SUSTech CS303 2023 Fall Project1 Report

Chunhui XU

November 18, 2023

1 Introduction

1.1 Problem Introduction

This project is originated from the age of information explosion. People are exposed to thousands of messages through our social network every day. But if only see one point of view, they will fall into an information cocoon. This project gives students a chance to simulate IEM problem: find the larger number of people either receive comprehensive information or remain obvious.

In order to characterize, let a DAG be the model of information transmission chain, and there are two conflicting viewpoints. Each node has its own probability to accepts the two viewpoints respectively. The problem should find a group of initial nodes to spread viewpoints, and let the number of nodes that accept both viewpoints or remain neutral be maximum.

1.2 Work Purpose

In this project, we need:

1. Understand how to model the IEM problem.
2. Find an efficient way to simulate and evaluate how good the initial nodes is.
3. Use greedy algorithm and evolutionary (or simulated annealing) algorithm to find the optimal solution to the IEM problem.

After the above steps, we have a preliminary understanding of using AI to model problems and seek their answers.

2 Preliminary

This section is originated from Project requirements document.

2.1 Notation Definitions

- **Social Network:** $G = (V, E)$, where $V = \{v_1, \cdots, v_n\}$ represents the node set, and $E = V \times V$ represents the edges between nodes.
- **Campaigns:** $C = \{c_1, c_2\}$, represents two campaigns; each campaign holds a viewpoint.
- **Initial Seed Set:** $I_i \subseteq V, i \in \{1, 2\}$ represents the initial seed set for campaigns c_i.
- **Balanced Seed Set:** $S_i \subseteq V, i \in \{1, 2\}$ represents the target seed set that you need to find for each campaign c_i.
- **Budget:** k represents the size of the target seed set; $|S_1| + |S_2| \leq k$.
- **Diffusion Probability:** $P_i = \{p^i_{(u,v)} | (u, v) \in E\}, i \in \{1, 2\}$ represents the edge weight associated with campaign c_i where $p^i_{(u,v)}$ represents the probability of node u activating node v under each campaign c_i.
- **Explored Result:** $r(U)$ represents the explore result of seed set U

2.2 Simulation Model

The famous Independent Cascade (IC) model is used. Each node $v \in V$ has two possible states, inactive and active.

- *active*: adopts new information being propagated through the network
- *inactive*: has not adopted new information yet, two kinds:
  - i Ever been attempted to be activated but NOT
  - ii Never been attempted to be activated

Each node can ONLY switch from inactive to active.
The **active nodes** and **inactive nodes in i** will be the explored seed set.

2.3 Result Expression

Given a social network $G = (V, E)$, two initial seed sets I_1 and I_2, and a budget k. The IEM is to find two balanced seed sets S_1 and S_2 , where $|S_1| + |S_2| \leq k$ and maximize the balanced information exposure, i.e.,

$$\begin{aligned} \max \Phi(S_1, S_2) = \max \mathbb{E}[|V \setminus (r_1(I_1 \cup S_1) \triangle r_2(I_2 \cup S_2))|] \\ \text{s.t.} \quad |S_1| + |S_2| \leq k \\ S_1, S_2 \in V \end{aligned}$$

3 Methodology

3.1 Greedy Algorithm

In greedy algorithm, we do these things:

1. For c_i, c_2, prune the original image respectively: delete all edges with $p^i_{(u,v)}$ less than 10^{-7}.
2. In the new graph, for each single node i, run the Monte Carlo Simulation for c_1, c_2, get node set $r_1(\{v_i\})$, $r_2(\{v_i\})$. Simulate three times for two campaigns respectively, take their intersection as the final result for this point.
3. Run the greedy algorithm as **Algorithm 1**, because we already got the $r(\{v\})$ for every node and campaign, we have the estimate:

   $$r_1(\widehat{S_1 \cup \{v_i\}}) = r_1(S_1) \cup r_i(\{v_i\})$$

   So we can quickly calculate $\Phi(S_1 \cup \{v\}, S_2)$ using set operations. Same for $\Phi(S_1, S_2 \cup \{v\})$.
4. When $|S_1|+|S_2| = k$, or there's no new node can improve current $\Phi(S_1, S_2)$ the greedy algorithm terminate and return S_1, S_2.

We need $O(|V| + |E|)$ to simulate every single node, and $O(k|V|)$ for greedy algorithm. So the overall time complexity is $O(k|V|)$, where V, E are from the pruned graph.

3.2 Evolutionary Algorithm

In evolutionary algorithm, we do these things:

1. Same steps as greedy algorithm steps 1, 2.
2. Generate the initial population randomly. The core idea: sort the influence $|r(\{v\})|$ of individual nodes. The higher influence the node has, the greater probability the node will in the initial population.

Algorithm 1 IEM Greedy

Require: $k > 0$
  $S_1, S_2 \leftarrow \emptyset$
  $\Phi_{max} \leftarrow \Phi(S_1, S_2)$
  while $|S_1| + |S_2| < k$ **do**
    $v_1^* \leftarrow \arg\max_v(\Phi(S_1 \cup \{v\}, S_2) - \Phi(S_1, S_2))$
    $v_2^* \leftarrow \arg\max_v(\Phi(S_1, S_2 \cup \{v\}) - \Phi(S_1, S_2))$
    if $\Phi(S_1 \cup \{v_1^*\}, S_2) = \Phi(S_1, S_2 \cup \{v_2^*\}) = \Phi(S_1, S_2)$ **then**
      break;
    else if $\Phi(S_1 \cup \{v_1^*\}, S_2) \geq \Phi(S_1, S_2 \cup \{v_2^*\})$ **then**
      $S_1 \leftarrow S_1 \cup \{v_1^*\}$
    else
      $S_2 \leftarrow S_2 \cup \{v_2^*\}$
    end if
  end while
  Output S_1, S_2

3. Sort the item in population by Φ and keep the first larger half.
4. Use the first half, pick two parents, cross over them to generate two new sons, and mutate them if they have the opportunity.
5. Let the number of parents and new sons be same, then continue from step 3
6. After certain times from step 3 to 5, terminate and return item that have the largest Φ in current population.

Algorithm 2 IEM Evolutionary Main

Require: $k > 0$, popNum
  $Pop \leftarrow$ randomChioce() * popNum
  for $i : 1 \rightarrow iterations$ **do**
    for $i : 1 \rightarrow popNum/2$ **do**
      $son1, son2 \leftarrow crossover(randomParents(Pop))$
      $son1 \leftarrow mutation(son1)$
      $son2 \leftarrow mutation(son2)$
      $Pop \leftarrow Pop \cup \{son1, son2\}$
    end for
    $Pop \leftarrow Sortby\Phi(S_1^i, S_1^2)$
    $Pop = Pop[0 : popNum/2]$
  end for
  Output Pop[0]

In cross over part, I use single point, two points and uniform cross over. In mutation, I use flip bit mutation and exchange mutation.

The data structure of item in population is two sets (S_1^i, S_2^i), represent one situation of the two balanced seed set. If $|S_1^i| + |S_2^i| > k$, there Φ will be **negative** to mark it as invalid.

Consider n as size of population and m as the iteration numbers. We need $O(|V| + |E|)$ to simulate every single node, $O(|V| \log |V|)$ pre-processing for the initial population, and $O(mn \log n)$ for evolutionary algorithm. So the overall time complexity is $O(mn \log n)$.

4 Experiments

4.1 Test Environment

In this part, I use the OJ result. The hardware/software as follow:

- Operation System: Debian 10
- Server CPU: 2.2GHz * 2
- Python version: 3.9.7

This algorithm can be evaluated as follows: when reaching the baseline, the shorter the time to get the result, the better

4.2 Greedy Experiments

Table 1: Data Set of Greedy

Case	Nodes	Edges	k	Baseline	My OJ Time
1	475	13289	10	430	1s
2	36742	49248	15	35900	75s
3	36742	29248	15	35900	49s
4	7115	103689			28s
5	3454	32140			68s

From this table, we can found that, if the nodes is large, the program will take long time, since it will traverse all nodes in each greedy process.

If the $|V|$ get larger and larger but k remain the same, the algorithm will take more time. So from this perspective, it is necessary to pick up effective nodes for greedy algorithms in advance.

4.3 Evolutionary Experiments

Table 2: Data Set of Evolutionary

Case	Nodes	Edges	k	Baseline	My OJ Time
1	475	13289	10	415	5s
2	13984	17319	14	13580	63s
3	13984	17319	14	13580	45s
4	3454	32140			221s
5	3454	32140			116s

From this table, we can found that, if the edges is large, the program will take long time, since it will need more iterations to generate and find the most effective node.

If the $|V|$ and $|E|$ are very close, maybe there will be some nodes has significant influence, it will be more easier to hold the good point in population.

5 Conclusions

5.1 Algorithm Evaluation

5.1.1 Greedy Evaluation

- Advantages:
  - The idea is simple and the code is easy to understand.
  - Obtained sufficiently effective results.
- Disadvantages:
  - There are very few optimization parts of the algorithm, only pruning.
  - There is still a lot of room for improvement in terms of ultimate efficiency.

5.1.2 Evolutionary Evaluation

- Advantages:
  - Because of the better initial population, the number of iterations can be lower.
  - Obtained high enough results.
- Disadvantages:
  - The initial population generation takes more time
  - The generality to other problems needs to be improved.

5.2 Algorithm Improvement

5.2.1 Greedy Improvement

In greedy algorithm, I used a basic algorithm idea and got enough results to pass the review. However, there is still room for optimization as follows:

- The criteria for filtering new nodes can be improved. For example, filter out all points with out-degree greater than a certain value and perform a greedy algorithm. This avoids wasting too much time by traversing all nodes.
- More appropriate pruning can be dynamically selected: to avoid pruning too many edges to cause information loss , which will significantly reduce the accuracy of the algorithm.

5.2.2 Evolutionary Improvement

In evolutionary algorithm, I chose a better initial population to achieve a higher base value. However, there is still room for optimization as follows:

- Generate the initial population completely randomly: This enables solving more general problems, rather than using a some strategy.
- Achieve more crossover and mutation processes: such as OX, PBX and Inversion mutation.
- Implement more ways to select parents: Set more ways to assign weights to parents with different rankings.

5.3 Learning summary

In this project, greedy algorithm and evolutionary algorithm were used to provide a solution to the IEM problem. I learned how to transform real world problems to abstract model. Finally, I try to use AI to solve the problem, and achieved the expected results.