International Journal of Innovative
Computing, Information and Control
Volume **10**, Number **2**, April **2014**

 ISSN 1349-4198
pp. **811–822**

DYNAMIC ROUTE GUIDANCE ALGORITHM BASED ON IMPROVED HOPFIELD NEURAL NETWORK AND GENETIC ALGORITHM

Na Lin and Hongdong Liu

School of Computer
Shenyang Aerospace University
No. 37, Daoyi South Ave., Daoyi Development Dist., Shenyang 110136, P. R. China
lin_na2000@163.com; 915981598@qq.com

Received April 2013; revised August 2013

Abstract. *To meet the needs of the traffic dynamic change in the dynamic route guidance system, aiming at the inherence weaknesses of the Hopfield neural network (HNN) which easily falls into local optimum and converges slowly, a method of applying genetic operator to the HNN is proposed to guarantee the global searching capability of the network. At the same time, dynamic learning rate is introduced into the HNN to accelerate the convergence rate of the neural network. The simulation experiments are composed of two parts. One is about the oliver30 problem where our algorithm is compared with traditional ones. It turns out that the algorithm proposed in this paper can easily gain a better optimal solution. The other is conducted in the MapX5.0+VC6.0 environment Under simulated real traffic situation our algorithm can get the routes which meet the requirements of drivers better.*

Keywords: Urban transportation, Hopfield neural networks, Genetic algorithms, Global optimization, Travel time, Dynamic route guidance

1. **Introduction.** "12th Five-Year Plan" in China is the critical period to promote the intelligent transportation. The coastal ports, highways, rural roads and other transport infrastructure constructions have made considerable developments. The road transport ability continues to be improving. As a consequence the frequent traffic congestion, traffic accidents and environmental pollution are becoming so serious in urban transportation that caused great distress to people's daily traffic, life and production. The urban transportation problems need to be solved as soon as possible [1]. By providing drivers with real-time traffic information, the dynamic route guidance system ensures the purpose of inducing trip and the vehicles to follow a reasonable, comfortable and efficient travel route with minimal total cost [2,3].

The problem of selecting the optimal path in dynamic route guidance system means to search for the optimal path between one origin and one destination in a given transport network and make the path-related total cost at minimum [4]. It is a typical combinatorial optimization problem. Due to the complexity and uncertainty of traffic flows in the urban traffic network, some algorithms of artificial intelligence are applied to studying on dynamic route guidance [5,6]. These intelligent algorithms include artificial neural network, evolutionary computation, swarm intelligence, artificial immune system and fuzzy system. And each of them has its own advantages and disadvantages and faces the inherent conflict of cost-optimization and time-performance. So far many works have been done and it is proved that there is not a single best method suitable for all situations in dynamic route guidance. Some of them focus only on limited evaluation criteria and regard the shortest path as the path which costs the shortest time or the shortest distance

[7]. Moreover, when comes to large-scale real-time transportation network system, the calculating time of certain methods turns to be very long and it is difficult to meet the needs of traffic flow guidance [8]. In this paper, we have found a way to combine two methods which well complement each other, the Hopfield Neural Network (HNN) and the Genetic Algorithm (GA), and have worked out a better solution for given situations in dynamic route guidance within a period of time acceptable. The research and analysis of the HNN and GA is as follows.

1.1. **Hopfield Neural Network.** Using Artificial Neural Network (ANN) to solve the combinatorial optimization problems with constraints is started by Hopfield and Tank. After they established Hopfield Neural Network (HNN) Model, they effectively addressed the TSP Issue, one of the NP Problems. Thus, HNN is considered to be one of the most effective ways to solve complex combinatorial optimization problems. HNN is a non-programming (or even a hardware implementation) which works in parallel and is able to learn. It also has a good degree of robustness and adaptive capacity. However, HNN itself has some shortcomings. The very first one is that the optimal solution it gets is likely to be a local optimal solution rather than a global one. Literature [9] applies Z-score model and logarithmic method to the Hopfield network in order to train and get a more appropriate input. Literature [10] proposes an adaptive weight learning method to adjusting the weighting coefficients dynamically.

1.2. **Genetic Algorithm.** Genetic Algorithms (GA) is an algorithm simulating biological evolution proposed by John Holland in 1975 [11], which is a calculation model simulating the natural selection of Darwinian biological evolution and the evolution process of the genetics mechanism. Its main advantages are the ability of global searching and the ease of expansion and little special knowledge is needed when applied, while the shortcomings are premature convergence, low computing speed, and the lack of local optimizing capacity [12]. Literature [13] proposes an inheritance pattern which uses adjacent crossover strategy, greedy forward mutation policy and steadily reproducing. It increases the population diversity which has a certain effect on avoiding premature convergence.

1.3. **The basic idea of hybrid optimization strategy of HNN and GA.** The principle of combining HNN and genetic algorithm to make the best of both for efficient global optimization capability is shown in Figure 1.

First of all, get a local optimal solution by HNN accelerating algorithm. Secondly do the genetic operations on it in order to jump out of the neighborhood of local optimal solution and increase the diversity of solutions. Then, make use of the partial and quick optimization capacity of the accelerated HNN algorithm to search and generate a new

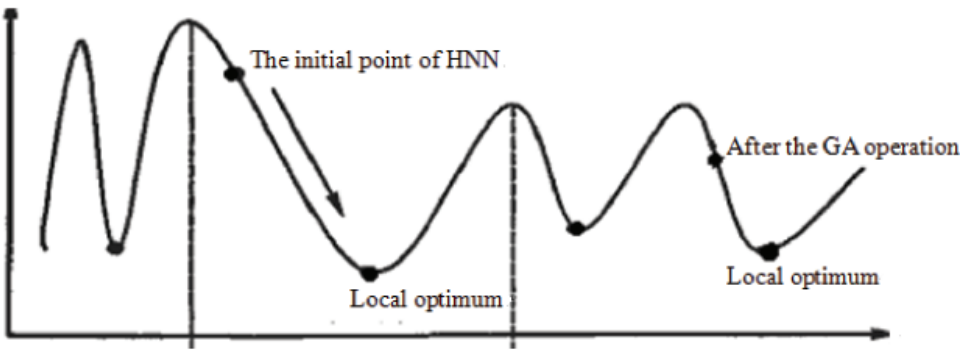

FIGURE 1. Algorithm schematic diagram

local optimal solution. Finally, compare the new local optimal solution with the historical local optimal solution to eliminate the less optimal one. After varying N generations a relatively stable solution will be generated.

2. The Hybrid Optimization Strategy Based on HNN and GA.

2.1. A description of the problem.

A given network is defined as a directed graph $G(N, A)$, N is the collection of n vertices A is the collection of m sides. Corresponding to each edge (i, j) from the point i to the point j, define a road resistance matrix $c = [c_{ij}]$ which means the generalized road resistance from point i to point j. If point i is not connected to point j then $c_{ij} = \infty$. P_{st} represents the route from the origin s to the destination t: $P_{st} = (s, i, j, k, \ldots, r, t)$, the total road resistance $tc_{st} = c_{si} + c_{ij} + c_{jk} + \ldots + c_{rt}$. Solving the optimal path problem means to look for the path with the smallest resistance P_{st}. Define a $n \times n$ matrix $V = [v_{ij}]$, n as the number of nodes. If the edge from point i to point j is part of the optimal path then $v_{ij} = 1$ and others are 0. Because each point in the shortest path cannot be used more than once, the "1" in each row and each column of the matrix V cannot be more than one, and the diagonal elements are all 0. To make the total road resistance of the path minimum, the objective function should be $\min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} v_{ij}$ to make sure the path from the origin s to the destination t is continuous then

$$\text{s.t.} \sum_{k=1,k\neq i}^{n} v_{ik} - \sum_{l=1,l\neq i}^{n} v_{li} = \begin{cases} 1 & i = s \\ -1 & i = t \\ 0 & \text{others} \end{cases} \tag{1}$$

2.2. Improved Hopfield Neural Network model.

2.2.1. *HNN and dynamic network map.* Using neural network to solve optimization problems is mostly to find the energy function that can describe the actual problems, and to achieve the optimum purpose through the adjustment of parameters. Make each element of the above-mentioned matrix V represents a neuron to get a $n \times n$ neuron matrix, and define the $\beta = [\beta_{ij}]$ matrix if there is no connection between point i and point j, then $\beta_{ij} = 1$ and others 0. According to all these we construct energy function E:

$$E = \frac{A}{2} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j\neq i}}^{n} c_{ij} v_{ij} + \frac{B}{2} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j\neq i}}^{n} \beta_{ij} v_{ij} + \frac{C}{2} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j\neq i}}^{n} v_{ij}(1 - v_{ij}) \tag{2}$$

where in A, B, C are all parameters, v_{ij} is the output of the Neuron (i, j). The first item is the target function of the shortest path problem which makes the total road resistance from the origin s to the destination t minimal the second item is to delete the side which does not exist, so as to ensure each row and each column of the neuron matrix has only one "1" at most, and the rest are 0 the third item makes sure the output eventually converges to 0 or 1. Thus, the minimum value of the energy function E corresponds to the global optimal solution of the optimal path.

2.2.2. *The principle of improving HNN.* As the same as the discrete Hopfield Network, Hopfield defined a computational energy function [10] with the meaning of Lyapunov for the continuous model:

$$E = -\frac{1}{2} \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j\neq i}}^{n} w_{ij} v_i v_j + \sum_{i=1}^{n} v_i \theta_i + \sum_{i=1}^{n} \frac{1}{\tau} \int_0^{v_i} f^{-1}(x) dx \tag{3}$$

where w_{ij} is the connection weight between the neuron i and neuron j, θ_i is the threshold, function $f(x)$ is a continuous S-type function.

From Formula (3) we can see that:

$$\frac{dE}{dv_i} = -\left(-\frac{u_i}{\tau} + \sum_{\substack{j=1 \\ j \neq i}}^{n} w_{ij}v_j - \theta_i\right) = -\frac{du_i}{dt} \tag{4}$$

where τ is the time and constant u_i is the input of the neuron i.

In the process of neurons updating, the i-th neuron updates itself in accordance with the dynamic equation:

$$u_i(t + \Delta t) = u_i(t) + \frac{du_i(t)}{dt} \times \Delta t \tag{5}$$

Obtained by the inverse function:

$$\begin{aligned} \Delta f^{-1}(v_i) &= f^{-1}(v_i(t + \Delta t)) - f^{-1}(v_i(t)) \\ &= u_i(t + \Delta t) - u_i(t) \\ &= \frac{du_i(t)}{dt} \times \Delta t \end{aligned} \tag{6}$$

$$\frac{dE}{du_i} = \frac{dE}{dv_i}\frac{dv_i}{du_i} = -\frac{du_i}{dt}\frac{dv_i}{du_i} = -\frac{\Delta f^{-1}(v_i)}{\Delta t} f'(u_i) \tag{7}$$

where in $v_i = f(u_i)$ is a Sigmoid Function (monotonic raising function), namely when $f(u_i) > 0$, its inverse function $u_i = f^{-1}(v_i)$ is a monotonic increasing function. From Formula (5) we can see the increment of the network input u_i with respect to the change of the energy function is a declining gradient, which provides a theoretical basis for improving its convergence rate.

2.2.3. *Improving the HNN algorithm.* To accelerate the convergence means to set the learning speed according to the situation. So we introduce the dynamic convergence rate η to determine convergence rates of the neurons.

When the convergence rate needs increasing, η will be expanded α times and $\alpha > 1$. When the convergence rate needs decreasing, η will be reduced β times and $0 < \beta < 1$. Where in α, β will be adjusted according to the experimental results. Here is the expression:

$$\eta_i(t) = \begin{cases} \eta_i(t-1) \times \alpha & \frac{dE}{du_i}(t-1) \times \frac{dE}{du_i}(t) > 0 \\ \eta_i(t-1) \times \beta & \frac{dE}{du_i}(t-1) \times \frac{dE}{du_i}(t) < 0 \\ \eta_i(t-1) & \text{others} \end{cases} \tag{8}$$

Adjustment rules:

(1) When the energy function and the partial derivative output consecutively have the same symbol twice, it indicates that energy is going down and convergence rate needs increasing.
(2) If two continuous energy function's partial derivative changes its symbol, it indicates that the step length is too long and it skips the extreme point. Convergence rate needs reducing and the process should go back to the previous state.

When the step length changes, the change of neuron i's internal state is as follows:

$$\Delta u_i(t) = \eta_i(t) \times \left(\frac{du_i(t)}{dt} \times \Delta t\right) \tag{9}$$

The output of the neuron i is:

$$v_i = f(u_i(t-1) + \Delta u_i(t)) \tag{10}$$

2.3. **Genetic algorithm.** As the genetic algorithm does not require the target function to be continuous, so its search is global. That means it can easily get a global optimal solution or a second-best solution that performs well. So we can make use of its global searching capability to optimize the local optimal solutions generated by the HNN.

(1) **Chromosome:** A path contains many nodes. So the chromosomes can vary in length and each gene represents a node. Take integer values which represents the node order as gene values – does not apply to binary encodings, then the first gene represents the starting node and the last gene represents the target node.

(2) **Population initialization:** Under the restrictions of the starting node represented by the first gene and the destination node represented by the last node, randomly generate individuals [14].

(3) **Fitness function:** As the smaller energy function in HNN is, the less the path cost is and the better the fitness is. So we use 100/E as the fitness function of chromosomes.

(4) **Selection operator:** Apply to the fitness value rank selection. First of all do the sorting according to the values of the fitness. Then rank each individually. The first place is for the optimal solution and the second place is for the second-best solution and so on. For example, if the choosing probability $P = 0.5$, in accordance with this method, the probability distributions of the individuals are $(0.5)^1 = 0.5$, $(0.5)^2 = 0.25$, $(0.5)^3 = 0.125$ [15]. Stochastic universal sampling is applied to as the population sampling method in order to avoid great floating of the number offspring.

(5) **Crossover operator:** In order to avoid infeasible solution after crossover, we use heuristic crossover operation. Namely the crossover operation is not blind and it can improve the searching results efficiency. Process: scan parents to see if there are identical gene pairs. If there is then keep these gene pairs in the offspring. Otherwise, check whether there are corresponding gene pairs in the parents whose weight is not zero. Namely this gene pair is connected. If such gene pair exists, randomly select a pair and do the single-point crossover and obtain the offspring [16].

(6) **Mutation operator:** The offspring generated from singlepoint mutation may be infeasible. So we randomly select two nodes from parents' chromosomes which are neither the origin nor the destination and substitute a randomly generated path between the two nodes for the parents' corresponding parts to produce new offspring. For example, the father individual is 2, 8, 10, 15, 14 we randomly select two nodes which do not contain 2 and 14, such as (8, 10, 15) that may vary into (8, M, 15), then after the mutation, chromosome becomes 2, 8, M, 15, 14 which ensures the connectivity of a path.

(7) **Patching algorithm:** Offspring obtained after crossover and mutation may contain duplicate nodes, so we need to run the patching algorithm to eliminate duplicate nodes. For example, if the chromosome after mutation is:

$$29,\ 0,\ 2,\ 1,\ 4,\ 6,\ 8,\ 9,\ 2,\ 15,\ 14,\ 11,\ 3$$

where there is a duplicate node "2" then the chromosome becomes: 29, 0, 2, 15, 14, 11, 13 after running the patching algorithm.

2.4. **Process of the GAHNN dynamic route guidance algorithm.** The process of the GAHNN is as follows:

(1) Randomly generate X paths along from the origin to the destination, which forms a certain scale of initial population by means of integer coding.
(2) Calculate the fitness value of each chromosome.

(3) Judge whether it is the end of the GA stage or not. Decide when fitness reaches a certain value or iteration runs for a certain number of times, if yes, go to (5) and run the HNN local searching.
(4) Choose father generation on the basis of the selection operator adapted. Then do crossover and mutation to guarantee the diversity of population.
(5) Select m ones from the GA solutions as the initial values of HNN, selection principle: make sure the solutions are as global as possible and avoid selecting one solution repeatedly.
(6) Take the selected m solutions respectively as the initial values for HNN to create neuron matrix and the neural network state equation.
(7) Update the neural network state equation, and at the same time adjust dynamic convergence rate η basing on the symbols of two continuous energy function and the output's partial derivative.
(8) Judge if it is the end of HNN by whether the neural network gets stable in a certain number of times of iterating. If not, go to (5) and choose the initial values for HNN one more time.
(9) Compare the results acquired and choose the best one as the solution to the problem.
(10) Follow the mapping rule and the acquired best solution. Select the final route from the neuron matrix.

The flow chat of GAHNN algorithm is shown in Figure 2.

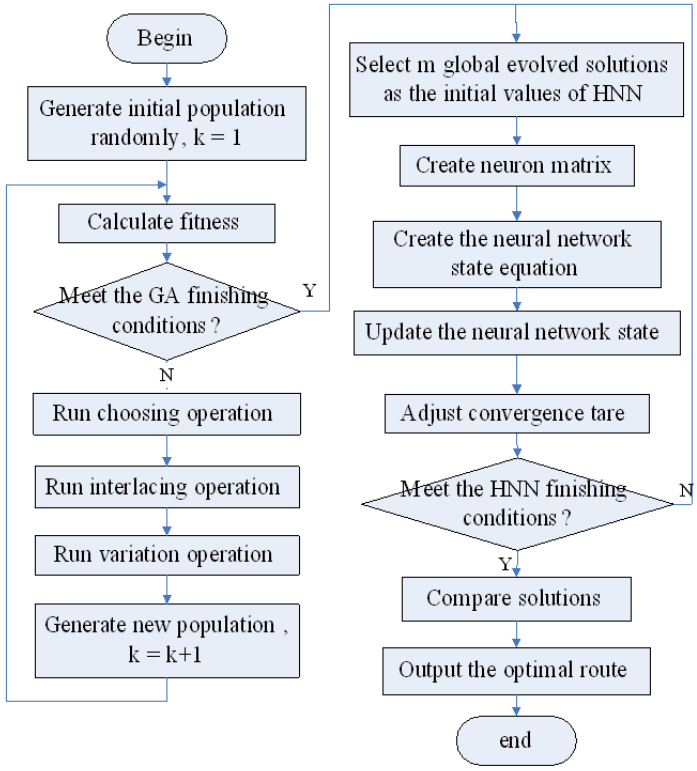

FIGURE 2. Flow chat of the GAHNN algorithm

3. Analysis of Simulation Instances.

3.1. **Simulation environment and network parameters.** The performance test of GAHNN hybrid optimization strategy algorithm is made of two parts: one is to solve the generally used standard oliver30 problem and to make comparisons among general GA

algorithm, HNN algorithm and the GAHNN hybrid optimization strategy proposed in this paper. The other is to verify the validity of our algorithm in the MapX5.0+VC6.0 environment. The hardware environment needed in this test is a PC with a processor of Intel(R) Pentium(R) Dual CPU E2180 @ 2.0 GHz and 2.0GB internal memory.

In all the tests, the HNN network parameters, such as A, B, C, μ_0 and so on are very sensitive to the change of network. Though Hopfield and Tank has already given the values: A = B = 500, C = 200, $\mu_0 = 0.02$, considering the impact of network parameters to the convergence rate, we actually choose A = B = 0.5, C = 0.2, $\mu_0 = 0.02$ in the program so as to reduce the difference of magnitude order of the energy function, thus decrease the value of emerges ΔE. As the judging basis of convergence of the program is this ΔE, choosing such values for the parameters precedes the convergence rate.

The crossover rate of the GA algorithm is chosen as 0.40 and the mutation rate 0.40 too. In the GAHNN hybrid strategy, the terminating condition of the GA searching is the fitness value is greater than 0.95 or the searching iterates over 50 times and the fitness value is greater than 0.90 [17]. When GA finishes, 10 sets of dissimilar solutions are orderly selected from the searching results by the fitness value as the initial values of the HNN so as to create a HNN network and do the optimizing respectively. At last we compare all the HNN optimized solutions and choose the best as the global optimal.

3.2. **Simulation results and analysis.**

3.2.1. *Oliver30 traveling salesman problem.* Solving the oliver30 TSP with GA, HNN and GAHNN hybrid strategy separately, the results are shown in Figures 3 to 5, wherein the coordinate axis represents only the numerical value with no dimension. In each figure there are 30 points representing 30 cities respectively with fixed coordinates. The point of the Oliver30 TSP is to find an optimal route with the minimal total route cost which will take the salesman to travel around all 30 cities without revisiting any. It can be seen from these figures that the GAHNN has obtained a better solution than the other two algorithms with fewer detours. Figure 6 shows how the iteration of each algorithm converges. As the times of iterating grow, the total route cost of all three algorithms decreases. The convergence rate of the GAHNN is a little slower than the GA and the HNN, but eventually it obtains a better total route cost than others.

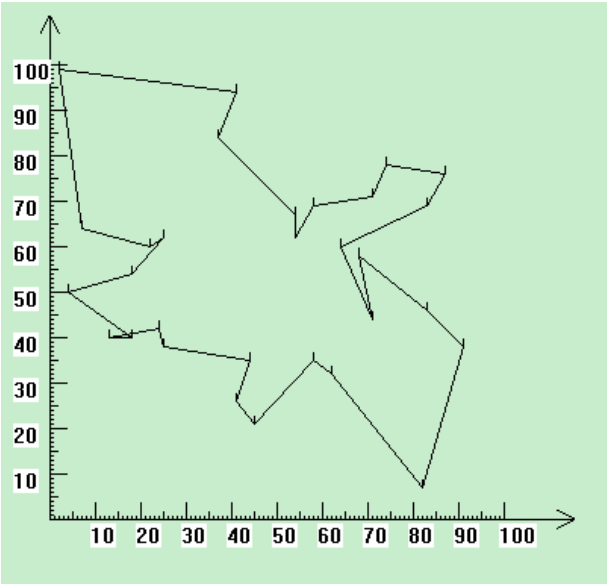

FIGURE 3. Optimal route GA obtained for the oliver30 TSP

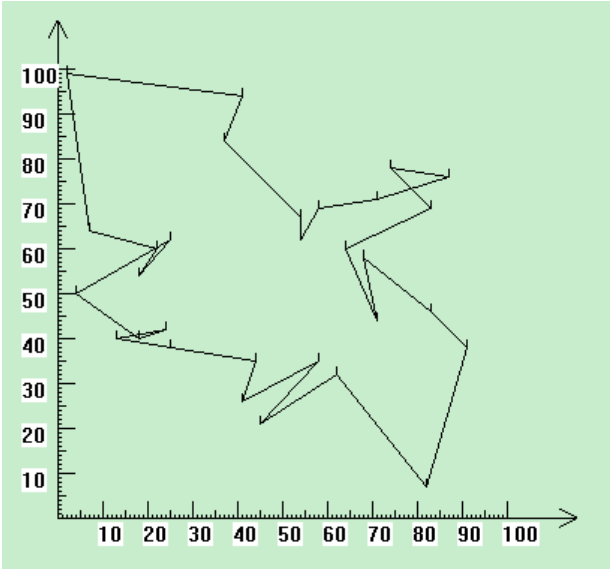

FIGURE 4. Optimal route HNN obtained for the oliver30 TSP

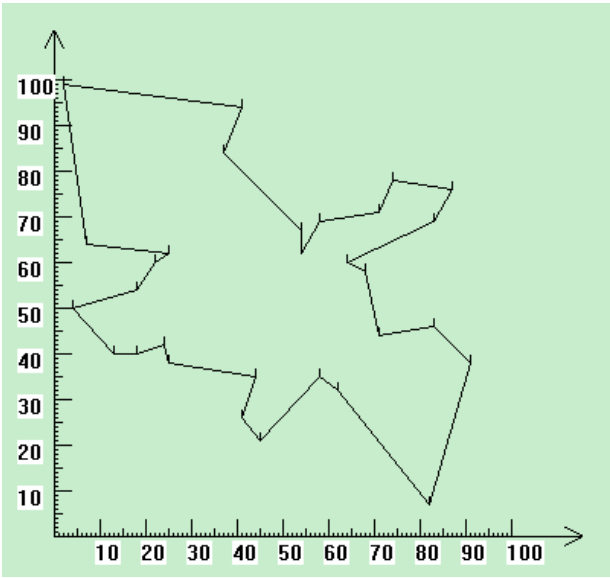

FIGURE 5. Optimal route GAHNN obtained for the oliver30 TSP

Table 1 shows the comparison results of the total route cost three different algorithms has achieved in the same experimental environment for many times. Dealing with the oliver30 TSP, the GAHNN algorithm has achieved an average total route cost of 423.74 which is better than the GA algorithm's 424.79 and the HNN algorithm's 426.60. Considering the convergence rate, it is clear from Table 1 and Figure 6 that three algorithms all converges after over 47 times of iterating. The GAHNN algorithm has computed 0.16 seconds more than GA and 0.262 seconds more compared with the HNN. Such increase on computing time is difficult to be sensed by users. The simulation results show that the GAHNN algorithm proposed in this paper has obtained a better total route cost of the optimal solution by sacrificing certain time. It is proved that our algorithm is able to acquire a better solution compared with the GA or the HNN, and is of better global searching ability.

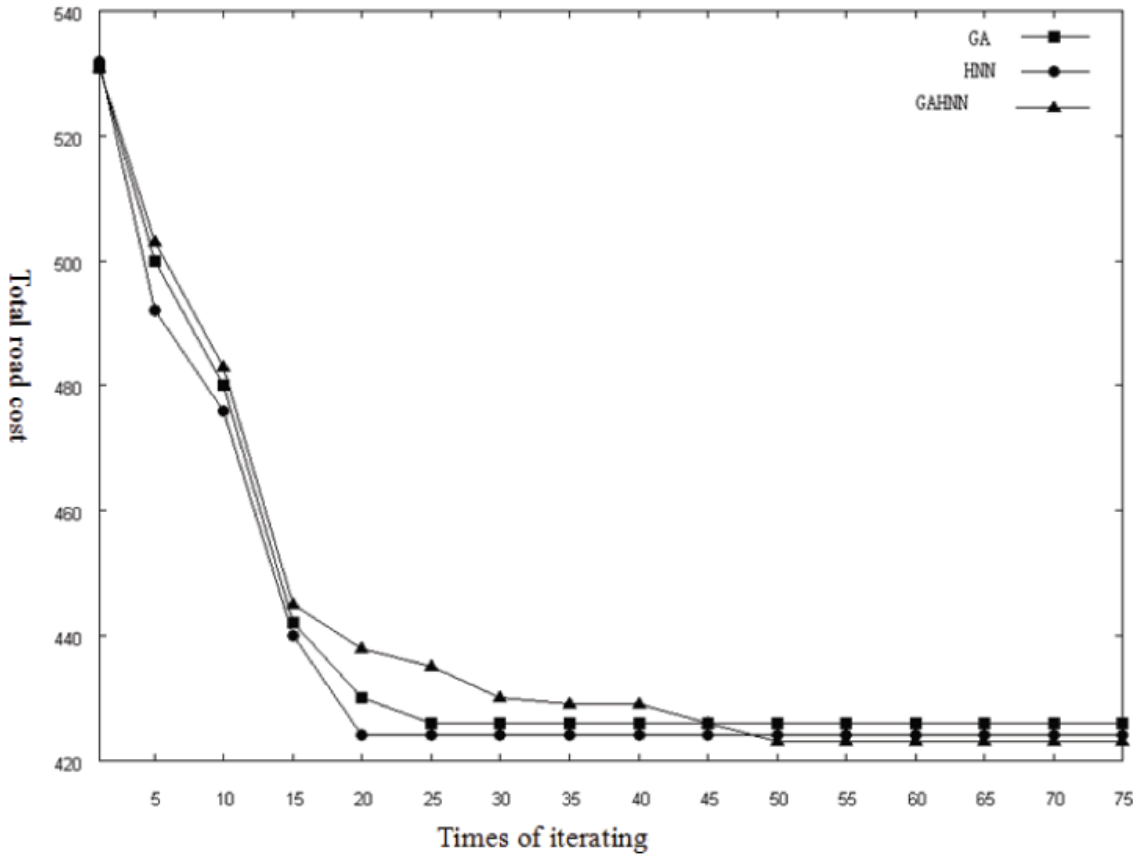

FIGURE 6. Iteration graph of the oliver30 TSP optimal solutions

TABLE 1. Oliver30 TSP simulation results of three algorithms

algorithm	optimal solution	Average iterations	Average computing time
GA	424.7963	21	0.593s
HNN	426.6002	18	0.391s
GAHNN	423.7406	47	0.653s

3.2.2. *Simulation on MapX.* We respectively adapted the GA, HNN and GAHNN on the vector map of Beijing to do the test. In the Visual C++6.0 and MapX5.0 environment, where the weather is assumed to be normal and streets are grade No.1, the simulation is performed in normal traffic situation and rush hour situation.

(1) Normal traffic situation

The parameters are defined as follows:

Average traffic volume: 30 per minute,

Hourly traffic volume: 40 per minute,

Planned hourly traffic volume: 35 per minute,

Vehicle speed: 50 km/h,

Vehicle density: 175 per hour,

Road congestion rate: 0.225,

Road congestion time: 10 minutes,

Red light time: 1 minute,

Split: 0.5.

Figure 7 shows the optimal routes acquired by three algorithms in normal traffic situation from the same origin to the same destination. It is clear that the GAHNN has

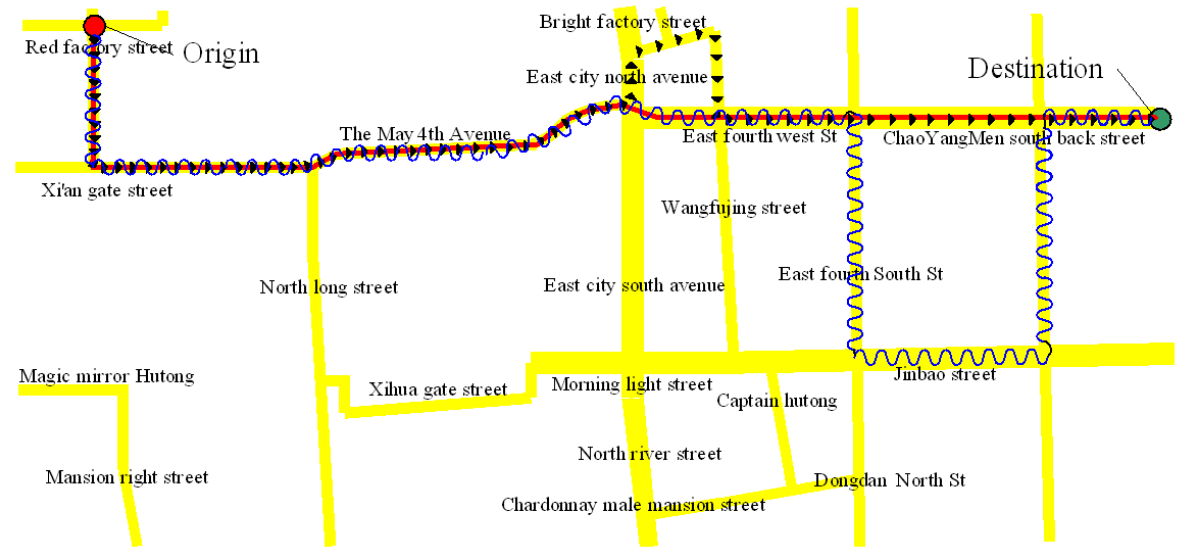

FIGURE 7. Routes acquired by three algorithms in normal traffic

TABLE 2. Results of three algorithms in normal traffic

algorithm	Symbols in Figure 7	Travel distance (km)	Travel time (h)	Computing time (s)
GA	triangle	3.55	0.299	8.91
HNN	wavy line	4.36	0.281	7.95
GAHNN	full line	3.10	0.243	10.2

acquired a better route without detour at all. Table 2 shows the actual data of computing results of three algorithms. It can be seen that the solution computed by GAHNN is the best one in both travel distance and travel time with only little longer computing time. When applied to the real world, our algorithm will provide users a more realistic route with both the shortest travel distance and the shortest travel time in normal traffic situation.

(2) Rush hour situation

The parameters are defined as follows:

Average traffic volume: 60 per minute,

Hourly traffic volume: 80 per minute,

Planned hourly traffic volume: 70 per minute,

Vehicle speed: 25 km/h,

Vehicle density: 350 per hour,

Road congestion rate: 0.5,

Road congestion time: 20 minutes,

Red light time: 2 minute,

Split: 0.5.

Figure 8 shows the optimal routes acquired by all three algorithms in rush hour situation from the same origin to the same destination. Detours happen in all three routes to avoid the traffic congestion in rush hour. Table 3 is a data comparison of three different results. It shows clearly that the optimal route GAHNN computed is significantly better than the other two on travel distance. While the travel time 0.466h is a little bit longer than the GA's 0.433h, yet better than the HNN's 0.490h. Though all three algorithms have their own advantages, the GAHNN shows a comprehensively good performance in completing the mission of dynamic route guidance.

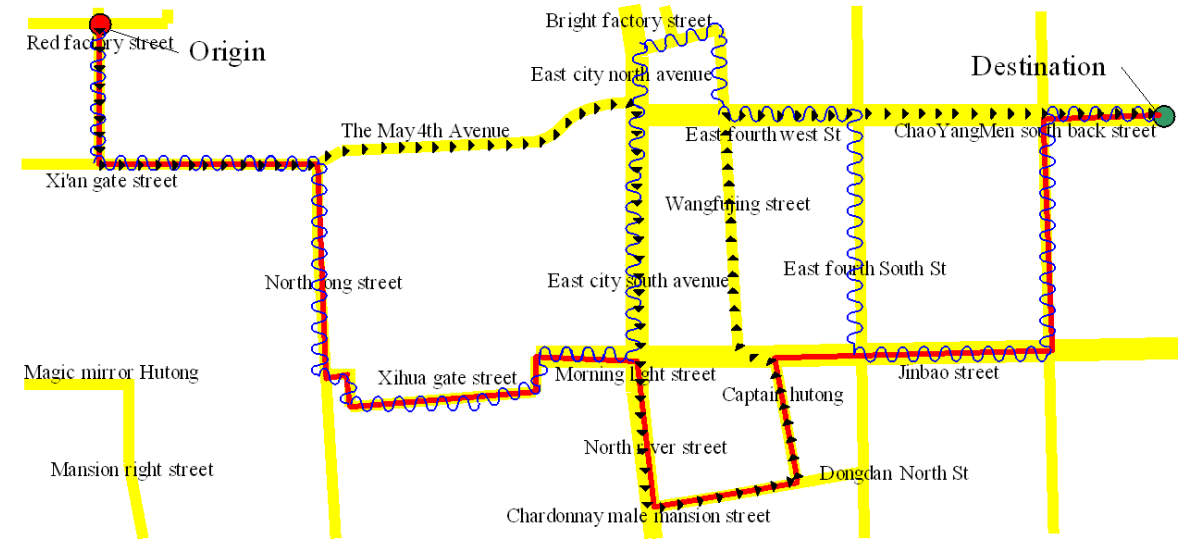

FIGURE 8. Routes acquired by three algorithms in rush hour situation

TABLE 3. The results of three algorithms in rush hour situation

algorithm	Symbols in Figure 8	Travel distance (km)	Travel time (h)	Computing time (s)
GA	triangle	6.71	0.433	10.9
HNN	wavy line	6.14	0.490	10.83
GAHNN	full line	5.29	0.466	12.8

The above-mentioned experimental results in two different situations show that the GAHNN algorithm gets a better route than the other two in both rush hour and normal traffic situation. The results have proved that the GAHNN has the advantage of applying in the real road network and is capable of benefiting users by providing a more realistic optimal route. Moreover, to some degree the HNN can be implemented in hardware chips that run much faster in a hardware environment than in a software one. Namely, when comes to real world applications, the route-guiding devices which equipped the GAHNN proposed in this paper will not only provide a good quality optimal route, but also perform much better in computation time than the other two algorithms.

4. **Conclusions.** This paper has analyzed the GA and HNN deeply, take advantage of GA's global searching ability to combining with HNN's local searching ability, and propose the GAHNN hybrid optimization strategy. The simulation results have shown that the GAHNN gets a global optimal solution within the time acceptable by users, and is effective when running in real road network. In this paper, we have not take into consideration the impact of the drivers' behavior on route choosing, the current dynamic allocation model does not show the drivers' possible reactions to the dynamic information, which is described in some references [18-20]. Such issues will be further studied in our future research.

Acknowledgment. The Project is supported by Natural Science Foundation of Liaoning Province (No. 20102175), Program for Liaoning Excellent Talents in University (LJQ2012 011), "Liaoning Bai-Qian-Wan Talents Program" (No. 2009921089, 2010921080), Liaoning Province Education Department Research Projects (No. L2010423) and Graduate Education of Liaoning Province Innovation Projects.

REFERENCES

[1] C. Wang, M. A. Quddus and S. G. Ison, Impact of traffic congestion on road accidents: A spatial analysis of the M25 motorway in England, *Accident Analysis and Prevention*, vol.41, no.4, pp.798-808, 2009.

[2] L. Gao, M. Liu, Z. Sun and B. Mao, Simulation on impact of information guidance on regional traffic flow, *Journal of Transportation Systems Engineering and Information Technology*, vol.8, no.4, pp.63-69, 2008.

[3] E. Ben-Elia, R. D. Pace, G. N. Bifulco and Y. Shiftan, The impact of travel information's accuracy on route-choice, *Transportation Research Part C: Emerging Technologies*, vol.26, pp.146-159, 2013.

[4] Y. Pavlis and M. Papageorgiou, Simple decentralized feedback strategies for route guidance in traffic networks, *Transportation Science*, vol.33, no.3, pp.264-278, 1999.

[5] A. Nazemi and F. Omidi, An efficient dynamic model for solving the shortest path problem, *Transportation Research Part C: Emerging Technologies*, vol.26, pp.1-19, 2013.

[6] S. Effati and A. R. Nazemi, Neural network models and its application for solving linear and quadratic programming problems, *Applied Mathematics and Computation*, vol.172, no.1, pp.305-331, 2006.

[7] S. Sen, R. Pillai, S. Joshi and A. K. Rathi, A mean-variance model for route guidance in advanced traveler information systems, *Transportation Science*, vol.35, no.1, pp.37-49, 2001.

[8] Y. (Marco) Nie, X. Wu, J. F. Dillenburg and P. C. Nelson, Reliable route guidance: A case study from Chicago, *Transportation Research Part A*, vol.46, pp.403-419, 2012.

[9] F. Jolai and A. Ghanbari, Integrating data transformation techniques with Hopfield neural networks for solving traveling salesman problem, *Expert Systems with Applications*, vol.37, no.7, pp.5331-5335, 2010.

[10] G. Serpen, Hopfield network as static optimizer: Learning the weights and eliminating the guesswork, *Neural Processing Letters*, vol.27, no.1, pp.1-15, 2008.

[11] S.-M. Chen and C.-Y. Chien, Parallelized genetic ant colony systems for solving the traveling salesman problem, *Expert Systems with Applications*, vol.38, no.4, pp.3873-3883, 2011.

[12] P.-C. Chang, W.-H. Huanq and C.-J. Ting, Dynamic diversity control in genetic algorithm for mining unsearched solution space in TSP problems, *Expert Systems with Applications*, vol.37, no.3, pp.1863-1878, 2010.

[13] Z. Yu, Y. Zhang, Z. Wang and M. Ni, Improved genetic algorithm for urban traffic flow guidance strategy, *Control and Decision*, vol.26, no.12, pp.1891-1894, 2011.

[14] X. Zhao, D. Yu and F. Liu, Based on genetic neural network of regional control evaluation method research, *China Science and Technology Information*, vol.4, pp.271-274, 2011.

[15] M. N. D. Nair, Comparing genetic algorithm and guided local search methods by symmetric TSP instances, *Proc. of the 10th Annual Conference on Genetic and Evolutionary Computation*, pp.1131-1132, 2008.

[16] L. Zou and J. Xu, Method of shortest path on dynamic network based on genetic algorithm, *Computer Applications*, vol.25, no.4, pp.742-744, 2005.

[17] X. Shi and M. Zhang, A human-imitating control based on neural network&GA, *Journal of System Simulation*, vol.16, no.8, pp.1835-1844, 2004.

[18] A. Paz and S. Peeta, Paradigms to deploy a behavior-consistent approach for information-based real-time traffic routing, *Netw. Spat. Econ.*, vol.9, pp.217-241, 2009.

[19] F. Gao and M. Wang, Route choice behavior model with guidance information, *Journal of Transportation Systems Engineering and Information Technology*, vol.10, no.6, pp.64-69, 2010.

[20] Z. Parvaneh, T. Arentze and H. Timmermans, Understanding travelers' behavior in provision of travel information: A Bayesian belief approach, *Proc. of the 15th Meeting of the EURO Working Group on Transportation*, pp.251-260, 2012.