Front. Electr. Electron. Eng. China (2006) 4: 445–450
DOI 10.1007/s11460-006-0085-1

RESEARCH ARTICLE

BAI Rong-gang, QU Yu-gui, LIN Zhi-ting, WANG Qing-hua, ZHAO Bao-hua

Tracking the events in the coverage of wireless sensor networks based on artificial neural-networks algorithms

Abstract Sensor deployment is an important problem in mobile wireless sensor networks. This paper presents a distributed self-spreading deployment algorithm (SOMDA) for mobile sensors based on artificial neural-networks self-organizing maps algorithm. During the deployment, the nodes compete to track the event and cooperate to form an ordered topology. After going through the algorithm, the statistical distribution of the nodes approaches that of the events in the interest area. The performance of the algorithm is evaluated by the covered percentage of region/events, the detecting ability and the energy equalization of the networks. The simulation results indicate that SOMDA outperforms uniform and random deployment with lossless coverage, enhancive detecting ability and significant energy equalization.

Keywords wireless sensor network, coverage, artificial neural-networks, self-organizing maps algorithm, genetic algorithm

1 Introduction

One fundamental issue in wireless sensor networks (WSN) is its deployment, which is a primary metric that provides an indication regarding quality-of-service. Recently, much research has been done about various issues related to the deployment problem. Most approaches use either a centralized solution as in the circle covering problem and the geometric problem [1–4]. Some scenarios consider uniformity as an important metric [5], and some algorithms' objective is to minimize the number of active nodes and the number of uncovered demand points in the area [6]. However, few or none of these studies note that placing the sensors in the region without events is wasteful, or the distribution of events may change every now and then. Uniformity may not provide more equal energy consumption or more exact events locations, and prearrangement of the topology couldn't adapt to the variation of the environment [6]. It seems that in some scenarios sensors should be deployed according to the distribution of events.

The purpose of this paper is to present a distributed self deployment algorithm for mobile sensors based on self-organizing maps algorithm (SOM) [7], which is classical in the artificial neural-networks area.

Our algorithm aims to improve the deployment for more exact event location and more equal energy consumption in the networks. We provide a competition and cooperation mechanism at each sensor. During the self-organization, the nodes organize themselves in an adaptive manner to track the event and form an ordered topology.

In the next section, we present our distributed self-spreading algorithm. Another required algorithm arranging the nodes into a lattice is presented in Sect. 3 followed by simulation results and performance estimation in Sect. 4. Some concluding remarks are provided in Sect. 5.

BAI Rong-gang, QU Yu-gui, LIN Zhi-ting, WANG Qing-hua
Department of Electronic Engineering and Information Science,
University of Science and Technology of China,
Hefei 230027, China

ZHAO Bao-hua(✉)
Department of Computer,
University of Science and Technology of China,
Hefei 230027, China
E-mail: bhzhao@ustc.edu.cn

Received date: July 16, 2006

2 Self-organizing map deployment algorithm

In this paper we will present a possible deployment implementation of the neural-networks SOM algorithms, which can be easily adopted to wireless sensor network platforms and will meet the requirements for sensor networks like: simple parallel distributed computation and data robustness [8–11]. As a result of the best features for the underlying distribution obtained from the input events, lower communication costs and more exact search can be acquired.

2.1 Introduction to self-organizing maps algorithm

The theoretical rationality of our algorithm is the SOM, which is an unsupervised learning method for categorization in artificial neural-networks. In SOM, the neurons are placed at a lattice that is usually one- or two-dimensional. The neurons become selectively tuned to various input patterns in the course of a competitive learning process. The locations of the neurons so tuned (i.e., the winning neurons) become ordered with respect to each other in such a way that a meaningful coordinate system for different input features is created over the lattice [12]. A self-organizing map is therefore characterized by the formation of a topographic map of the input patterns in which the spatial locations of the neurons in the lattice are indicative of intrinsic statistical features contained in the input patterns.

2.2 Assumptions and restrictions

The assumption in this paper is that all sensor nodes have capabilities for sensing, communication, computation and mobility. Sensing and communication coverage areas of each node are assumed to be circular and equal. The events don't occur frequently in the network. The nodes could approximately estimate the distance and direction to the neighbor and event. Another assumption is that only a few nodes instead of all know their location by the method such as GPS [13]. These assumptions could be satisfied by most practical networks.

2.3 Algorithm detail

We call the deployment algorithm the SOMDA. In the algorithm, the inputs are the locations of the events, and the outputs are the locations of the sensor nodes.

To begin with, a specified number of nodes are uniformly distributed in a given region, for instance, inside a rectangle. The sensing range (R_s) and the communication range (R_c) are assumed to be given. The relation between sensing and communication ranges is discussed by Wang et al. [14]. By geometric analysis, they conclude that, when the sensor range is less than half of the radio range, if we assure the area coverage we also assure the network connectivity. Therefore, we do not consider connectivity in our work. After the initialization, there are four essential processes involved in SOMDA, as summarized here:

1) Competitive process When an event happens, the nodes (j=1, 2, ⋯, k) in R_s will detect it and compute their respective values of a discriminant function, which is defined as the distance from the nodes to the event according to Eq. (1). The Euclidean distance (D_j) can be estimated by the sensing intensity of the event.

$$D_j = \left\| \boldsymbol{X} - \boldsymbol{W}_j \right\| \tag{1}$$

where $\boldsymbol{X}$ stands for the location of the event, $\boldsymbol{W}_j$ stands for the location of jth node, n denotes the time step. They are both two-dimensional coordinates vectors. Sensing information with timestamp (n) (i.e., D_j) would be exchanged in the neighborhood. After receiving packets from its neighborhood, a node can identify the winner by searching for the minimum D_j. If we use the index $i(\boldsymbol{X})$ to identify the node that best matches the input vector $\boldsymbol{X}$, we may then determine $i(\boldsymbol{X})$ by applying the condition:

$$i(\boldsymbol{X}) = \arg\min_j \left\| \boldsymbol{X} - \boldsymbol{W}_j \right\| \tag{2}$$

Node i is called the winning node for the input vector $\boldsymbol{X}$.

2) Cooperative process In this process, the winning node determines the spatial location of a topological neighborhood of excited nodes, thereby providing the basis for cooperation among such neighboring nodes.

Let $h_{j,i(X)}$ denote the topological neighborhood centered on winning node $i(\boldsymbol{X})$, and encompass a set of excited node j, a typical one of which is denoted by j. Let $d_{i,j}$ denote the lateral distance between winning node i and excited node j. Then we choose the Gaussian function:

$$h_{j,i(X)}(n) = \exp\left(-\frac{{d_{j,i}}^2}{2\sigma^2(n)}\right) \tag{3}$$

$$\sigma(n) = \sigma_0 \exp\left(-\frac{n}{\tau_1}\right) \tag{4}$$

where $\sigma(n)$ is the width of the Gaussian function and it decreases as the time n increases. τ_1 and σ_0 are constants.

3) Adaptive process This last mechanism enables the excited node to adjust its location to be close to the input event. The adjustments made are such that the response of the winning and excited nodes to the subsequent application of an adjacent input event is enhanced. Such as SOM, we may express the position change of node j as follows:

$$\Delta\boldsymbol{W} = \boldsymbol{W}_j(n+1) - \boldsymbol{W}_j(n) = \eta(n) h_{j,i(X)}(n)[\boldsymbol{X}(n) - \boldsymbol{W}_j(n)] \tag{5}$$

$$\eta(n) = \eta_0 \exp\left(-\frac{n}{\tau_2}\right) \tag{6}$$

where $\eta(n)$ is the learning-rate parameter and it decreases gradually with increasing time n. τ_2 and η_0 are constants of the SOM algorithm.

After calculating the adjustment at current step, the excited nodes decide their next movement. As a result of SOMDA, the node location feature could reflect the statistics distribution of the input events.

If we assume the competitive, cooperative and adaptive processes as the base operation, the computational complexity is $O(n)$.

4) Stopping criteria When the nodes should stop their movements is an important issue according to the monitoring environment. If the event distribution is stable, the nodes should move in the self-organization stage and stop moving gradually. In this situation, we introduce an artificial stopping criteria, if every node moves less than L (e.g., L = 4) for the time duration S (e.g., S is the time consumed by 1 000 events in the networks), the network could be considered stable and stop moving.

3 Genetic algorithm

In SOMDA, the randomly distributed nodes need to be arranged in a two-dimensional lattice first. We consider 100 randomly placed nodes in a square region of size 100×100 as shown in Fig. 1. Let R_s = 12.5 and R_c = 25, thus every node has 16.28 neighbors on average.

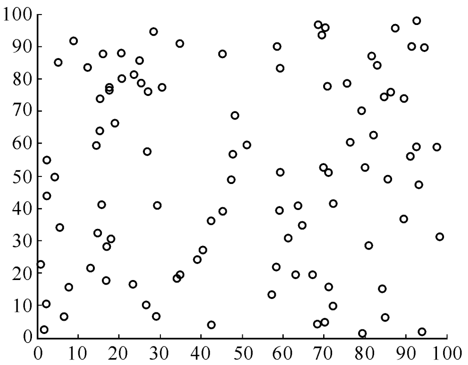

Fig. 1 Initial distribution of 100 sensor nodes

If each node randomly selects a position on the 10×10 lattice, the training of the networks would move the nodes over a long distance (Fig. 2).

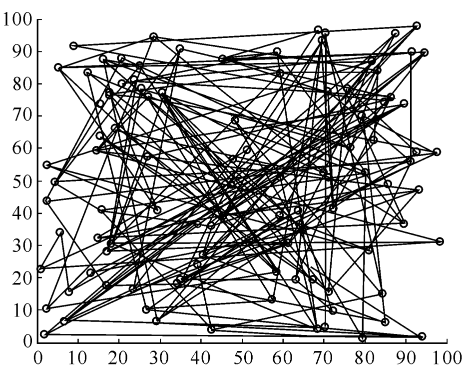

Fig. 2 Random arrangement in a two-dimensional lattice

To accelerate the training and shorten the distance of the movement, it is best to arrange the lattice-spatial location close to the actual position. We assume that only a few nodes (4 nodes in the corner) know their exact position, and every node knows their neighborhood. We use the centralized genetic algorithm (GA) to arrange the lattice. The evolutionary algorithm runs fast, and in many situations finds good solutions in a feasible time. The parameter of the GA is shown as follows:

1) Population: as the traveling salesman problem (TSP) [15], we give each node a numeric name, e.g., 1, 2, 3, ⋯, 100, which represents its position in the lattice (e.g., number 37 denotes row = 3, column = 7); the candidate is made of an ordering of these 100 digits corresponding to a topology of the lattice. Then randomly generate an initial population $M(20)$ with 20 candidates. The four corner nodes choose 1, 10, 91, 100 number immovably.

2) Fitness function: for the nodes only knowing their neighbors, we define:

$$\text{fitness} = K - \sum_{i=1}^{100} \sum_{j \in L(i)} N_{\mathrm{D}}(i, j) - \sum_{i=1}^{100} \sum_{j \in N(i)} L_{\mathrm{D}}(i, j) \tag{7}$$

where $L(i)$ is the lattice neighbors set of node i. $N(i)$ is the natural neighbors set of node i. $N_{\mathrm{D}}(i, j)$ is the natural distance between node i and j. $L_{\mathrm{D}}(i, j)$ is the lattice distance between node i and j. K is a constant. The minimum of the fitness function can approximately satisfy the condition that the lattice–spatial location is close to the actual position.

3) Select operator: as a natural evolution, the reproducing probabilities $p(m)$ for each individual m is proportional to fitness(m).

4) Crossover operator: as the order crossover [16], pieces of subsequences are passed on from one parent, p_1, to a child, c_1, while the ordering of the remaining pieces of c_1 is inherited from the other parent, p_2. These pieces of subsequence c_1 inherited from p_1 are some lattice positions of a set of nodes and their neighbors.

5) Mutation: randomly exchange two numbers in the ordering of the individual with probability 10 %.

6) Satisfying solution: repeat the circle for 2 000 rounds. The solution can be approximately assumed to be the best one.

7) Result: as shown in Fig. 3, a good result has been obtained where the lattice–spatial location of the node is close to its actual position.

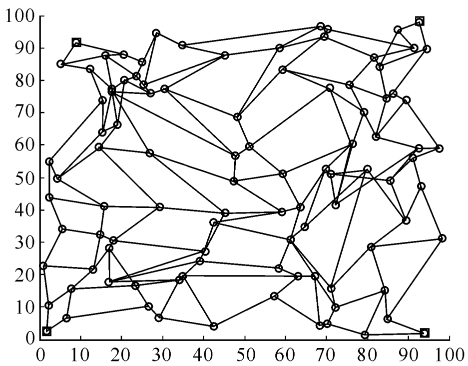

Fig. 3 Arrange 100 nodes in the lattice with GA

4 Simulation results

4.1 The coverage after running SOMDA

We consider the same initialized topology and region in Sect. 3. The nodes had already been arranged into a 10×10 lattice as shown in Fig. 3. We test the effect of SOMDA in two scenes: events distribute uniformly and irregularly. In the first scene (Fig. 4(a)), the points represent the uniform distribution of the events. In the second scene (Fig. 4(b)), the events are sparse in some parts and dense in some others. This is exactly the case where coverage improvement is required.

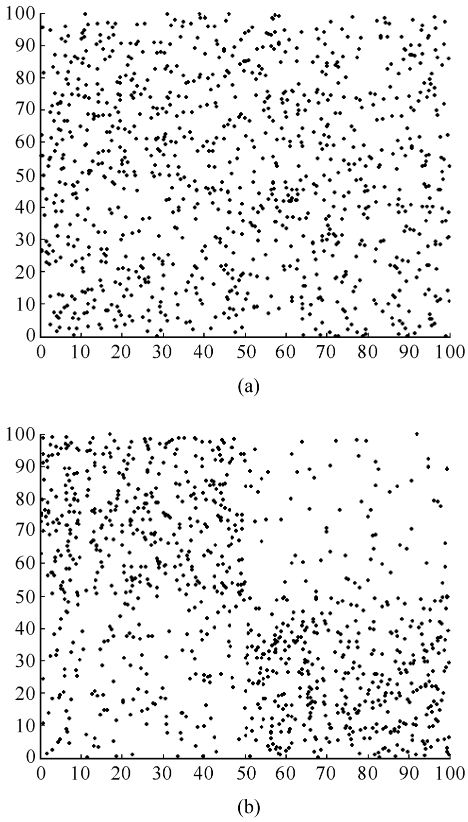

Fig. 4 Distribution of 1 000 events. (a) Uniform distribution of 1 000 events; (b) Irregular distribution of 1 000 events

The essential ingredients/parameters of the SOMDA are:

1) Initialization Place 100 nodes in a square region randomly and arrange them into a lattice with GA described in Sect. 3. Generate a continuous input space of activation patterns in accordance with a certain probability distribution such as Fig. 4. Define a time-varying neighborhood function $h_{j,i(\boldsymbol{X})}(n)$ around a winning node $i(\boldsymbol{X})$ as Eq. (3).

2) Ordering phase There are 10 iterations in this stage. During the phase, $\sigma(n)$ starts with an initial value σ_0=8 and then shrinks to about 1, and $\eta(n)$ starts with an initial value η_0 = 0.1 and then decreases to 0.003 7. These desirable values are satisfied by τ_1 = 4.55 and τ_2 = 10.

3) Convergence phase There are 500 iterations in this stage. During the phase, the $\sigma(n)$ starts with an initial value σ_0 = 8 and then shrinks linearly to 0, and $\eta(n)$ starts with an initial value η_0 = 0.1 and then decreases linearly to 0.

Figures 5(a) and (b) corresponding to Figs. 4(a) and (b) respectively show the topology after running the SOMDA. The statistical distribution of the nodes in the region approaches that of the input events, except for some edge effects. Comparing the final feature map with the input distribution, we see that the tuning of the map has captured the local irregularities in the input distribution.

In the true networks, since the nodes cannot measure the event's location very accurately, we introduce a 10 % error into the estimated distance. In this case, results of the training that the topology is extraordinarily similar to Fig. 5(b) shows significant data robustness of SOMDA.

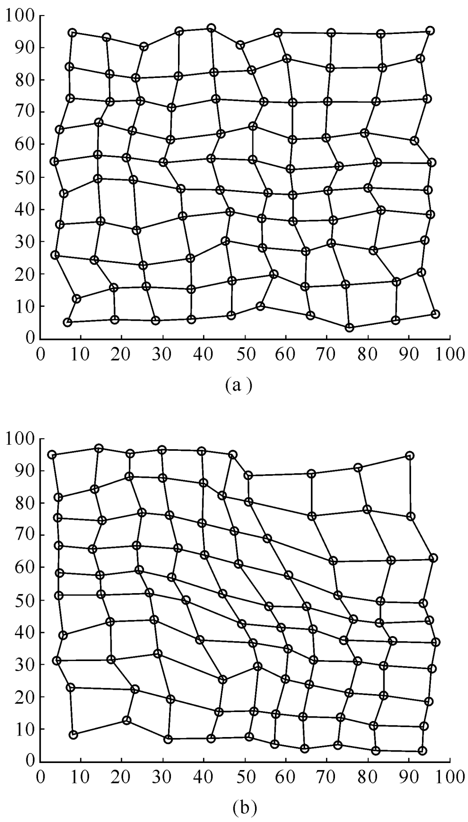

Fig. 5 (a) Deployment under uniform distribution; (b) Deployment under irregular distribution

4.2 The performance of SOMDA

The performance of our algorithm is evaluated in terms of four metrics: the percentage of region covered, the percentage of events covered, the inspecting ability of the events, and the equalization of the networks. The number of nodes varies from 16 to 225 in the same area and the results are averaged over 20 runs. We compare the performance of

SOMDA against perfect uniform (UNIFOM) and random deployment (RANDOM). UNIFOM is the objective of Ref. [5].

Figure 6 shows the percentage of region covered under different network size. When network size is small, the performance of SOMDA is a little worse than UNIFOM, but it performs as well as UNIFOM when the network is large enough. In the next three tests, 10 000 events happen in the area following the distribution shown in Fig. 4(b) and the other parameters are the same as the sections before.

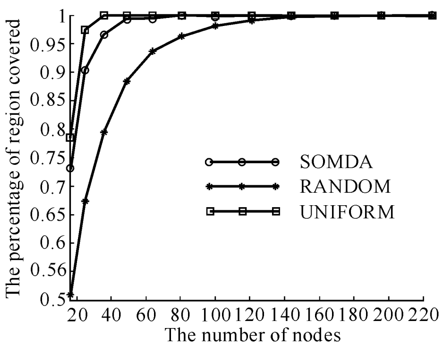

Fig. 6 The coverage of area vs. network size

Figure 7 shows the percentage of events covered under different network sizes. Both SOMDA and UNIFOM have quite similar performance. SOMDA doesn't weaken the main function of the sensor networks inspecting events.

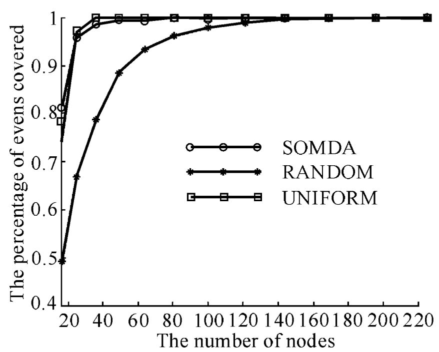

Fig. 7 The coverage of events vs. network size

The average number of the nodes detecting a certain event is important to detect the goal. Figure 8 shows that SOMDA has better performance than the others. This is helpful to sense and locate the event, and the advantage increases with the number of nodes.

The biggest advantage of our algorithm is the contribution to the equalization of the energy consumption. We suppose that the main energy consumption is the data transmission. The nodes that detect the event will fuse their similar data and send it with one node. When an event occurs, the nodes in the sensing range detect it and choose one of them to send it. After 10 000 events happening in the

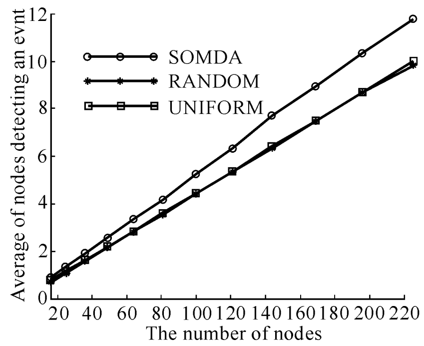

Fig. 8 Average of nodes detecting an event vs. network size

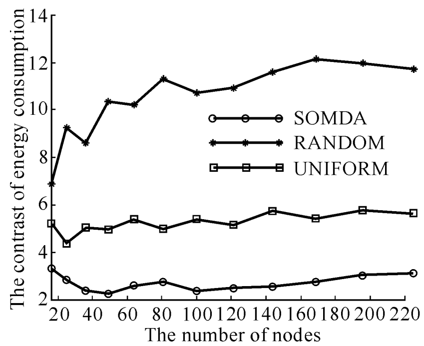

Fig. 9 Equalization of energy consumption vs. network size

area, the packets sent by every node were counted. The sum of the data sent by the maximum fifth of nodes divided by the minimum fifth is shown in Fig. 9. The lower value means the more equally the nodes consume. Therefore, SOMDA will keep the energy consumption of nodes more equal, and the networks life will be prolonged.

5 Conclusions

In this paper we have presented a possible adaptation of one popular artificial neural networks SOM algorithm in the coverage problem of wireless sensor networks. In SOMDA the nodes will be arranged into a lattice, then track the events and excite their neighborhood. Finally, the statistical distribution of the nodes in the region approaches that of the input events. The features of SOM algorithms such as parallel distributed computation, distributed storage, data robustness and density matching is suitable for the WSN platform. The training result of SOMDA provides lossless coverage of area or events but enhancive detecting ability in the same network size. Since only fused data are passed to the Base Station instead of crude, the SOMDA has more energy equalization than the uniform and random deployment when the events occur irregularly. This deployment idea will be more useful especially where initial events distribution is quite uneven or instable and redeployment is too costly or too risky.

Acknowledgements This paper was supported by the National Natural Science Foundation of China (No. 60241004), National Basic Research Program of China (No. 2003CB314801), the State Key Laboratory of Networking and Switching Technology.

References

1. Meguerdichian S., Koushanfar F., Potkonjak M. et al., Coverage problems in wireless Ad-hoc sensor networks, IEEE INFOCOM, 2001, 3: 1380–1387
2. Dhillon S. S., Chakrabarty K., Sensor placement for effective coverage and surveillance in distributed sensor networks, IEEE WCNC 2003, 2003, 3: 1609–1614
3. Lin Zhi-ting, Qu Yu-gui, Zhai Yu-jia, 3-dimensional sensor model in the sensor network, Chinese Journal of Electronics, APR 2006, 15 (2): 324–328 (in Chinese)
4. Lin Zhi-ting, Qu Yu-gui, Zhai Yu-jia, Placement algorithm for the wireless sensor network, Chinese Journal of Electronics, 2006, 15 (1): 179–182 (in Chinese)
5. Heo N., Varshney P. K., A distributed self spreading algorithm for mobile wireless sensor networks, IEEE WCNC, 2003, 3: 1597–1602
6. Quintao F. P., Nakamura F. G., Mateus G. R., Evolutionary algorithm for the dynamic coverage problem applied to wireless sensor networks design, the 2005 IEEE Congress on Evolutionary Computation, 2005, 2: 1589–1596
7. Kohonen T., The self-organizing map, Proceedings of the Institute of Electrical and Electronics Engineers, 1990, 78: 1464–1480
8. Elaine C., Kristof V. L., Martin S., Self-organization in Ad-hoc sensor networks: an empirical study, Proceedings of the Eighth International Conference on Artificial Life, 2002: 260–263
9. Kulakov A., Davcev D., Distributed data processing in wireless sensor networks based on artificial neural-networks algorithms, Proceedings of 10th ISCC, 2005, 353–358
10. Kulakov A., Davcev D., Trajkovski G., Application of wavelet neural-networks in wireless sensor networks, SNPD/SAWN 2005: 262–267
11. Kulakov A., Davcev D., Tracking of unusual events in wireless sensor networks based on artificial neural-networks algorithms, ITCC 2005, 2005, 2: 534–539
12. Simon H., Neural networks: a comprehensive foundation (second edition), Prentice-Hall, Inc, New Jersey: 1999
13. Kaplan E. D., Understanding GPS: principles and applications, Artech House, Inc, America: 1996
14. Wang X., Xing G., Zhang Y. et al., Integrated coverage and connectivity configuration in wireless sensor networks, SenSys '03: Proceedings of the 1st International Conference on Embedded Networked Sensor Systems, ACM Press (2003): 28–39
15. George F. L., Artificial intelligence: structures and strategies for complex problem solving (fifth edition), Pearson Education, Inc, Iowa: 2005
16. Oliver I. M., Smith D. J. , Holland J. R. C., A Study of permutation crossover operators on the traveling salesman problem, Proceedings of the Second International Conference on Genetic Algorithms, Hillsdale, NJ: Erlbaum & Assoc, 1987: 224–230