Influence Maximization on Yelp user review network
Harvard AM 207 Final Project

Chang Liu, Richard Kim and Joseph Palin

May 12, 2015

1 Abstract

In today's digital economy, numerous social network sites take advantage of "network effects" to fuel their large-scale successes. One common strategy called viral marketing incentivizes a few prominent users to try a new product hoping they will make recommendations that influence other users to use follow suit. But who should be incentivized to optimize this scheme?

This problem is an instance of a general NP-hard problem coined "The Influence Maximization Problem." In our data set, a collection of reviews released by Yelp for their data set challenge, we first build an "Independent Cascades Model" for measuring the influence an initial set of incentivized users will have on the network as a whole. Then, we implement two stochastic algorithms, one simulated annealing algorithm and one genetic algorithm, and compare their performances with two baseline popular approaches, the general greedy algorithm and a high-degree node-selection heuristic, that are prominent in the relevant academic literature.

We find that our stochastic algorithms outperform the heuristic approach and match the greedy algorithm in terms of the estimated influence of their outputs. More importantly, the efficiencies of our algorithms are asymptotically faster than that of the greedy algorithm.

2 Introduction

2.1 Motivations

In recent years, we have witnessed a surge of online sites that serve as large-scale social networks platforms for businesses to promote their products and services; Yelp is one of the most successful examples. To promote the effect of "word of mouth", Yelp's business model relies on its enormous user base (116.6 million monthly active users globally [1]) to generate an unparalleled number of reviews of local businesses (67 million reviews [2]), and these reviews in turn influence other users on their decisions to try out a business. In fact, a recent Nielsen survey shows that, "82 percent of Yelp users visit the site when preparing to spend money on a product or service." [3] 41% respondents always or frequently make purchases and 52% do so occasionally, 44% of users regard the review text as crucial to their decision to make a purchase, 26% prefer ratings, and 17% prefer number of reviews on a business [4].These reasons suggest that whether a user writes a review or not is a good measure of the user being influenced by reading previous reviews and making a purchase; we will later use this measure to quantify the effect of "word of mouth" over the user network.

In addition, as a part of its marketing campaigns, Yelp incentivizes a subset of its users (called "Elite Yelpers") to generate high-quality reviews to attract other users to visit and review more businesses. Thus, information about businesses originate from a few initial users and cascade down the user network. With such powerful purchasing influence stemming from the reviews, 95% of Yelp's revenue comes local advertising [3]. Consequently, Yelp would be interested in two problems: 1) measuring how effective is the influence of a set of reviews over the entire user network, and 2) discovering who among the users are able to engage and influence the largest number of other users. With these problems in mind, we conduct an empirical study of Yelp's released dataset [5], by proposing an Independent Cascades Model to problem 1 and applying approximation algorithms for Influence Maximization Problem to solve problem 2 .

2.2 Related Work

In 2003, Kempe et al. [6] showed that the influence maximization problem is a discrete optimization problem modeled in a directed multi-graph: a node has some directed edges toward other nodes, each edge with a certain probability to influence or "activate" its neighbors. The problem is to find k initial nodes that maximizes the expectation of the number of nodes activated at the end of a stochastic cascade process, which is called the influence function $f(S_1)$ for a k-size set S_1 of initial nodes. Kempe et al. show that the problem is NP-hard.

They present a greedy algorithm and proved it a 63%-approximation algorithm, which guarantees the output of the algorithm to be within 63% of the optimal solution. However, how to best compute the stochastic influence function is still an open question. Kempe et al. run Monte-Carlo simulations and obtain accurate estimates of the influence function. The disadvantage is that it is unscalable on a large network due to the large number of simulations (10,000 trials for each estimate of the influence function). Few subsequent studies addressed the issue of efficiency. In 2007, Leskovec et al. proposed a "Cost-effect Lazy Forward" algorithm that improves efficiency up to 700 times to that of the original greedy algorithm [7]. But the algorithm could still take hours to finish on a moderately large graph with 15,000 nodes. In 2009, Chen et al. [9] discovered a "degree discount heuristic" that finds a solution close to that of the best greedy algorithm, but in milliseconds, less than one-millionth of time of the fastest greedy algorithm. This is the fastest algorithm we know so far.

The greedy algorithm developed by Kempe et al. [6] takes advantage of the submodular nature of the influence function in the independent cascade model. A prominent feature of the greedy algorithm is that it outperforms classical heuristics such as selecting high-out-degree nodes or high-centrality nodes. The study of degree and centrality in influence cascades has been discussed in the field of social network analysis [10].

However, if we do not take into account the submodular nature of the influence function (because it might not be warranted in some alternative models), could we introduce more sophisticated techniques that improve stochastic optimization? In this paper, we design two stochastic optimization algorithms, a genetic algorithm and a simulated annealing, and compare them against the greedy algorithm and a high-degree heuristic method.

2.3 Libraries used

For this project, we utilized: numpy, matplotlib, networkx, and pandas.

3 Methods

3.1 The Independent Cascade Model

Graph Development To measure the effect of "word of mouth" in Yelp's network, we model the problem as a directed graph $G(V, E)$: A node v_1 in the set V represents a Yelp user, and a directed edge between two users v_1 and v_2, denoted (v_1, v_2), represents v_1 influencing v_2. In our model, influence of v_1 on v_2 is a function of the number of businesses v_1 has reviewed, and the number of businesses that v_2 has subsequently reviewed within a time period.

In searching for a function to model the influence edge weights, we needed a function that returned values between 0 and 1 to serve as probabilities. The Gaussian was ruled out because it extended beyond that desired range. The standard uniform distribution provides the desired range, and functions well for testing, but it yields edge weights that are consistently high enough to eliminate the need for any optimization.

The next improvement was to draw edge weights from a Beta distribution with parameters α, the number of positive reviews a neighbor node wrote within 90 days after the root node, and β, the total number of reviews written by the neighbor node. This provided edge weights that were usually closer to 0 than to 1, mimicking the general expectation of one Yelper's influence over others.

Additionally, choosing α and β as described also encapsulates that users routinely following other users are more susceptible to influence, and users with more regular reviews (possible food critics) are less susceptible to influences by others.

As a final revision to the edge weight function, the square root of the Beta edge weight value was returned. This was done because, while the uniform distribution gave values that were too high to need optimization, the Beta distribution itself yielded values that were too low to need optimization.

Independent Cascade Once we create the graphs modeling the Yelp network, we use it to measure the influence that different collections of Yelp users have over the graph as a whole over time. Just as any user's review may influence another user or succession of users to write a review, one node in a graph may affect many other nodes through a tree of directed edges. The starting node and the succession of affected neighbor nodes forms an Independent Cascade. The more generalized version of the Independent Cascade starts with one or more nodes, and tabulates the total number of nodes eventually influenced by the starting set.

The starting set of nodes is denoted S_1. With probability proportional to the directed edge weights between nodes and neighbors, the set of affected or "activated" nodes in time period one. Each successive set of activated nodes is determined from the set of nodes activated in the immediately prior time period. The final set of all sequentially activated nodes is expressed $I(S_1)$, where S_1 can be any initial set, and the function $I()$ returns the set of all initial nodes and all nodes activated in one independent cascade.

Influence Function The purpose of the Independent Cascade model is to find, for any starting set size k, the initial nodes that have the highest expected number of activated nodes after t discrete time steps. Because activating a neighbor of an activated node is dependent on a probability defined by the edge weight, even with identical starting sets, repeated cascades will not necessarily return the same number of activated nodes. To get an estimate of the expectation of a set of activated nodes, we use Monte-Carlo simulations: each initial set's cascade is run N times, and the average of the runs is used for the expectation of the cascade, expressed

$$f(S_1) = E[|I(S_1)|]$$

3.2 Optimization and Stochastic Optimization

With the graph built and the influence function defined, we turn to the question of which set of k initial users maximizes the influence.

$$\arg\max_{S_1 \subset N} f(S_1)$$

$$s.t. |S_1| = k$$

This Influence Maximization problem, as it is commonly called in the literature, is an NP-hard problem. Therefore, it is prohibitively expensive to find the actual solution, and we turn to various approximation algorithms and compare their performance in terms of estimated influence.

The heuristic method and greedy algorithm are compared with the genetic algorithm and simulated annealing algorithm. The heuristic method will serve as a baseline.

Heuristic Method The high-degree heuristic is the simplest and the fastest optimization technique. It sorts the nodes by out degree and selects the k nodes with highest out degree.

Greedy Algorithm The greedy algorithm starts by initializing an empty set S. Then, it repeats k times: for each node i in $V \setminus S$, find the node that maximizes $f(S \cup \{i\}) - f(S)$. We add the optimal node to the set S. The algorithm terminates with k nodes in S. However, a set of k local optimal does not guarantee that the set S is the global optimum because the algorithm commits to each choice of i without the ability to backtrack and consider sets that are not constructed as local optima that might be the foundation for a global optima. However, theoretically, it has been proved by Kempe et al. that such greedy strategy guarantees the answer is at least 63% of the optimal solution.

Simulated Annealing The simulated annealing approach takes a starting set of nodes in the graph, and then swaps out one of the nodes for a new node as it walks around the search space of possible user sets. As time progresses, the expected influence of each user set is annealed upon, and it becomes harder for the algorithm to walk the current user set into a relatively low influence set. The number of nodes that can be changed with any step is subject to re-configuration, but it is not set at the complete set size as that would mimic the results of an algorithm that picked a completely new random set of nodes at every phase, only slower due to sporadic rejection.

Genetic Algorithm The genetic algorithm approach starts with an initial population of possible solutions. Each of those possible solutions is scored by its expected influence and recorded. After the initial population is set, two phases occur, crossover and mutation. In the crossover phase, the first node in each set is swapped with another node from the population. In the mutation phase, with probability proportional to the mutation rate, each node in the first node set of the population is randomly changed. The population is repeatedly perturbed in this manner until the prescribed number of iterations are done and an optima is returned.

3.3 Parameter values

To test how the various algorithms worked over different node set sizes k, we ran the algorithms from $k = 1$ to $k = 20$. The number of time iterations the cascade is allowed to perform for influence calculations was capped at $t = 10$. We set $N = 30$ because it yielded small enough standard deviation of f while not costing an inordinate amount of computation time for a function that would be called more than one hundred thousand times.

3.4 Data

Due to our limited computing resources, we worked on the North Carolina subgraph of the Yelp Dataset Challenge dataset [5]. We subset further on positive reviews (4 stars or more) to model cascading effects of positive reviews. The ensuing sub-graph is further subsetted keeping businesses with over 150 reviews and users

with over 60 reviews. This last subsetting keeps users who are highly connected while also keeping the edges in the subgraph from being too dense or too sparse. This subgraph had 182 user and 3402 directed edges.

4 Results

We give a visualization of the simulated influence cascade of k=5 users after 10 iterations of time over the North Carolina subgraph in **Figure 1**. We sample the stochastic processes multiple times to get a fairly accurate estimate of the influence function f. To determine a suitable sample size N to run our algorithms, we sample the processes for $N = 10, 20, 30$ ... up to 400, and repeat 10 trials for each N. We plot a graph of the standard deviations of f over the 10 trials against sample size N in **Figure 2**. As a measure of justifying the running time for each N, we plot the standard deviations per unit time on a log scale $\gamma = \frac{log(std(f))}{T}$ where T is the average time of 10 trials, in **Figure 3**. We find that $N = 30$ with standard deviation around 1 is suitable because 1 standard deviation of f can be interpreted as a 1 user difference in a discrete optimization problem, and also because we are close to the limit of using our computational resources economically.

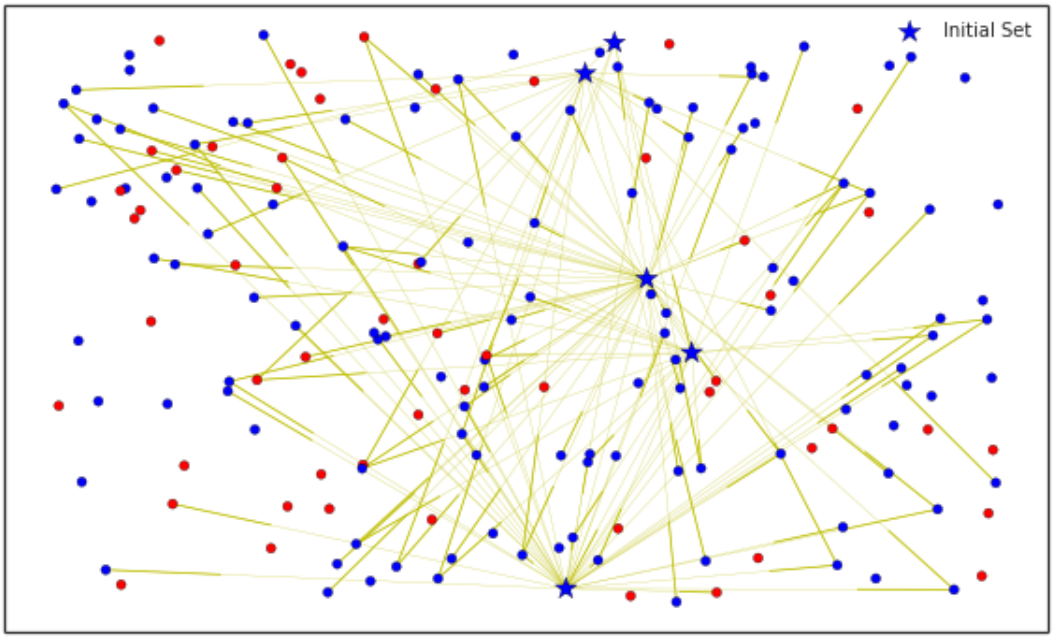


Figure 1: Simulated influence cascade of $k = 5$ initial users over the Yelp user network in North Carolina. Stars represent initial users, blue and red dots activated and unactivated users, and yellow lines the neighbors of the initial users.

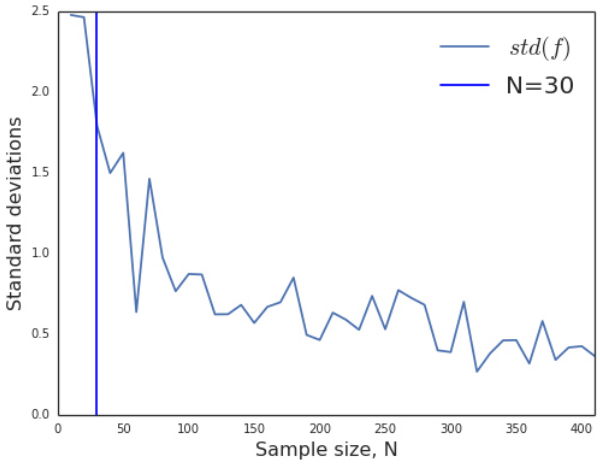


Figure 2: Standard deviations of the simulated f with respect to sample size N

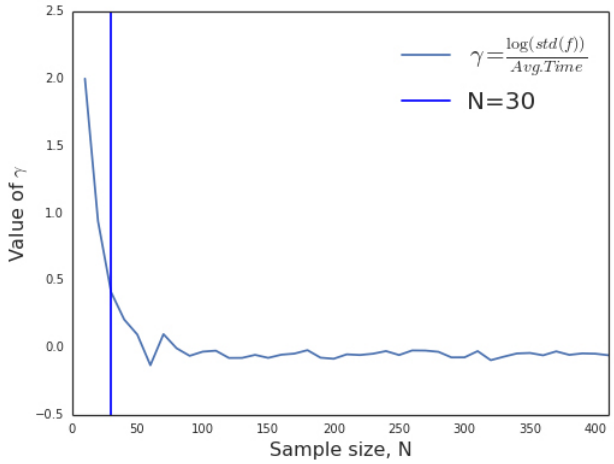
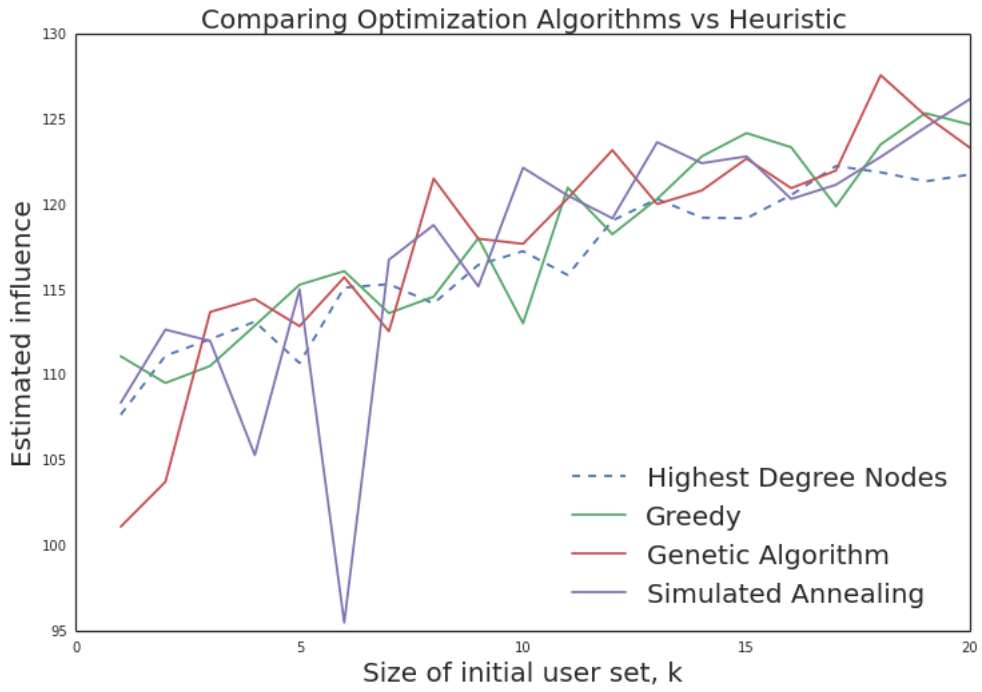


Figure 3: Standard deviations of the simulated f with respect to running time



(Figure is not drawn to scale.)

Figure 4: Estimated influence of high-degree heuristic, the greedy algorithm, simulated annealing, and genetic algorithm, on the Yelp user graph under the Independent Cascade Model with the Beta distribution defined above. ($|V| = 182, |E| = 3402, p_{uv}^2 \sim Beta(r_{uv}, r_v)$)

Lastly, using $N = 30$, we then run a simulation from $k = 1$ to $k = 20$, comparing the estimated influence from the output of the greedy algorithm, high-out-degree heuristic, simulated annealing, and genetic algorithm. **Figure 4** shows the influence performance of these algorithms on the North Carolina subgraph. Using heuristic as a baseline and comparing estimated influence for $k = 1...20$, we observe that the greedy algorithm outperforms heuristic 12 out of 20 times; genetic algorithm outperforms heuristic 15 out of 20 times; and simulated annealing outperforms heuristic 14 out of 20 times.

5 Discussion

We find that the genetic algorithm yields the best solution in our simulation. When k is very small, e.g. k<3, the heuristic performs better than the genetic algorithm since the optimization problem becomes quite trivial for our relatively small graph. In line with results in the literature [6] [8] [9], we expect high-degree heuristic to perform worse for a larger input graph and larger initial set size k, because it fails to take into account high-out-degree nodes having similar sets of neighbors.

We are also interested in the efficiency of the algorithms. For running time analysis, we use the following variables: p, a constant parameter of our stochastic algorithms, N, number of times we sample the influence function, k, number of users to start an independent cascade, n, total number of users in a particular graph, t, the number of iterations before we terminate an independent cascade.

Using these variable definitions, and a non-trivial derivation, we find the run time complexity of both the genetic algorithm and simulated annealing is $O(pNtkn)$, or just $O(Ntkn)$ because p is a constant. The run time complexity of the greedy algorithm is $O(Ntk^2n^2)$. Asymptotically, simulated annealing and genetic algorithm runs $O(nk)$ times faster than the greedy algorithm. This was borne out in simulation when k above 15 caused the greedy algorithm's run time to grow quickly from tens of minutes to a few hours. Meanwhile, the stochastic optimization approaches maintained stationary run times in the low tens of minutes range, even as k grew. It will be interesting to compare the performance of the fastest greedy algorithm on a large input graph with our stochastic algorithms.

6 Conclusions

In this paper, we provide an empirical study of applying the influence maximization problem to a real-world social network, Yelp's user network. We propose an independent cascade model to measure the influence of

initializing users to post reviews on Yelp's user network in North Carolina; we also design stochastic algorithms that are competitive with the general greedy algorithm and outperforms the high-degree heuristic in terms of the estimated influence. As far as we are aware, no study of stochastic algorithms has been done on the influence maximization problem.

More importantly, our study concludes that stochastic algorithms are asymptotically faster algorithm than the general algorithm, which could provide a more scalable solution on a large graph. It is important to note the stochastic algorithms are independent of the submodularity of the problem and thus applicable to a broader range of similar problems.

There are several future directions to pursue. Given the prolific data that Yelp offers, we plan to refine our independent cascade model by incorporating more relevant information into the original graph such as influence of "Elite Yelper" status, influence of "friends" network, and user-based or business-based similarity. Another improvement is to fine-tune the stochastic algorithms including designing a parallel tempering variant as an alternative to simulated annealing. Lastly, we will work on scaling up our project to experiment on the entire Yelp user network with 366 thousand users and 2.9 million social edges [5].

References

[1] URL: `https://www.quantcast.com/yelp.com?country=US`.

[2] URL: `http://expandedramblings.com/index.php/yelp-statistics/`.

[3] URL: `http://www.nielsen.com/us/en/insights/news/2013/the-reviews-are-in--yelp-users-are-four-star-consumers.html`.

[4] URL: `http://yelp.typepad.com/.a/6a00d83452b44469e20192ab90efab970d-pi`.

[5] URL: `http://www.yelp.com/dataset_challenge`.

[6] J. M. Kleinberg D. Kemp and E. Tardos. "Maximizing the spread of influence through a social network". In: *Proceedings of the 9th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. 2003, pp. 137–146.

[7] C.Faloutsos J.Leskovec A.Krause et al. *Cost-effective outbreak detection in networks*. 2007.

[8] David Parkes and Sven Sueken. *Economics and Computation*. Cambridge University Press, 2016. Chap. 24. Information, Games and Network, pp. 611–631.

[9] Y. Wang W. Chen and S. Yang. "Efficient influence maximization in social networks". In: *KDD* (2009).

[10] S. Wasserman and K. Faust. *Social Network Analysis*. Cambridge University Press, 1994.