Computer Communication & Collaboration (Vol. 5, Issue 1, 2017)
ISSN 2292-1028(Print) 2292-1036(Online)
Submitted on Jan. 06, 2017
DOIC: 2292-1036-2017-01-002-80

Genetic Algorithm for Energy-Aware QoS Multicast Routing in MANETs[1]

Ting Lu*, Shan Chang**, Wenjing Guo, Qiubo Huang

Department of Computer Science and Technology, Donghua University

2999 North Renmin Road, Songjiang District, Shanghai, China

E-mail: * luting@gmail.com; ** changshan@dhu.edu.cn

Abstract: In mobile ad hoc networks (MANETs), mobile nodes rely on batteries for proper operation. Battery energy of each node is finite and represents one of the greatest constraints for designing multicast routing protocols. Considering limited battery energy in supporting multicast service, some studies have proposed several techniques for designing a power-saving network layer. These techniques include minimizing the total power to route packets in path selection, optimizing the size of control headers, reducing the transmission of control messages, and etc. This paper discusses the quality-of-service multicast routing problem in MANETs which is NP-complete. In order to prolong the lifetime of a source-based multicast tree, we present an Energy-Aware Multi-Constrained Genetic Algorithm (EAMC_GA) to resolve this problem that depends on: (1) bounded end-to-end delay; and (2) minimum battery cost of the multicast tree. Furthermore, we use a data structure of tree to encode a multicast tree, thereby simplifying the encoding/decoding operations of genetic algorithm. Experiment results show that our proposed algorithm is efficient and effective.

Keywords: Mobile ad hoc Networks, Multicast Tree, Quality of Service, Genetic Algorithm, Residual Battery Capacity.

1. Introduction

Unlike wired and cellular networks, Mobile Ad Hoc Networks (MANETs) [1-3] have no fixed networking infrastructure. MANETs consist of multiple mobile nodes that maintain network connectivity through wireless communication. If two mobile nodes are within the transmission range of each other, they communicate directly. Otherwise, they need a packet forwarding operation using a multi-point hopping method. MANETs have applications in areas where it is not economically practical or physically possible to provide a wired networking infrastructure, e.g., battle field and disaster recovery. In such situations MANETs offer a promising solution.

In MANETs, mobile nodes are typically powered by batteries with limited energy supply. When a node exhausts its battery energy, it ceases to function. This shut down may potentially result in partitioning of the entire network. Therefore, the most important issue related to mobile nodes in MANETs is that of operation in limited-energy environments. In general, the design of QoS

[1] This research is funded by National Natural Science Foundation of China, No. 61402101, and No. 61672151; Shanghai Municipal Natural Science Foundation, No. 14ZR1400900, Fundamental Research Funds for the Central Universities, No. 2232014D3-42.

multicast routing protocol [4-7] with multi-constrained metrics has not always taken into consideration energy consumption. Without any consideration of energy consumption in MANETs, some nodes often have problems with being overused and therefore die of battery exhaustion soon. Once these selected nodes in the multicast tree run out of residual battery energy during packet forwarding, there are numerous interruptions that can occur in selected packet forwarding paths [8]. The multicast service for applications will not be maintained continuously until the completion of packet forwarding. To address this problem, packets should be routed through nodes that have sufficient remaining energy. As mentioned above, the QoS items which are called as metrics include end-to-end delay, available bandwidth, packet loss ratio, delay jitter and so on. Since metrics can be characterized by the mathematical operator that is used to compute the metric value of a path, they are classified into three categories, namely additive (e.g., end-to-end delay), multiplicative (e.g., packet loss ratio) and concave (e.g., available bandwidth). Many studies have addressed multicast routing usually with the general QoS metrics such as the bandwidth, transmission delay and packet loss ratio.

Genetic algorithm (GA) is a random searching algorithm [9-12]. It derives inspiration from the optimization process that exists in nature. The "survival of fittest" filter is applied to the population, which is constructed by crossover and mutation operators. In each generation, new population of solutions is generated by exchanging and combining the information obtained from the solutions of previous generation. Crossover operator merges two selected chromosomes to generate new offspring by exchanging some genes of two chromosomes. Mutation inducts new genetic structures into the population by randomly modifying some of the genes. Mutation keeps the search algorithm away from local optimum and prevents converging too fast. The entire population is ordered on the basis of the fitness values of individuals and the best individuals are retained. When generations are produced in an iterative manner, the average fitness of each generation is expected to be improved.

In this paper, we study the delay-constrained least-cost multicast routing problem. Our proposal is based on GA that begins with a set of random multicast Steiner tree. We use the tree structure encoding scheme to improve the encoding/decoding mechanism. By this encoding method, the coding space is greatly reduced and the encoding/decoding operation is omitted. Moreover, remaining battery capacity (i.e., the energy left in the battery) is considered in the cost function and the cost is minimized in route selection. In this way, the energy consumption is distributed among all nodes in a balanced manner. Thus, the overusing situation of nodes can be improved. We utilize an improved crossover and energy-efficient mutation mechanism. The proposed mutation method is applicable in extending the lifetime of a multicast tree by replacing the bottleneck node with minimum energy by other nodes with higher energy. The simulation results show that our proposal is an efficient and effective algorithm for multicast route selection.

The remainder of this paper is organized as follows: Section 2 describes the network model and the multicast routing problem. Section 3 presents the Energy-Aware Multi-Constrained Genetic Algorithm (EAMC_GA). Section 4 gives the analysis of convergence of EAMC_GA and the comparison with other GAs. Section 5 evaluates the performance of EAMC_GA. Section 6 concludes this paper.

2. Problem Formulation

In general, a QoS multicast routing problem can be involved in several constraints, e.g., end-to-end delay, cost, available bandwidth, packet loss ratio, and delay jitter. If all the constraints are considered in the design of routing algorithm, the corresponding algorithm will be too complicated to be applicable in practice. In this paper, we consider only two QoS constraints in our algorithm and present a feasible model for QoS multicast routing. Most multimedia applications are delay-sensitive. Therefore, the delay constraint is considered. In MANETs, nodes rely on batteries

for proper operation. Battery energy is the basic requirement for transmitting application's information. Therefore battery energy is also considered.

An ad hoc network can be modeled as a graph $G=(V,E)$, where V and E denote the set of nodes and links, respectively. Each node $u \in V$ represents a mobile host, and each link $e \in (u,v) \in E$ indicates that nodes u and v are within the transmission range of each other and can communicate directly. Any link $e \in E$ has a delay $delay(e): E \to R^+$ associated with it, and any node $u \in V$ has a cost $\text{cost}(u): V \to R^+$ associated with it, where R^+ is the set of positive real numbers.

Let $s \in V$ be the multicast source and $D \subseteq V - \{s\}$ the set of multicast destinations. A multicast tree $T(s,D)$ is a tree rooted at s and spanning all members of D. The delay of a path from s to a destination $t \in D$ on the tree T, denoted as $delay(P_T(s,t))$, is described as follows:

$$delay(P_T(s,t)) = \sum_{e \in p_T(s,t)} delay(e) \tag{1}$$

The total cost associated with a multicast tree is defined as follows:

$$cost\left(T\left(s,D\right)\right) = \sum_{u \in T(s,D)} cost\left(u\right) \tag{2}$$

Definition 1. (Delay-Constrained Least-Cost Multicast Routing Problem). Given a network $G(V,E)$, a source node $s \in V$, a destination node set $D \subseteq V - \{s\}$, a positive link delay function $delay(\cdot) \in R^+$, a positive node cost function $\text{cost}(\cdot) \in R^+$, the delay-constrained least-cost multicast routing problem is to find a minimum cost tree T which spans s and D subject to:

$$delay\left(p_T\left(s,t\right)\right) \le DDCF\left(t\right), \quad \forall t \in D \tag{3}$$

where $p_T(s,t)$ is the path from s to t on T, and $DDCF(\cdot)$ is the delay constraint function that assigns an upper bound to the path delay from the source to each destination. It is noted that $DDCF(u)$ may be different from $DDCF(v)$ for $u \neq v$, $u,v \in D$. In the special case, $DDCF(\cdot)$ assigns the same delay bound to all destinations, i.e., $DDCF(t) = \Delta$, $\forall t \in D$.

3. The Proposed Multicast Routing Algorithm

There are two main genetic operators in GA. Crossover identifies the common links between two "parent" individuals and retains them in the child individual, since the common links represent the "good traits" and should be passed on to the child. Mutation simply adds genes to the individuals. Mutation introduces new genetic structures in the population by modifying some of the genes, and keeps the search algorithm escaping from the local optimum. In this paper, we design a source-tree-based routing algorithm and adopt the tree structure encoding scheme to simplify encoding/decoding operations. In order to use node energy in a balanced manner, we utilize a cost function based on the node's remaining battery capacity. Figure 1 shows the flowchart of the EAMC_GA. The corresponding pseudo code is given in Figure 2.

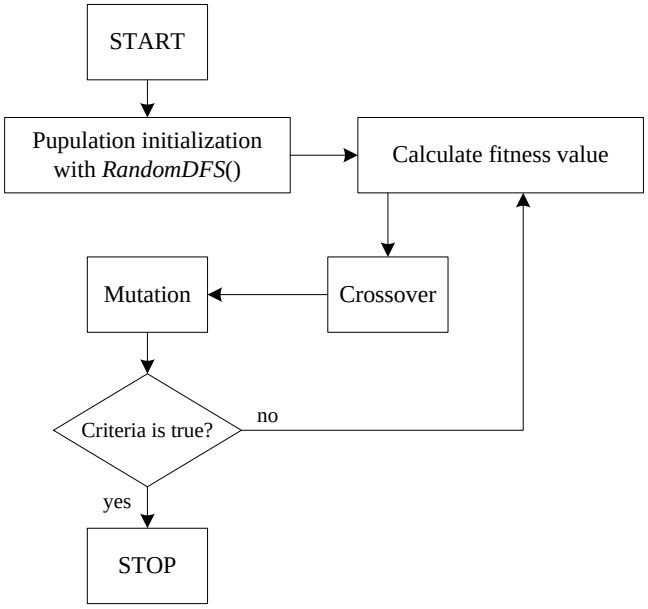

Figure 1. The flowchart of EAMC_GA

EAMC_GA (*G*, *s*, *D*)
{
1. *for* (*i*=1*; i*<= N_p*; i*++) { // N_p: population size
2. *Chromosome*(*i*) = *RandomDFS*(*G*, *s*, *D*); // *RandomDFS*(): random depth-first search algorithm
   }
3. *for* (*j*=1*; j*<=N_g*; j*++) { // N_g: the number of generations
4. *select the best individuals and copy them into the next generation;*
5. *for* (*k*=1; *k*<=N_p-$N_{optimal}$; *k*++) { // $N_{optimal}$: the number of the best individuals
6. T_M=*MSTSelect* (*Chromosome*)
7. T_N= *MSTSelect* (*Chromosome*)
8. T_C=*Crossover*(T_M, T_N);
9. *if* (*rand*() < p_m) // p_m: mutation probability
10. *Mutation*(T_C);
    }
   }
11. *Select the best individual and output it*;
}

Figure 2. The pseudo code of EAMC_GA.

3.1. Encoding

How to encode a multicast tree into a chromosome is a key issue for GA. A better encoding method would make the genetic operation (e.g., crossover and mutation) more efficient. A $N \times N$ one-dimensional binary encoding scheme is adopted in [4], where N represents the number of nodes in the network. However, in this encoding scheme, back and forth transformation between genotype

and phenotype space is very complicated, especially for large networks. This weakness causes its low efficiency. Another representative encoding scheme for a tree is the Prüfer number used in [13,14]. This method encodes a tree consisting of n nodes into n-2 integers. The disadvantage of this scheme is that it exhibits relatively low locality. Changing even one element of Prüfer number can drastically change the structure of the tree. In our algorithm, we choose the tree structure encoding method. Any data structure of a tree [15] can be used to describe the chromosome structure. Each chromosome represents a multicast tree. By this method, the encoding and decoding operations are omitted.

3.2. Initial population

In general, two issues should be considered for population initialization: (1) population size Np; (2) the procedure to initialize the population. The population size Np is set by the system. The way to generate the initial population is the same as that presented in [16]. A random depth-first search algorithm was employed to generate a random multicast Steiner tree. The searching process begins at the source s and randomly selects an unvisited node at each node for the next visit. The algorithm terminates when all destination nodes have been visited.

3.3. Fitness function

Fitness function interprets the chromosome in terms of physical representation and evaluates its fitness based on traits of being desired in the solution [17]. In other words, the fitness function should reflect the individual performance: the "good individual" has bigger fitness than the "bad one". Therefore, the fitness function is defined as follows:

$$\begin{aligned} f(T) &= \frac{\alpha}{f_{battery\_cost}(T)} \prod_{t \in D} \Phi\left(delay\left(p_T(s,t)\right) - DDCF(t)\right), \\ \Phi(Z) &= \begin{cases} 1, & Z \le 0 \\ \gamma, & Z > 0 \end{cases} \qquad 0 < \gamma < 1 \end{aligned} \tag{4}$$

where $f(T)$ represents the fitness value of a chromosome T, α is the positive real coefficient, $f_{battery\_cost}(T)$ is the battery cost function for T, and $\Phi(Z)$ is the penalty function. For any $t \in D$, if the path $p_T(s,t)$ satisfies the delay bound $DDCF(t)$ (i.e., $delay\left(p_T(s,t)\right) \le DDCF(t)$), the value of $\Phi(\cdot)$ is 1, or else γ. The value of γ determines the degree of penalty: the smaller the value of γ, the higher the penalty. In our study, we set γ=0.5.

In order to use node energy in a more balanced manner, we utilize a cost function based on the node's remaining battery capacity. Let c_u^t denote the residual battery capacity of node $u \in V$ at time *t*, and $c_u^t = c_u^{full}/n$, where c_u^{full} is the full capacity of *u* (assuming $c_u^{full} = 1$) and *n* is the integer ranging from 1 to 100. We suppose that the willingness of each node to forward packets is inversely proportional to its remaining battery capacity. The less battery capacity it has, the more reluctant it is. The battery cost function of *u* is given by $f_{battery\_cost}(u) = \frac{1}{c_u^t}$. Then the total battery cost of a tree is defined as follows:

$$f_{battery\_cost}(T) = \sum_{u \in T} \frac{1}{c_u^t} \tag{5}$$

This battery cost function prevents mobile nodes from being overused by finding the multicast tree with sufficient remaining battery capacity, thereby consuming node energy in a more balanced manner and maximizing the network lifetime. However, a multicast tree including a node with little remaining battery capacity may still be selected, because only the summation of battery cost of each node is considered in route selection. To address this issue, in the mutation phase, we remove the bottleneck node (i.e., the node with the minimum residual battery capacity in a multicast tree) from the tree and use nodes with higher battery capacity instead.

3.3. Selection of parents

In our GA, the elitist model [18] is adopted as the selection operator. Through this model, we first select the optimal chromosomes and copy them into the next generation directly. Then, we select the rest by the proportional model [19]. The selection probability $p_s(i)$ that a parent i is selected is given by:

$$p_s(i) = \frac{f(T_i)}{\sum_{j=1}^{N_p} f(T_j)} \tag{6}$$

3.4. Crossover scheme

The crossover operator used in our work is an improved scheme proposed in [20]. Two multicast trees are selected by proportional model as the parents to produce an offspring. This process is repeated $N_p - N_{optimal}$ times, where $N_{optimal}$ is the number of the best individuals, which are selected and directly copied into the next generation.

According to the crossover probability of 1, the crossover operator generates a child T_c by identifying the same links of two parents (e.g., T_M and T_N) and retaining them in T_c. Since the individual with higher fitness value is selected as parent with higher probability, the same links of selected parents are more likely to represent the "good traits". It is helpful to select the same links and pass them on to the child for quicker convergence of the algorithm. It should be noted that these same links may be in some separate sub-trees and links are needed to be added to transform these sub-trees into a tree.

The process of connecting separate sub-trees to be a multicast tree is as follows. Two separate sub-trees are randomly selected among these sub-trees. The two selected sub-trees are connected with the least-delay path. Therefore, a new sub-tree consisting of the two sub-trees and the path is formed. The newly generated sub-tree will be among the separate sub-trees for next selection. This process continues until a multicast tree is constructed. In order to find the least-delay path between two sub-trees, we add two nodes. One is connected to all the nodes of one sub-tree with links, which have zero delay associated with them. In the same way, the other one is connected to all the nodes of the other sub-tree with zero-delay links. Hence the problem of finding the least-delay path between two sub-trees is equivalent to the problem of finding the least-delay path between the two added nodes. Clearly there is no routing loop in the multicast tree by this connection scheme.

Figure 3 shows an example of crossover procedure. As shown in Figure 3, the same links of T_M and T_N are retained in T_c. Then, all sub-trees are connected with least-delay paths which are denoted as dot lines in T_c.

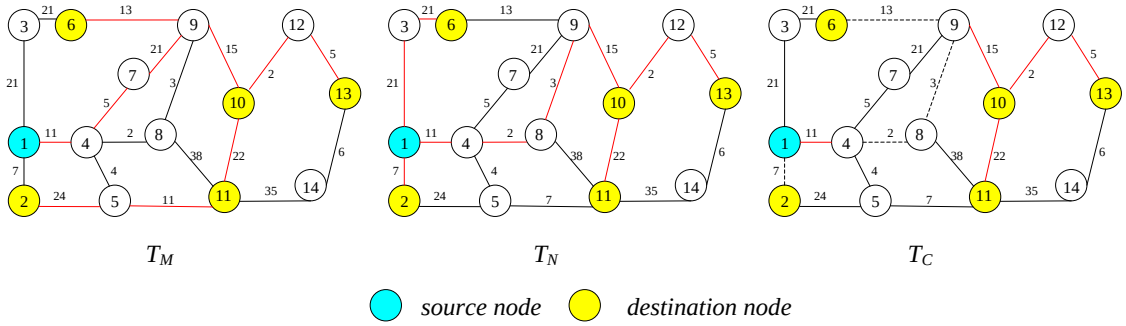

Figure 3. Example of the crossover procedure

3.5. Mutation

When a new offspring is produced, the mutation operation is performed according to the mutation probability pm. In order to extend the lifetime of the multicast tree, our mutation process is as follows. First, check whether the bottleneck node $x \in D$. If $x \notin D$, go to the second step; otherwise, do nothing. Second, remove x from the tree T, remove the link between x and its father node, and remove the links between x and its child nodes. After the removing operations, some separate sub-trees are left. Third, randomly select two sub-trees and connect them with the least-cost path to generate a new sub-tree. This selection and connection process continues until a multicast tree is constructed. In order to find the least-cost path between two sub-trees, we also add two nodes as described in Section 3.5. One is connected to all the nodes of one sub-tree and the other one is connected to all of the nodes of the other sub-tree. The cost associated with the two newly added nodes is zero. Similarly, the problem of finding the least-cost path between two sub-trees is equivalent to the problem of finding the least-cost path between the two added nodes

4. Analysis of the Proposed Algorithm

4.1. Analysis of convergence

The characteristics of EAMC_GA are as follows: (1) crossover probability $p_c \in (0,1]$; (2) mutation probability $p_m(0,1)$; and (3) adopting the elitist model for selection. According to the Theorem 2.7 in [21], EAMC_GA could finally convergence to the global optimal solution.

For a large-scale network, it is time-consuming to obtain the optimal solution to the multicast routing problem with multiple QoS constraints, which is NP-complete. Furthermore, GAs may not be promising candidates for supporting delay-sensitive applications in MANETs because they involve a large number of iterations in general. This problem can be overcome by the hardware implementation of GAs (e.g., field-programmable gate array (FPGA) chips) [22], which is very fast. In addition, GAs are not very sensitive to network size [23]. In this regard, EAMC_GA is quite promising for multicast routing in MANETs.

4.2. Comparisons with other genetic algorithms

Compared with the GAs presented in [4,13,14], EAMC_GA adopts tree structure encoding scheme. This encoding scheme overcomes the weaknesses of one-dimensional binary code [4] and Prüfer number [13,14]. The back and forth transformation between genotype and phenotype space in binary encoding is very complicated, especially for large networks. The drawback of Prüfer number

encoding is that it does not preserve the locality. Compared with the GA presented in [24], the crossover and mutation of EAMC_GA will not form illegal trees (i.e., tree with non-existent links). However, the crossover and mutation in [24] may generate illegal trees and repair process is needed, which will increase the computation complexity. Compared with GA in [16], EAMC_GA adopts the elitist model for selection to avoid the pre-maturation convergence occurred in [16].

5. Experiments

The computational simulation is implemented in MS VC++ 6.0 uses the GAlib which is a C++ Library of GA. The simulation environment was carried out on an Intel Pentium Dual-core CPU with 2.50GHz, 2GB RAM, and Windows XP Professional Operating System. GAlib provides interfaces to specify the crossover and mutation probability, the number of generations and the population size. We modified the crossover and mutation operators and used the fitness function formula mentioned earlier. The simulation was performed using random network. Every node is within the maximum transmission range of at least one other node in the network, i.e., the network is connected. The source and the destinations are randomly generated. $DDCF(t)$ for destination t is uniformly distributed in the range of [30*ms*, 160*ms*], the delay of each link is uniformly distributed in [0*ms*,50*ms*].

Our experiments aim to test the convergence ability, the convergence speed and the convergence procedure of the GA. We have study the following three performances: (1) Average Success ratio (SR); (2) Convergence process; (3) Running time. The SR is defined as follows:

$$SR = \frac{N_{accepted}}{N_{request}} \tag{7}$$

where $N_{accepted}$ is the average number of multicast routing accepted and $N_{request}$ is the total number of multicast routing requests. If the multicast tree constructed by the proposed algorithm satisfies the delay constraints, we consider a successful routing request is accepted.

5.1. Success ratio

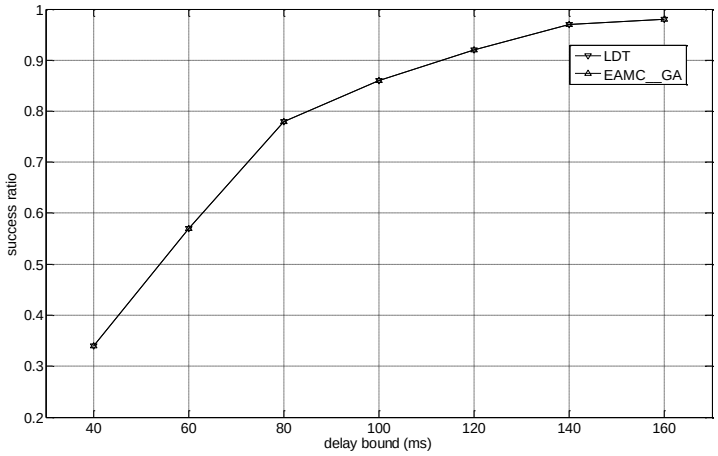

Figure 4. Comparison of average success ratio for EMAC_GA and LDT

Since the least-delay multicast tree algorithm (LDT) has the highest routing success ratio among all the delay-constrained multicast routing algorithms, we compare the EAMC_GA with LDT in this experiment. Figure 4 shows the comparison of success ratio between EAMC_GA and LDT in a

network with 14 nodes. From Figure 4, we can observe that the two algorithms have the same success ratio. This proves that EAMC_GA is capable of constructing a delay-constrained multicast tree if one exists.

5.2. Convergence process

Figure 5 shows the convergence processes of EAMC_GA under parameters: $\Delta = 20$, $N_g = 18$, $N_p = 15$, $p_m = 0.05$, and Figure 6 under parameters: $\Delta = 25$, $N_g = 18$, $N_p = 15$, $p_m = 0.05$. For simplification, we set the same delay bounds to all destinations. From Figure 5 and Figure 6, it is obvious that the algorithm can converge to the global solution quickly.

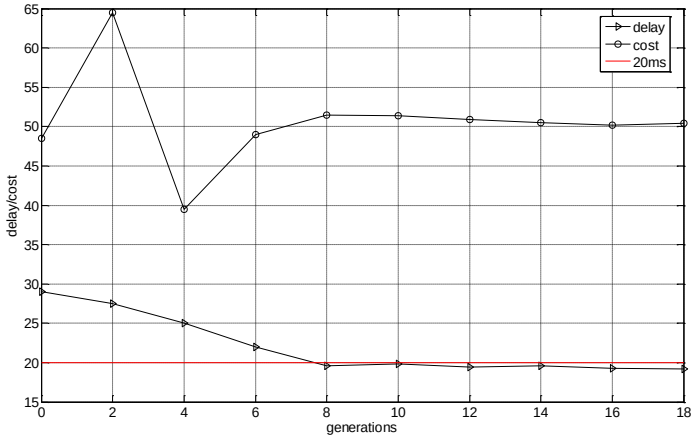

Figure 5. Convergence of EAMC_GA for $\Delta = 20$

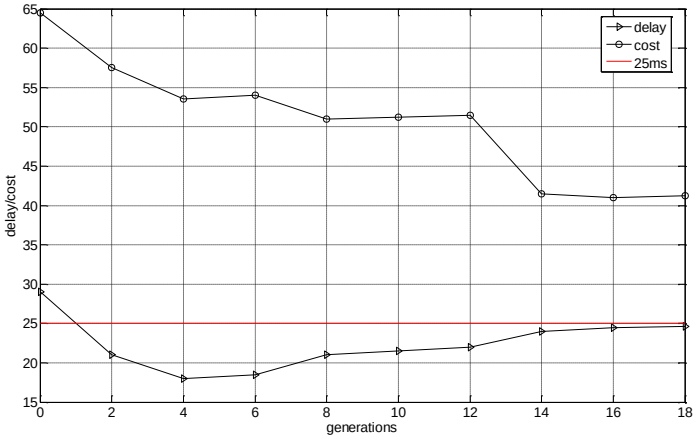

Figure 6. Convergence of EAMC_GA for $\Delta = 25$

5.3. Running time

Figure 7 shows the computation time for networks with different size. Form Figure 7, it is obvious that the running time grows slowly with the size of the network. For a large-scale network with 140 nodes, the running time is fairly desirable.

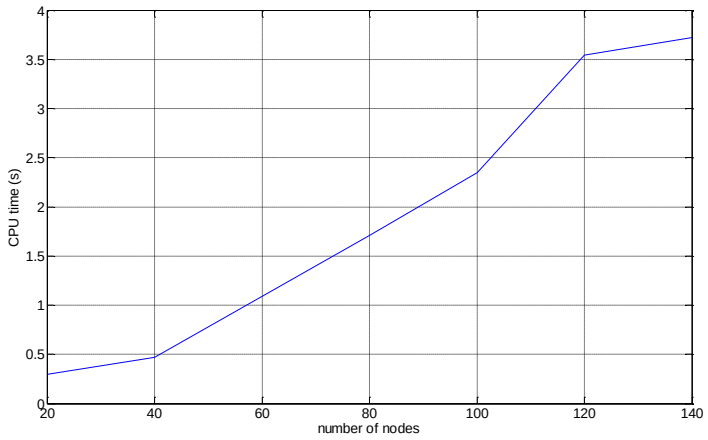

Figure 7. Running time of EAMC_GA for networks with different size

6. Conclusions

We focus on energy consumption efficiency which generally means selecting nodes with low energy consumption in route selection. Although this mechanism can reduce the total power consumption of the overall network, it may generate a situation in which some nodes with low power consumption are overused. Thus, these nodes will die of battery exhaustion soon. Such excessive usage situation will result in partitioning of the entire network and shorten the lifetime of the multicast service. We adopt a strategy to balance energy consumption for multicast in the entire work. EAMC_GA has the following characteristics. (1) Tree structure encoding scheme simplifies the encoding operation and omits the encoding/decoding process. (2) Cost function based on node's remaining battery capacity balances energy consumption in the network. (3) Heuristic crossover mechanism speeds up the convergence. (4) Energy-aware mutation technique extends the lifetime of multicast service. Our research on genetic algorithms for multicast routing problem is more suitable for MANETs.

References

[1]. Lim, H. and Kim, C. (2001), "Flooding in wireless ad hoc networks", *Comput. Commun.*, 24(3-4), 353-63.

[2]. Toh, C.K. (2001), "Maximum battery life routing to support ubiquitous mobile computing in wireless ad hoc networks", *IEEE Commun. Mag.*, 39(6), 138-147.

[3]. Jia, X., Zhi, L. and Qianmu, L. (2008), "An adaptive clustering routing transition protocol in ad hoc networks", *Comput. Commun.*, 31(10), 1952-1960.

[4]. Xiang, F., Junzhou, L., Jieyi, W. and Guanqun, G. (1999), "QoS routing based on genetic algorithm", *Comput. Commun.*, 22(15-16), 1392-1399.

[5]. Haghighat, A.T., Faez, K., Dehghan, M., Mowlaei, A. and Ghahremani, Y. (2003), "GA-based heuristic algorithms for QoS based multicast routing", *Comput. Commun.*, 16(5-6), 305-312.

[6]. Sanna, R.L. and Atzori, L. (2007), "Group multicast routing problem: a genetic algorithms based approach", *Comput. Netw.*, 51(14), 3989-4004.

[7]. Forsati, R., Haghighat, A.T. and Mahdavi, M. (2008), "Harmony search based algorithms for bandwidth-delay-constrained least-cost multicast routing", *Comput. Commun.*, 31(10), 2505-2519.

[8]. Wang, B. and Gupta, S.K.S. (2004), "Extending the lifetime of multicast trees in WANETs", *J. Inf. Sci. Eng.*, 20(3), 425-447.

[9]. Li, L. and Li, C. (2003), "Genetic algorithm-based QoS multicast routing for uncertainty in network parameters", In: Proceedings of APWeb 03, Xian, China, April, Springer-Verlag, Berlin, 430-441.

[10]. Ci, S., Guizani, M., Chen, H.H. and Sharif, H. (2006), "Self-regulating network utilization in mobile ad-hoc wireless", IEEE Trans. Veh. Technol., 55(4), 1302-1310.

[11]. Tseng, S.Y., Huang, Y.M. and Lin, C.C. (2006), "Genetic algorithm for delay and degree-constrained multimedia broadcasting on overlay networks", *Comput. Commun.*, 29(17), 3625-3632.

[12]. Chiang, T.C., Liu, C.H. and Hung, Y.M. (2007), "A near-optimal multicast scheme for mobile ad hoc networks using a hybrid genetic algorithm", *Expert Syst. Appl.*, 33(3), 734-742.

[13]. Zhou, G. and Gen, M. (1998), "An effective genetic algorithm approach to the quadratic minimum spanning tree problem", *Eur. J. Oper. Res.*, 25(3), 229-247.

[14]. Haghighat, A.T., Faez, K., Dehghan, M., Mowlaei, A. and Ghahremani, Y. (2002), "A genetic algorithm for steiner tree optimization with multiple constraints using Prüfer number", In: Proceedings of EurAsia-ICT 2002, Shiraz, Iran, October, Springer-Verlag, Berlin, 272-280.

[15]. Mark, A.W. (1996), *Data Structure and Algorithm Analysis in C*. Addison Wesley Press.

[16]. Ravikunmar, C.P. and Bajpai, R. (1998), "Source-based delay-bounded multicasting in multimedia networks", *Comput. Commun.*, 21(2), 126-132.

[17]. Chang, W.A. and Ramakrishna, R.S. (2002), "A genetic algorithm for shortest path routing problem and the sizing of populations", *IEEE Trans. Evol. Comput.*, 6(6), 566-579.

[18]. Rudolph, G. (1994), "Convergence analysis of canonical genetic algorithms", *IEEE Trans. Neural Netw.*, 5(1), 96-101.

[19]. Dirk, T. and David, G. (1994), "Convergence models of genetic algorithm selection schemes", In: Proceedings of Parallel Problem Solving from Nature-PPSN III, Jerusalem, Israel October, Springer-Verlag, Berlin, 119-129.

[20]. Wang, Z.Y., Shi, B.X. and Zhao, E. (2001), "Bandwidth-delay-constrained least-cost multicast routing based on heuristic genetic algorithm", *Comput. Commun.*, 24(7-8), 685-692.

[21]. Guoliang, C., Xufa, W., Zhenquan, Z. and Dongsheng, W. (1996), *Genetic Algorithm and its Application*. People's Posts and Telecommunications Press.

[22]. Scott, S.D., Samal, A. and Seth, S. (1995), "HGA: a hardware-based genetic algorithm", In: Proceedings of FPGA 95, Monterey, CA, USA, February, ACM, USA, 53-59.

[23]. Tufte, G. and Haddow, P.C. (1999), "Prototyping a GA pipeline for complete hardware evolution", In: Proceedings of EH 99, IEEE Computer Society, USA, 18-25.

[24]. Yunsheng, Y., Yikung, C., Hanchieh, C. and Jonghyuk, P. (2008), "A genetic algorithm for energy-efficient based multicast routing on MANETs", *Comput. Commun.*, 31(4), 858-869.