Neighbor-Hop Mutation for Genetic Algorithm in Influence Maximization

Aviral Chawla
achawla1@uvm.edu
University of Vermont
Burlington, Vermont, USA

Nick Cheney
ncheney@uvm.edu
University of Vermont
Burlington, Vermont, USA

ABSTRACT

The spread of contagions has been a central component of research on social networks. An abundance of literature shows that few nodes in the network contribute significantly to the final magnitude of the outbreak. The problem of finding the set of k most influential nodes is called the Influence Maximization Problem. This problem is NP-Hard. Considering the usefulness and difficulty of this problem, there has been a lot of scholarly efforts to solve it. In this paper, we propose a metaheuristic approach for this problem. Specifically, we enhance the simple Genetic Algorithm with a Neighbor-Hop Mutation and demonstrate that it outperforms the Greedy algorithm in all test cases. Furthermore, our proposed algorithm converges in a significantly shorter time, taking only one to two hundred generations compared to the one to four thousand generations required by the simple Genetic Algorithm.

KEYWORDS

social networks, influence maximization, viral marketing, random walk, genetic algorithm, neighbor-hop

ACM Reference Format:
Aviral Chawla and Nick Cheney. 2023. Neighbor-Hop Mutation for Genetic Algorithm in Influence Maximization. In *Genetic and Evolutionary Computation Conference Companion (GECCO '23 Companion), July 15–19, 2023, Lisbon, Portugal.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3583133.3590755

1 INTRODUCTION

The study of networks play a crucial role for understanding complex social behavior. One of the main challenges in this field is identifying which nodes in a network are important. Researchers have developed numerous methods of measuring this centrality of a node. This paper specifically focuses on a node's importance in terms of its overall "influence" on the network.

Influence is the measure of the magnitude of spread of infection, such as an idea or a virus, by a given node to other network nodes. There are many ways to model contagion behavior in a network, some of which are discussed in Sec. 3. Given any cascading model, the problem of Influence Maximization involves choosing k nodes from a network such that the resulting contagion, propagated through these nodes, will generate the largest outbreak possible. This problem is intrinsically tied to the complex topology of each network. Consequently, the problem was proven NP-Hard [8].

The Influence Maximization problem helps us understand dynamical processes on networks. It also offers us interesting insights into the emergent behavior of complex systems. In addition to its academic value, Influence Maximization has can potentially be productive in the field of epidemiology and marketing. [6]

1.1 Contributions

We propose an evolutionary algorithm approach to solve the influence maximization problem (IMP). We work with a Genetic Algorithm (GA) initially introduced for the IMP by [3]. But we substitute the random mutation with a *Neighbor-Hop mutation*. Our results demonstrate that the Genetic Algorithm with Neighbor-Hop (NHGA) outperforms both GA and the Greedy algorithm for the Independent Cascade (IC) and Weighted Cascade (WC) models on several tested social networks for k = 10, 20, 30, and 40. These results indicate that, to the best of our knowledge, NHGA is the first algorithm that consistently outperforms the Greedy algorithm and with fast speeds.

2 RELATED WORK

The foundational paper for the Influence Maximization by [8] serves as the reference for works and vocabulary around the problem. In it, Kempe et al. define different methods to calculate spread such as General Cascade and Linear Threshold. They also prove that the problem is NP-Hard and provides a general Greedy method to solve the problem.

Since the influence of a seed set is monotonic and submodular, the greedy algorithm demonstrates a (1 - $\frac{1}{e}$) approximation to the optimal solution. This approximation guarantee makes Greedy algorithm the gold standard. However, the Greedy algorithm is slow. The time complexity of the Greedy algorithm is $O(knRm)$ [4], where n is the number of nodes, m is the number of edges, and R is the number of rounds of simulation. For a fixed edge density, the algorithm scales quadratically with the size of the graph. As a result, many algorithms have been proposed to improve the efficiency of the Greedy algorithm without compromising the approximation guarantees- such as CELF[11], DGS [10], SMG [7], etc.

Alongside Greedy-like approaches, centrality measures such as Degree, Eigenvector, and Betweenness have been explored for identifying influential nodes. However, these measures do not consider the dynamics on the network, and therefore may not accurately capture the properties of the diffusion process. [2] Other heuristic-based approaches, such as DD, MixedGreedy, [5] and TIM, [14] have been proposed to solve the problem. Compared to Greedy-based solutions, these heuristics are typically faster, but do not come with

GECCO '23 Companion, July 15–19, 2023, Lisbon, Portugal

ACM ISBN 979-8-4007-0120-7/23/07.
https://doi.org/10.1145/3583133.3590755

any theoretical guarantees, except for TIM. And as far as we know, none of them universally outperform the greedy-based solutions.

Metaheuristic-based approaches show true promise in this field. [13] [15][1] Evolutionary algorithms provide us with creative ways to explore the problem space for superior results. They also produce not just one, but a variety of good results at a time.

Our research focuses on the GA proposed by [3]. This algorithm holds two main advantages: firstly, it has been demonstrated to perform on par with, and occasionally better than, the Greedy algorithm. Secondly, its simplicity makes it highly accessible and provides valuable insights into the mechanics of influence maximization. Our aim is to enhance this algorithm by introducing a minor modification that significantly improves its performance. The resulting algorithm retains the simplicity and accessibility of the original GA while providing superior results.

3 METHODOLOGY

3.1 Independent Cascade

The Independent Cascade (IC) model is a commonly used model for diffusion in a network. In this model, the transmission of information or disease from one node to another occurs with a fixed probability, β. This means that if a node is infectious, the probability of infecting a connected node is solely dependent on this fixed probability, and not on any other metric.

To measure the final spread, we simulate the probabilistic process in discrete time steps. The initial set of nodes to be measured is denoted as I_0 and has a size of k. Starting with an infected set I, we infect their neighbors u with a probability $p(v,u) = \beta$, where $v \in I$. We then move all the infected nodes v from I to R, where they can no longer be re-infected or infect others. All the newly infected neighbors are moved to I. This process keeps unfolding till I is empty. The size of set R when I is empty, denoted $\bar{R}$, is the final spread.

Since the IC model is stochastic , the calculation for the spread may differ with each simulation. To account for this, we run each cascade 100 times and average the results to obtain a more stable estimate of the influence spread.

3.2 Weighted Cascade

The Weighted Cascade (WC) model is another method we employ to measure the spread in the network. Similar to the IC model, we use a stochastic process to simulate the spread. However, the probability of infection differs in this model: $p(v,u) = \frac{1}{in-degree(u)}$. In contrast to the IC model, where the infection probability is fixed. As with the IC model, we run 100 simulations and average the results to estimate the spread.

3.3 Greedy Algorithm

The general Greedy algorithm maximizes the influence by iteratively selecting the node with the highest potential for additional influence until the set is full. First, we calculate $\bar{R}(\{v\})$, $\forall\, v \in G$. From this, we select the node, n_1, with the highest measured influence. Next, we calculate $\bar{R}(\{n_1, v\})$, $\forall\, v \in G - \{n_1\}$, and then pick the set with the highest influence, $\{n_1, n_2\}$. We repeat this process till our set is of size k.

3.4 Genetic Algorithm

We reproduce the Genetic Algorithm with Tournament Selection and Elitism introduced in the paper by [3]. The algorithm has a genome of k unique indices of nodes $v \in G$. The two fitness functions used are: 1) Spread with Independent Cascade, $\bar{R}_{IC}$, and 2) Spread with Weighted Cascade , $\bar{R}_{WC}$. Final fitness measure averages the spread using a given type of cascade over 100 runs. Bucur et al. use random mutation operator to mutate a genome. A random sample of nodes from the genome is replaced by a random sample from the graph. One-point crossover is used.

3.5 Neighbor-Hop Mutation

Our proposal suggests mutating the genome by replacing one of its nodes with a random neighbor of that node. Neighbors include both in and out-neighbors of that node. This mutation can be likened to a random walk on the graph. The concept of random walks on a network has previously been used to improve upon the efficiency and performance for influence maximization.[9] [13] The neighbor-hop mutation is not a traditional random walk as it consists of a group of nodes taking separate steps in random order rather than a single trajectory from node to node. Even though it is not a traditional random-walk, we suspect that it still exposes more influential nodes in the network by taking advantage of the high assortativity in the social network. The mutation is simple and does not add to the runtime of the GA in any substantive way.

4 EXPERIMENTS

4.1 Networks

We ran experiments on 4 social networks. These social networks are - soc-Slashdot0811, wiki-Vote, soc-Epinions, and email-Enron. The network data came from SNAP (Stanford Large Network Dataset Collection). [12] All networks are constructed as directed unipartite graphs in our experiments. The primary motivation behind using directed graphs is to be able to use the Weighted Cascade model.

4.2 Parameters

The genetic algorithm with Neighbor-Hop mutation was run with 4 networks listed above. It ran under IC and WC for k = 10, 20, 30, 40 with β = 0.01. Each condition set was run 10 times. For example, wiki-Vote network with IC model and k = 20 was run 10 times, and the fitness calculation at each step in the optimization process was run 100 times.

NHGA features were the same as the ones used by [3]. We used 2 elites, 100 individuals in a population, single mutation, one-point crossover, and tournament size 5. The optimization process for each run was 200 generations long, in contrast to 100×k generations used by [3].Reduced number of generations was used to save on computation, and as mentioned in section 5, NHGA converges very quickly. For comparison, GA was also run on the 4 networks with the same features and condition set, repeated 10 times, for 200 generations. It is important to note that the results of the GA here are not reflective of the true potential of GA as demonstrated in the original paper, since they are meant to be run for many more generations.

5 RESULTS AND DISCUSSION

5.1 Comparative Analysis

Fitness	k	GA	Greedy	NHGA	δ
Enron	10	443.809	425.32	496.664	16.77%
	20	510.094	462.47	574.109	24.14%
	30	545.876	486.53	631.180	29.7%
	40	589.148	505.55	670.460	32.62%
Wiki-Vote	10	95.466	104.6	111.738	6.8%
	20	147.504	153.83	166.032	7.93%
	30	187.538	199.27	202.623	1.68%
	40	216.538	232.24	238.008	2.5%
Epinion	10	274.983	372.35	418.582	12.4%
	20	355.199	457.86	506.770	10.7%
	30	387.590	477.065	566.013	18.644%
	40	425.907	496.11	613.370	23.6%
Slashdot	10	820.620	805.08	928.891	15.37%
	20	912.166	860.79	1024.566	19.02%
	30	964.184	895.12	1097.183	22.57%
	40	1014.612	939.05	1152.482	22.72%

Table 1: Average final fitness values for each of the recorded runs under Independent Cascade. δ measures the difference in percentage between NHGA and Greedy

Figure 1, Table 1, and Table 2 show the overall results for GA and NHGA against greedy. In the figure and tables, each data point for GA and NHGA is represented as the average of 10 runs under the specific parameters. In general, NHGA performs better under IC than under WC. though it has a marginal to a significant advantage over greedy for both. NHGA also performs better on larger networks compared to smaller ones. But at the same time, GA also outperforms greedy on larger networks. However, NHGA outperforms by a significantly higher margin.

5.2 Generations

Figures 2 and 3 shows fitness over generations. This pattern is identical for all the runs. As we can see, NHGA converges much sooner than GA. Ideally, GA needs 100×k generations to fully converge to the greedy, but NHGA does so in 200 or less. In addition, the confidence interval of NHGA is much tighter than that of GA at any given generation. These observations indicate that NHGA quickly and consistently converges on a local optimum.

5.3 Future Work

The simple augmentation of Neighbor-Hop mutation to the Genetic Algorithm opens new possibilities for the use of Evolutionary Algorithms for the Influence Maximization problem. Our paper demonstrates that NHGA outperforms greedy on all test cases, consistently, with varying margins. To the best of our knowledge, this is the first algorithm to do so. In addition, this algorithm is agnostic to network topology and significantly faster than simple GA. The next step would be to expand on the current results by including more networks and parameter sets. It will also be interesting to introduce diversity maintenance methods such as Novelty Search to see if it finds better solutions in the long run.

Fitness	k	GA	Greedy	NHGA	δ
Enron	10	4203.844	5646.9	6081.67	7.70%
	20	5357.319	8080.91	8258.687	2.2%
	30	6530.653	9423.93	9548.530	1.32%
	40	6997.837	10129.02	10607.579	4.73%
Wiki-Vote	10	265.344	296.08	316.08	6.8%
	20	396.415	424.177	440.221	3.78%
	30	474.491	518.87	524.901	1.16%
	40	552.972	584.73	598.13	2.29%
Epinion	10	3315.502	6385.25	6555.511	2.67%
	20	4342.508	8382.91	8496.289	1.35%
	30	5454.45	9180.61	9721.624	5.89%
	40	5807.565	10033.18	10526.514	4.9%
Slashdot	10	3098.257	5536.495	6307.881	13.93%
	20	4327.521	6404.59	7698.973	20.21%
	30	4732.469	7114.88	8682.603	22.03%
	40	5344.345	7307.73	9353.305	21.87%

Table 2: Average final fitness values for each of the recorded runs under Weighted Cascade. δ measures the difference in percentage between NHGA and Greedy

ACKNOWLEDGMENTS

Computations were performed on the Vermont Advanced Computing Core (VACC) supported in part by NSF award No. 1827314. Special thanks to William Thompson, Kameron Bielawski, and Samuel Rosenblatt for their support.

REFERENCES

[1] Elaf Adel Abbas and Huda Naji Nawaf. 2021. Influence Maximization based on a Non-dominated Sorting Genetic Algorithm. *Karbala International Journal of Modern Science* 7, 2 (June 2021). https://doi.org/10.33640/2405-609X.2891

[2] Zhong-Kui Bao, Chuang Ma, Bing-Bing Xiang, and Hai-Feng Zhang. 2017. Identification of influential nodes in complex networks: Method from spreading probability viewpoint. *Physica A: Statistical Mechanics and its Applications* 468 (Feb. 2017), 391–397. https://doi.org/10.1016/j.physa.2016.10.086

[3] Doina Bucur and Giovanni Iacca. 2016. Influence Maximization in Social Networks with Genetic Algorithms. In *Applications of Evolutionary Computation*, Giovanni Squillero and Paolo Burelli (Eds.). Vol. 9597. Springer International Publishing, Cham, 379–392. https://doi.org/10.1007/978-3-319-31204-0_25 Series Title: Lecture Notes in Computer Science.

[4] Wei Chen, Yajun Wang, and Siyu Yang. 2009. Efficient influence maximization in social networks. *Proceedings of the 15th ACM SIGKDD international conference on*

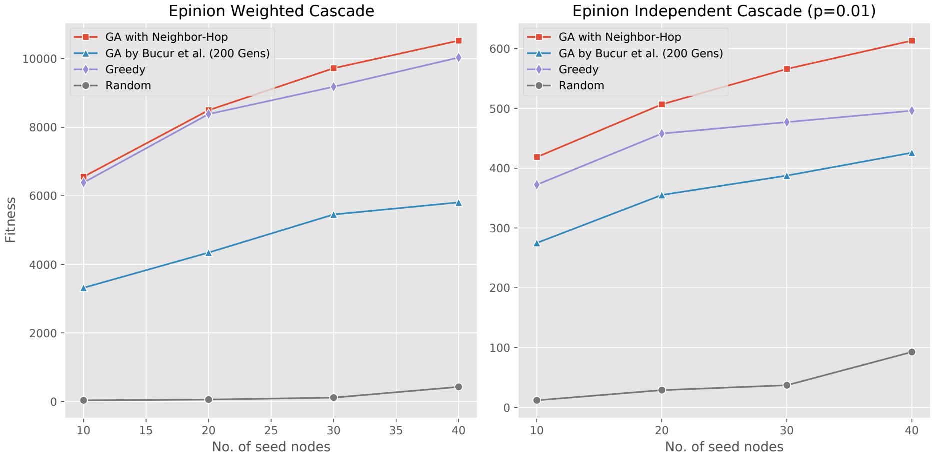

Figure 1: Comparative performance results for the three algorithms as tested on Epinion network.

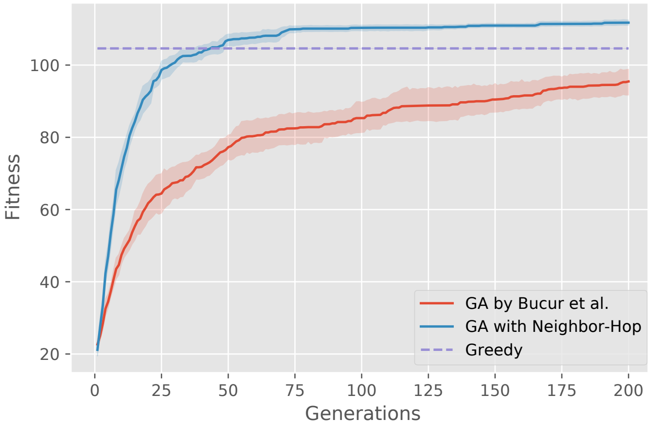

Figure 2: Fitness over generations on Wiki-Vote network for k = 10 under Independent Cascade Model.

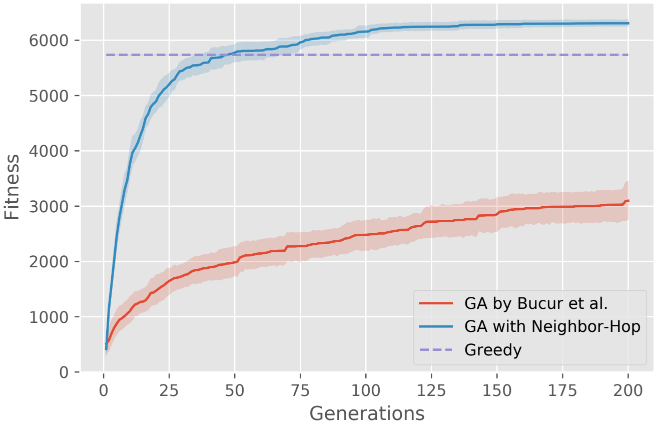

Figure 3: Fitness over generations on Slashdot network for k = 10 under Weighted Cascade Model.

Knowledge discovery and data mining - KDD '09 (2009). https://doi.org/10.1145/1557019.1557047

[5] Wei Chen, Yajun Wang, and Siyu Yang. 2009. Efficient influence maximization in social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, Paris France, 199–208. https://doi.org/10.1145/1557019.1557047

[6] Judith A Chevalier and Dina Mayzlin. 2006. The Effect of Word of Mouth on Sales: Online Book Reviews. *JOURNAL OF MARKETING RESEARCH* (2006).

[7] Mehdi Heidari, Masoud Asadpour, and Hesham Faili. 2015. SMG: Fast scalable greedy algorithm for influence maximization in social networks. *Physica A: Statistical Mechanics and its Applications* 420 (Feb. 2015), 124–133. https://doi.org/10.1016/j.physa.2014.10.088

[8] David Kempe, Jon Kleinberg, and Éva Tardos. 2003. Maximizing the spread of influence through a social network. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, Washington, D.C., 137–146. https://doi.org/10.1145/956750.956769

[9] Seungkeol Kim, Dongeun Kim, Jinoh Oh, Jeong-Hyon Hwang, Wook-Shin Han, Wei Chen, and Hwanjo Yu. 2017. Scalable and parallelizable influence maximization with Random Walk Ranking and Rank Merge Pruning. *Information Sciences* 415-416 (Nov. 2017), 171–189. https://doi.org/10.1016/j.ins.2017.06.018

[10] Suman Kundu and Sankar K. Pal. 2015. Deprecation based greedy strategy for target set selection in large scale social networks. *Information Sciences* 316 (Sept. 2015), 107–122. https://doi.org/10.1016/j.ins.2015.04.024

[11] Jure Leskovec, Andreas Krause, Carlos Guestrin, Christos Faloutsos, Jeanne VanBriesen, and Natalie Glance. 2007. Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, San Jose California USA, 420–429. https://doi.org/10.1145/1281192.1281239

[12] Jure Leskovec and Andrej Krevl. 2014. SNAP Datasets: Stanford Large Network Dataset Collection. http://snap.stanford.edu/data.

[13] Jianxin Tang, Ruisheng Zhang, Ping Wang, Zhili Zhao, Li Fan, and Xin Liu. 2020. A discrete shuffled frog-leaping algorithm to identify influential nodes for influence maximization in social networks. *Knowledge-Based Systems* 187 (Jan. 2020), 104833. https://doi.org/10.1016/j.knosys.2019.07.004

[14] Youze Tang, Xiaokui Xiao, and Yanchen Shi. 2014. Influence maximization: near-optimal time complexity meets practical efficiency. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*. ACM, Snowbird Utah USA, 75–86. https://doi.org/10.1145/2588555.2593670

[15] Chun-Wei Tsai, Yo-Chung Yang, and Ming-Chao Chiang. 2015. A Genetic New-Greedy Algorithm for Influence Maximization in Social Network. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, Kowloon Tong, Hong Kong, 2549–2554. https://doi.org/10.1109/SMC.2015.446