An Improved Particle Swarm Optimization Algorithm and Its Application in the Community Division

Hao JIANG^{1, a}, Liu-Yi ZHANG^{2, b}

School of Computer Science and Engineering, Southeast University, Nanjing, 211189, China
^aemail: hjiang@seu.edu.cn, ^bemail: dtzly888@163.com

Abstract. With the deepening of the research on complex networks, the method of detecting and classifying social network is springing up. In this essay, the basic particle swarm algorithm is improved based on the GN algorithm. Modularity is taken as a measure of community division [1]. In view of the dynamic network community division, scrolling calculation method is put forward. Experiments show that using the improved particle swarm optimization algorithm can improve the accuracy of the community division and can also get higher value of the modularity in the dynamic community

Among the classic community division algorithms, GN algorithm and Newman algorithm are the most representative. The GN algorithm is to delete the highest weight of the network edge each time. It stops when each node is degenerated into a community. The GN algorithm is based on the greedy strategy. The advantage of this algorithm is the low space complexity, meanwhile, the defect of the algorithm lies in the fact that the termination of the community division is unknown when the result of division is uncertain. The Newman fast algorithm is improved based on the GN algorithm which has a good performance in the static network. However, social networks is interactive and dynamic as the relationship in the network changes over time. In order to solve the problems above, this essay refers to the concepts of modularity and designs the objective function based on particle swarm optimization algorithm. By using the penalty function method and other constraints strategy, the particles which do not meet the requirements will be timely eliminated. Through the mutation strategy and the optimization of convergence factor, the convergence of the algorithm is improved. This essay also adopts a scrolling calculation method to update the dynamic network of community structure.

1 An Improved Particle Swarm Optimization Algorithm

1.1 Description of particle swarm optimization

Particle swarm optimization algorithm is proposed by Kennedy and Eberhart in 1995[2]. This is a kind of evolutionary computation technique based on the swarm

intelligence which is inspired by the behavior of the birds. This algorithm eventually converges to the global optimal particle through the cooperation between the particles. These particles adjust their speed and position in order to find the local optimal and global optimal particles. The process is described as follows. First of all, the value of each particle variable is assigned including the position vector and velocity vector of the searching space. Then the iterative calculation is started. The particles renew themselves through the two "extreme values" at each time of iterations. One "extreme value" is the optimal solution called pbest which is found by each particle itself and the other is found by the whole particles called gbest. Then, the particles update their speed and position vector according to the following formula.

$$\left. \begin{aligned} v_{id} &= wv_{id} + c_1r_1(pbest - x_{id}) + c_2r_2(gbest - x_{id}) \\ x_{id} &= x_{id} + v_{id} \end{aligned} \right\} \quad (1)$$

In formula (1), the learning factor c_1 and c_2 are the nonnegative constants, r_1 and r_2 are the random numbers in the interval (0,1), inertial factor w is often set no greater than 0.

The basic particle swarm optimization algorithm finds the best target by the cooperation between particles. However, it is easy to fall into the local optimum in the real problem. So we often need to make some improvements on it.

1.2 Improvements on particle swarm optimization

1.2.1 Optimization based on the parameter

In formula 1, the parameter which affects the search ability of the algorithm is the inertia weight w . The parameters which affect the convergence rate of the algorithm is the accelerate weights C^1 and C^2 . How to set parameters has no unified standard which just depends on experiences. Reference [3] points out that the greater w is, the weaker the global search ability is and the smaller the w is and the stronger global search ability is. So we need to set up a proper inertia weight to balance the global search and local search ability. Meanwhile, the convergence speed of the PSO algorithm is determined by acceleration weight C^1 and C^2 . This essay assigns the value to C^1 and C^2 respectively because this is advantageous for speeding the convergence. Formula is as follows:

$$\left. \begin{aligned} w &= a - e^{(-(iteration/IT_{max})^{0.2})}, \\ c_1 &= b - \gamma / (iteration / IT_{max})^{0.2} \\ c_2 &= b + \gamma / (iteration / IT_{max})^{0.2} \end{aligned} \right\} \quad (2)$$

In formula 2, the inertia weights C^1 and C^2 are becoming smaller as the time of iterations increases, IT_{max} is the largest time of iterations and $iteration$ is the current time of iterations, $iteration / IT_{max}$ is in the interval (0, 1]. As the time of iterations increases, C^1 is reduced and C^2 is increased gradually. a , b and γ are the constants which are set between 1 and 2.

1.2.2 Optimization based on particle diversity

As the iteration times are increased, the similarity between the particles will get higher in the PSO algorithm. If the optimal solution is unable to be further optimized, this algorithm is very easy to fall into the local optimum. For this kind of situation, the diversity of the particles should be optimized. This optimization is mainly revealed in two aspects. One is to take random variation for each particle variable. Each particle variable contains the increase or decrease of the community ID which is represented in 1 or -1. During the process that each particle finds the optimal solution, those two integer variables are taken complementary mutation. This will be further elaborated in the part of community division. The other aspect is to take the random search for the current global optimum, g_{best} . As the particles converge towards the global optimal particle eventually, if g_{best} is the local optimal particle, the algorithm can't search for another space. Reference [4] puts forward a formula which is described as follows:

$$\text{if } (rand() < c) \text{ then } x_{id} = random(l_d, u_d) \quad (3)$$

In formula (3), the value c is the noise factor which is set in the interval [0,1], $random(l_d, u_d)$ is a random

number which is between l_d and u_d , the value l_d and u_d are the part of position vector.

In this essay, combined with the conditions in social networks, the formula is designed as follows:

$$\text{if } (rand() < c) \text{ then } x_{id} = 1 - x_{id} \quad (4)$$

Using the formula 4, the value g_{best} is randomly chosen a bit inversion and assigned to the position vector after being calculated. Experiments prove that this kind of random mutation can maintain the diversity of particles.

2 Community division

2.1 Particle fitness function

In order to add the termination conditions, this essay makes reference to the concept of modularity. Assuming that the network is divided into k communities, a $k \times k$ matrix $E = (e_{ij})$ is defined. In this matrix, e_{ij} represents the proportion of the edges connect two community nodes i and j in all edges. The sum of the elements on the diagonal of the matrix $\text{Trace}(E) = \sum_i e_{ii}$ is defined. In this matrix, the sum of the elements in each row is $a_i = \sum_j e_{ij}$. It represents the proportion of the edges linked to the i th community of all edges. On this basis, the fitness function is:

$$Q = \sum_i (e_{ii} - a_i^2) = \text{Trace}(E) - \|e^2\| \quad (5)$$

In formula 5, Q represents the proportion of the inside edges minus the expectations of the edges which connect both arbitrary sides. If the proportion of the edge inside is no greater than expected for any connection, $Q = 0$. The upper limit of Q is 1. The closer Q gets to the value 1, the more obvious the community structure is.

2.2 Particle coding

Reference [5] puts forward that each particle variable stores the ID number of the current community. Because the number of the divided community is unknown, the initialized ID number of the community is set randomly from 1 to the total number of the nodes. If the community division is reasonable, the convergence of the algorithm can be accelerated. So this essay proposes a new particle coding scheme. Each particle variable stores the current ID number of the community and the increase or decrease of the community ID. The increase or decrease of the community ID is represented by 1 and -1. -1 represents that the current community ID minus 1 and 1 represents that the current community ID plus 1. At each period of time, the modularity is decided by both of them. These two variables are coded into the coding chains. For an example, if there are eight nodes, each particle on the search space at each time is represented as follows:

87287425-11-11-111-1

In this coding chain, 87287425 is the distribution of the ID numbers in eight communities, 11-11-111-1 is the increase or decrease of the community ID direction.

2.3 Penalty function strategy

During the community division, the decision variables are affected by various constraints such as community ID, modularity and so on. Under these constraints, if the algorithm runs beyond the scope of constraint a little bit, the optimal solution will not be reached. By using the penalty function strategy, the problem is transformed into no constraint problems.

2.3.1 Penalty function

In the community division, if the modularity does not meet the prescribed requirements, some isolated points are generated. Isolated points are dissimilar to the majority of the nodes and their appearance is as a result of improper community convergence direction. In this essay, isolated points are avoided by adjusting the penalty factor. The core idea is to increase the weight coefficient as the error between the modularity and the expected modularity is increased. When the error meets the maximum limit, the punishment will be infinite in order to timely eliminate those particles.

Its mathematical formula is as follows:

$$\left. \begin{aligned} s &= 1000 \\ \delta(X) &= Q_0 - Q \\ f &= s * \delta(X) = 1000 * (Q_0 - Q) \end{aligned} \right\} \quad (6)$$

In formula 6, Q_0 represents expected modularity, Q represents the actual modularity, s represents penalty factor of the isolated points.

2.3.2 Integer Penalty function

In the improved PSO algorithm, the projection of the random solution found by particles should be 1 or -1. If the increase or decrease of the community ID direction during the random solution is not 1 or -1, integer penalty function is designed in order to eliminate the particles which do not meet the requirements. The formula is as follows:

$$con = Kch * |(y+1) * (y-1)| \quad (7)$$

In formula 7, con represents the integer penalty function, Kch represents integer penalty coefficient, $|(y+1) * (y-1)|$ represents the situation when y is not -1 or 1.

Taking the two custom penalty functions into consideration, the extended objective function is improved as follows:

$$F = Q + f + con \quad (8)$$

In formula 8, F represents the extended target function, Q represents the current

modularity, con represents integer penalty function, f represents the penalty function.

2.4 Mutation strategy

In the process of finding the optimal solutions, taking random mutations for the particles can enhance the search efficiency. This paper puts forward mutation strategy for the increase or decrease of the community ID. For example, the increase or decrease of the community ID is 1 1 1 1 -1 -1 -1 -1, this series of the integer number is -1 -1 -1 -1 1 1 1 1 after taking the bit-inversion mutation. In this paper, when taking the particle mutation is determined by fitness change rate delta. Fitness change rate is the rate of best position found by particles in the history and the mathematical expression is as follows:

$$\Delta = \frac{F_{Pbest(t)} - F_{Pbest(t-M)}}{F_{Pbest(t)}} \quad (t > M) \quad (9)$$

When the Delta is less than a certain threshold, the mutation strategy is taken. This strategy can save the cost of time and space because of the continuous iterations.

2.5 Scrolling calculation for the Dynamic community

Traditional community divisions are mostly applied in the static social network which is an analysis of the static network. However, the interaction among the members changes over time in real social networks. Static network is usually a snapshot of the social network but the structure of real social networks often changes and connection between the nodes is dynamic. The newest community structure can no longer achieved. For this kind of situation, this essay proposes a daily scrolling calculation which is applied to analyze the evolution of the network according to the separate community structure in the discrete time.

According to the current social network structure, the nodes in the community are monitored and current value of modularity is calculated. In Fig.1, operation 1 is monitoring current nodes && calculating current modularity and operation i is monitoring nodes && calculating modularity.

Fig.1 Scrolling Calculation For The Dynamic Community

Algorithm is described as follows:

Step 1 (Initializing the community): Traversing the network and assigning the community ID to each node. Coding the particle and initializing the population size m and the number of iterations (generally 3 to 4 times of population size).

Step 2 (Finding the optimal solution): Duplicating the particle's current location to the historical location and

calculating the fitness degree of the current position. Finding the particles which have the highest fitness degree and making their position vector as the global optimal position.

Step 3 (Iterative evolution): In every update of the iteration, the particles update themselves according to the two "extreme values". One is the optimal solution that the particle finds by itself. The other is the global optimal position.

Step 4 (The terminating conditions): If the current number of iterations is no less than the maximum number of iterations which is set from the start or meets requirements of minimum error, the iteration is terminated. Otherwise, the algorithm goes into step (3) to continue the iteration.

3 Experiments and Result Analysis

In order to verify the effectiveness of the community division, this paper adopts two commonly used data sets including Zachary data set and arXiv data set. Zachary data set reflects the true relationship in a karate club. This network contains 34 nodes and 78 edges.

Zachary data set is a commonly used dataset in the community division, so the correctness of its community classification can be measured. This paper uses the following formula to measure the correctness:

$$\gamma = \frac{w_s - w_0}{w_s} \times 100\% \quad (10)$$

In formula 10, w_s represents the number of nodes in the correct division, w_0 represents the total number of the nodes. This paper will make comparisons among the three algorithms below. Reference [6] mentioned that Fast - Newman algorithm is the quick Newman algorithm based on the modularity.

Table1. The Accuracy Compared with Three Algorithms

Algorithm ^o	Accuracy ^o	Q ^o	Number of the divided community ^o
Basic PSO ^o	0.94 ^o	0.390 ^o	2 ^o
Fast-Newman ^o	0.96 ^o	0.371 ^o	3 ^o
Improved PSO ^o	0.97 ^o	0.470 ^o	2 ^o

As can be seen from the table1, the best result comes from the improved PSO. In its results, the correct divided nodes accounted for 97% and the value Q meets 0.47 closed to the maximum.

Then, we set Zachary data as the input data and run the algorithms. The tendency of the modularity is recorded in Fig.2.

Fig.2 Comparisons of Modularity in the Static Community

As can be seen from the Fig.2, the value of modularity is increased during the iteration. The modularity is steady at the end of the iteration. The improved PSO this paper proposed has a higher value of the modularity at each time of iterations.

Zachary data set is a static community but the real community is dynamic. This paper uses the airXiv data set as the initial data set and adds the behavior of network changes in the community. We gather 24 time in a day and start the community division. The CNM algorithm is used as the comparison. Reference [7] mentions that CNM algorithm is developed based on Newman fast algorithm.

Fig.3 Comparisons of Modularity In The Dynamic Community

As the Fig.3 can be seen, this algorithm can obtain a better value of the modularity according to the latest community structure. When calculating modularity at each point of the time, the result of the improved PSO is higher than CNM algorithm does.

4. Conclusion and prospect

In this paper, the particle variables are designed and coded creatively on the basis of the basic particle swarm optimization algorithm and other community division algorithms. The fitness function of the particle swarm is established by referring to the concept of modularity. We also put forward the strategy of the penalty function to eliminate the particles which do not meet the requirements. Furthermore, we adopt the mutation strategy to improve the efficiency of the algorithm. In order to extend the scope of the application of this algorithm to the dynamic community, scrolling calculation method is put forward which is to monitor the nodes of the current network and calculate the value of the modularity. But due to the

limitation of the theoretical level, there are still many aspects to be improved such as the simple particle fitness function and the value of the modularity can be calculated by every hour and so on.

Acknowledgement

This work was supported by Collaborative Innovation Center of Wireless Communications Technology and the National Natural Science Foundation of China (Grant No.6157060279) us

References

- [1] M.E.J. Newman, Proceedings of the national academy of sciences,23, 8577-8582, (2006)
- [2] J.Kennedy, Encyclopedia of machine learning. Springer US, 760-766,(2011)
- [3] M.E.J. Newman, Physical review E, 69,6(2004)
- [4] XF.Xie, WJ.Zhang, ZL.Yang, arXiv preprint cs,(2005)
- [5] U.N.Raghavan,R.Albert, S.Kumara,Physical Review E,76,3,(2007)
- [6] M.Tasgin, A.Herdagdelen, H.Bingol, arXiv preprint arXiv,(2007)
- [7] K.Wakita, T.Tsurumi, Proceedings of the 16th international conference on World Wide Web. ACM, 1275-1276 ,(2007)
- [8] K.Wakita, T.Tsurumi, Proceedings of the 16th international conference on World Wide Web. ACM, 1275-1276 ,(2007)