Influence Maximization Problem

Ziqiang LI 李子强(11510352)
Department of Computer Science and Engineering,
Southern University of Science and Technology
Email: 11510352@mail.sustc.edu.cn

1. Preliminaries

Influence Maximization Problem is the problem of finding a small subset of nodes(seed nodes) in a social network,that could maximize the spread of influence. The IMP is NP-hard and the influence spread computation is $^{\#}P$-hard under the definitions shown in the introduction. I improve Degree Discount IC Algorithm [1], which can pick up the parent of high impact node. Besides, I also implement influence spread estimator with independent cascade (IC) and linear threshold (LT) models.

2. Methodology

2.1. Data structures

- **Network**: nested dictionary, storing both adjacent matrix and inverse adjacent matrix.
- dd_v: priority queue

2.2. Main functions

TABLE 1. IMPORTANT VARIABLES USED IN THE PAPER

Variable	Descriptions
AS	Activity Set
W_{sn}	weight between s and n
d_v	degree of vertex v in G
p	propagation probability in the IC model
t_v	number of neighbors of vertex v already selected as seeds
$argmax_v$	maximum value for vertex v

Algorithm 1 and Algorithm 2 show how to take one sample by the respective model [2]. In general, I take 10 thousand samples to finally estimate a seed's influence.The process of ISE is really time-consuming,so , when I evaluate the seeds of the output of Algorithm 3, I will take just hundreds or thousands samples, which the error for smaller sample is admissible, and I will re-evaluate the seeds seems performing well with more iteration.

A way to make Algorithm 2 more faster is that initialize vertex's threshold when it is used instead of initialize them all at the beginning.

Algorithm 1 IC(G, S)

Input: network G, seed set S
Output: the number of nodes influenced

initialize $AS \leftarrow S$
$count \leftarrow |AS|$
while $AS \neq \emptyset$ **do**
  initialize $newAS \leftarrow \emptyset$
  for each seed $s \in AS$ **do**
    for each inactive neighbour n **do**
      s tries to activate n by using W_{sn}
      if n is activated **then**
        $newAS \leftarrow newAS \cup \{n\}$
      end if
    end for
  end for
  $count \leftarrow count + |newAS|$
  $AS \leftarrow newAS$
end while
return $count$

Because computing w_total is time-consuming, LT model is slower than IC about 3 times.

Algorithm 2 LT(G, S)

Input: network G, seed set S
Output: the number of nodes influenced

initialize $v.thresh \leftarrow rand()$ for all $v \in V$
initialize $AS \leftarrow S$
$count \leftarrow |AS|$
while $AS \neq \emptyset$ **do**
  initialize $newAS \leftarrow \emptyset$
  for each seed $s \in AS$ **do**
    for each inactive neighbour n **do**
      compute w_total for n
      if $n.w\_total \geqslant n.thresh$ **then**
        set n is activated
        $newAS \leftarrow newAS \cup \{n\}$
      end if
    end for
  end for
  $count \leftarrow count + |newAS|$
  $AS \leftarrow newAS$
end while
return $count$

Algorithm 3 Degree_Discount(G, k)

Input: network G, selecting size k
Output: S
1: initialize $S \leftarrow \emptyset$
2: **for** each vertex v **do**
3: $d_v \leftarrow \sum W_{vu}$ for each neighbour u of v
4: $dd_v \leftarrow d_v$
5: initialize $t_v \leftarrow 0$
6: **end for**
7: **for** $i = 1$ to k **do**
8: select $u \leftarrow argmax_v\{dd_v \mid v \in V \setminus s\}$
9: **while** u has a big parent m **do**
10: $u \leftarrow a$
11: **end while**
12: $S \leftarrow S \cup \{u\}$
13: **for** each neighbour v of u and $v \in V \setminus s$ **do**
14: $t_v \leftarrow t_v + W_{uv}$
15: $dd_v \leftarrow d_v - 2t_v - (d_v - t_v)t_v p$
16: **end for**
17: **end for**
18: **return** S

Algorithm 3 always choose vertex v with the greatest degree discount into seeds S (Line 8) [1], but there is a case that v has a big parent m that can completely activate v, if m is activated, but m is not chose as seeds. $big\ parent$ (Lines 9-11) can fix this case. $big\ parent$ means that a node has a edge to its neighbour with the weight of 1, and what should be noticed is that whether it traps into infinite loop to find $big\ parent$.

3. Empirical Verification

In practical, Algorithm 3 also can add some stochastic process in select u (Line 8) to get a better seed, for example randomly choosing u by the distribution of weight dd_v.

I test Algorithm 3 with given network file. Without multiprocessing optimization, It can get a great seed with 4 vertex in 2 seconds which just run hundreds of iteration.

3.1. Performance

In below shows the performance. Table 2 gives the time for each ISE model run ten thousands times to give a estimated influence for a seed. Table 3 gives the time for generate ten thousands seeds, evaluate them and return the best seeds.

TABLE 2. EXPERIMENTAL RESULTS OF ISE

Seed size	4	10
IC	0.33 s	0.479 s
LT	0.84 s	1.05 s

TABLE 3. EXPERIMENTAL RESULTS OF IMP

Seed size	4	10
IC	4.66 s	12.20 s
LT	8.54 s	15.726 s

References

[1] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* - KDD 09, 2009.

[2] P. Shakarian, A. Bhatnagar, A. Aleali, E. Shaabani, and R. Guo, *Diffusion in social networks*. Cham: Springer, 2015.