Modern Optimization Methods Final Project

HO Ching Ru / R09946006 / Data Science Degree Program

December 15, 2020

Abstract

The main goal of this final project is to apply algorithms learnt from this course, and to gain the ability to judge and compare these algorithms to the existing ones.

1 Social Network Analysis

(a) **Social Network** refers to a network set of relationships. A network (in mathematical terms, **graph**) usually contains a set of objects (in mathematical terms, **nodes**), and a mapping or description of relationships (in mathematical terms, **edges**) between the objects.

(b) **A center of a network** refers to the most influential node in a social network. On the other hand, the **centrality** indicators which identify the most important nodes within a graph. If A_{ij} is an adjacency matrix, where:

$$A_{ij} = \begin{cases} 1, \text{if node } i \text{ and } j \text{ have a relationship.} \\ 0, \text{otherwise.} \end{cases} \tag{1}$$

and the **degree centrality** K_i is measured by the total amount of direct links with the other nodes:

$$K_i = \sum_j A_{ij} \tag{2}$$

(c) **Community** refers to a specific group of nodes who all hold relationship in common. If each node will be separated into different community, $c_i = \pi_0$ refers to node i is in the community π_0. Thus, two nodes i and j are in the same community if and only if $c_i = c_j$. The probability of both two nodes in one community is:

$$\frac{\sum_{ij} A_{ij}\delta(c_i, c_j)}{\sum_{ij} A_{ij}} = \frac{1}{2m}\sum_{ij} A_{ij}\delta(c_i, c_j), \tag{3}$$

where m is the number of edges (in other words, total links) of whole graph:

$$m = \frac{1}{2}\sum_{ij} A_{ij} \tag{4}$$

and $\delta(c_i, c_j)$ is the Kronecker delta function:

$$\delta(c_i, c_j) = \begin{cases} 1, \text{if they are in the same community, meaning } i = j \\ 0, \text{otherwise} \end{cases} \tag{5}$$

then, we can define **community modularity** $Q \in [-\frac{1}{2}, 1]$ as:

$$Q = \frac{1}{2m}\sum_{ij}\left(A_{ij} - \frac{K_i K_j}{2m}\right)\delta(c_i, c_j) \tag{6}$$

(d) Each time when we separate nodes into different community, we'll get a new modularity Q. The optimization target of **Community Detection Problem** is trying to find a maximum modularity Q as possible, meaning the community structure having the high accuracy.

In Newman (2014), the researchers pointed out if Q is in the range of 0.3 to 0.7, we can say the social network has a good community structure generally.

2 Community Detection via Metaheuristics

Please refer to the **Appendix (I)** for the further source code details. Noticed that the objective function is modularity-Q.

(a) **Genetic Algorithm**:

* Iteration times = 10 (in Karate network), and = 100 (in Co-authorship network)
* Community Number = 11 (in the constraint of "the number of members in a community must be less than 10% of the total number of nodes in the network" mentioned in here), and = 3 (in both network) Due to the description of final project mentioned, a community must have at least 2 nodes, but the maximum number of nodes is restricted to be less than 50% of the total number of nodes.

  Which means, for instance, in the Karate network, the number of community should between 2 (34 nodes totally with 50% nodes in a community) and 17 (34 nodes totally with only 2 node in a community). It's hard to decide how many communities we should separate to. Though there have some methods that can handle dynamic community separation, but I want to decide it at first to reduce the computational resource.

  In here, I assumed that the less community we separate, the modularity-Q might be larger. If I set the number of communities to 2, though the expectation of number of member in each community equals to 17 (50% of nodes), but we can not ensure the result of initialization will always follow.

  In fact, we can pick up 17 nodes in one community and others in another one, but in here, I'll set the number of communities to 3. The expectation of number of member in each community equals to about 11 (33% of nodes). The result of initialization will not follow, but we can ensure the result might not far from the restriction mentioned in the first paragraph.
* Probability of reproduction = 100%. Because I want to ensure that there always have offspring candidates to select in one generation.
* Probability of mutation = 10%. It is the most generally parameter used for mutation.

(b) **Simulated Annealing**:

* Iteration times = 10 (in Karate network), and = 100 (in Co-authorship network)
* Community Number = 11 (in here), = 3 (in both network). The reason is the same as above mentioned.
* Initial temperature (T_0) = 50000. Because $\Delta f(x)$ is the difference of modularity, where $-\frac{1}{2} \leq Q \leq 1$. If the temperature is a small number, the probability of deciding to reject or accept, $\mathbb{P} = \min\{1, \exp(-\Delta f/T)\}$, is tend to be 1.0, make every "not better case" easily be acceptable in the iteration. Thus, I set a big number for the initial temperature.
* Temperature decay factor (k) = 0.8. It is the most generally parameter used for decaying.

(c) **Tabu Search**:

* Iteration times = 10 (in Karate network), and = 100 (in Co-authorship network, the reason is the same as above mentioned.)
* Community Number = 11 (in here), = 3 (in both network). The reason is the same as above mentioned.

* Tabu list size (length of tabu) = 10. Though some research said that the size of tabu list = 7 is better than 10. However, I think 10 is more custom because I used 10 in homework 2 before.

(d) **Particle Swarm Optimization** (You cannot use SIB method):

* Sorry, I have no idea to solve a discrete problem (social network) without using non-discrete problem in this final project.

(e) **Ant Colony Optimization**:

* Iteration times = 10 (in Karate network), and = 100 (in Co-authorship network, the reason is the same as above mentioned.)
* Community Number = 11 (in here), = 3 (in both network). The reason is the same as above mentioned.
* Pheromone evaporation factor (decay factor) = 0.5. In each time, the pheromone will decay a half. I think this proportion is suitable because if the factor is close to 1.0, meaning the pheromone will decay slowly, the result might not convergence. But if the factor is closed to 0.0, the speed of convergence will become too quickly.

3 A Small Test on the Karate Network

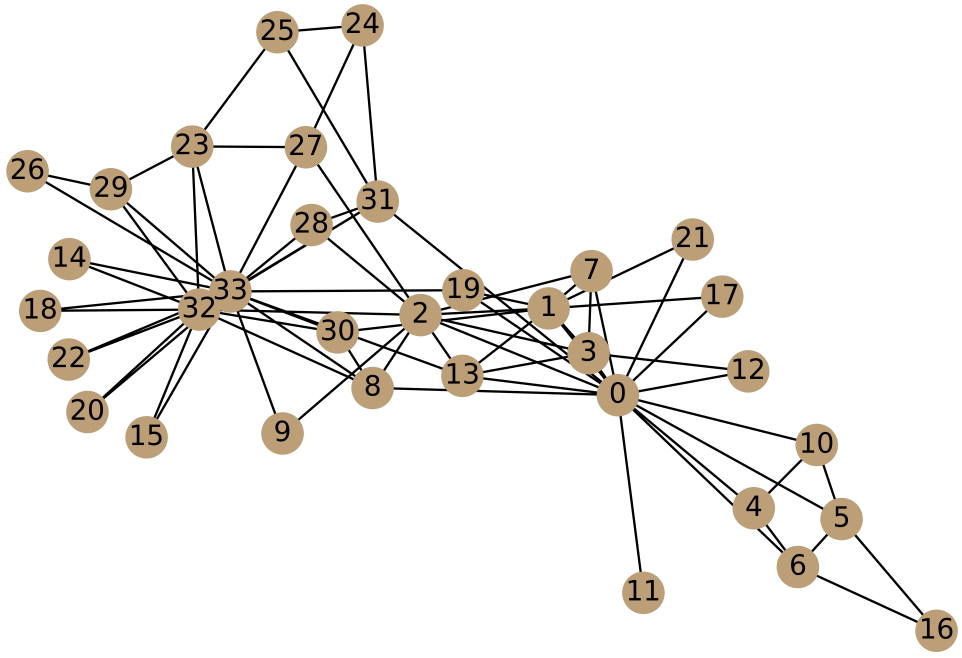

Figure 1: Visualization of the Zachary Karate Club Network

(a) Formal Statement of the optimization problem in Karate Network:

$$
\begin{aligned}
&\exists G = (V, E) \\
&V = \{0, 1, \cdots, 33\},\ |V| = 34 \\
&G = \{(0,1), (0,2), \cdots, (32,33)\},\ |E| = 78 \\
&\max Q = \frac{1}{2m} \sum_{ij} \left(A_{ij} - \frac{K_i K_j}{2m} \right) \delta(c_i, c_j)
\end{aligned}
\tag{7}
$$

where G is the set of nodes, E is the set of edges, A_{ij}, K_{ij}, m and $\delta(\cdot,\cdot)$ as mentioned above.

(b) The combination problem is equivalent to the integer (number of nodes) partition with the constraint. However, the constraint in these problem is **a community must have at least 2 nodes, but the maximum number of nodes is restricted to be less than 50% of the total number of nodes.** Denote the number of nodes equals to n, the potential combinations is $\sum_{a_i \in s,\, s \in S_n}$, where a and s follow:

$$\begin{aligned} &\exists s = (a_1, a_2, \cdots, a_j) \in S_n \\ &\quad a_1 + a_2 + \cdots + a_j = n \\ &\quad a_1 \geq a_2 \geq \cdots \geq \ a_j > 0 \\ &\quad a_i \neq 1(\text{at least 2 nodes}) \wedge a_i \leq \frac{n}{2}(\text{be less than 50\%}),\ \forall i \in \{1, j\} \end{aligned} \tag{8}$$

For example, if the network has 8 nodes, and 8 can be partitioned into $S_8 = ((2,2,2,2),(2,2,4),(2,3,3),(4,4))$, in this case, $|S_8| = 4$. Thus, all potential combinations is:

$$\sum_{a_i \in s,\, s \in S_8} \left(\frac{8!}{a_1! a_2! \cdots a_n!} \right) = \left(\frac{8!}{2!2!2!2!} \right) + \left(\frac{8!}{2!2!4!} \right) + \left(\frac{8!}{2!3!3!} \right) + \left(\frac{8!}{4!4!} \right) \tag{9}$$

By the same concept, the Karate Network has 34 nodes, and 34 can be partitioned into S_{34}, thus all potential combinations can be represented as:

$$\begin{aligned} &\sum_{a_i \in s,\, s \in S_{34}} \left(\frac{34!}{a_1! a_2! \cdots a_n!} \right) \\ &= \left(\frac{34!}{\underbrace{2! \cdots 2!}_{17}} \right) + \left(\frac{34!}{\underbrace{2! \cdots 2!}_{15} 4!} \right) + \left(\frac{34!}{\underbrace{2! \cdots 2!}_{14} 3!3!} \right) + \cdots + \left(\frac{34!}{17!15!2!} \right) + \left(\frac{34!}{17!17!} \right) \end{aligned} \tag{10}$$

(c) Please refer to the **Appendix (II)** for the further source code details. The best modularity-Q applied by each algorithm on Karate Network is as below:

(1) **Genetic Algorithm**: 0.392176199868507 (**the best overall**)

(2) **Simulated Annealing**: 0.28328402366863825

(3) **Tabu Search**: 0.16789940828402367 (**the worst overall**)

(4) **Particle Swarm Optimization**: No idea

(5) **Ant Colony Optimization**: 0.22066074950690337

(d) Check the performance using a graphical representation:

(1) Each algorithm's performance on modularity-Q in 10 iterations:

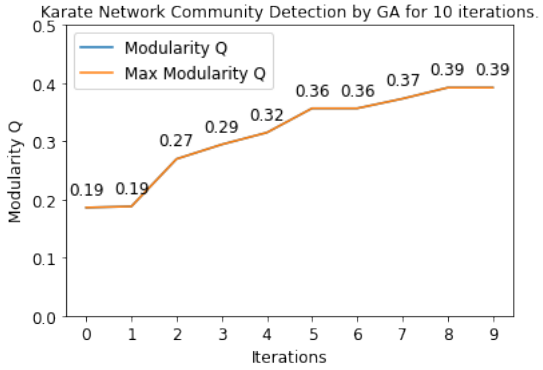

Figure 2: Genetic Algorithm for 10 iterations

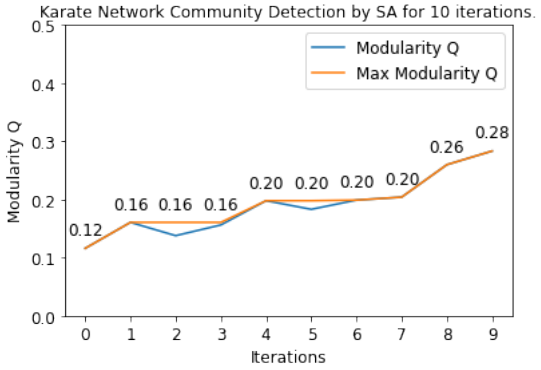

Figure 3: Simulated Annealing for 10 iterations

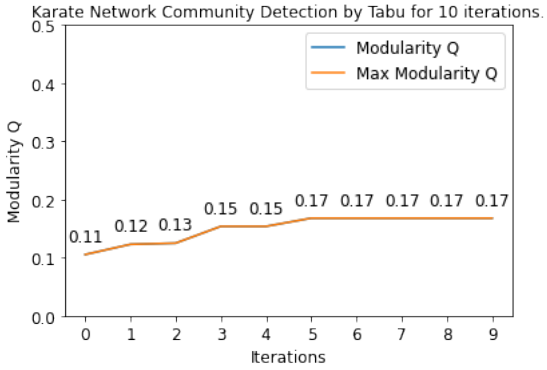

Figure 4: Tabu Search for 10 iterations

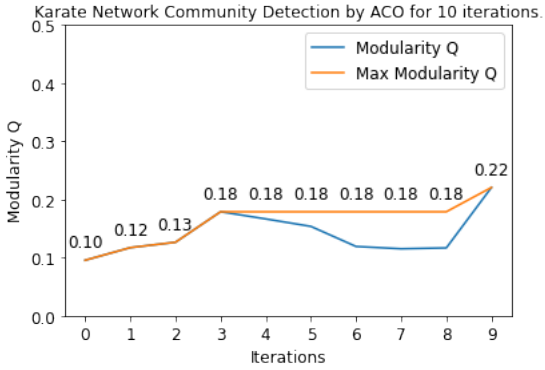

Figure 5: Ant Colony Optimization for 10 iterations

(2) Visualization the communities by each algorithm, the different colors of node represent different communities:

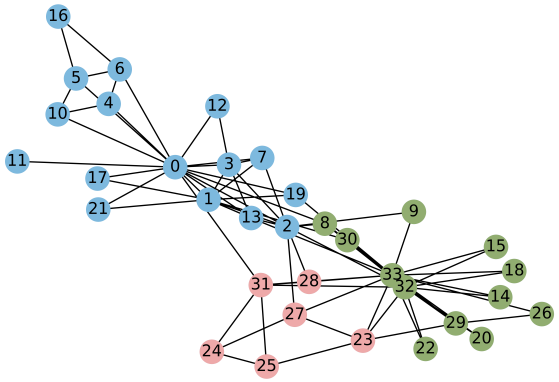

Figure 6: Using Genetic Algorithm ($Q \simeq 0.39$)

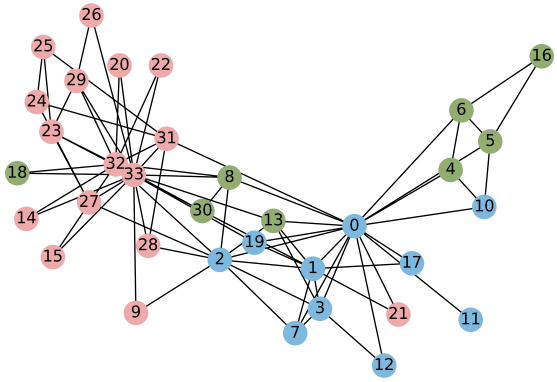

Figure 7: Using Simulated Annealing ($Q \simeq 0.28$)

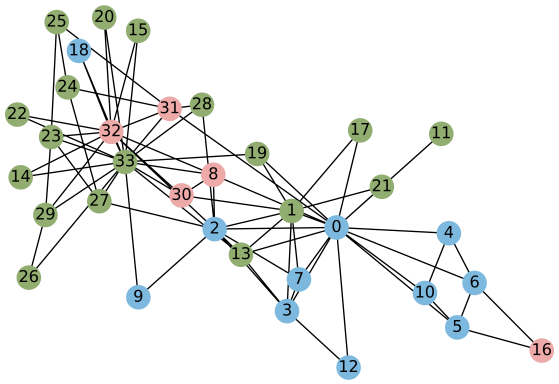

Figure 8: Using Tabu Search($Q \simeq 0.17$)

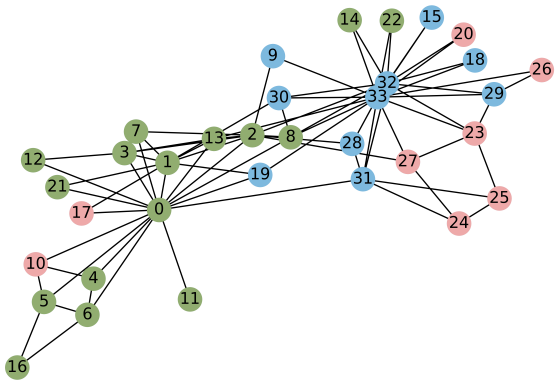

Figure 9: Using Ant Colony Optimization. ($Q \simeq 0.22$)

4 The Community Detection on the Co-authorship-dblp Network

After running your algorithms with a small testing network, we will study a large-scale citation.

(a) Omitted.

(b) List the basic statistics of the Co-authorship-dblp Network, e.g., the number of nodes and the number of edges in the network:

* **Nodes**: 540486
* **Edges**: 15245729
* **Density of the adjacency matrix**: $15245729/(540486 \times 540486) = 0.000052189$

(c) Due to some technical issues, my computer is lack of memory and computational resource to create a full adjacency matrix. Thus, I have no idea to run each algorithms on the full data. Instead of running on the full Network, **I sample 1000 nodes (authors) from the Co-authorship-dblp Network** to show the performance of each algorithms.

* **Nodes**: 1000 (random sample from the whole network)
* **Edges**: 55

* **Density of the adjacency matrix**: $55/(1000 \times 1000) = 0.000055$ (close to the raw data)

Please refer to the **Appendix (III)** for the further source code details. The best modularity-Q applied by each algorithm is as below:

* **Genetic Algorithm**: 0.27801652892562423
* **Simulated Annealing**: 0.3026446280991788 (**the best overall**)
* **Tabu Search**: 0.25586776859504756
* **Particle Swarm Optimization**: No idea
* **Ant Colony Optimization**: 0.24776859504132756 (**the worst overall**)

(d) Check the performance using a graphical representation:

(1) Each algorithm's performance in 10 iterations

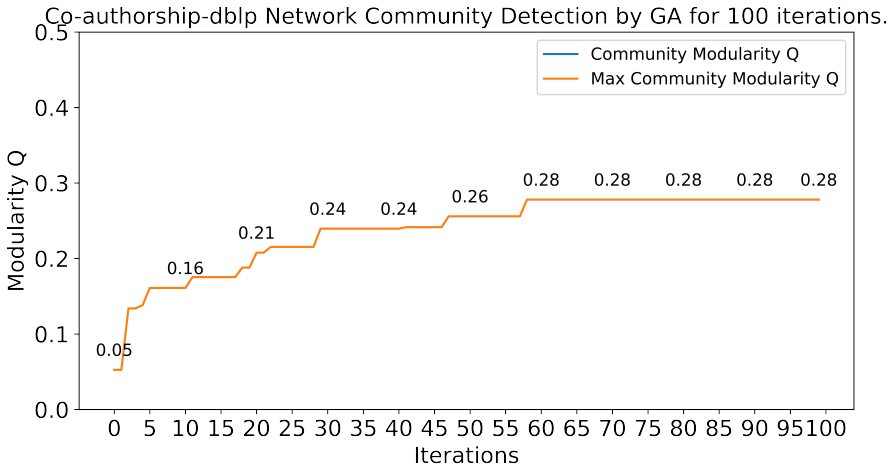

Figure 10: Genetic Algorithm for 100 iterations on Co-authorship-dblp Network ($Q \simeq 0.28$)

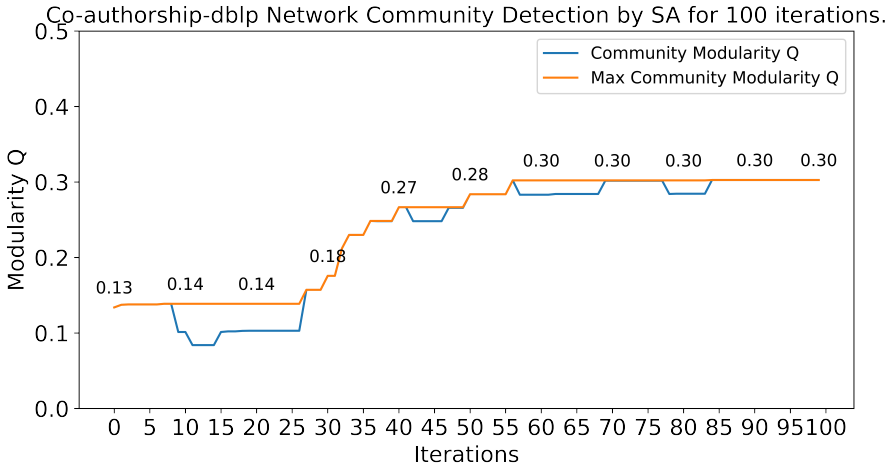

Figure 11: Simulated Annealing for 100 iterations on Co-authorship-dblp Network ($Q \simeq 0.30$)

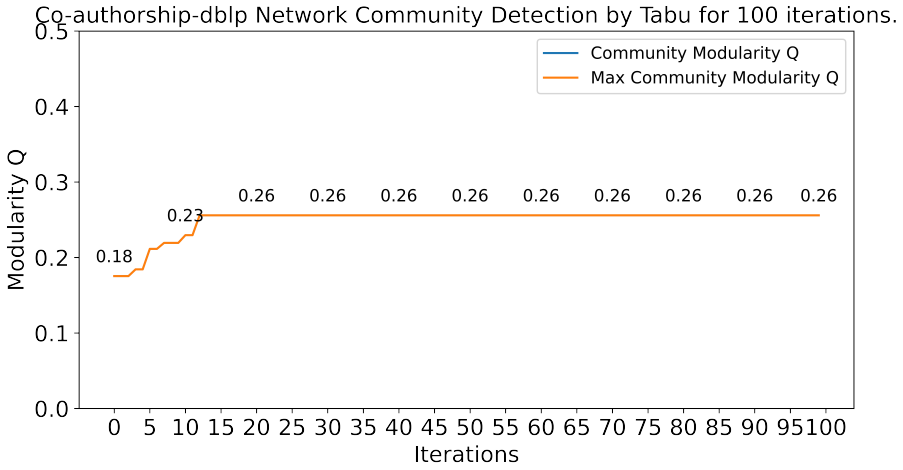

Figure 12: Tabu Search for 100 iterations on Co-authorship-dblp Network ($Q \simeq 0.26$)

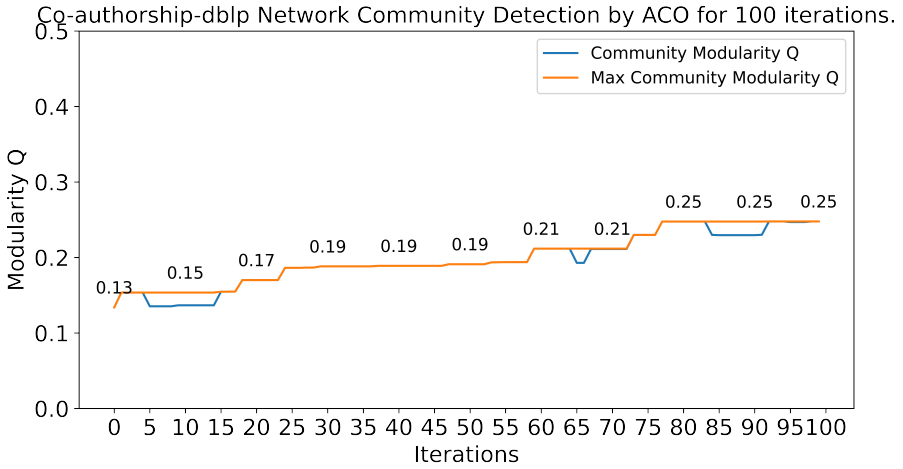

Figure 13: Ant Colony Optimization for 100 iterations on Co-authorship-dblp Network ($Q \simeq 0.25$)

5 Conclusion

In the last part of this final project, please provide the following information:

(a) Summarize your comparison results on four methods of your choice.

(1) In the short term (Karate Network, 10 iterations), the Genetic Algorithm has the best performance, and the Tabu Search has the worst performance.

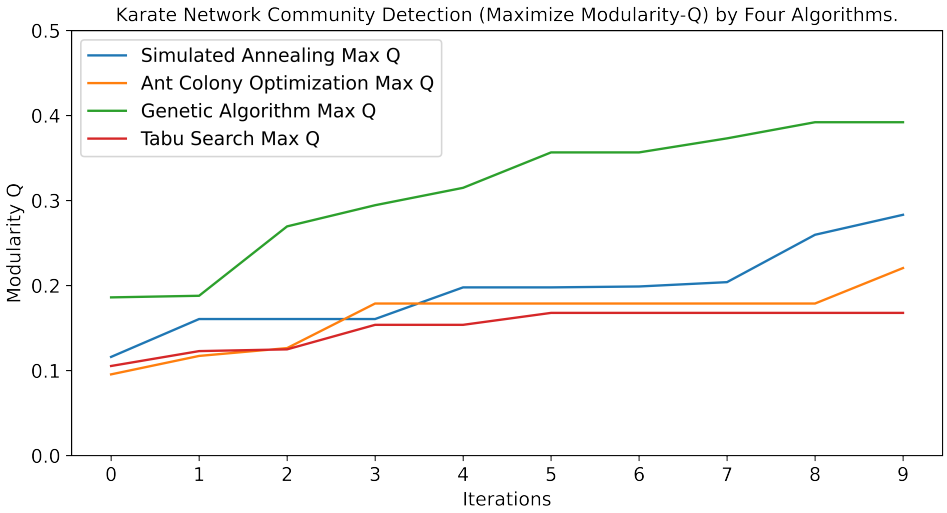

Figure 14: Karate Network Community Detection (Maximize Modularity-Q) by Four Algorithms for 10 iterations (in the short term)

(2) In the long term (Co-authorship-dblp Network, 100 iterations), the Simulated Annealing has the best performance, and the Ant Colony Optimization has the worst performance.

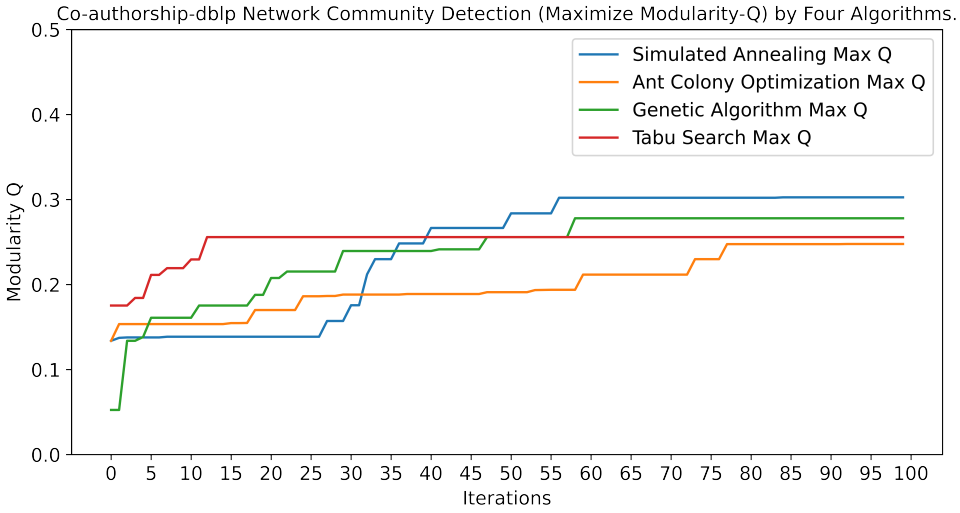

Figure 15: Co-authorship-dblp Network Community Detection (Maximize Modularity-Q) by Four Algorithms for 100 iterations (in the long term)

(b) The **Advantages** and **Disadvantages** of all four methods shown in this application to Community Detection Problem.

(1) **Genetic Algorithm**:

* **Advantages**: In the reproduction period, I use each two candidates in the parents generation to crossover, so there are totally $\binom{n}{2}$ offspring. This can generate many possible candidates, making Genetic Algorithm finding the better solution than other algorithms.

* **Disadvantages**: Due to the same reason as above, it really takes many, many time to reproduct and mutate. The time cost is very high. For instance, in Karate network, the Genetic Algorithm costs more 10 times than others. And in Co-authorship-dblp Network, due to the lacking of computational resource, I only set the particle as 10, so the performance is not very good enough.

(2) **Simulated Annealing**:

* **Advantages**: The time cost is less than others, but maybe because I only use the simplest mutation method, single bit flip (SBF), in my programming.
* **Disadvantages**: In the short term (e.g. only 10 iterations), Simulated Annealing will usually accept "not the better case" in one iteration due to $\mathbb{P} = \min\{1, \exp(-\Delta f/T)\}$. This will make the objective function, modularity-Q, become worse. Though sometimes Q will increase after decreasing (leave the local optimal solution), sometimes it will keep decreasing. However, in the long term, the problem can be optimized.

(3) **Tabu Search**:

* **Advantages**: Depending on using the tabu list, it will increase in the long term, and making the result not too bad.
* **Disadvantages**: It should need to spend many time to try which length of tabu list should use. Though I use 10 in my programming, but maybe it can be replaced by other one. But I have no time to try every possible value.

(4) **Ant Colony Optimization**:

* **Advantages**: It can leave the local optimal solution, and find other neighbors as the candidate points.
* **Disadvantages**: Similar to Simulated Annealing, the performance of Ant Colony Optimization in the short term is not good enough, because it will try another route though it had reached a "maximum" point to avoid trapping in local optimal solution. However, in the short term, this advantage is not clear to show its feature. Another problem is, as the number of ants increasing, this algorithm takes more time cost for iteration.

(c) Potential improvement on the best method to make the algorithm even better on this problem

* As above mentioned, I assumed that the less community we separate, the modularity-Q might be larger before. Then set the number of community as 3 at first due to the number of member restriction. However, the number of community can be dynamic, rather than a fixed value given at first. This can be one potential improvement.
* According to He (2106), one possible method is to calculate the **Similarity Metric** first, i.e.:

$$S_{ij} = \sum_{z \in nb(i) \cap nb(j)} \frac{1}{K_z} \tag{11}$$

where $z \in nb(i) \cap nb(j)$ are the common neighbors of nodes i and j. The normalization of the Similarity Metric is:

$$S^*_{ij} = \frac{\sum_{z \in nb(i) \cap nb(j)} \frac{1}{K_z}}{\sqrt{\sum_{z \in nb(i)} \frac{1}{K_z} \sqrt{\sum_{z \in nb(j)} \frac{1}{K_z}}}} \tag{12}$$

If nodes i and j have no common neighbors, $S^*_{ij} = 0$; if nodes i and j have the same neighbors, S^*_{ij} is close to 1. We can compute the similarity of its two nodes i and j according the normalization of Similarity Metric.

* Given a node i, if $\forall j \in nb(i)$, $S^*_{ij} = 0$, node i is still a single node in community ϕ_0. If $\exists j \in nb(i)$, $S^*_{ij} > 0$, and if S^*_{ij} is maximal among the neighbors of node i, nodes i and j belong to the same community in π_0.

* By this method, we can initialize a better statement rather than only use the random method. And the result of initialization can decide how many community we should separate in the algorithms.

References

[1] Ranjan Kumar Behera et al. "Genetic algorithm-based community detection in large-scale social networks". In: *Neural Computing and Applications* (2019), pp. 1–17.

[2] Jialin He, Duanbing Chen, and Chongjing Sun. "A fast simulated annealing strategy for community detection in complex networks". In: *2016 2nd IEEE International Conference on Computer and Communications (ICCC)*. IEEE. 2016, pp. 2380–2384.

[3] Chang Honghao, Feng Zuren, and Ren Zhigang. "Community detection using ant colony optimization". In: *2013 IEEE Congress on Evolutionary Computation*. IEEE. 2013, pp. 3072–3078.

[4] Charles Kadushin. "Introduction to social network theory". In: *Boston, Ma* (2004).

[5] P Mazur, K Zmarzlowski, and AJ Orlowski. "Genetic algorithms approach to community detection". In: *Acta Physica Polonica-Series A General Physics* 117.4 (2010), p. 703.

[6] Mark EJ Newman and Michelle Girvan. "Finding and evaluating community structure in networks". In: *Physical review E* 69.2 (2004), p. 026113.

[7] Ryan A. Rossi and Nesreen K. Ahmed. "The Network Data Repository with Interactive Graph Analytics and Visualization". In: *AAAI*. 2015. URL: `http://networkrepository.com`.