Using the complex network for solving TSP problem

Yi-ying Chen(1st)
Department of Information Engineering
Shijiazhuang University of Economics
Shijiazhuang, China
E-mail: mrs.cyy@163.com

Ze-xing Zhang(2nd)
Department of Information Engineering
Shijiazhuang University of Economics
Shijiazhuang, China
E-mail: wwweee111@126.com

Wen-bin Li(3rd)
Department of Information Engineering
Shijiazhuang University of Economics
Shijiazhuang, China
E-mail: mr.liwb@emails.bjut.edu.cn

***Abstract*—This paper puts forward a new algorithm (Complex Network Generic Algorithm, CGA) for solving TSP problem, which is an variant of genetic algorithm. There are three differences between GA and CGA. First CGA does not use selection operator. Second, individuals of CGA interact with each other in specific environment determined by the complex network. Third, to some extent, the interaction of individuals in CGA is structured by the given complex network. Experimental results invalidate this new method on the one hand. On the other hand, they show that CGA is an effective algorithm.**

Keywords- Gemeric algorithm, TSP, complex network

I. INTRODUCTION

The root of traveling salesman problem can be traced back to the knight's tour problem presented by Euler in 1759. At present, TSP has become a classic NP problem in the modern field of combinatorial optimization [1]. When the problem scale is large enough, it is very difficult and even impossible to solve. Now, a lot of algorithms are used for solving the TSP problem, such as Hopfield neural network[3], simulated annealing algorithm [3], genetic algorithm[4], ant colony algorithm[5][6], particle swarm algorithm [7][8], immune algorithm [9] and leapfrog algorithm[2]. Hopfield and Tank used Hopfield network to solve *N=30* TSP problem in 1985. They found a suboptimal solution within 0.2 seconds using a network consisting of 900 neurons. It is simple, standard and fast to use Hopfiled network to solve TSP problem. At the same time there are some flaws. For example, it is not stable, it is difficult to solve large scale TSP problem and it is highly sensitive to parameters *etc* [3]. N.Metropolis *et.al*. proposed a simulated annealing algorithm in 1953. S.Kivkpatrick et.al. first simulated annealing algorithm for solving combinatorial optimization problems in 1983. This algorithm is an effective way to solve the TSP problem, which can handle the number of city more than 6000. However the disadvantage of it is the convergence time is longer. As you know, there are a lot of literatures for TSP problem [10-13].

The paper puts forward a new algorithm (Complex Network Generic Algorithm, CGA) for solving TSP problem based on complex network, referenced to genetic algorithm. The experimental results show the CGA is effective.

II. GENETIC ALGORITHM AND ANALYSIS

There are many steps to design the genetic algorithm: to determine code schemes, to determine the fitness function, to design selection strategy, to design genetic operators (crossover and mutation operator) and to determine the algorithm's termination rules. Algorithm 1 shows the pseudo code of traditional genetic algorithm.

Algorithm 1. the pseudo code of genetic algorithm

```
Input：population number, fitness function, encoding mode
Output：optimal solution
Processing:
    Initializing population;
    Calculating individual fitness
    DO
       evolving individual using genetic operator;
       calculating individual fitness;
       generating new population by choosing best;
    WHILE(termination conditions);
    Output the individuals which has the best fitness;
```

Algorithm 1 shows that the individuals of $(i+1)^{th}$ generation are always associated with the individuals of i^{th} generation. So from the 1st generation to the last one, we connect the individuals which have relationship with edges. A graph will be got, shown as Fig.1. In other words, the genetic algorithm executed each time will form a corresponding graph. This prompted us to thinking: could we give the structure of the graph first, in which the edges represent genetic or selection operation, the nodes interacting with each other through the "edge" achieve evolution.

The genetic algorithm and others mentioned algorithms previously are called bionic algorithm, such as ACO algorithm, PSO algorithm, immune algorithm, artificial fish-swarm algorithm, hybrid frog-jumping algorithm etc. These

algorithms are designed through simulating the approaches about animals and plants dealing with the problems. For example, the genetic algorithm was presented according to Darwin's theory of evolution. Ant colony algorithm was presented by imitating the ant colony's methods of searching for food, which always choose the shortest path. These kinds of algorithms have great similarities in the run mode. Some of them are: The group consisted of the individuals; Individual evolves based on specific rules; Through iteratively, the updating groups are generated (such as genetic algorithm, ant colony algorithm), or the new individual positions are generated (such as particle swarm algorithm, artificial fish-swarm algorithm, hybrid frog-jumping algorithm); The optimal solution is emerged with the continuous evolution of the group or movement [14].

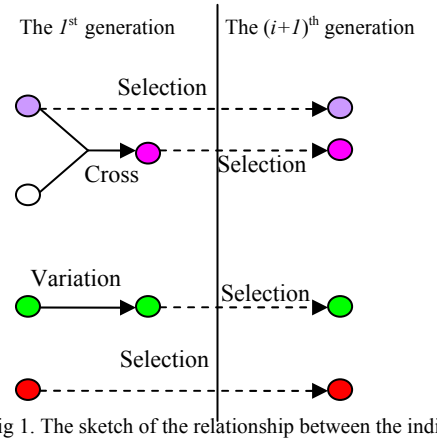


Fig 1. The sketch of the relationship between the individuals

III. SOLVING TSP BASED ON COMPLEX NETWORK

The solution process of the simulation algorithm can not be separated from the interaction among individuals. Let's assuming that the nodes of complex network are individuals, the sides of complex network are relationships between the individuals. Then the individual evolution rules might be designed. Based on the rules the solution methods of whole complex network's evolution can be got. According to this consideration, the paper designed CGA algorithm.

A. CGA Algorithm Overall design

Here are some definitions:

[**Definition 1**] Individuals: One of the complex network nodes;

[**Definition 2**] Individual Coding: representation of solutions;

[**Definition 3**] Fitness: The real value representing the excellent degree of solutions.

[**Definition 4**] Neighbor: The other individuals which direct contact with this individual.

[**Definition 5**] Small Population: An individual and its all neighbors constitute small population.

[**Definition 6**] Small environment: For an individual, small population constitutes its small environment.

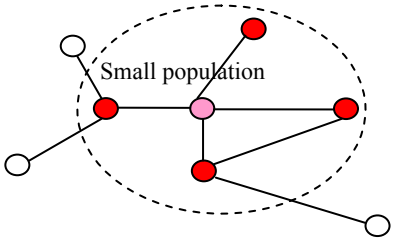


Fig 2 The sketch of small populaiton

[**Definition 7**] Big environment: For an individual, the environment outside the small environment is called the big environment.

The solution process of CGA algorithm depends on a complex network which has been built in advance. In complex networks, each node represents a solution. There are several encoding methods, such as integer mode, floating-point mode and mixed mode [15]. Every node connected its neighbor with edges form a small population, shown as Fig 2. Each small population evolved once in one time iteration. In fact, it is the individuals' evolution in the population. There are two methods of individual evolution, variations and hybrids, with the same as the genetic algorithm. Individual evolution can be effected only by a small environment, or both small environment and big environment. After many times iterations, all of the small populations' evolutions have been stopped in the complex network.

The following is CGA algorithm.

Algorithm 2. the CGA algorithm

Input: complex network, encoding mode, fitness function
Output: optimal solution
Processing:
Initializing the encoding of each node in the network;
Calculating node fitness
DO
FOR every node
finding all neighbors of the current node(small population);
small population evolution(including the current node);
ENDFOR;
WHILE(termination conditions);
Output the node which has the best fitness;

B. STRATEGY DESIGN OF ENCODING, GENETIC OPERATORS AND THE SMALL POPULATION EVOLUTION

For TSP problem, the encodings of each node are represented as $N+1$ dimensional row vector: $<c_1, c_2, ...,c_N, c_1>$, in complex network (N is the number of cities) . In which each component c_i using real coding mode is used to represent a number of the city. In the front of N-dimensional components, every one is not equal to each other. The $N+1$ dimension component is equal to the first one. This means that it starts

from c_1 city to c_2 city, … to c_N city, and finally back to c_1 city. The individual's fitness formula is defined as:

$$F(c_1,...,c_N,c_1)=1/(d(c_N,c_1)+\sum_{x=1}^{N-1}d(c_x,c_{x+1}))$$

In the above formula, the $d(c_x,c_y)$ represents the weight of edge(c_x, c_y). In another words, it represents the distance of city c_x and city c_y, $x \neq y$, x、$y \in \{1, 2, ...N\}$.

The following algorithm 3 is the small population evolution strategy, in which the evolution of the individual is affected only by small environment. The new individual evolution strategy can be designed, which is affected both by small environment and big environment, if you want to.

Algorithm 3. the small population evolution strategy

```
Input：complex network, the current node
Output：nothing
Processing：
  nh1 =the best fitness neighbor;
  nh2 =the second fitness neighbor;
  nh3 =the worst fitness neighbor;
  IF nh1 or nh2 is worse than current node
      nh1 cross nh2, then generate two children nodes;
      replace the current node with the best child;
      replace nh3 with the second better child;
  ELSE
      IF nh1 is better than nh2
          IF nh2  variation is effect
              update nh2;
          END
      ELSE
          IF nh1 variation is effect
              update nh1;
          END
      END
      IF nh3 variation is effect
          update nh3;
      END
  END
```

Let's assume that the encoding of the parent nodes are: <c_1, c_2, ...,c_N, c_1> and <c'_1, c'_2, ...,c'_N, c'_1>, crossover node is I、J（I、$J \in [1,N]$）, two offsprings are: <c'_1,..., c'_{I-1}, c_I, ... c_J, c'_{J+1},c'_N, c'_1> and <c_1,..., c_{I-1}, c'_I, ... c'_J, c_{J+1},c_N, c_1>. For TSP problem, there are repeated cities should be deal with in the two offsprings.

There are two methods of mutation. Let's assume the individual coding is <c_1, c_2, ...,c_N, c_1>, variation nodes are I、J（I、$J \in [1,N]$）. The first method is transposing I、J dimensional component, getting <c_1,..., c_{I-1}, c_J, c_{J-1},... , c_{I+1} ,c_I, c_{J+1},c_N, c_1>. The second one is interchanging I、J dimensional component, getting <c_1,..., c_{I-1}, c_J, c_{I+1},... , c_{J-1} ,c_I, c_{J+1},c_N, c_1>.

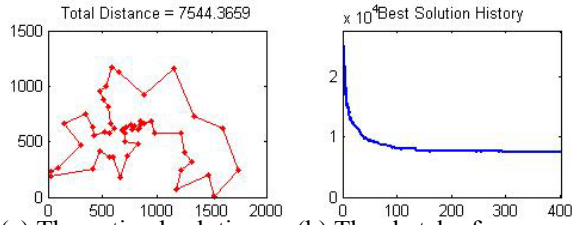
The termination condition is fixing the number of iterations.

IV. THE EXPERIMENTAL RESULTS

A. *The Expermental Results*

This paper carries out a series of simulations experiments. The hardware platform is: **I**ntel Core 2 Duo Processor T5500、1G memory. The software platform is: Windows XP + Matlab 7.1. The data sets come form TSPLIB[1].

Experiment 1: The complex network is scale-free network. The relevant parameters are: the initial number nodes n0=8, the increase number of edges m=7, the total number of nodes n=300. The CGA parameters are: the number of iterations is 400. The data sets: berlin52.tsp.

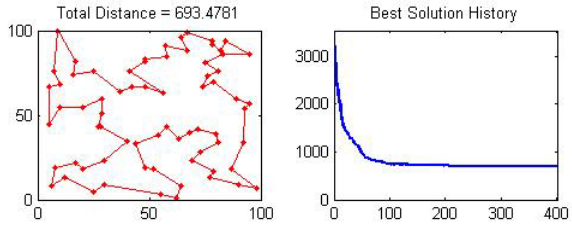


(a) The optimal solution (b) The sketch of convergence

Fig 3. The optimal solution and its sketch of convergence in berlin52 data sets(BA network)

The Fig 3 shows the optimal solution and its convergence by using CGA algorithm in berlin52 data sets after 10 repetitions experiments. At present, the latest optimal solution is 7542 published by TSPLIB in berlin52.tsp data sets. In the domestic literature, the optimal solution is 7544 or greater than it in this data sets. The optimal solution got by using CGA algorithm meets or exceeds the level of the domestic literature.

Experiment 2: The complex network is scale-free network. The relevant parameters are: the initial number of nodes n0=8, the increase number of edges m=7, the total number of nodes n=300. The CGA parameters are: the number of iterations is 400. The data sets: st70.tsp.



(a) The optimal solution (b) The sketch of convergence

Fig 4. The optimal solution and its sketch of convergence in st70 data sets(BA network)

The Fig 4 shows the optimal solution and its convergence by using CGA algorithm in st70 data sets after 10 repetitions experiments. At present, the latest optimal

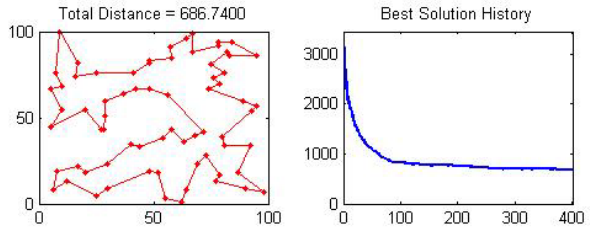
http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

solution is 675 published by TSPLIB in st70.tsp data sets. The optimal solution got by using CGA algorithm is 693, which close to the level of reported data in the world.

The above two experiments show that the CGA algorithm is feasible and effective.

B. DISCUSSION

The network structure is an important factor to affect the solving efficiency and effectiveness from the description of the CGA algorithm. It determines the number of small populations and the number of individuals. Further it determines the quality of the small population's evolution, which has been proved by large of experiments. The following experiment 3(shown as Fig.5) adopts NW network rather than scale-free network. The optimal solution is 686 in ST70 data sets after 10 repetitions experiments, which has been very close to the optimal solution published by international reports. In addition using the NW network, the optimal solution is same as experiment 1. in berlin52 data sets after 10 repetitions experiments.



(a) The optimal solution (b) The sketch of convergence

Fig 5. The optimal solution and its sketch of convergence in st70 data sets(NW network)

Experiment 3: The complex network is NW network. The relevant parameters are: the total number of nodes n=300，K=10, p=0.9. The CGA parameters are: the number of iterations is 400. The data sets: st70.tsp.

Fig.3(b), Fig.4(b) and Fig 5(b) (including other experimental results) show the CGA convergence process does not exist prematurity. Pre-convergence is faster and post-convergence is slower. The reason is that in the initial stage, the majority of individuals are in the "bad" states which have the more evolution possibility. In later stage most of individuals evolve slowly, or it is difficult to find the better individuals than itself. So the speed of convergence is slower. The reason not existing prematurity is the relative worst individuals must mutate in our designed small population evolution strategy. This means that a number of individuals will mutate in each iteration. This variation is not necessarily in the right direction, maybe changes in the bad direction. This can be seen as a "disturbance" in the entire evolutionary process, which can effectively prevent premature.

V. CONCLUSION

In this paper, we propose a CGA algorithm based on a fixed network structure after analyzing many other genetic algorithms. The genetic algorithm design parameters generally include: population size, the selection probability, crossover probability, and mutation probability and termination conditions. Different from it our design parameters include: network architecture, node coding, fitness function and small populations evolutionary strategies.

We did some experiments on the two data sets, the results show that: 1) The CGA algorithm is effective. 2) The algorithm can effectively avoid premature. Although the CGA algorithm did not achieve the best results better than international coverage on the TSP problem, this is a new attempt for us. We are confident that we can achieve the better results after studying deeply in the following two aspects: 1) the theoretical and experimental research about the CGA algorithm performance influenced by network structure. 2) Analysis and experimental study of small populations' evolutionary strategy.

REFERENCES

[1] Zhou kang, Qiang Xiao-li,Tong Xiao-jun,Xu jin. "Algorithm of TSP. Computer Engineering and Applications", 2007,43(29),pp. 43-47.

[2] Chen Yi-ying,Li Wen-bin, Wang Duo,Zhu Qun-ying. "Using discrete search space-oriented leaping frog algorithm to solve TSP", Computer Engineering and Applications, 2009,45(27),pp. 729-738.

[3] Jiao Li-cheng. "Neural network computing", Xi An:Xi'an University of electronic science and technology Press,1993.

[4] John Henry Holland. "Adaptation in Natural and Artificial Systems", The MIT press 1975.

[5] A. Colorni, M. Dorigo et V. Maniezzo, "Distributed Optimization" by Ant Colonies, actes de la première conférence européenne sur la vie artificielle, Paris, France, Elsevier Publishing, 134-142, 1991

[6] M. Dorigo, "Optimization, Learning and Natural Algorithms", PhD thesis, Politecnico di Milano, Italie, 1992.

[7] Kennedy, J.; Eberhart, R. Particle "Swarm Optimization". Proceedings of IEEE International Conference on Neural Networks.IV. 1995, pp.1942–1948doi:10.1109/ICNN.1995.488968.

[8] Shi, Y.; Eberhart, R.C. "A modified particle swarm optimizer". Proceedings of IEEE International Conference on Evolutionary Computation. 1998, pp. 69–73.

[9] Wang Lei, Pan Jin,Jiao Li-cheng. "The Immune Algorithm". Acta Electronica Sinica,2000,28(7):pp.74-78.

[10] Ji Jun-zhong, Huang Zhen, Liu Chun-nian, Dai Qi-guo. "An Ant Colony Algorithm Based on Multiple-Grain Representation for the Traveling Salesman Problems". Journal of Computer Research and Development, 2010, 47(3):pp.434-444.

[11] Ji Jun-zhong, Huang Zhen,Liu Chun-nian. "A Fast Ant Colony Optimization Algorithm for Traveling Salesman Problems". Journal of Computer Research and Development, 2009, 46(6):pp.968-978.

[12] Wan Ying-yu,Zhou Zhi,Chen Guo-liang,Gu Jun. "Sizescale:New Algorithms for The Traveling Salesman Problem". Journal of Computer Research and Development, 2002, 39(10):pp.1294-1302.

[13] Wang Jian-wen,Dai Guang-ming,Xie Bai-qiao,Zhang Quan-yuan. "A Survey of Solving the Traveling Salesman Problem". Computer Engineering & Science,2008,30(2):pp.72-74.

[14] http://www.cnhup.com/index.php/archives/biological-modelling-intelligence-optimization-algorithm-intro/

[15] Chen Yong-jian, Zhou Yan. "Application of Coding Genetic Algorithm and its Improvement Research". Journal of TaiYuan Normal University (Natural Science Edition). 2008,vol 7(2):pp.76-78.