Data & Knowledge Engineering 129 (2020) 101852

Contents lists available at ScienceDirect

Data & Knowledge Engineering

journal homepage: www.elsevier.com/locate/datak

An analytical model for information gathering and propagation in social networks using random graphs

Samant Saurabh [a,*], Sanjay Madria [b], Anirban Mondal [c], Ashok Singh Sairam [d], Saurabh Mishra [e]

[a] Department of Information Technology and Decision Sciences, Indian Institute of Management, Bodh Gaya, Bihar, 824234, India
[b] Department of Computer Science, Missouri University of Science and Technology, Rolla, MO, 65409, USA
[c] Department of Computer Science, Ashoka University, Sonipat, Harayana, 131029, India
[d] Department of Mathematics, Indian Institute of Technology, Guwahati, Assam, 781039, India
[e] Department of Computer Science, Shiv Nadar University, Uttar Pradesh, 201314, India

ARTICLE INFO

Keywords:
Social networks
Data models
Information gathering
Data sharing
Business intelligence
Node discovery in graphs

ABSTRACT

In this paper, we propose an analytical model for information gathering and propagation in social networks using random sampling. We represent the social network using the Erdos–Renyi model of the random graph. When a given node is selected in the social network, information about itself and all of its neighbors are obtained and these nodes are considered to be discovered. We provide an analytical solution for the expected number of nodes that are discovered as a function of the number of nodes randomly sampled in the graph. We use the concepts of combinatorics, probability, and inclusion–exclusion principle for computing the number of discovered nodes. This is a computationally-intensive problem with combinatorial complexity. This model is useful when crawling and mining of the social network graph is prohibited. Our work finds application in several important real-world decision support scenarios such as survey sample selection, construction of public directory, and crowdsourced databases using social networks, targeted advertising, and recommendation systems. It can also be used for finding a randomized dominating set of a graph that finds applications in computer networks, document summarization, and biological networks. We have evaluated the performance both analytically as well as by means of simulation, and the results are comparable. The results have an accuracy of around 96% for random graphs and above 87% for the power-law graphs.

1. Introduction

Information gathering and information propagation [1] are two of the major activities that occur in any social network. Information gathering can be conducted through various methods like surveys, observation, and examination to collect relevant data from the participants of a given study. In online social networks, information gathering can also be done via the third-party app downloads and installation. When we install apps on our mobile or install third-party apps on Google, Facebook, and Twitter accounts, we are asked for permission to let the apps access our contact information like phone numbers and email ids. True-Caller [2,3] was able to create one of the largest public phone-directory by using the crowd-sourced data from a social network of our mobile using the aforementioned strategy.

* Corresponding author.Samant Saurabh
E-mail addresses: samant@iimbg.ac.in (S. Saurabh), madrias@mst.edu (S. Madria), anirban.mondal@ashoka.edu.in (A. Mondal), ashok@iitg.ac.in (A.S. Sairam), sm609@snu.edu.in (S. Mishra).

https://doi.org/10.1016/j.datak.2020.101852
Received 16 July 2019; Received in revised form 26 June 2020; Accepted 3 September 2020
Available online 8 September 2020

The collected information can be used for business intelligence to investigate the characteristics, behaviors, and opinions of a group of people. It can also be used to obtain information regarding gender, religion, ethnicity, experience, opinions, income, social status, psychological, geographical, and physical characteristics of the people. The collected information can be used for effectively facilitating decision making in many important and diverse applications such as city management, transportation, governance, education, health care, tourism, and so on. In a social network, information can be gathered fast because, in many situations, information about the neighbors can also be obtained from the queried user.

On the other hand, information can also be propagated and shared in a given social network [4] in case of targeted advertising, promotions, notifications, and recommendation systems. One of the major advantages of a social network is that each individual is connected to many other individuals. Hence, by informing a user in the network, information can also be propagated to the neighbors. Thus, providing information to just a few members, we can potentially propagate it to a much larger audience due to the connectivity in the social network. In this paper, we propose an analytical model for information gathering and propagation in a social network graph using simple random sampling and node discovery.

When a node is selected, the node along with all of its neighbors are considered to be discovered. In the node discovery optimization problem, we determine the minimum number of nodes (in the graph) that needs to be selected such that all the nodes of the graph are discovered. In the context of social networks, this problem is often referred to as the influence maximization problem [5]. We could find the minimum number of nodes that need to be queried in order to discover the whole graph by mapping this problem to the dominating set problem [6].

As the number of sampled nodes increases, our model constructs the dominating set [7] of the graph. A vertex v in a graph G covers a vertex u if $v = u$ or if v is adjacent to u. A subset of vertices of G is a dominating set if it collectively covers all vertices in the graph [6]. A connected dominating set (CDS) [8] is a dominating set in which all the nodes are connected through some path.

They have many applications in the field of computer networks, wireless sensor networks, and mobile ad hoc networks [6,9]. At the data link layer, clustering using the dominating set can increase the spatial reuse of the spectrum, minimize collision, and provide QoS guarantees. CDS can create a virtual network backbone for packet routing and control. The efficiency of multicast/broadcast routing can also be increased by CDS because it reduces the forwarding of duplicate copies of the same messages in the network. Nodes in a wireless network are energy-constrained. CDS plays an important role in power management by increasing the number of nodes that may be allowed to be in sleep mode while still preserving the capability of the network to forward messages.

Multi-document text summarization is another important area of research where the minimum dominating set can be used [10]. A sentence graph is a graph where the sentences are represented by vertices and similar sentences are attached with edges. Given a sentence graph that is generated from a set of documents, the summary can be extracted by using the concept of the minimum dominating set of the graph.

The minimum dominating set (MDS) approach is rapidly emerging as a promising algorithmic method to analyze complex biological networks integrated with human body functioning and related disorders [11]. By identifying important genes (driver nodes), network controllability-based on the minimum dominating set (MDS) provides a new way to study these networks and discover the driver nodes and important genes.

Our work of finding the dominating set using the randomized sampling would be helpful in the above-mentioned scenarios because the complexity of this algorithm is low. The analytical formula proposed by us provides the expected number of nodes that we need to select in order to cover the whole graph. In distributed algorithms for finding the dominating set, our scheme would be even more helpful because nodes can select themselves to be part of the dominating set randomly with a given probability p. The value of p would be the ratio of the domination number provided by our analytical model and the total number of nodes in the graph. This scheme would requires less communication overhead and still would be able to construct a dominating set with a high probability.

The novelty of our work is that it can predict the number of nodes that are discovered as a function of the number of nodes randomly selected. Besides, it assists in predicting the domination number of a graph and in the construction of its dominating set. This formula gives accurate results for both the ER model and the random graph models for social networks. It could help the distributed algorithms in estimating the expected number of nodes required to construct the dominating set. This number can be taken as an input to devise a distributed algorithm with a lesser number of message exchanges and reduced time complexity.

For large graphs, finding this minimal set is computationally intensive with combinatorial complexity [12]. Moreover, finding this minimal set of nodes requires us to have the information about all the nodes of the graph as well as the structure of the entire graph. In practice, such data is typically not readily available due to the following reasons: (i) Finding this information might be generally prohibitively expensive (ii) Crawling or accessing the graph to extract information is generally restricted by the host company of the social network to prevent loss of competitive advantage (iii) It is generally time-consuming to mine such data.

Hence, in this work, we consider the randomized node discovery problem. In this method, the selection of nodes is performed randomly for discovering the graph. We aim at finding the average number of nodes that need to be selected randomly to discover all the nodes of the graph or a given percentage of it. In our work, we model the social network as the Erdos–Renyi (ER) model of the random graph [13]. In this model of the random graph *G(n,p)*, there are n nodes, and any two given nodes are connected with a probability p. We find the analytical solution for the expected number of nodes that are discovered in the *Erdos–Renyi* graph after randomly querying m nodes. We use the concepts of inclusion–exclusion principle [14], probability, and graph theory to compute this value.

The main contributions of this work are as follows:

- **Randomized Node Discovery Algorithm:** We propose a simple but efficient randomized node discovery scheme for uncovering information about all of the nodes in the graph.

- **Analytical Results for Node Discovery:** We model the graph using the *Erdos–Renyi* model of the random graph and provide an exact analytical solution for the expected number of nodes that are discovered after querying a given number of nodes, which are randomly selected.
- **No Requirement of Complete Graph:** In our work, we do not require the information about the complete graph. As previously discussed, this data is not available due to various reasons such as privacy and competitive advantage. Our model works even if we know the total number of nodes in the graph and the linkage probability among the nodes.
- **Analytical Results for Common Neighbors :** We render several important analytical results regarding the number of neighbors shared by the different number of nodes in a random graph.
- **High Accuracy of Theoretical Results:** We derive tight upper and lower bounds for the analytical solution and perform simulations to find the accuracy of the proposed analytical formula. We find that both our theoretical and simulation results match and the accuracy of our results is around 96% for the Erdos–Renyi random graph and around 87% for the power-law graph models.
- **Applicable for other graphs:** The analytical formula derived for the random graph can be used even for other graphs like the power-law graphs (social network graph, citation graph, Internet graph). We provide an expression to compute the linkage probability for a general graph, using which we can get the expected number of nodes that are discovered by randomly querying m nodes of the graph. The accuracy for the power-law graphs is also around 87%.

The remainder of the paper is organized as follows. In Section 2, we discuss related works. In Section 3, we formulate the problem statement and provide the randomized node discovery algorithm. In Section 4, we find an analytical expression for the expected number of nodes discovered as a function of the number of nodes queried. In Section 5, we provide the performance evaluation of our analytical formulas and provide the accuracy of our results using simulation results. In Section 6, we provide the managerial implications of our model. Finally, in Section 7, we conclude our work with directions for future work.

2. Related work

The node discovery problem is equivalent to the dominating set [6] in graph theory. The dominating set of a graph is the minimum number of nodes required to cover all the nodes of the graph. Finding the dominating set is an NP-hard problem. Several heuristic schemes exist in the literature to obtain an approximate solution to this problem.

A greedy heuristic algorithm in [7] finds an approximation for the dominating set of a graph. Initially, the dominating set D is empty. At each iteration of the algorithm, a vertex is added to the dominating set which would cover the maximum number of previously uncovered nodes. If d is the size of the dominating set found by the algorithm, the total complexity of the algorithm is $O(nd + m) \leq O(n^2)$. Here, n is the number of nodes in the graph and m is the number of edges.

The greedy algorithm may be tweaked into its randomized version [7,15]. In the randomized greedyheuristic algorithm, the probability that a vertex is chosen in a given iteration is proportional to the number of additional vertices it would cover. Hence, it is not always the one that covers the maximum number of vertices. The complexity of this algorithm is also $O(n^2)$.

The heuristic greedy vote [15,16] chooses the next node in a more sophisticated manner. In this algorithm, importance is given to the specific nodes that would get covered, and to whether or not they could be covered in some alternate way. To implement this idea, they use the concept of $vote(v) = \frac{1}{degree(v)}$ for each vertex v. The quantity $1 + degree(v)$ represents the number of vertices that could cover v (including itself). Then for each vertex v, define $weight(v)$ to be the sum of all the quantities $vote(w)$ such that w is a vertex which has not yet been covered, and which v could cover. A vertex v with maximum value of $weight(v)$ is chosen at each iteration. The complexity of this algorithm is also $O(n^2)$.

The node discovery or the dominating set problem plays a pivotal role in solving the influence maximization problem. It is the problem of finding a subset of nodes in a social network for maximizing the spread of influence. The work in [17] investigates the efficient influence maximization from two complementary directions. In particular, it uses a simple greedy algorithm wherein each iteration, it selects the node that maximizes the social influence. Moreover, it proposes a new degree discount heuristic that improves the influence spread. The work in [18] identifies the influencers in a social network using the Shapley value method. It uses referral behavior data to improve the selection of the top influencers. It also considered the influence of two-hop neighbors for improving the influencer selection.

The work in [19] provides a suspected infected (SI) model of diffusion for social media influence maximization. It proposed a multi-threading approach for the implementation of the algorithm for improving the rate of the influence spread. However, diffusion models typically assume that information would slowly percolate to the whole graph; however, this is extremely time-consuming.

The node discovery problem can also be mapped to the set cover problem [20] as shown in Section 3. Any solution to the set cover optimization problem would essentially also become a candidate for the solution of the node discovery problem. However, the set cover problem is NP-complete. The work in [21] proposes a heuristic-based algorithm to provide a $O(n^3)$ solution for addressing the approximate set cover problem. In the proposed solution, the number of sets of the set cover did not exceed the maximum number of sets covering an element time the optimal value. In [12], a threshold of $\ln(n)$ has been proposed for approximating the set cover solution; it also proved that $(1 - 2o(1))\ln$ is a threshold below which set cover cannot be *efficiently* approximated.

Besides finding the solution to the node discovery problem, it is also important to test our model on different social network graph models. In [22–24], the authors provide the random graph models for the social network with arbitrary degree distributions. Social networks have three distinct properties (i) They exhibit small-world effect which means that the average distance between any two seemingly far off nodes is small. In most of the social network graphs, the value is small i.e. between 6 to 10 [25]. (ii)

The social network graphs display the clustering effect. It implies that the probability of an edge between two nodes is much higher if they have one or more mutual acquaintances [25]. (iii) The last but the most important one is that the social network graphs have power-law degree distribution which means the nodes have highly skewed degree distribution [26]. The random graph model proposed by [22] can model graphs with any random degree distribution using the generating function $G(x)$. The model provides exact solutions for finding (i) the average degree of the nodes in the graph (ii) Finding if the graph consists of many small clusters of connected components or does it have a giant component (iii) the fraction S of the network that belongs to the giant component (iv) the average size of the connected components of the graph excluding the giant component (v) average path length among the nodes of the graph (vi) average number of nodes at a given distance from a node in the graph (vii) the value of the average clustering coefficient. It also shows that the average path length (distance) of the nodes in a social network graph increases logarithmically with the size N i.e. the number of nodes in the graph. However, on adding the feature of high clustering in the graph, the model becomes complex, and finding the exact solution becomes much harder.

We can utilize the analytical model given by [23] to compute the average degree of nodes and use it in our analytical model of node discovery which requires the average degree of the nodes of the graph as an input. Further, we might estimate if the graph will require a large number of nodes to discover the whole graph by observing the average size of the connected components and whether a giant component exists or not in the graph. This can help us in finding if the influence can spread fast or not in the network. By computing the average clustering coefficient, we can also get to estimate the number of nodes that would be discovered by randomly sampling any m given nodes.

The Erdos–Renyi model exhibits a Poisson degree distribution for large graphs. Even though the power-law degree distribution is not captured by this model, still the model is handy in many ways. One of the major advantages of this model is it is simple and exactly solvable and a lot of quite accurate analytical results can be derived using this model which is not possible otherwise.

No analytical solution has been provided for finding the expected number of nodes that are discovered as a function of the number of nodes that are randomly selected and queried. To our knowledge, this is the first work to find an analytical answer for the number of nodes that need to be queried to discover a given percentage of nodes in the graph.

3. Problem statement and context of the problem

In this paper, we consider the problem of node discovery in a social network graph where the nodes are selected at random. *A node is said to be discovered if it has been randomly selected or if it is the neighbor of an already selected node.* In the case of information gathering, a selected node is supposed to provide knowledge about itself and all its neighbors. In the case of information dissemination, when a node receives any information, we assume that it would propagate it to all its neighbors. We find the exact analytical solution for the expected number of nodes that are discovered when we have randomly selected m out of the total of n nodes in the graph. We derive an exact solution for the Erdos–Renyi model of the Random Graph $G(n, p)$. In a random graph $G(n, p)$, there are n nodes in the graph and there exists a probability p that two nodes are connected to each other.

In our work, we do not require information about the complete graph. It means we do not need the adjacency matrix or the adjacency list of the graph. This data is not available due to various reasons such as privacy and competitive advantage. In practice, such data is typically not readily available due to the following reasons: (i) Finding this information might be generally prohibitively expensive (ii) Crawling or accessing the graph to extract information is generally restricted by the host company of the social network to prevent loss of competitive advantage (iii) It is generally time-consuming to mine such data. Our model works even if we just know the total number of nodes in the graph and the linkage probability among the nodes. Mostly, this information about the total number of nodes in the graph and the average degree of nodes is published by the companies and the authorities because it does not reveal much information about the graph and hence is not a privacy concern. As the nodes of the graph are randomly sampled, there is no need for an initial seed node in our algorithm.

3.1. Context of the problem

In Fig. 1, we provide an example to illustrate the context of the problem. Let us assume that we want to collect the phone number of all the people in a social network. The social network graph for the community is provided in Fig. 1 where the phone number for each person along with their height in centimeter is provided. If we query person A, we come to know about the phone number of A herself and her neighbors B, E and J. We assume that the information is stored as $\langle name, value \rangle$ pair so that the list does not contain any repeated element. For example, if we query nodes *E, G, H, M, and A*, then we would discover the node-sets or the phone-number sets $S_E = \{E, A, C, F, G\}$, $S_G = \{G, F, K, L, M, H, E\}$, $S_H = \{H, G, C, D, I\}$, $S_M = \{M, G, L, N, O, I\}$ and $S_A = \{A, J, E, B\}$ respectively. Clearly the union of $S_E \cup S_G \cup S_H \cup S_M \cup S_A = G$ which means that we would collect the phone number of all the people in the graph. Here, S_i denotes the set of nodes that are discovered after querying node i and G denotes all the nodes of the graph. However, finding the minimum set of nodes is an NP-complete problem and becomes highly inefficient for the large graphs [5]. Moreover, many times this complete data is not readily available. Hence, we propose to randomly select nodes in the graph and query them to discover all the nodes of the graph and construct the graph ourselves. If we randomly select nodes *G, E, I, D, M, B, and A*, then also we cover the whole graph. However, the number of nodes that are queried is more than the optimal number. In most of the social networks, nodes know the information about their neighbors [27]. Hence, we do not need to collect information from the whole population. If we randomly sample any 7 nodes, with relatively high probability, it would be sufficient to cover all the nodes for the graph given in Fig. 1.

The node discovery problem can be mapped to the set cover problem. Let the set of all the nodes of the graph represent the universal set S i.e. the set that needs to be covered. We have $S = \{v_1, v_2, \dots, v_n\}$ where v_i are the nodes of the graph. Let us define

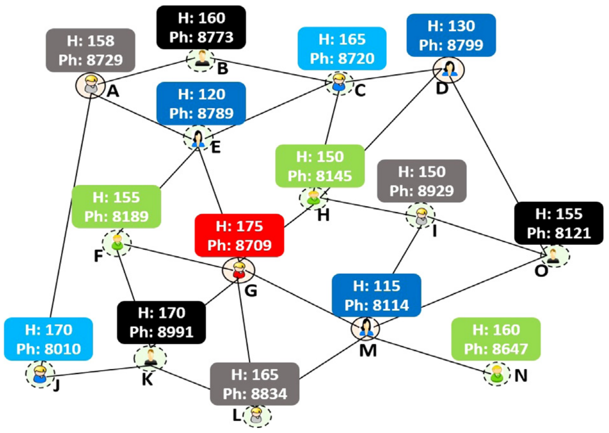


Fig. 1. Social graph showing height and phone number associated with each individual.

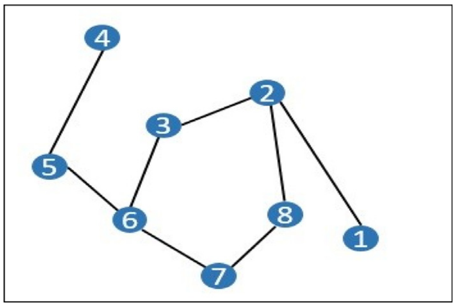


Fig. 2. A sample random graph.

n sets which can be numbered from s_1 to s_n. The set s_1 contains node v_1 and all its neighbors, set s_2 contains node v_2 and all its neighbor and similarly set s_n contains node v_n and all its neighbors. Then, the set cover problem is to find a subset from the sets $s_1, s_2, \dots, s_n$ whose union is the set S. The optimization problem for the set cover problem would be to find the minimum number of subsets r that can cover the set S.

3.2. Number of nodes to select using inclusion–exclusion principle

In this work, we employ a random selection of the nodes of the graph for node discovery.

Fig. 2 illustrates the node discovery problem. We have $s_1 = \{v_1, v_2\}$, $s_2 = \{v_2, v_3, v_8, v_1\}$, $s_3 = \{v_2, v_3, v_6\}$, $s_4 = \{v_4, v_5\}$, $s_5 = \{v_4, v_5, v_6\}$, $s_6 = \{v_3, v_5, v_6, v_7\}$, $s_7 = \{v_6, v_7, v_8\}$, and $s_8 = \{v_2, v_7, v_8\}$. As can be seen from Fig. 2, if nodes v_1 and v_4 are queried, then nodes v_1, v_2, v_4, v_5 are discovered while if node v_2 and v_4 are selected then $v_1, v_2, v_3, v_4, v_5, v_8$ are discovered.

We use the Inclusion–Exclusion principle to derive the number of nodes that are discovered after querying m out of the total n nodes in the graph. Let $N_{dis}(m)$ denote the number of nodes that are discovered after querying m nodes. Let us query node v_1, v_2 and v_3 respectively in that order for the graph given in Fig. 2. When we query v_1, then $N_{dis}(1) = 2$ as we have discovered v_1, v_2. When we query v_2, then $N_{dis}(2) \neq 2 + 3$ because $s_1 \cap s_2 \neq \phi$. Here $N_{dis}(2) = 4$. To find the correct answer, we need to apply the inclusion exclusion principle [28] for the two sets s_1 and s_2 as shown in Eq. (1).

$$\begin{aligned} |s_1 \cup s_2| &= |s_1| + |s_2| - |s_1 \cap s_2| \\ &= 2 + 3 - 1 = 4 \end{aligned} \tag{1}$$

Similarly, when node v_3 is queried, then $N_{dis}(3) \neq 2+3+3 = 8$ but it is 5. Again, we can find $N_{dis}(3)$ by the Inclusion–Exclusion principle as shown in Eq. (2).

$$\begin{aligned} |s_1 \cup s_2 \cup s_3| &= |s_1| + |s_2| + |s_3| - (|s_1 \cap s_2| + |s_2 \cap s_3| + |s_3 \cap s_1|) + |s_1 \cap s_2 \cap s_3| \\ &= 2 + 4 + 3 - 2 - 2 - 1 + 1 \\ &= 5 \end{aligned} \tag{2}$$

Similarly, if we want to find $N_{dis}(m)$ where $1 \leq m \leq n$, then $N_{dis}(m)$ would be given by the generalized Inclusion–Exclusion principle given by Eq. (3).

$$N_{dis}(n) = \left| \bigcup_{1 \leq i \leq n} s_i \right| = \sum_{1 \leq i_1 \leq n} \left| s_{i_1} \right| - \sum_{1 \leq i_1 \leq i_2 \leq n} \left| s_{i_1} \cap s_{i_2} \right| + \sum_{1 \leq i_1 \leq i_2 \leq i_3 \leq n} \left| s_{i_1} \cap s_{i_2} \cap s_{i_3} \right| - \cdots + (-1)^{n+1} \left| \bigcap_{i=1}^{n} s_i \right| . \tag{3}$$

3.3. Randomized algorithm for node discovery in $G(n,p)$

In this Section, we provide the randomized algorithm that we use for the node discovery in random graph $G(n,p)$. Algorithm 1 is basically the random sampling of nodes in a social network graph. The input to Algorithm 1 is the graph $G(n,p)$ and the number of nodes m to be randomly selected in the graph. The output is the set of the discovered nodes. In line 1, we create an instance of the random graph with n nodes and linkage probability p. In line 2, we initialize the set of discovered nodes and variable i. In the *while loop* from line 4 to 12, we randomly sample m nodes of the graph $G(n,p)$. In line 5, we select a random node v. In the next line, we find all the neighbors of v and store them in a set $Neigh$. The set $uniqueNodes$ is the set of nodes that are the neighbors of v and those that were not discovered till then. In the $forloop$ from line 8 to 10, we append the unique nodes found to the $discovered$ set. Finally, after the algorithm ends, we have the set of nodes that is discovered by randomly selecting m nodes from the graph $G(n,p)$.

Algorithm 1: Algorithm for Randomized Node Discovery

Input: Random Graph G(n,p), number of nodes m to be randomly selected

Output: set of nodes of G that is discovered

```
1: G = Randon_Graph(n,p);
2: discovered = { }
3: $i \leftarrow 1$
4: while ( i ≤ m) do
5:    v = G.selectRandomNode()
6:    Neigh = G.neighbors(v)
7:    uniqueNodes = [ $u \in Neigh \cap u \notin discovered$ ]
8:    for  n ∈ uniqueNodes do
9:       discovered.append(n(< $key, value$ >))
10:   end for
11:   $i \leftarrow i + 1$
12: end while
```

4. Mathematical analysis: Number of nodes discovered as function of nodes queried

In our work, we assume that the random graph has a total of n nodes and p is the probability of an edge existing between two nodes. We call p as the linkage probability. Eq. (4) shows the adjacency matrix for the random graph $G(n,p)$ where $A(i,j) = 1$ if an edge exists between the two nodes i and j and $A(i,j) = 0$ otherwise.

$$A = \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & \dots & 1 \\ 0 & 1 & 1 & 0 & 1 & \dots & 0 \\ 0 & 1 & 1 & 1 & 1 & \dots & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 1 & 0 & 1 & 1 & 0 & \dots & 1 \end{bmatrix} \tag{4}$$

In this section, we first provide some properties regarding the expected number of common neighbors being shared by two or more nodes in a random graph $G(n,p)$. Then we proceed to find the expected number of nodes discovered as a function of the number of nodes queried in the graph using the results that we have derived.

4.1. Important results for common neighbors in random graph $G(n,p)$

Lemma 1. *The expected number of nodes (neighbors) connected to a single node (degree of a node) is given by Eq. (5).*

$$AD_{node} = (n-1)p \tag{5}$$

Proof. From the adjacency

matrix shown in Eq. (4), we find that it would be equal to the number of 1's in a single row leaving out the column for the node itself. This value would be $\sum_{i=1}^{n-1} p = (n-1)p$.

Lemma 2. *The expected number of common neighbors, CN_2, shared by two given nodes apart from themselves in G(n,p) is given by Eq. (6).*

$$CN_2 = (n-2) * p^2 \tag{6}$$

Proof. Without the loss of generality, let us assume that the two given nodes are node 1 and node 2. Node 3 will be a common neighbor of both node 1 and node 2 with probability $p * p = p^2$ i.e. both A(1,3) and A(2,3) are equal to one. This means that the expected value of node 3 being a common neighbor of both node 1 and node 2 is p^2. By the linearity of expectation, the expected number of common neighbors for both node 1 and node 2 would be given by Eq. (7).

$$\begin{aligned} CN_2 &= P(A(1,3)=1) * P(A(2,3)=1) \\ &+ P(A(1,4)=1) * P(A(2,4)=1) \\ &+ \cdots + P(A(1,n)=1) * P(A(2,n)=1) \\ &= \sum_{i=3}^{n} p^2 \\ &= (n-2)p^2. \end{aligned} \tag{7}$$

Lemma 3. *Expected value of the number of common neighbors shared by k given nodes in $G(n,p)$ apart from themselves, CN_k, is given by Eq. (8).*

$$CN_k = (n-k) * p^k \qquad \textit{where } 2 \le k \le n \tag{8}$$

Proof. Without the loss of generality, let us assume that the k given nodes are node 1 to node k. Again, by the linearity of expectation, the expected number of common neighbors for nodes from node 1 to node k would be given by Eq. (9).

$$\begin{aligned} CN_k &= P(A(1,k+1)=1) * \cdots * P(A(k,k+1)=1) \\ &+ P(A(1,k+2)=1) * \cdots * P(A(k,k+2)=1) \\ &+ \cdots + P(A(1,n)=1) * \cdots * P(A(k,n)=1) \\ &= \sum_{i=k+1}^{n} p^k \\ &= (n-k)p^k. \end{aligned} \tag{9}$$

Corollary 1. *The expected value of the sum of the number of common neighbors, SCN_2, shared by any 2 nodes selected from any m given nodes in $G(n,p)$ is given by Eq. (10).*

$$SCN_2 = {}^{\mathrm{m}}C_2(n-m) * p^2 \tag{10}$$

where the common neighbors are outside the group of m nodes.

Here, $(n-m) * p^2$ is the expected value of the number of common neighbors of any pair of nodes selected from the m nodes. ${}^{\mathrm{m}}C_2$ is the number of possible combinations of the 2 nodes selected from the m given nodes.

Corollary 2. *The Expected value of the sum of the number of common neighbors, SCN_k, shared by any k nodes selected from m given nodes in $G(n,p)$ is given by Eq. (11).*

$$SCN_k = {}^{\mathrm{m}}C_{\mathrm{k}}(n-m) * p^k \tag{11}$$

where the common neighbors are outside the group of m nodes.

Here, $(n-m) * p^k$ is the expected value of the number of common neighbors of any k nodes selected from the m nodes. ${}^{\mathrm{m}}C_{\mathrm{k}}$ is the number of possible combinations of the k nodes selected from the m given nodes.

Corollary 3. *The expected value of the sum of the number of common neighbors shared by any k nodes selected from m given nodes in $G(n,p)$ is given by Eq. (12).*

$$SCN'_k = {}^{\mathrm{m}}C_{\mathrm{k}}(n-k) * p^k \tag{12}$$

where the common neighbors can be even from any (n-k) nodes in the graph.

Here, $(n-k) * p^k$ is the expected value of the number of common neighbors of any k nodes selected from any (n-k) nodes in the graph. ${}^{\mathrm{m}}C_{\mathrm{k}}$ is the number of possible combinations of the k nodes selected from the m given nodes.

We will use the results provided in Eqs. (5) to (12) to derive the expected number of nodes that are discovered on querying m nodes in $G(n,p)$.

4.2. Analytical expression for the number of nodes discovered

Based on the results obtained in Section 4.1, we want to find the number of nodes discovered when m out of the n nodes are queried. By using the inclusion–exclusion principle, we first find the number of nodes discovered by taking all the m nodes separately, one at a time. We would discover ${}^{\mathrm{m}}C_1(n-m)p$ number of nodes. However, they would be sharing common neighbors taken two at a time. We need to subtract these numbers based on the inclusion–exclusion principle. There are ${}^{\mathrm{m}}C_2$ ways of choosing 2 nodes among the m given nodes. The expected number of neighbors shared by any two nodes would be ${}^{\mathrm{m}}C_2(n-m)p^2$. Now, we need to add the number of common neighbors shared by nodes taken three at a time. This number would be ${}^{\mathrm{m}}C_3(n-m)p^3$ and so on.

For the m given nodes when we apply the inclusion–exclusion principle, it will generate a series with alternate positive and negative terms which can be obtained from the Eqs. (5) to (10). We define $f(m)$ as the number of nodes that are discovered after randomly selecting m nodes excluding the selected nodes. The series can be written as:

$$\begin{aligned} f(m) &= {}^{\mathrm{m}}C_1(n-m)p - {}^{\mathrm{m}}C_2(n-m)p^2 + {}^{\mathrm{m}}C_3(n-m)p^3 + - \cdots + (-1)^{(\mathrm{m}-1)\mathrm{m}}C_{\mathrm{m}}(n-m)p^{\mathrm{m}} \\ &= (n-m)[{}^{\mathrm{m}}C_1 p - {}^{\mathrm{m}}C_2 p^2 + \cdots + (-1)^{(\mathrm{m}-1)\mathrm{m}}C_{\mathrm{m}} p^{\mathrm{m}}] \\ &= (n-m)[{}^{\mathrm{m}}C_0 - {}^{\mathrm{m}}C_0 + {}^{\mathrm{m}}C_1 p - {}^{\mathrm{m}}C_2 p^2 + {}^{\mathrm{m}}C_3 p^3 - \cdots + (-1)^{(\mathrm{m}-1)\mathrm{m}}C_{\mathrm{m}} p^{\mathrm{m}}] \\ &= (n-m)[1 - ({}^{\mathrm{m}}C_0 - {}^{\mathrm{m}}C_1 p + {}^{\mathrm{m}}C_2 p^2 - {}^{\mathrm{m}}C_3 p^3 - \cdots + (-1)^{(\mathrm{m}+1)\mathrm{m}}C_{\mathrm{m}} p^{\mathrm{m}})] \\ &= (n-m)[1-(1-p)^m] \end{aligned} \tag{13}$$

Now whenever any node is queried, it will also be discovered. However, we have not taken it into account in Eq. (13). Hence, we need to add it to the count. As a result, the total number of nodes discovered, $F(m)$ would be

$$F(m) = (n-m)[1-(1-p)^{\mathrm{m}}] + m \tag{14}$$

To check the correctness of the formula, we verify the boundary conditions. When m = 0, F(0) = 0, which means that when we have not selected any node, the number of nodes discovered would be zero. When $m = n$, $F(n) = n$, this implies that when we have selected all the nodes of the graph, we would discover all the nodes of the graph. The case $p = 0$, $F(m) = m$ implies that when no nodes are connected in the graph then whichever nodes we select, they are the only nodes that are discovered. When $p = 1, F(m) = n$ for all values of m because each node is connected to all the other nodes in the graph. These boundary values for $F(m)$ are shown in Eq. (15).

$$F(m) = \begin{cases} 0 & \text{for} \quad m = 0 \\ n & \text{for} \quad m = n \\ m & \text{for} \quad p = 0 \\ n & \text{for} \quad p = 1 \end{cases} \tag{15}$$

4.3. Upper bound

To find the upper bound of F(m), when choosing common neighbors among any k given nodes in the counting process of $F(m)$, we allow neighbors to be chosen from all the $(n-k)$ nodes. We do not exclude the m nodes that we have already sampled. This allows for the double counting and hence gives us the upper bound instead of the exact count.

$$\begin{aligned} ub(m) &= {}^{\mathrm{m}}C_1(n-1)p - {}^{\mathrm{m}}C_2(n-2)p^2 + {}^{\mathrm{m}}C_3(n-3)p^3 - \cdots + (-1)^{(\mathrm{m}+1)\mathrm{m}}C_{\mathrm{m}}(n-m)p^{\mathrm{m}} + m \\ &= n[{}^{\mathrm{m}}C_1 p - {}^{\mathrm{m}}C_2 p^2 + {}^{\mathrm{m}}C_3 p^3 - \cdots + (-1)^{(\mathrm{m}+1)\mathrm{m}}C_{\mathrm{m}} p^{\mathrm{m}}] - [{}^{\mathrm{m}}C_1 p - 2{}^{\mathrm{m}}C_2 p^2 + \cdots + (-1)^{(\mathrm{m}+1)} m{}^{\mathrm{m}}C_{\mathrm{m}} p^{\mathrm{m}}] + m \\ &= n(1-(1-p)^m) - mp(1-p)^{(m-1)} + m \end{aligned} \tag{16}$$

$$ub(m) = \begin{cases} 0 & \text{for} \quad m = 0 \\ 2n & \text{for} \quad m = n \\ m & \text{for} \quad p = 0 \\ 2n & \text{for} \quad p = 1 \end{cases} \tag{17}$$

We can also find another upper bound by using the inclusion exclusion principle itself and by using just the first term.

$$\begin{aligned} F(m) &= |\bigcup_{i=1}^{m} A_i| \\ &\leq \sum_{i=1}^{m} |A_i| \\ &= {}^{\mathrm{m}}C_1(n-1)p \\ &= m(n-1)p \end{aligned} \tag{18}$$

However, the Upper Bound given by Eq. (16) is much tighter compared to the Upper Bound given by Eq. (18). This can be observed from the Results provided in Section 5. The lower bound for $F(m)$ is m. Whenever we select any node, at least that node would be discovered even if, it does not have any neighbor or any undiscovered node as its neighbor.

4.4. Performance on other graphs

In this section, we study how can we apply our analytical result for different kind of graphs. For a graph $G(n,m)$ where $n = |V|$ and $m = |E|$, we calculate the expected value of the average linkage probability $\hat{p}$. For a graph *G(n,m)*, let the degree of vertex v_i be $deg(v_i)$ and $\hat{p}_i$ be the linkage probability of node v_i, then the average linkage probability for all the nodes of the graph is given by Eq. (19) given below.

$$\begin{aligned} \hat{p} &= \frac{1}{n}[\hat{p}_1 + \hat{p}_2 + \hat{p}_3 + \cdots + \hat{p}_n] \\ &= \frac{1}{n}[\frac{deg(v_1)}{n} + \frac{deg(v_2)}{n} + \frac{deg(v_3)}{n} + \cdots + \frac{deg(v_n)}{n}] \\ &= \frac{1}{n^2}[deg(v_1) + deg(v_2) + deg(v_3) + \cdots + deg(v_n)] \\ &= \frac{2m}{n^2} \text{ (sum of degree of nodes in undirected graph)} \end{aligned} \tag{19}$$

Hence, using this value for $\hat{p}$, we can approximate any graph $G(n,m)$ to the random graph $G(n, \frac{2m}{n^2})$ using Eq. (19). In general, if the probability, $P(k)$ of a node having degree k is known, then $\hat{p}$ can also be computed from Eq. (20) given below.

$$\hat{p} = \frac{1}{n}\sum_{k=1}^{n} k * P(k) \tag{20}$$

where n is the total number of nodes and $P(k)$ is the degree distribution function for node of having degree k. For example, we know that for social network graph which follows power law distribution of $P(k) \propto k^{-\gamma}$ where γ is in between $(2,3)$ [29], if we know γ and n, then we can compute $\hat{p}$. Similarly, for a random graph $G(n,p)$ the probability that a node has degree k is given by $P(k) = \binom{n-1}{k}p^k(1-p)^{n-1-k}$.

4.5. Estimation of linkage probability when a complete graph is not known

As discussed in Section 1, in many real-life scenarios, information about the complete graph might not be known. In that case, we cannot compute the linkage probability using Eq. (19). However, we can estimate the linkage probability from the sampled nodes. Let there be a total of tot sampled nodes $sn_1, sn_2, sn_3, \dots, sn_{tot}$. Let the degree of the sampled nodes be $dsn_1, dsn_2, \dots, dsn_{tot}$ respectively. Then the estimated value $\hat{p}$ of the linkage probability p is given by Eq. (21) given below.

$$\hat{p} = \frac{1}{tot}\sum_{k=1}^{tot} \frac{dsn_k}{n} \tag{21}$$

The total number of nodes n in the social network is usually well known as this data is usually published by the owner.

5. Performance evaluation

In this section, we present the results that we obtained theoretically and by using simulation on random graphs. We find the mean percentage error of our analytical model for node discovery for various random graph models viz. (i) Erdos–Renyi (ii) power law cluster (iii) Barabasi–Albert (iv) Newman–Watts–Strogatz (v) Watts–Strogatz and (vi) dual Barabasi–Albert graphs. Next, we compare the number of nodes that are required to discover the whole graph using the experimental and theoretical model proposed in this paper with state of art approximation algorithms present in the literature.

We used Python networkX library to create a random graph and simulated the randomized node discovery using Algorithm 1. We analyze these results to study the trends that we obtain when we vary the number of nodes and the linkage probability of the random graph besides changing the graph models themselves.

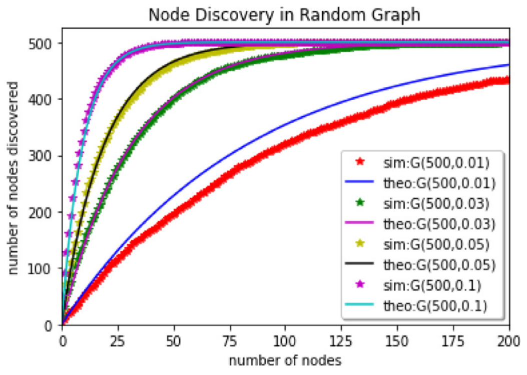


Fig. 3. Expected number of nodes discovered as a function of number of nodes queried for $G(500, p)$.

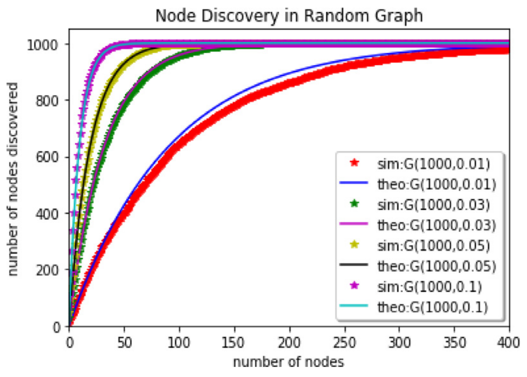


Fig. 4. Simulation results for the network.

Table 1
Node discover for G(500).

% of nodes discovered	p = 0.01	0.03	0.05	0.10
25%	27	9	5	2
50%	71	22	13	6
75%	140	44	27	13
100%	500	244	137	85

5.1. Simulation and comparative study

First, we analyze the performance of our model for the Erdos–Renyi model of random graphs. We simulate the graph $G(n, p)$ with value of n being n = {100, 500, 1000, 5000} nodes having different linkage-probability values of p = {0.01, 0.03, 0.05, 0.1} respectively. First, we keep the number of nodes as constant and vary the linkage probabilities to find the trend of the number of nodes discovered as a function of the number of nodes randomly selected for the query.

Fig. 3 provides the results for the number of nodes discovered as the number of nodes queried. From Fig. 3, we find that as the linkage probability increases from $p = 0.01$ to $p = 0.1$, the complete graph is discovered at a much faster rate for $G(500, p)$ for various values of p. As probability increases, the accuracy of our analytical expression increases. For $p = 0.01$, after query 200 nodes, only around 430 nodes are discovered. For $p = 0.1$, it just takes 85 nodes to discover the whole of 500 nodes.

Fig. 4, describes the number of nodes discovered as a function of number of nodes queried for $G(1000, p)$. As the linkage probability increases, a lesser percentage of nodes are required to discover the whole graph. It just takes around 72 nodes to discover the complete graph for G(1000,0.1). It takes around 195 nodes to discover G(1000,0.05). It takes 280 and 785 nodes to discover G(1000,0.03) and G(1000,0.01) respectively. The accuracy of our formula also improves as n increases.

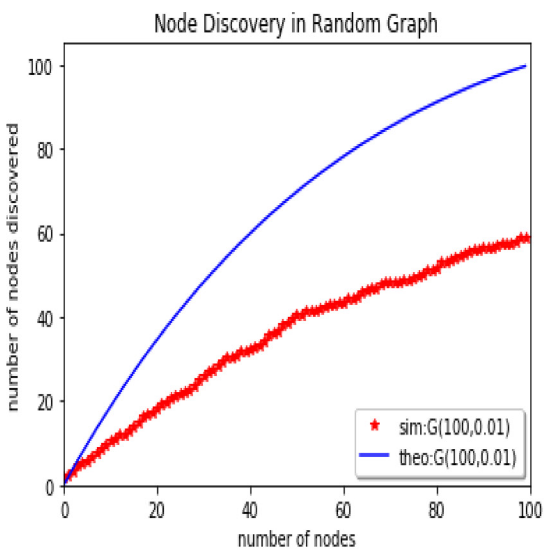
Table 1, shows that number of nodes that needs to be queried to get 25%, 50%, 75% and 100% of the nodes in the graph for G(500, p = 0.01, 0.03, 0.05, 0.1) respectively.

Table 2, shows the number of nodes that needs to be queried to get 25%, 50%, 75% and 100% of the nodes in the graph for G(1000, p = 0.01, 0.03, 0.05, 0.1) respectively.

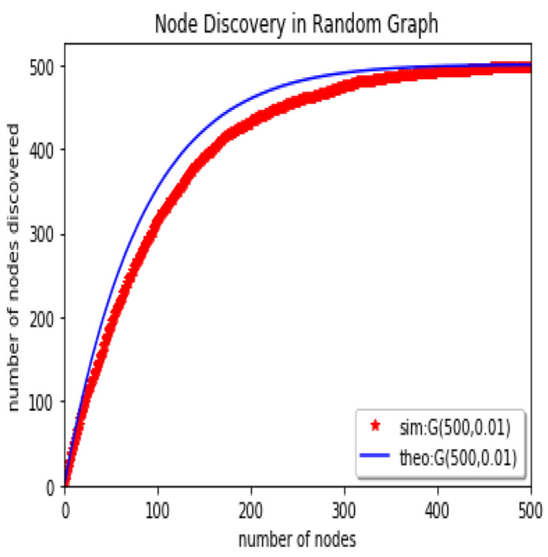
Again from Tables 1 and 2, we can observe that as the linkage probability increases, the number of nodes that needs to be queried decreases drastically.

Table 2
Node discover for G(1000).

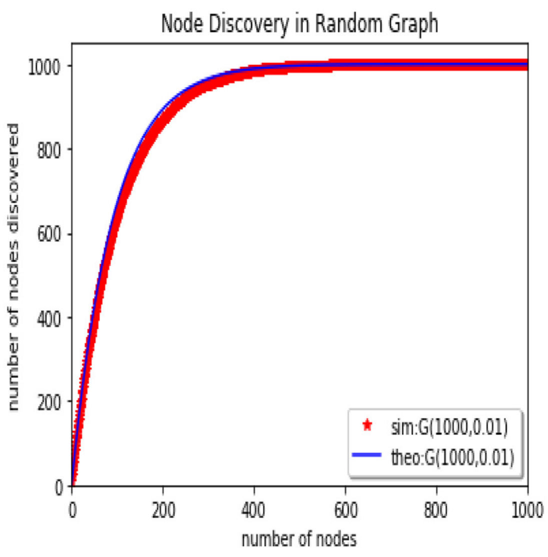
% of nodes discovered	p = 0.01	0.03	0.05	0.10
25%	27	9	5	2
50%	71	22	13	6
75%	140	44	27	13
100%	785	280	195	72



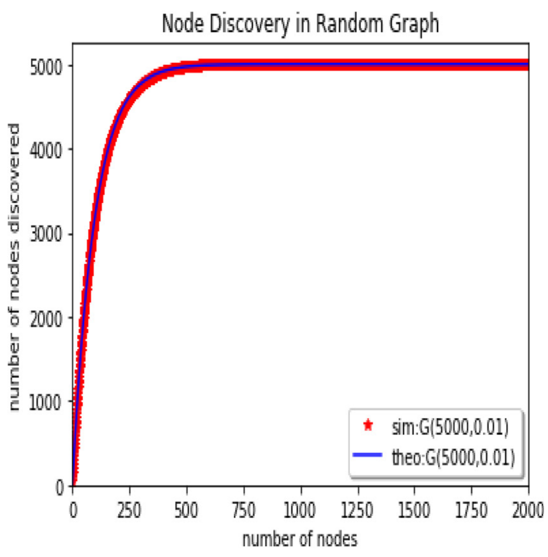
(a) Nodes discovered for **G(100,0.01)**



(b) Nodes discovered for **G(500,0.01)**



(c) Nodes discovered for **G(1000,0.01)**



(d) Nodes discovered for **G(5000,0.01)**

Fig. 5. Nodes discovered as a function of nodes queried.

In this experiment, we want to observe the effect on the number of nodes discovered as a function of a number of nodes queried when linkage probability is constant. In this experiment, we simulate the graph for 100, 500, 1000 and 5000 nodes using a probability of 0.01

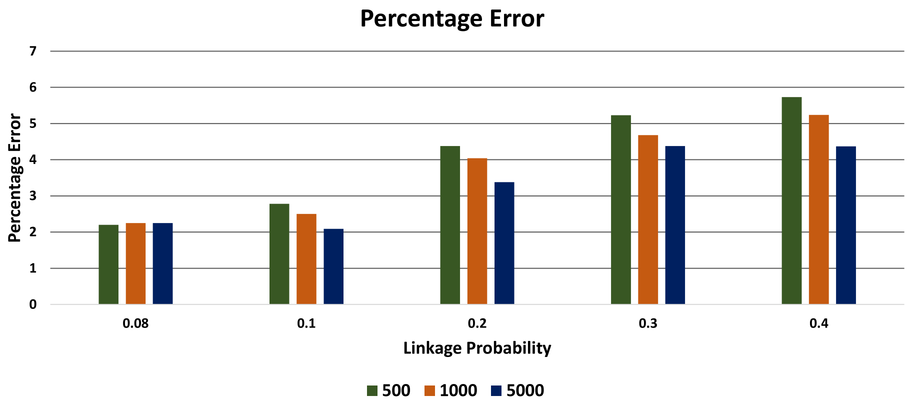


Fig. 6. Mean error between the theoretical and the simulation results.

Fig. 5, describes the number of nodes discovered as a function of nodes for (a) 100, (b) 500, (c) 1000, and (d) 5000 nodes for linkage probability of 0.01. It is observed that for large graphs i.e., graphs having a greater number of nodes our theoretical (from the formula) results have better accuracy, and the rate of node discovery is also improved.

From the results shown above, we observe that if the total number of nodes is greater, then a greater number of nodes would be discovered with the same percentage of the total number of nodes queried. We did the comparative study of four graphs of hundred, five hundred, thousand and five thousand nodes having the same linkage-probability of 0.01. We found that our formula is giving approximately the same result as that of the simulation of the random graph. The accuracy of our theoretical formula increases as the number of nodes in the graph increases for the same value of linkage probability.

In Fig. 6, we find the mean percentage error (MPE) in value for the simulation and theoretical results. The mean percentage error is calculated by Eq. (22).

$$MPE = \frac{1}{clip}\sum_{i=0}^{clip}\frac{\mid sim_i - theo_i \mid}{sim_i} * 100 \tag{22}$$

The value of $clip$ is given by Eq. (23)

$$clip = \max\{dom\_num_{exp}, dom\_num_{theo}\} \tag{23}$$

Note that the mean percentage error in Eq. (22) is computed for index $i = 0$ to $i = clip$, the value of clip is given using Eq. (23). The domination number used in Eq. (23) of a graph is the minimum number of nodes required to discover the complete graph. Both the experimental and the theoretical values converge to the total number of nodes in the graph (when they discover all the nodes in the graph). The complete discovery of the graph usually occurs at a much lower value compared to the total number of nodes present in the graph. We call these numbers $domNum_{exp}$ and $domNum_{theo}$ respectively for the experimental and theoretical values. These are the domination numbers found by the experimental and theoretical methods. After $\max\{domNum_{exp}, domNum_{theo}\}$, number of nodes discovered $F(m)$ remains constant at the value of the total number of nodes in the graph i.e. n. If we include $F(m)$ for $m > clip$, then it would drastically reduce the value of mean percentage error for that long stretch of equal values in the two arrays of experimental and theoretical values. Hence, we do not include these values in the mean percentage error computation.

It can be observed from Fig. 6, that as the linkage probability increases, the mean absolute error increases. The maximum error is around 5.7% for p = 0.4 and n = 500 error goes below two percent for the lower values of p. It is also observed that as the number of nodes in the graph increases for the same value of the linkage probability, the mean percentage error decreases in the range of 0.08 to 0.4 of linkage probability.

In the next result, we compare the performance of our upper and lower bounds in Fig. 7. We find that upper bound 2 is tighter than the upper bound 1. In fact,

$$theo \leq ub_2 \leq 2 * theo \tag{24}$$

5.2. Performance on power-law graphs

In this section, we present the performance of our analytical formula for node discovery for graphs other than the Erdos–Renyi model of the random graphs. We have carried out this evaluation in two steps. First, we have experimented with four different real-world graphs and then we have experimented with around 80 random graphs that model the social networks. These random graph models include (i) Power-law cluster graph (ii) Barabasi–Albert graph (iii) Newman–Watts Strogatz graph (iv) Watts Strogatz graph (v) Dual Barabasi Albert graph.

In the real world graph, the first one is a collaboration graph crawled from arXiv.org, High Energy Physics Theory section [30], from the year 1991 to the year 2003, called the hep graph as shown in Fig. 8a. The second graph is a social network graph of

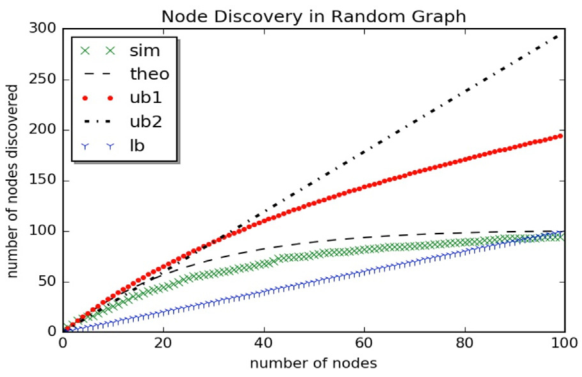


Fig. 7. Lower and upper bounds.

Facebook as shown in Fig. 8b. This graph consists of 'circles' (or 'friends lists') from Facebook. Facebook data [31] was collected from survey participants using this Facebook app. The third graph is a sensor network graph of 130-node sensor testbed consisting of WirelessHART [32] as shown in Fig. 8c. The last graph is a collaboration graph crawled from arXiv.org, Physics section known as phy. It is shown in Fig. 8d.

We observe that our analytical formula performs sufficiently well for all the graphs given in Fig. 9 by approximating linkage probability with Eq. (19). In fact, we observe that our analytical formula works sufficiently well for power-law graphs of collaboration and social networks given in Figs. 9a, 9b, Figs. 9c and 9d. The accuracy is best for the linkgain graph (Fig. 9c), followed by Hep graph (Fig. 9a), Facebook Graph (Fig. 9b) and Phy Graph (Fig. 9d).

We do the performance evaluation of our analytical model on various random graphs that have been proposed to model the social network and power-law graphs in the literature. The three important factors that differentiate the social network graphs from the Erdos–Renyi model of the random graph are (i) the power-law degree distribution of nodes and (ii) the clustering coefficient of the graph and (iii) the small world phenomenon.

Several models of power-law graphs are available in the literature. In Tables 3 and 4, we evaluate our work for various kinds of power-law graph models viz (i) power_law_cluster_graph (ii) barabasi_alberta_graph (iii) newman_watts_strogatz_graph (iv) watts_strogatz_graph (v) dual_barabasi_albert graph for modeling social network graph besides the erdos_renyi_graph model.

We give a brief description of these random graph models for modeling social networks.

1. Power law cluster graph: This graph model was given by Holme and Kim [33] for the generation of graphs with power-law degree distribution and a given approximate average clustering coefficient. The function powerlaw_cluster_graph(n, m, p) function provided in the networkx library in python is used for generating these graphs where n is the number of nodes in the graph, m is the number of random edges to be added for each new node and p is the probability of adding a triangle after adding a random edge.

2. Barabasi Albert graph: This model generates a random graph according to the BarabasiAlbert theory of preferential attachment [34] model. A graph containing n nodes is generated by attaching a new node with m edges. The nodes are attached preferentially to the existing nodes with higher degree.

3. NewmanWattsStrogatz graph: The Newman–Watts–Strogatz graphs generate a graph according to the small world phenomena. The function newman_watts_strogatz_graph(n, k, p) in the networkx library of python returns small-world graph [35] where n is the total number of nodes. Each node is joined with its k nearest neighbors in a ring topology and p is the probability of adding a new edge for each edge.

4. Watts Strogatz graph: The function watts_strogatz_graph(n, k, p) returns a WattsStrogatz small-world graph [25]. Here, n is the number of nodes in the graph. Each node is joined with its k nearest neighbors in a ring topology and p is the probability of rewiring each edge.

5. Dual Barabasi Albert graph: The function dual_barabasi_albert_graph(n, m_1, m_2, p) returns a random graph according to the dual BarabasiAlbert preferential attachment model [36]. Here, n is the total number of nodes, m_1 is the number of edges to attach from a new node to existing nodes with probability p, m_2 is the number of edges to attach from a new node to existing nodes with probability $(1 - p)$ and p is the probability of attaching m_1 edges (as opposed to m_2 edges)

From Table 3, we show the mean percentage error calculated using Eq. (22) for different random graphs having 400 nodes. We observe that the mean error for the power-law cluster graphs is 13.59%. The mean error decreases with an increase in the number of edges added for each node while the probability of adding a triangle being constant. We can observe from the table that for the power-law graphs increasing the probability of linkage does not affect much the error value. The mean error for the Barabasi Albert graphs is 13.07%, and it decreases with an increase in the number of edges added for each new incoming node while other variables of the graph remaining constant. For the Erdos–Renyi model, the mean error is 3.26% and it increases as the linkage probability increases. For the Newman–Watts–Strogatz graphs, the mean error is 14.46% and the mean error for Watts Strogatz graphs is 17.33%. For both of these graphs, the mean error decreases with the increase in p i.e. the probability of rewiring each edge. The mean error also decreases as the number of nearest neighbors joined with increases. For the Dual Barabasi Albert graphs,

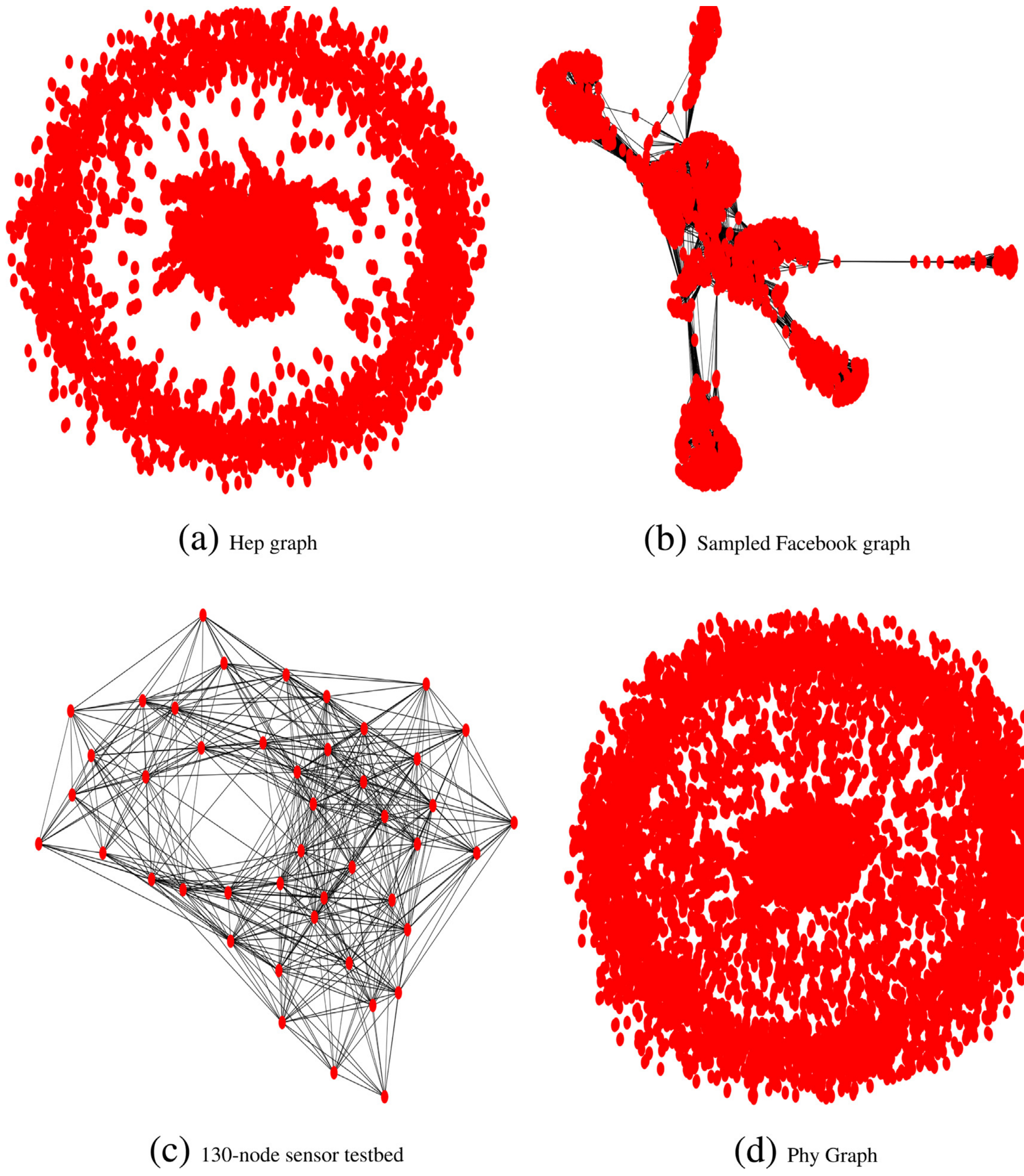
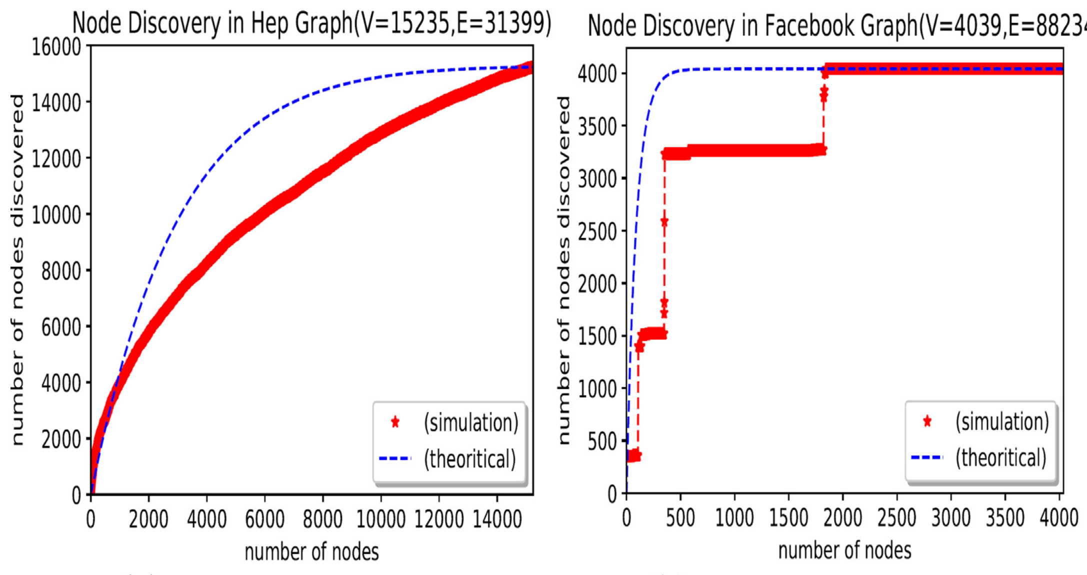


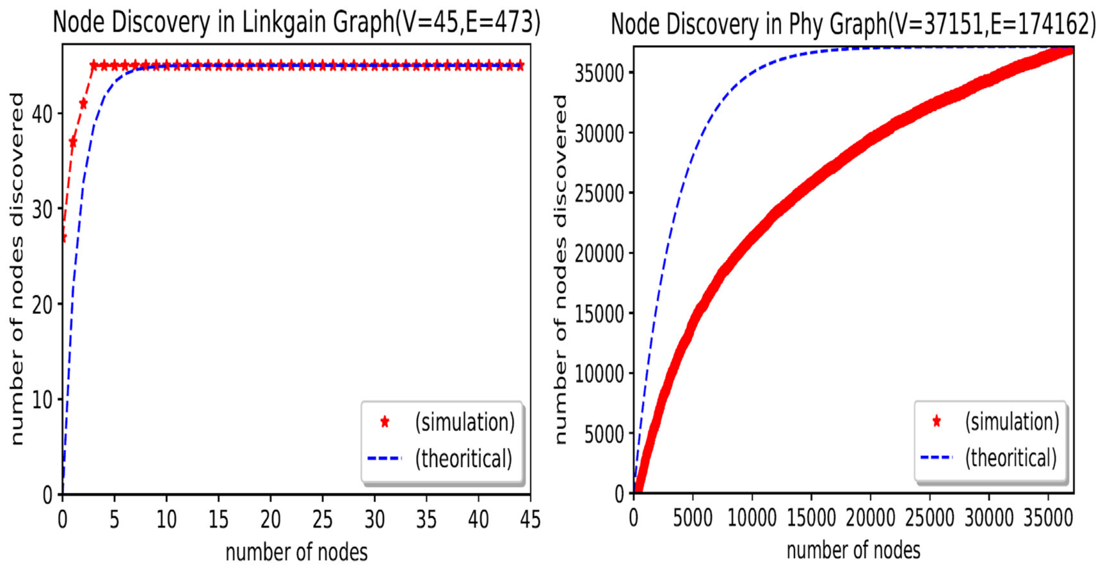
Fig. 8. Power law and other graphs.

the mean error is 12.8%. The mean error for all the social network graph models (including the Erdos–Renyi model) is 12.80% and excluding the Erdos–Renyi graph model is 14.31%. Hence, we observe that our analytical model can predict the number of nodes discovered quite well for a whole range of social network graph models that exhibit both power-law degree distribution and clustering effect.

In Table 4, we conduct the same set of experiments, but on graphs with $n = 800$ nodes. We observe that the mean error for power-law cluster graphs is 13.63% and it decreases as the number of random edges added to increases and increases as the linkage probability increases. The mean percentage error is 14.26% for the Barabasi Albert graph and it decreases with the increase in the number of nodes a new node is preferentially attached to. The mean error is 2.93% for the Erdos–Renyi graph and it increases with the increase in the linkage probability. The mean percentage error is 11.13% for the Newman–Watts–Strogatz graph, and 14.74% for the Watts Strogatz graphs. The mean error decreases with an increase in the probability of rewiring each edge. For both Newman–Watts Strogatz and Watts Strogatz graphs as number of nearest node connected to a node i.e., k increases the error decreases. The mean error for Dual Barabasi Albert graph is 9.42%, for the higher value number of nodes a node is attached it



(a) Nodes discovered in Hep graph (b) Nodes discovered in facebook graph



(c) Nodes discovered in linkgain graph (d) Nodes discovered in Phy Graph

Fig. 9. Performance in power law graphs.

shows lower error. The mean error for all the social network graph models including the Erdos–Renyi model is 10.01% and the mean error excluding the Erdos–Renyi graph model is 11.78%. We observe that as the nodes in the graph have increased, the mean error decreases.

In Table 5, we compare the minimum number of nodes required to discover the whole graph (called the domination number of the graph) computed by various algorithms and models. We have compared the domination number using the following algorithms

(i) Greedy approximation algorithm: This is a well-known greedy heuristic for finding the dominating set of a graph [7]. Initially, the dominating set D is empty. At each iteration of the algorithm, a vertex is added to the dominating set which would cover the maximum number of previously uncovered vertices.

(ii) Randomized greedy approximation algorithm: The algorithm GreedyRan is similar to the previous approximation algorithm [7]. The probability that a vertex is chosen in a given iteration is proportional to the number of additional vertices it would cover and it is not always the one that covers the maximum number of vertices.

(iii) Dominating Set by Abdol-Hossein Esfahanian (AHEDS) : This algorithm is an implementation of algorithm 7 in [37] which finds a dominating set, not necessarily the smallest one.

Table 3
Mean percentage error for various random network models for social network graphs for n = 400 nodes.

Graph	Error	Graph	Error
power_law_cluster(400,10,0.1)	18.56	newman_watts_strogatz(400,60,0.1)	18.7
power_law_cluster(400,20,0.1)	13.57	newman_watts_strogatz(400,70,0.1)	16.3
power_law_cluster(400,30,0.1)	10.22	newman_watts_strogatz(400,70,0.2)	10.29
power_law_cluster(400,40,0.1)	11.43	newman_watts_strogatz(400,70,0.3)	7.92
power_law_cluster(400,50,0.1)	13.49	newman_watts_strogatz(400,70,0.4)	6.21
power_law_cluster(400,50,0.2)	13.69	newman_watts_strogatz(400,70,0.5)	4.5
power_law_cluster(400,50,0.3)	13.86	watts_strogatz(400,40,0.1)	31.41
power_law_cluster(400,50,0.5)	13.96	watts_strogatz(400,50,0.1)	25.98
barabasi_albert(400,5)	18.31	watts_strogatz(400,60,0.1)	23.04
barabasi_albert(400,10)	17.19	watts_strogatz(400,70,0.1)	20.41
barabasi_albert(400,20)	10.43	watts_strogatz(400,70,0.2)	14.78
barabasi_albert(400,30)	8.07	watts_strogatz(400,70,0.3)	10.43
barabasi_albert(400,40)	11.37	watts_strogatz(400,70,0.4)	7.34
erdos_renyi(400,0.03)	1.79	watts_strogatz(400,70,0.5)	5.28
erdos_renyi(400,0.05)	1.49	dual_barabasi_albert(400,5,10,0.4)	8.6
erdos_renyi(400,0.1)	3.1	dual_barabasi_albert(400,5,20,0.4)	6.64
erdos_renyi(400,0.2)	4.57	dual_barabasi_albert(400,10,20,0.4)	7.85
erdos_renyi(400,0.3)	5.23	dual_barabasi_albert(400,10,20,0.2)	14.96
newman_watts_strogatz(400,40,0.1)	27.8	dual_barabasi_albert(400,30,50,0.2)	15.84
newman_watts_strogatz(400,50,0.1)	23.97	dual_barabasi_albert(400,30,50,0.4)	16.05
		dual_barabasi_albert(400,30,50,0.5)	15.46

Table 4
Mean percentage error for various random network models for social network graphs for n = 800 nodes.

Graph	Error	Graph	Error
power_law_cluster(800,10,0.1)	19.71	newman_watts_strogatz(800,140,0.1)	13.79
power_law_cluster(800,20,0.1)	17.3	newman_watts_strogatz(800,140,0.2)	8.56
power_law_cluster(800,30,0.1)	14.42	newman_watts_strogatz(800,140,0.3)	6.51
power_law_cluster(800,40,0.1)	11.59	newman_watts_strogatz(800,140,0.4)	5.08
power_law_cluster(800,50,0.1)	9.72	newman_watts_strogatz(800,140,0.5)	4.25
power_law_cluster(800,50,0.2)	11.19	watts_strogatz(800,80,0.1)	24.26
power_law_cluster(800,50,0.3)	12.18	watts_strogatz(800,100,0.1)	21.25
power_law_cluster(800,50,0.5)	12.95	watts_strogatz(800,120,0.1)	18.64
barabasi_albert(800,5)	18.21	watts_strogatz(800,140,0.1)	19.15
barabasi_albert(800,10)	18.68	watts_strogatz(800,140,0.2)	12.72
barabasi_albert(800,20)	15.72	watts_strogatz(800,140,0.3)	10.29
barabasi_albert(800,30)	11.01	watts_strogatz(800,140,0.4)	7.01
barabasi_albert(800,40)	7.41	watts_strogatz(800,140,0.5)	4.61
erdos_renyi(800,0.03)	1.04	dual_barabasi_albert(800,5,10,0.4)	12.89
erdos_renyi(800,0.05)	1.3	dual_barabasi_albert(800,5,20,0.4)	5.51
erdos_renyi(800,0.1)	2.6	dual_barabasi_albert(800,10,20,0.4)	12.46
erdos_renyi(800,0.2)	4.31	dual_barabasi_albert(800,10,20,0.2)	13.72
erdos_renyi(800,0.3)	5.41	dual_barabasi_albert(800,30,50,0.2)	7.76
newman_watts_strogatz(800,80,0.1)	21.1	dual_barabasi_albert(800,30,50,0.4)	6.54
newman_watts_strogatz(800,100,0.1)	16.02	dual_barabasi_albert(800,30,50,0.5)	7.11
newman_watts_strogatz(800,120,0.1)	13.8		

(iv) Minimum weight dominating set (MWDS): This algorithm computes an approximate minimum weighted node dominating set for a given graph [38]. In our case, as all the nodes have the same weight, hence it computes the smallest node dominating set.

(v) Randomized selection: In this algorithm, we randomly select nodes from the graph and find the dominating set as given in Algorithm 1 of this paper.

(vi) Theoretical model: In this, we use our analytical model to predict the domination number as proposed in this paper.

From Table 5, we observe that Greedy heuristic is given in [7] finds the dominating set of the smallest size for all the various social network graph models. For most of the graphs, the domination number increases in the following order for these algorithms: Domination Number(Greedy random) < domination number(AHEDS) < domination number(MWDS).

As expected, the size of the dominating set is higher when it is found by random sampling and that by the theoretical model because the theoretical model is also based on the random sampling of the Erdos–Renyi model of random graphs. However, we observe that the domination number provided by the analytical model is just around 20%–30% more than those provided by Greedy Random, AHEDS, and MWDS algorithms.

We observe that our model performs quite well for the Erdos–Renyi, Barabasi–Albert, Dual Barabasi–Albert, and the power-law cluster graph models. However, the model is not so accurate when applied to the Newman–Watts Strogatz and the Watts Strogatz graph models. Our analytical model underestimates the domination number in the above two cases.

Mostly, the domination number found for the analytical model is 3–4 times more than that found by the Greedy heuristic algorithm named Greedy Random.

Table 5
Domination number for power law graphs using different algorithms.

Random graph	Greedy	Greedy random	AHEDS	MWDS	Experimental	Theoretical
erdos_renyi(400,0.01)	114	154.1	162	368	120	310
erdos_renyi(400,0.03)	49.8	83	85.4	141.6	215	138
erdos_renyi(400,0.05)	33.2	59	59.9	90.82	147	86
erdos_renyi(400,0.1)	19.7	35.5	36.4	48.32	61	43
erdos_renyi(400,0.2)	11.5	20.5	20.78	23.68	25	21
erdos_renyi(400,0.3)	8.3	14.1	14.08	15.9	17	13
erdos_renyi(400,0.5)	5	8.2	8.56	8.22	9	7
barabasi_albert(400,5)	39.94	79.8	96	104	149	164
barabasi_albert(400,10)	23	54.6	66	60	74	88
barabasi_albert(400,15)	18	42.4	49	42	49	60
barabasi_albert(400,20)	14.9	36.8	39	31	38	45
barabasi_albert(400,30)	11.16	27	34	22	37	31
power_law_cluster(400,5,0.1)	37.58	72.8	94.32	99.96	146	164
power_law_cluster(400,10,0.1)	22.5	50.9	58	59	62	88
power_law_cluster(400,15,0.1)	16.58	41.1	58	57	47	60
power_law_cluster(400,20,0.1)	13.3	32.9	57	53	32	46
power_law_cluster(400,30,0.1)	10.2	25.1	57	53	34	31
power_law_cluster(400,40,0.1)	8.14	22.7	58	53	41	24
newman_watts_strogatz(400,40,0.1)	13	17	18.44	23	277	39
newman_watts_strogatz(400,50,0.1)	11	14	15	17.86	225	31
newman_watts_strogatz(400,60,0.1)	9	9	12	14.86	193	26
newman_watts_strogatz(400,70,0.1)	8	10	10.98	13.14	169	22
newman_watts_strogatz(400,70,0.2)	8	11	10.74	14.98	83	20
newman_watts_strogatz(400,70,0.3)	8	9	10.5	15.52	53	8
newman_watts_strogatz(400,70,0.4)	8	9	10.18	16.38	39	17
newman_watts_strogatz(400,70,0.5)	7	9	10.02	16.66	28	16
watts_strogatz(400,40,0.1)	16	25	28.02	29.48	267	151
watts_strogatz(400,50,0.1)	13	20	23.66	24.42	229	34
watts_strogatz(400,60,0.1)	12	17	20.36	21.28	200	28
watts_strogatz(400,70,0.1)	11	16	17.88	19.28	171	24
watts_strogatz(400,70,0.2)	12	16	20.86	21.02	92	24
watts_strogatz(400,70,0.3)	11	19	22.66	23.52	63	24
watts_strogatz(400,70,0.4)	12	22	23.64	25.78	49	24
watts_strogatz(400,70,0.5)	13	23	24.42	25.82	38	24
dual_barabasi_albert(400,5,10,0.4)	34	66	87.62	98.8	75	108
dual_barabasi_albert(400,5,20,0.4)	33	58	84.96	101.04	78	64
dual_barabasi_albert(400,10,20,0.4)	23	45	55.72	54.04	39	57
dual_barabasi_albert(400,10,20,0.2)	17	45	49.14	53.32	34	51
dual_barabasi_albert(400,30,50,0.2)	9	21	50.24	18.5	50	21
dual_barabasi_albert(400,30,50,0.4)	9	22	50.74	20.18	51	23
dual_barabasi_albert(400,30,50,0.5)	11	27	50.82	19.94	51	24

6. Discussion

In this section, we discuss the key implications of our proposed work and the analytical model.

- **Information Gathering and Propagation:** The analytical model provided by our decision support system can be used for providing the expected number of users that need to be randomly sampled to discover information about all the users in the social network. It can also be used to compute the expected number of users that need to be randomly sampled to propagate information to all the members in the social network in one iteration.
  - The collected information can be used to investigate the characteristics, behaviors, and opinions of a group of people. It can also be used to obtain information regarding gender, religion, ethnicity, experience, opinions, income, social status, psychological, geographical, and physical characteristics of the people. The collected information can be used for effectively facilitating decision making in many important and diverse applications such as city management, transportation, governance, education, health care, tourism, and so on. In a social network, information can be gathered fast because, in many situations, information about the neighbors can also be obtained from the queried user.
  - Information propagation in the social network can be used for targeted advertising, promotions, notifications, and recommendation systems. Our analytical model can provide the number of people to whom we should send the information for propagation in the social network. Estimating this number accurately can result in monetary benefit to the advertisers because of the advertisement campaign charges for each advertisement that is displayed.
- **Accessibility and Privacy Concerns:** This method of discovering nodes of the graph is relatively fast and requires relatively minimal costs. A key value proposition of our proposed method is that we do not need to acquire knowledge about the whole graph; we just need to know the total number of nodes in the graph and the average linkage probability. Observe that our

proposed model can work even if mining or crawling of the whole graph is not allowed. Notably, accessibility concerns are becoming increasingly important these days for ensuring user privacy and for sustaining competitive advantage in business settings.

- **Usefulness for the Power-Law Graphs:** We can model various graphs as a random graph using the formula given in Section 4.4. We observe that the accuracy of our formula is above 96% for random graphs with more than 500 nodes and our formula has an accuracy of around 87% for the power-law graphs. Most of the real-world graphs follow the power-law distribution. For example, Facebook, Linked In, Twitter, Instagram, and Snapchat graphs can all be modeled as power-law graphs. As our model works with an accuracy of around 87% on these graphs, it would be highly beneficial in predicting the number of users that we should randomly sample to collect data or disseminate information to the whole network.

7. Conclusion

In this paper, we have proposed a simple randomized node discovery algorithm and derived the expected number of nodes that are discovered by randomly querying m nodes of the Erdos–Renyi model of the random graph $G(n, p)$. We have also computed an analytical expression for the number of common neighbors that are shared by a given set of nodes in the graph. Tight lower and upper bounds have also been provided for the expected number of nodes that are discovered after querying the m number of nodes. We have compared our analytical formula with simulation results and obtained an accuracy of approximately 96%. The accuracy of our results improve when the number of nodes increases. Furthermore, we showed that our analytical formula could be used with other types of graphs such as power-law graphs. We also provided a scheme for computing the linkage probability for any general graph so that our model could be used in conjunction with them. We demonstrated that our analytical formula has an accuracy of around 87% for even the power-law graphs. Our work can potentially find application in a wide gamut of scenarios such as survey and data collection in a social network, fast information dissemination in a social network, targeted advertising, crowdsourced public directory service construction, and construction of recommendation systems. Besides, it can also be used in finding a randomized dominating set of a graph that finds application in computer networks, wireless networks, document summarization, and analysis of biological networks.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] C.-M. Chiu, T.-P. Liang, E. Turban, What can crowdsourcing do for decision support? Decis. Support Syst. 65 (2014) 40–49.
[2] D.S. Benco, P.C. Kanabar, J.C. Nguyen, H. Song, Caller ID Spoofing, Google Patents, 2008, US Patent App. 11/581, 634.
[3] R.D. Sadafule, Mobile app development for the Indian market, IEEE Softw. 31 (3) (2014) 17–20.
[4] M.M. Mostafa, More than words: Social networks' text mining for consumer brand sentiments, Expert Syst. Appl. 40 (10) (2013) 4241–4251.
[5] Y. Li, J. Fan, Y. Wang, K.-L. Tan, Influence maximization on social graphs: A survey, IEEE Trans. Knowl. Data Eng. 30 (10) (2018) 1852–1872.
[6] F. Kuhn, R. Wattenhofer, Constant-time distributed dominating set approximation, Distrib. Comput. 17 (4) (2005) 303–310.
[7] L.A. Sanchis, Experimental analysis of heuristic algorithms for the dominating set problem, Algorithmica 33 (1) (2002) 3–18.
[8] J. Albath, M. Thakur, S. Madria, Energy constraint clustering algorithms for wireless sensor networks, Ad Hoc Netw. 11 (8) (2013) 2512–2525.
[9] J. Blum, M. Ding, A. Thaeler, X. Cheng, Connected dominating set in sensor networks and MANETs, in: Handbook of Combinatorial Optimization, Springer, 2004, pp. 329–369.
[10] C. Shen, T. Li, Multi-document summarization via the minimum dominating set, in: Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), 2010, pp. 984–992.
[11] P.G. Sun, Y.N. Quan, Q.G. Miao, J. Chi, Identifying influential genes in protein–protein interaction networks, Inform. Sci. 454 (2018) 229–241.
[12] U. Feige, A threshold of ln n for approximating set cover, J. ACM 45 (4) (1998) 634–652.
[13] W. De Nooy, A. Mrvar, V. Batagelj, Exploratory Social Network Analysis with Pajek, Cambridge University Press, 2018.
[14] S.M. Ross, Introduction to Probability and Statistics for Engineers and Scientists, Academic Press, 2014.
[15] A.K. Parekh, Analysis of a greedy heuristic for finding small dominating sets in graphs, Inf. Process. Lett. 39 (5) (1991) 237–240.
[16] S.E. Nikoletseas, P.G. Spirakis, Near-optimal dominating sets in dense random graphs in polynomial expected time, in: International Workshop on Graph-Theoretic Concepts in Computer Science, Springer, 1993, pp. 1–10.
[17] W. Chen, Y. Wang, S. Yang, Efficient influence maximization in social networks, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2009, pp. 199–208.
[18] I. Roelens, P. Baecke, D.F. Benoit, Identifying influencers in a social network: The value of real referral data, Decis. Support Syst. 91 (2016) 25–36.
[19] S. Peng, A. Yang, L. Cao, S. Yu, D. Xie, Social influence modeling using information theory in mobile social networks, Inform. Sci. 379 (2017) 146–159.
[20] D. Kempe, J. Kleinberg, É. Tardos, 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, 2003, pp. 137–146.
[21] D.S. Hochbaum, Approximation algorithms for the set covering and vertex cover problems, SIAM J. Comput. 11 (3) (1982) 555–556.
[22] M.E. Newman, D.J. Watts, S.H. Strogatz, Random graph models of social networks, Proc. Natl. Acad. Sci. 99 (suppl 1) (2002) 2566–2572.
[23] M.E. Newman, et al., Random graphs as models of networks, in: Handbook of Graphs and Networks, Vol. 1, Wiley Online Library, 2003, pp. 35–68.
[24] M.E. Newman, Coauthorship networks and patterns of scientific collaboration, Proc. Natl. Acad. Sci. 101 (suppl 1) (2004) 5200–5205.
[25] D.J. Watts, S.H. Strogatz, Collective dynamics of 'small-world'networks, nature 393 (6684) (1998) 440–442.
[26] S.-H. Yook, H. Jeong, A.-L. Barabási, Modeling the internet's large-scale topology, Proc. Natl. Acad. Sci. 99 (21) (2002) 13382–13386.
[27] X. Dimitropoulos, D. Krioukov, A. Vahdat, G. Riley, Graph annotations in modeling complex network topologies, ACM Trans. Model. Comput. Simul. (TOMACS) 19 (4) (2009) 17.
[28] J. Kahn, N. Linial, A. Samorodnitsky, Inclusion-exclusion: Exact and approximate, Combinatorica 16 (4) (1996) 465–477.

[29] O. Diallo, J.J. Rodrigues, M. Sene, J. Niu, Real-time query processing optimization for cloud-based wireless body area networks, Inform. Sci. 284 (2014) 84–94.
[30] J. Leskovec, J. Kleinberg, C. Faloutsos, Graph evolution: Densification and shrinking diameters, ACM Trans. Knowl. Discovery Data (TKDD) 1 (1) (2007) 2.
[31] J. Leskovec, J.J. Mcauley, Learning to discover social circles in ego networks, in: Advances in Neural Information Processing Systems, 2012, pp. 539–547.
[32] V.P. Modekurthy, A. Saifullah, S. Madria, Distributed graph routing for wirelesshart networks, in: Proceedings of the 19th International Conference on Distributed Computing and Networking, ACM, 2018, p. 24.
[33] P. Holme, B.J. Kim, Growing scale-free networks with tunable clustering, Phys. Rev. E 65 (2) (2002) 026107.
[34] A.-L. Barabási, R. Albert, Emergence of scaling in random networks, Science 286 (5439) (1999) 509–512.
[35] M.E. Newman, C. Moore, D.J. Watts, Mean-field solution of the small-world network model, Phys. Rev. Lett. 84 (14) (2000) 3201.
[36] N. Moshiri, The dual-Barab\'asi-Albert model, 2018, arXiv preprint arXiv:1810.10538.
[37] A.-H. Esfahanian, Connectivity algorithms, in: Topics in Structural Graph Theory, Cambridge University Press, 2013, pp. 268–281.
[38] V.V. Vazirani, Approximation Algorithms, Springer Science & Business Media, 2013.

Dr. Samant Saurabh Samant Saurabh is Assistant Professor in the Department of Information Systems and Data Analytics at Indian Institute of Management Bodh Gaya. He received his Ph.D. in Computer Science from Indian Institute of Technology, Patna. He has done his Masters from the University of Massachusetts, Amherst and B. Tech from Indian Institute of Technology Guwahati in Electronics and Communication Engineering. His area of interest is Algorithms, Network Security and Data Mining.

Dr. Sanjay Kumar Madria Sanjay Kumar Madria received his Ph.D. in Computer Science from Indian Institute of Technology, Delhi, India in 1995. He is a Professor, Department of Computer Science, at the Missouri University of Science and Technology (formerly, University of Missouri-Rolla), USA and Site Director, NSF Industry/University Center on Net-centric System Software. Earlier he was Visiting Assistant Professor in the Department of Computer Science, Purdue University, West Lafayette, USA. He co-authored a book entitled "Web Data Management: A Warehouse Approach" published by Springer-Verlag. He guests edited WWW Journal. His research interests are in cloud computing, mobile p2p, security and data management. He served as a general co-chair of the IEEE Mobile Data Management conference and IEEE Symposium on Reliability in Distributed Systems. He received faculty excellence award in 2007, 2009, 2011 and 2013, Japanese Society for Promotion of Science invitational fellowship in 2006, and Air Force Research Lab's visiting faculty fellowship from 2008 to 2015. He is IEEE Senior Member, IEEE Golder Core Member and an ACM Distinguished Visitor program.

Dr. Anirban Mondal Anirban Mondal is an Associate professor at Asoka University India. Prior to this, he had worked as a senior researcher in Xerox research and was an Associate Professor at the Indraprastha Institute of Information Technology, Delhi (IIITD), INDIA. He had been working for the University of Tokyo for a long tenure of seven years. He received his Ph.D. in Computer Science from the School of Computing, National University of Singapore (NUS) in 2003. He had completed his Bachelor of Technology (with Honors) in Computer Science and Engineering at the Indian Institute of Technology (IIT), Kharagpur, India in 1998. His research interests include Peer-to-Peer environments, mobile environments, mobile-P2P networks, GRID computing, distributed data management, spatial data and Geographic Information Systems, indexing and load-balancing for multidimensional databases, DataBanking and autonomic computing. He had been awarded the prestigious JSPS (Japanese Society for Promotion of Science) Fellowship from September 2003–September 2005. He has published several papers and given invited talks at several key international database conferences/workshops.

Dr. Ashok Singh Sairam Ashok Singh Sairam is working as Associate Professor in the Department of Mathematics at Indian Institute of Technology Guwahati since August 2017. Prior to this, he had been working for eight and half years in the Department of Computer Science and Engineering at Indian Institute of Technology Patna. His research interest is in the broad areas of computer networks and network security. Currently, he is working in software-defined networks, crowdsourcing techniques, and location-based privacy. He received his B.Tech degree in Computer Science from NIT Silchar and his M.Tech and a Ph.D. degree from IIT Guwahati. He had undertaken and worked on various projects of the Department of Science and technology and Department of information technology, Government of India. He is a Senior Member-IEEE and Member-ACM. He has published several papers and given invited talks at several key international conferences/workshops.

Mr. Saurabh Mishra Saurabh Mishra is a Ph. D. scholar at the Shiv Nadar University, where he works on mobile data management and data mining and crowdsourcing. He earned his M.Tech. degree in System Analysis and Computer Applications from National Institute of Technology Surathkal, India.