample space of the algorithm is uniform. At first, we define the number of offspring equal parent group and then we select *pop_size* individuals, which have a high fitness as the next-generation groups from the parent and offspring using roulette algorithm [16] and in accordance with the principle of survival of the fittest. In addition, taking into account the principle of the best individual should be retained, the best individual of the parent copied to the next generation directly is not used for crossover with other individual.

E. Crossover

The crossover operation is the core part of the genetic progress. The crossover operation is the main way to produce individuals in the next generation, it is necessary to passed the good character in the population to the next generation, while maintain the diversity of individuals in the population. Through the crossover operation, the search capability of genetic algorithm is able to leap to improve. In this algorithm, we draw on algorithm thinking from the literature [8] to complete the crossover operation through merging trees and we set the crossover probability 0.5. Randomly selected two individuals: $P_F(s,M)$, $P_M(s,M)$, produce an offspring individual $P_G(s,M)$. First, each tree $T(s,M)$ is decomposed into a set $\{P_T(s,d), d \in M\}$ which contains path from source node to destination node. Then, we select the two cross-object $P_F(s,M)$ and $P_M(s,M)$. We calculate $P_F(s,M)$ and $P_M(s,M)$ from the source node s to destination node d path fitness value $F_F(s,d)$ and the $F_M(s,d)$, we define:

$$F_T(s,d) = \frac{1}{\sum_{\{P_T(s,d)\}} c(e)}$$

We select $P_F(s,M)$ or $P_M(s,M)$ randomly as the path of offspring $P_G(s,M)$ in order to avoid premature local convergence of populations. After determine the path set of the offspring $P_G(s,M)$, we finally merge the path and delete ring, constraints judgment.

F. Mutation

Mutation operation in the evolutionary process is an important part. On the one hand, mutation accelerates the move to the optimal solution by small local variations. On the other hand, fresh blood is injected into the populations' non-stop to maintain the freshness of individuals in the population and avoid falling into a local optimum. In this paper, we use a combination of the two variants. The first mutation, small-scale variation occurred in parent individuals. First, we select the outstanding individual as

the variation object by replacing a small number of the path from destination node to the source node to complete the first mutation and the probability is set to 0.2 that is 20% of the parent individual producing offspring individual. For the second variation, we use multi-point mutation and the mutation probability is set to 0.3 that is 30% of the parent individual producing offspring individual. According to our encoding, we generate a new individual by replace some edge in the chromosome by its allele and the probability of the variation in edge is set to 0.2. After variation, the variation of individual $T(s,M)$ will be divided the tree into a series of subtrees $\{T_{S_1}(s,M),T_{S_2}(s,M)\cdots T_{S_n}(s,M)\}$ and then merge the subtree into the mutated offspring. First of all, each of subtrees delete its loop and subtree in set of subtrees merged with other tree until there is one subtree $T_{S_{last}}(s,M)$ in the set of subtrees. Finally, we need to traverse every destination nod, determine whether the node have joined the multicast tree $T_{S_{last}}(s,M)$. Because the process of merging trees may come across failing of subtree's merging due to the degree-constraint conditions. For the failure destination node in the tree, we then generated path which from the source node to the destination node by the random path algorithm and then merge the path to the multicast tree $T_{S_{last}}(s,M)$.

The first mutation operation is described as follows:
Procedure: MutationProcess_a ( )

Begain：

Initialize the destination node subset of M ';
// the individual of $T(s,M)$or mutation.

Select the outstanding individual $T(s,M)$ using the roulette wheel;

Select a leaf node meanwhile a destination node $d_i \in T(s,M)$ randomly and record the path $P_T(s,d_i)$;

Node d_i joins destination node subset of M '.

Travers node in the path $P_T(s,d_i)$ from the node d_i in reverse order, until the node v is fork node and record the path $P_T(v,d_i)$. At the same time, determine whether the node v is destination node, if so, and then add the node v into M '.

For each d_i in M' do

{

/*Call the random path generation RandomPathGenerater($T(s,M)$) to generate the path from the source node s to destination node d_i . */

Generate a random path from the source node s to destination node d_i the $Path(s,d_i)$randomly;

Travers node in the path $P_T(s,d_i)$from destination node in reverse order, until the node already belongs to the tree $T(s,M)$ and then the merge $Path(s,d_i)$ into the tree $T(s,M)$;

}

End

The second mutation operation is described as follows:
Procedure: MutationProcess_b ()

Begain:

$T(s,M)$; // individual $T(s,M)$ for variation.

Mutate $T(s,M)$, and thus obtained a series of subtree $\{T_{s_i}(s,M)\}$;

Subtree which contains source and destination nodes get rid of loop and then prune the subtree. At the same time, we should mark these subtrees that is mark subtree which contains the source node: $Mark\left(T_{s_i}(s,M)\right) = 1$ and subtree which contains the destination node subtree labeled as $2,3\cdots$ in turn. Compute the degree of node in the entire subtree $T_{s_i}(s,M)$;

Filter out invalid subtree not containing the source node or destination node and then get the set of subtree finally:

$$subtree = \{T_{S_1}(s,M),T_{S_2}(s,M)...T_{S_n}(s,M)\};$$

// Start to merge subtree in the set of subtree and judge degree-constraint

Choose $T_{s_i}(s,M)$ from subtree which $Mark\left(T_{s_i}(s,M)\right)= 2$;

While the number of the set of subtree is greater than 1 do

{

Select another subtree $T_{s_j}(s,M)$ from the set of subtree randomly;

If $Mark\left(T_{s_i}(s,M)\right) > Mark\left(T_{s_j}(s,M)\right)$ then

Generate the random path $Path(T_{s_i}(s,M),T_{s_j}(s,M))$ which connects the tree of $Mark\left(T_{s_i}(s,M)\right)$ to the tree $Mark\left(T_{s_j}(s,M)\right)$.

If all the node in $Path(T_{s_i}(s,M),T_{s_j}(s,M))$ satisfy the degree-constrained condition then

Merger tree $T_{s_i}(s,M)$ into $T_{s_j}(s,M)$;

Modify, $Mark\left(T_{s_i}(s,M)\right)= Mark\left(T_{s_j}(s,M)\right)$;

Else

Delete tree $T_{s_j}(s,M)$ from the set of subtree;

Else

Generate the random path $Path(T_{s_i}(s,M),T_{s_j}(s,M))$, which connects the tree of $Mark\left(\Gamma_{s_i}(s,M)\right)$ to the tree $Mark\left(\Gamma_{s_j}(s,M)\right)$.

If all the node in $Path(T_{s_i}(s,M),T_{s_j}(s,M))$ satisfy the degree-constrained condition then

Merger tree $T_{s_j}(s,M)$ into $T_{s_i}(s,M)$;

Modify, $Mark\left(\Gamma_{s_j}(s,M)\right)=Mark\left(\Gamma_{s_i}(s,M)\right)$;

Else

Delete tree $T_{s_j}(s,M)$ from the set of subtree;

}

Remove the last tree of $T_{s_{last}}(s,M)$ from the set of subtree;

While there are still the purpose of node, d_i, is not joined T_ (s_last) (s, M) do

{

Generate $Path(s,d_i)$ from the source node s to the destination node d_i randomly;

If all the node in the path $Path(s,d_i)$ satisfy the degree constraint then

Merge the path $Path(s,d_i)$ to the tree of $T_{s_{last}}(s,M)$;

Else

Calls the function which generate random $Path(s,d_i)$ from the source node s to destination node d_i recursive;

}

End

G. End Condition

This algorithm, we set the termination condition for the generic: the variance of the best individual's fitness value in last N group less than their mathematical expectation of 0.01 times. The value of N should be set dynamically with the constant changes of network size. When this condition is reached, the group has evolved to a stationary state; the optimal individual is close to the global optimal solution most. According to the experimental results, we can get more desirable results when the algorithm reaches a convergence state.

III. Simulation Results

In this paper, we simulate our genetic algorithm in MRSIM. We realized the degree-constraint SPH algorithm in order to analyze the results by comparing genetic algorithm with SPH algorithm. Conducting comparative analysis, In order to unified simulation environment of the two algorithms, we set the value of Degcon(v) $\frac{2Deg(v)}{3}$ that is limiting the degree-constraint of node is 2/3 times as lager as the value of the actual degree of the nodes in the random network topology. Meanwhile we set Degcon (v) greater than or equal to 1. Population size of *pop_size* and termination conditions of *N* is set with the network size dynamically.

Figure 1 shows the cost of best individual in each generation changes in line chart. We set the number of nodes is 40 and the destination nodes is 8. The genetic manipulation achieves convergence after 61 generation. From the figure, we can find that the cost of the best individual in each generation decreasing until a stable value, namely, the state of convergence.

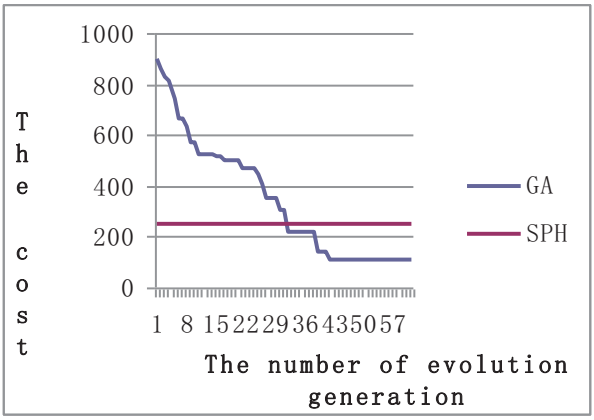

Figure 1. Cost of best individual in each generation in GA and SPH

Figure 2 depicts the cost of best individual in the generation under the comparison between the genetic algorithm and the SPH algorithm in different network size when the state is convergence. We use the average value of the 10 costs of best individual in the generation generated by 10 times of execution of algorithm. We set the number of the destination nodes is 20% of the number of the all nodes in different network size. We can see the cost of the optimal tree generated by the genetic algorithm is always less than the SPH algorithm under different network size. And the larger the random network size is, the more obvious advantages of genetic algorithms are.

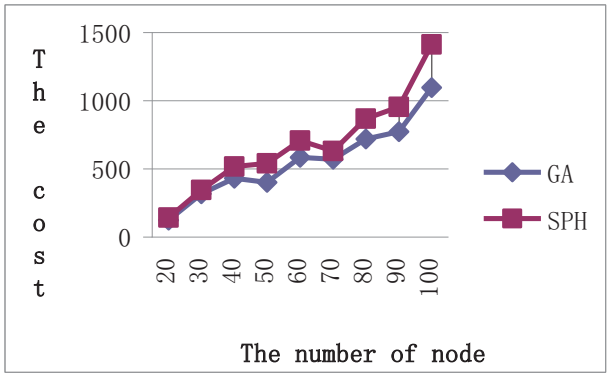

Figure 2. Cost of best individual in the generation of genetic algorithm and the SPH algorithm in different network size

Figure 3 illustrates the convergence time of comparison between the genetic algorithm and the SPH algorithm under different number of nodes. We set the number of the nodes is 20% of the number of the all nodes in different network size. We simulate 10 times in each network size and calculate the average time. We can see as the network size increases, the convergence time also increases. This is because as the number of node becomes large, the search space becomes larger and we need to enlarge the number of iterations in order to obtain better multicast tree and for the convergence time grows too. While, the convergence time of SPH algorithm is much faster than the genetic algorithm. In terms of convergence rate, the SPH is better than the genetic algorithm.

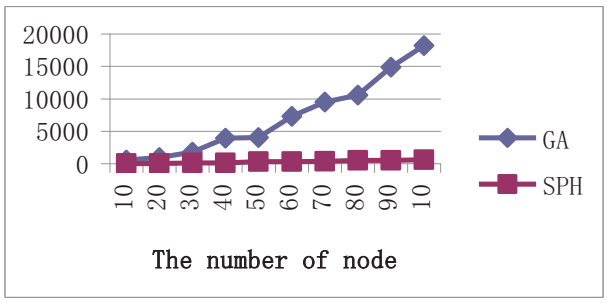

Figure 3. Impact of genetic algorithm and the SPH algorithm in convergence time

Figure 4 shows the number of generations when the state achieves convergence in the different network size. We set the number of the nodes is 20% of the number of the all nodes in different network size. We simulate 10 times in each network size and calculate the average time. We can see, see as the network size increases, the number of generations when the state achieves convergence increases. This is because as the network size increasing, result in the search space larger and the search harder.

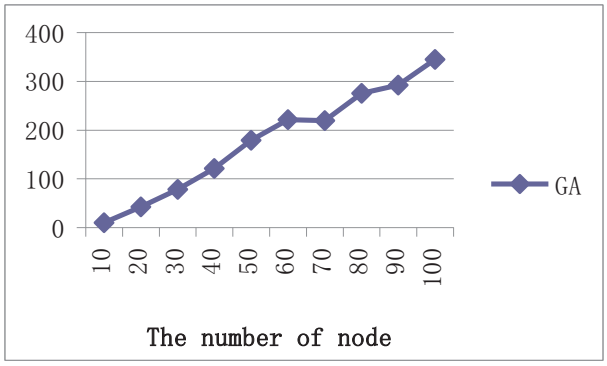

Figure 4. Number of generations when the state achieves convergence in the different network size.

IV. CONCLUSIONS

In this paper, we proposed a degree-constrained multicast routing algorithm based on genetic algorithm. The algorithm is based on Darwin's genetic theory of evolution, according to the natural law of survival of the fittest. First, this paper improved random path generation algorithm, increasing tree generation efficiency. Second, the paper uses a combination of two variants to better simulate the variation process in order to adjust the variation parameter expediently. Finally, we realized our algorithm in MRSIM and then evaluate the simulation results.

Not the same as with ordinary heuristic search algorithm, genetic algorithm is a stochastic global search algorithm which focusing on achieving global parallel search with a large search space and adjust the search direction in order to find to the optimal solution or quasi-optimal solution through the search process. General heuristic algorithms are poor search algorithms or local search algorithms mostly. Such as the SPH algorithm, it just concentrated in the shortest edge while not consider the overall situation. From the simulation results, the results of multicast routing algorithm based on genetic algorithm obtained usually better than common heuristic algorithms such as the SPH algorithm. Especially the larger network size, our algorithm tends to find a multicast tree with a lower cost than the average heuristic algorithm.

However, the genetic algorithm also has its own shortcomings. In the case of convergence time, genetic algorithm needs a longer time. Because the genetic algorithm is global search, while the SPH algorithm is only concentrated in the shortest edge. From the experimental results, this convergence time of the algorithm increases but the performance decreased with the increasing in network size and nodes.

ACKNOWLEDGEMENTS

This work was supported by the National Natural Science Foundation of China No.60903176 and No.61173078, the Program for New Century Excellent Talents in University No.NCET-10-0863, the Science and

Technology Development Program of Shandong Province No.2011GGX10116, the Natural Science Foundation of Shandong Province No.ZR2010FQ028 and No.ZR2011FL021, the Program for Youth and mild-life scientist's award fund in Shandong province under Grant No.BS2009DX037 and the Program for Youth science and technology star fund of Jinan No.TNK1108.

REFERENCES

[1] Ying Liu, You-jian Zhao,Jian-ping Wu. Degree-Constrained Multicasting Algorithm [J]. Journal of Chinese Computer Systems, 2004, vol. 3: 1216-1218

[2] K. Vik, P. Halvorsen, C. Griwodz.Evaluating Steiner-tree heuristics and diameter variations for application layer multicast Original Research Article.Computer Networks,2008, Vol52(15): 2872-2893

[3] Jin Zhang, Liang Ma, Liantang Zhang. Algorithms for degree-constrained Euclidean Steiner minimal tree[J]. Journal of Systems Engineering and Electronics, 2008, Vol 19(4): 735-741

[4] Maria Angelova, K. Atanassov, T. Pencheva. Purposeful model parameters genesis in simple genetic algorithms[J]. Computers & Mathematics with Applications, 2012, Vol 64(3): 221-228

[5] A Sadrzadeh. A genetic algorithm with the heuristic procedure to solve the multi-line layout problem Original Research Article[J].Computers & Industrial Engineering, 2012, Vol 62(4): 1055-1064

[6] W. Yang, Felix T.S. Chan, V. Kumar. Optimizing replenishment polices using Genetic Algorithm for single-warehouse multi-retailer system[J]. Expert Systems with Applications, 2012, Vol 39(3): 3081-3086

[7] B. Ramkumar, M. P. Schoen, Feng Lin. Hybrid enhanced continuous tabu search and genetic algorithm for parameter estimation in colored noise environments[J]. Expert Systems with Applications, 2011, Vol 38(4): 3909-3917

[8] Yun Wen, Hua Xu, Jiadong Yang. A heuristic-based hybrid genetic-variable neighborhood search algorithm for task scheduling in heterogeneous multiprocessor system[J]. Information Sciences, 2011, Vol 181(3): 567-581

[9] M.J. Abedini, M. Nasseri, D.H. Burn. The use of a genetic algorithm-based search strategy in geostatistics: application to a set of anisotropic piezometric head data[J].Computers & Geosciences, 2012, Vol 41: 136-146

[10] F. A. Samman, T. Hollstein, M. Glesner. Planar adaptive network-on-chip supporting deadlock-free and efficient tree-based multicast routing method[J]. Microprocessors and Microsystems, 2012, Vol 36(6): 449-461