Front. Electr. Electron. Eng. China 2007, 2(3): 326–329
DOI 10.1007/s11460-007-0061-4

RESEARCH ARTICLE

SUN Lijuan, GUO Jian, LU Kai, WANG Ruchuan

Topology control based on quantum genetic algorithm in sensor networks

Abstract Nowadays, two trends appear in the application of sensor networks in which both multi-service and quality of service (QoS) are supported. In terms of the goal of low energy consumption and high connectivity, the control on topology is crucial. The algorithm of topology control based on quantum genetic algorithm in sensor networks is proposed. An advantage of the quantum genetic algorithm over the conventional genetic algorithm is demonstrated in simulation experiments. The goals of high connectivity and low consumption of energy are reached.

Keywords sensor network, topology control, power control, genetic algorithm, quantum genetic algorithm

1 Introduction

An Ad-hoc network is a multi-hop temporary self-control system which consists of a set of mobile terminals with wireless receivers and transmitters. All nodes in an Ad-hoc network are equal, eliminating the need to install any central control node. Each node can act as both the host and router. As a host, a user-oriented application runs on the terminal; as a router, a corresponding routing protocol runs on the terminal, thereby the node has the function of routing and packet forwarding. In this network, due to the limited transmission scope of a wireless terminal, two nodes without direct means of communication may be connected with the help of a number of intermediate nodes. Therefore, it is also known as a multi-hop wireless network, a self-organized network, an infrastructureless network, or a peer-to-peer network. An Ad-hoc network, characterized as non-centered and self-organized, has a wide range of potential applications, particularly in such situations as battlefield communications, disaster relief, and in the expedition field, etc. [1–5].

Translated from *Journal on Communications*, 2006, 27(12): 1−5 [译自: 通信学报]

SUN Lijuan, GUO Jian, LU Kai, WANG Ruchuan (✉)
College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
E-mail: wangrc@njupt.edu.cn

A wireless sensor network is a kind of static Ad-hoc network, the nodes of which remain fixed or move slowly. The nodes' physical location is easy to collect in such networks as fire monitoring or meteorological observation networks. Generally, the radius of each node radiation signal is set for the same value and cannot be adjusted. In this way it is simple to process. Links generated are symmetrical and topology control is usually based on the actual needs, analyzed with graph theory and implemented by adjusting the distance between nodes. This method results in unreasonable topology, a waste of energy and short life of nodes. Currently, some researches [2] have appeared on the topology control in a network of which nodes' radiation radius is adjustable.

In this paper, a solution of topology control based on quantum genetic algorithm (QGA) in sensor networks is proposed. An advantage of the quantum genetic algorithm over the conventional genetic algorithm is demonstrated in simulation experiments, and the goals of high connectivity and low consumption of energy fulfilled.

2 Topology control in sensor network

The main goal of topology control in sensor networks is to design a network with high connectivity and low power consumption. Specifically, there are two objectives: first, there exists a path between any two nodes; second, power consumption of all nodes is the lowest. In addition, if the radiation radius of a node exceeds a certain threshold, its energy will rapidly be exhausted. Therefore, the number of these nodes should be as small as possible. We illustrate the topology control of sensor networks as follows.

Suppose that a sensor network has n nodes. For arbitrary node i, $1 \leqslant i \leqslant n$, the coordinates of which are (x_i, y_i), radius is r_i, and power consumption is e_i. $\mathrm{R}_{\max}$ denotes the maximum radiation radius of the node, and the threshold radius is R_v. Network adjacency matrix is $\boldsymbol{C}$, the total energy is E, and the number of nodes exceeding the radius threshold R_v is f. Obviously, we arrive at

1) $r_i \leqslant R_{\max}$ $(1 \leqslant i \leqslant n)$
2) $\boldsymbol{C} = [c_{ij}]_{n\times n}$
3) $E = e_1 + e_2 + \ldots + e_n$

where $\boldsymbol{C}$ is an $n \times n$ matrix, and

$$c_{ij} = \begin{cases} 1, & \text{node } j \text{ is in the radiation radius of node } i \\ 0, & \text{otherwise} \end{cases}.$$

Then, the goal is to find out a set of suitable radius r_i ($1 \leqslant i \leqslant n$), meeting the below

1) for all c_{ij} in $\boldsymbol{C}^{(n)}$, $1 \leqslant i \leqslant n$, $1 \leqslant j \leqslant n$, $c_{ij} = 1$
2) min (E)
3) min (f).

3 Quantum genetic algorithm (QGA)

The quantum genetic algorithm [6–10] is the product of the combination between genetic algorithm (GA) and quantum computing. It is based on the quantum vectors, representing a chromosome by qubit coding, and updating the chromosome by quantum rotation gate and quantum non-gate. Eventually, the optimal solution will be found.

3.1 Qubit encoding

A qubit is the smallest unit of information in QGA. A qubit may be in either the '1' or '0' state, or in any superposition of the two, i.e. a qubit could be $|0>$, $|1>$, or in the in-between state. Therefore, it can be expressed as

$$|\Psi> = \alpha|0> + \beta|1> \quad (1)$$

where α and β are two complex numbers satisfying $|\alpha|^2 + |\beta|^2 = 1$. $|\alpha|^2$ and $|\beta|^2$ denote the probabilities of $|0>$ and $|1>$ respectively.

In QGA, qubits are used to represent a gene which expresses all probable information instead of a set of definite information. Any operation carried out on this gene may exert influence on all possible information simultaneously. Furthermore, a chromosome can be encoded as

$$\left(\begin{array}{c|c|c|c|c|c|c|c|c|c|c|c|c} \alpha_{11} & \alpha_{12} & \cdots & \alpha_{1k} & \alpha_{21} & \alpha_{22} & \cdots & \alpha_{2k} & \cdots & \alpha_{m1} & \alpha_{m2} & \cdots & \alpha_{mk} \\ \beta_{11} & \beta_{12} & \cdots & \beta_{1k} & \beta_{21} & \beta_{22} & \cdots & \beta_{2k} & \cdots & \beta_{m1} & \beta_{m2} & \cdots & \beta_{mk} \end{array}\right) \quad (2)$$

Here, k denotes the number of qubits in each gene, while m is the number of genes in each chromosome. α_{xy} and β_{xy} ($1 \leqslant x \leqslant m$, $1 \leqslant y \leqslant k$) are two complex numbers satisfying $|\alpha_{xy}|^2 + |\beta_{xy}|^2 = 1$.

3.2 Evolutionary operation

Quantum rotation gate is the implementation of evolution operation. It is operated as

$$\begin{pmatrix} \alpha_i' \\ \beta_i' \end{pmatrix} = \begin{pmatrix} \cos\theta_i & -\sin\theta_i \\ \sin\theta_i & \cos\theta_i \end{pmatrix} \begin{pmatrix} \alpha_i \\ \beta_i \end{pmatrix} \quad (3)$$

where $(\alpha_i, \beta_i)^{\mathrm{T}}$ and $(\alpha_i', \beta_i')^{\mathrm{T}}$ are the ith ($1 \leqslant i \leqslant mk$) qubits of the chromosome before and after the updating respectively. θ_i is the rotation angle, the value and direction of which can be adjusted by some strategies. A general adjustment strategy [10] is shown in Table 1.

Table 1 A general adjustment strategy

x_i	b_i	$f \geqslant f_b$	$\Delta\theta_i$	$S(\alpha_i, \beta_i)$	
				$(\alpha_i, \beta_i) > 0$	$(\alpha_i, \beta_i) < 0$
0	0	False	0	–	–
0	0	True	0	–	–
0	1	False	δ	+1	–1
0	1	True	δ	–1	+1
1	0	False	δ	–1	+1
1	0	True	δ	+1	–1
1	1	False	0	–	–
1	1	True	0	–	–

x_i and b_i are the ith bit of the observation of the current chromosome and target value respectively. f and f_b are their corresponding fitting degrees. $S(\alpha_i, \beta_i)$ and $\Delta\theta_i$ represent the direction and the angle of rotation, which can be retrieved from Table 1. The rotation angle can be demonstrated by $\theta_i = S(\alpha_i \beta_i)\Delta\theta_i$. There are two strategies of rotation angle adjustment: static and dynamic. During the algorithm process, if the δ value remains unchanged, static adjustment strategy will be adopted. If it changes, dynamic adjustment strategy can start in motion. There is a close relationship between δ and the performance of the algorithm. If δ is too large, it is prone to become premature, and only the local convergence to the optimal solution will be achieved; on the contrary, if the value is too small, it is slow to update the chromosome, and the algorithm will be in a state of stagnation. To remove these shortcomings, dynamic adjustment is usually adopted. That is, in the early part of running, in order to quickly approach the optimal solution, a greater δ value should be provided; with the increase of the generations of the algorithm, δ decreases, thus self-adaptive adjustment works.

In QGA, δ is generally no more than 0.1 π. In this paper, δ is defined as a variable with generations of evolution according to $\delta = 0.02\pi\cos\left(\dfrac{t}{\max t}\right)$, where t is the generation of evolution, max t is a constant based on the specific problem. δ can be adjusted according to the generation of evolution.

3.3 Workflow of QGA

The process of our algorithm [10] is detailed as follows:

1) Initialize population $Q(t_0)$. Initialize all genes of chromosomes with $\left(1/\sqrt{2}, 1/\sqrt{2}\right)$, which means that a chromosome represents the linear superposition of all possible states with the same probability.

2) Take the initial observation for all individuals, and obtain a group of solutions $p(t) = \left\{p_1^t, p_2^t, \ldots, p_n^t\right\}$. p_j^t denotes the jth solution of the tth population (namely the observation result of the jth individual). It is represented as an m-bit

binary string, each bit of which is either 0 or 1. The observation begins with the generation of a random number in [0, 1]. If it is greater than the square of the probability amplitude, set the bit as 1, otherwise 0. Then each solution of the group is valued respectively, and the best solution is saved as the target value of the next evolution.

3) Iterative section of the algorithm. First, see whether the condition to end the iteration is met. If the condition is met, then the algorithm ends. Otherwise, take another observation for all individuals of the current population, and obtain the corresponding solutions and fitting degree.

4) Based on the current target value, we update the population with quantum rotation gate, and get the children population. The process of adjustment includes observing individuals and calculating their fitting degree f. After comparing f with the fitting degree f_b of the current target value, we adjust the corresponding individual qubit. When $f < f_b$, showing that the fitting degree of the current individual is insufficient, we should evolve $(\alpha_i, \beta_i)^{\mathrm{T}}$ towards the direction in which it is helpful to gain b_i. Otherwise, we should evolve towards the direction in which it is helpful to gain x_i.

5) Compare the optimal solution of the current generation with the target value, choose the better one as the target of the next evolution, and jump back to Step 3).

4 Algorithm realization

We realize our algorithm by object-oriented approach. We design three entity classes and one operation class, which are described as follows. Interface classes can be generated by the development environment.

4.1 Entity classes

Chromosome class: every feasible solution corresponds to a chromosome. Thus, the key data member of the chromosome is a two-dimensional array, which stores codes of qubits. Observe() is a member function. It is used to observe chromosomes and acquire corresponding solutions.

Population class: a population consists of a number of chromosomes. The main member functions include MakePt(), StorBestSolution() and Update(). MakePt() obtains solutions of all chromosomes, StorBestSolution() stores the best individual of each generation, and Update() updates the population with quantum rotation gate.

Network class: a sensor network is an instance of network class. The key data member is the information matrix, which is a two-dimensional array containing coordinates (x, y) of all nodes.

4.2 Operation class

Quantum genetic algorithm class: this class defines the whole process of solving problems with QGA. Its main function is to accomplish the algorithm above.

5 Simulation experiments

Conventional genetic algorithm (CGA) is compared with quantum genetic algorithm in our simulation experiment. The subject is a sensor network with 20 randomly generated nodes. In order to eliminate the influence on the assessment of algorithms caused by randomness, an average of 100 simulation results are considered.

The sensor network topology generated by the conventional genetic algorithm is shown in Fig. 1.

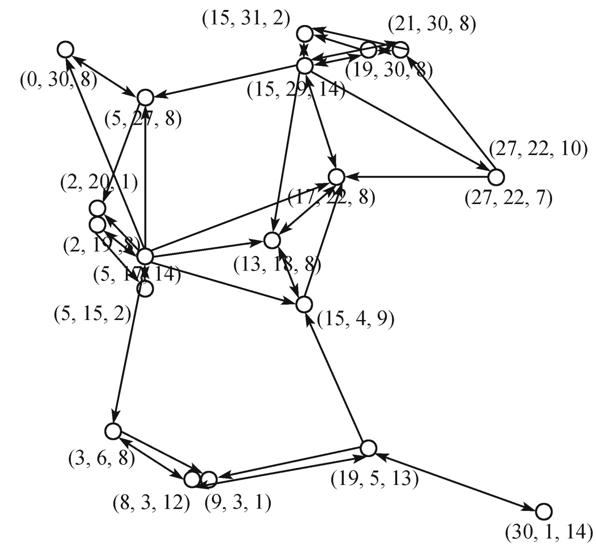

Fig. 1 Topology generated by CGA

The sensor network topology generated by the quantum genetic algorithm is shown in Fig. 2.

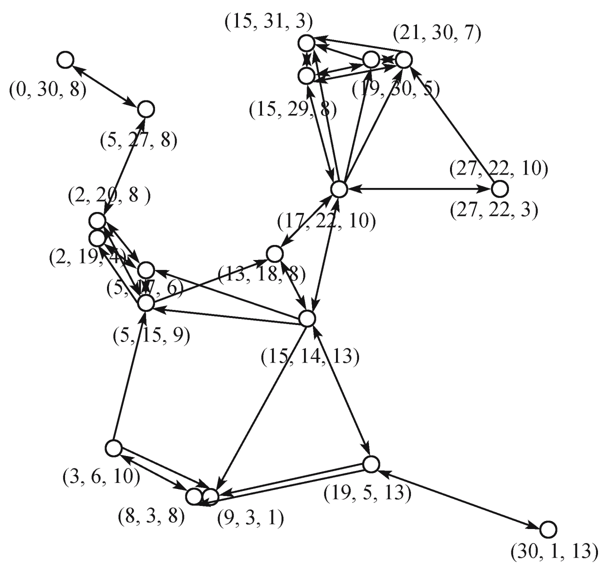

Fig. 2 Topology generated by QGA

Comparative results are as follows:

1) Comparison of computing performance

The success rates are shown in Table 2. Maximum generation of evolution is set to 90. If the optimal solution is found by the algorithm in the limit of the evolutionary generation set before, then it is considered successful. The success rate of QGA is higher than that of CGA.

Table 2 Comparison between success rates of algorithms

Algorithm	Success rates/%
CGA	54
QGA	99

2) Comparison of results

Table 3 shows the average of solutions. It can be found that the optimal solutions are found in fewer generations by QGA than by CGA, and the solutions are better as well.

Table 3 Comparison between results of algorithms

Algorithm	Earliest generation finding the solution	Ending generation	Total power the radius threshold	Nodes exceeding	Fitting degree
CGA	47.0	74.0	175	10	86.7
QGA	28.1	44.3	167	8	91.5

3) Comparison of optimal solution

Table 4 shows the comparison of optimal solutions gained by different algorithms. The optimal solution by QGA is better than that by CGA.

Table 4 Comparison between optimal solutions

Algorithm	Total power	Nodes exceeding the radius threshold	Fitting degree
CGA	163	7	89.6
QGA	155	7	93.0

6 Conclusion

QGA features fast convergence and short computing. Therefore, it can be widely applied to such situations as energy saving, routing and QoS supporting in sensor networks.

In this paper, QGA is adopted to solve the problems of topology control, and a good result is achieved. It not only extends the life of the network, but also ensures a route. In future works, we will improve our algorithm by taking in power control, and balance the load of all nodes in the generated network. Thus, fairness can be secured and the arrival of network splitting can be delayed as much as possible.

Acknowledgements The work was supported by the National Natural Science Foundation of China (Grant No. 60573141 and 70271050), the Natural Science Foundation of Jiangsu Province (No. BK2005146), the High Technology Research Program of Jiangsu Province (No. BG2004004 and BG2005038, BG2006001), the Hi-Technology Research and Development Program of China (No. 2006AA01Z219), Foundation of National Laboratory for Modern Communications (No. 9140C1101010603), the High Technology Research Programme of Nanjing (No. 2006RZ105), the Foundation of Jiangsu Key Laboratory of Computer Information Processing Technology (No. kjs050001 and kjs0606).

References

1. Zheng Shaoren, Wang Haitao, Zhao Zhifeng, et al. Ad-hoc Network Technology. Beijing: Posts & Telecom Press, 2005 (in Chinese)
2. Zhang Jian, Yang Hui, Zhu An. Topology controlment based on genetic algorithm in static Ad-hoc network. Journal of Wuhan University (Natural Science Edition), 2005, 51(3): 327–333 (in Chinese)
3. Wang Haitao. Architectures of Ad-hoc network and its designs. China Data Communications, 2003, 8: 70–76 (in Chinese)
4. Xiao Yongkang, Shan Xiuming. Wireless Ad-hoc network and its research challenges. Telecommunications Science, 2002, 6: 12–14 (in Chinese)
5. Xu Binhui, Zhou Keqin. Study on Ad-hoc networking. China Data Communications, 2002, 7: 66–70 (in Chinese)
6. Xie Guangjun, Zhuang Zhengquan. Quantum neural network. Computer Science, 2001, 28(7): 1–6 (in Chinese)
7. Han K H. Genetic quantum algorithm and its application to combinatorial optimization problems. In: Proceedings of the 2000 IEEE Congress on Evolutionary Computation. Piscataway: IEEE Press, 2000, 1354–1360
8. Narayanan A, Moore M. Quantum inspired genetic algorithm. In: Proceedings of the 1996 IEEE International Conference on Evolutionary Computation (ICEC96). Piscataway: IEEE Press, 1996, 61–66
9. Yang Jun'an, Zhuang Zhenquan. Research of quantum genetic algorithm and its application in blind source separation. Journal of Electronics (China), 2003, 20(1): 62–68 (in Chinese)
10. Yang Jun'an, Zhuang Zhenquan. Actuality of research on quantum genetic algorithm. Computer Science, 2003, 30(11): 13–15 (in Chinese)