Ddensest subgraph discovery

Project submitted to

FACULTY OF ENGINEERING AND TECHNOLOGY

JADAVPUR UNIVERSITY

In partial fulfillment of the requirements for the degree of

MASTER OF COMPUTER APPLICATIONS, 2019

BY

Somen Seal

Examination Roll: MCA196026

Registration No: 137318 of 2016-2017

Roll No : 001610503010

Under the guidance of

Prof. Nirmalya Chowdhury

Department of Computer Science Engineering

Jadavpur University, Kolkata-700032

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

FACULTY OF ENGINEERING AND TECHNOLOGY

JADAVPUR UNIVERSITY

TO WHOM IT MAY CONCERN

I hereby recommend that the project entitled "Densest Subgraph Discovery" prepared under my supervision and guidance at Jadavpur University, Kolkata by SOMEN SEAL (Reg. No. 137318 of 2016 – 17, Class Roll No. 001610503010 of 2016-17), may be accepted in partial fulfillment for the degree of Master of Computer Applications in the Faculty of Engineering and Technology, Jadavpur University, during the academic year 2018 – 2019. I wish him every success in life.

……..

Prof. (Dr.) Mahantapas Kundu
Head of the Department
Department of Computer Science and Engineering
Jadavpur University, Kolkata – 700032.

……..…

Prof. (Dr.) Nirmalya Chowdhury
Project Supervisor,
Department of Computer Science and Engineering
Jadavpur University, Kolkata – 700032.

……

Prof. (Dr.) Chiranjib Bhattacharjee
Dean, Faculty council of Engg. & Tech.
Jadavpur University, Kolkata – 700032

DECLARATION OF ORIGINALITY AND COMPLIANCE OF ACADEMIC PROJECT

I hereby declare that this project contains literature survey and original research work by the undersigned candidate, as part of his MASTER OF COMPUTER APPLICATIONS studies. All information in this document have been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material results that are not original to this work.

NAME: SOMEN SEAL

ROLL NUMBER: 001610503010

PROJECT TITLE: **DENSEST SUBGRAPH DISCOVERY**

SIGNATURE WITH DATE:

JADAVPUR UNIVERSITY
FACULTY OF ENGINEERING AND TECHNOLOGY

CERTIFICATE OF APPROVAL

The forgoing project is hereby accepted as a credible study of an engineering subject carried out and presented in a manner satisfactory to warrant its acceptance as a prerequisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or approve any statement made, opinion expressed or conclusion drawn therein, but approve the project only for the purpose for which it is submitted.

FINAL EXAMINATION FOR
EVALUATION OF PROJECT: 1._____

2._____

(Signature of Examiners)

ACKNOWLEDGEMENT

I express my honest and sincere thanks and humble gratitude to my respected teacher and guide *Prof.(Dr.) Nirmalya Chowdhury*, Professor of the Department of Computer Science & Engineering, Jadavpur University, for his exclusive guidance and entire support in completing and producing this project successfully. I am very much indebted to him for the constant encouragement, and continuous inspiration that he has given to me. The above words are only a token of my deep respect towards him for all he has done to take my project to the present shape.

I would like to thank *Mr. Ritam Sarkar* for valuable support and suggestions to the activities of the project.

Finally, I convey my real sense of gratitude and thankfulness to my family members, specially my elder sister, for being an endless source of optimism and positive thoughts; and last but not the least, my father & mother for their unconditional support, without which I would hardly be capable of producing this huge work.

SOMEN SEAL

Examination Roll: MCA196026

Class Roll: 001610503010

Registration No: 137318 of 2016 – 2017

CONTENTS

- Algorithm for densest subgraph-example

- Greedy 2-approximation for d(G)
- Structure in LP

❖RELATED WORKS

❖APPLICATION OF DENSEST SUBGRAPH DISCOVERY

- Fraud Detection

❖EXPERIMENTAL OBSERVATION

❖CONCLUSION AND FUTURE WORKS

❖ **APPENDIX**

- References

Abstract :

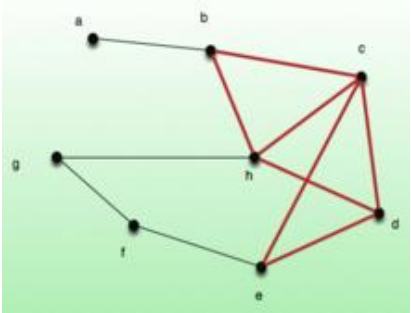
Finding a subgraph with maximum density is a very important graph-mining task with many applications. It is given that the direct optimization of edge density is not meaningful, as even a single edge achieves maximum density for a graph. In present times, research has focused on optimizing alternative density function. A very popular among those functions is the average degree, whose maximization leads to the well-known densest subgraph notion. Surprisingly, densest subgraphs are typically large graphs having a small edge density and large diameter. Here we will discuss the problem of finding highly connected subgraphs for undirected graphs. For undirected graphs, the average degree of subgraph is used for the notion of density of a subgraph. We study the optimization problems of finding subgraphs maximizing these notions of density for undirected graphs. This project gives a simple greedy approximation algorithm for these optimization problems. The dense subgraph discovery is very related to the clustering. Though the two problems also have the number of differences. For example, in one side the problem of clustering is mostly concerned with that of finding fixed partition in the data whereas in the other hand the problem of finding dense subgraph discovery, the dense components are defined in much more flexible way. Without any size constraints, a subgraph having maximum density can be found in polynomial time. But when we require a subgraph having a specified size, the problem of finding a maximum density subgraph becomes NP-hard. In this paper i will focus on developing fast polynomial time algorithm for several variation of dense subgraph problem for undirected graphs.

Discovering the densest subgraph is a very important in graph analysis, which has wide-ranging applications from the social network community mining to the discovery of biological network modules. As a result, it may lead to isolated subgraphs in the output though they aim to find one connected dense subgraph from the graph. Also, there are lack some of efficient algorithms for big natural graphs, especially considering the datasets become increasingly larger in this era of Big Data.

Introduction

What is density of a subgraph ?

Given an undirected graph G = (V, E), here the density of subgraph on a vertex set S is defined as d(S) = $\frac{|E(S)|}{|S|}$, where E(S) is the set of edges in the subgraph induced by S and ISI is the number of elements in S. The problem of finding densest a subgraph of a given graph G can be solved optimally and it can be done in polynomial time, despite the fact that there are exponentially many subgraphs can be considered.



This is an example of a graph G with density d= 1.375 and the subgraph with maximum density induced by the vertices b, c, d, e and h (in red) with density 1.4

Let G = (V, E) be a undirected graph with vertices V and edges $E \subseteq V \times V$ here, E(V) stands for the set of edges induced by V , i.e

$E(V) = \{(i, j) \in E : i \in V, j \in V\}$

Then the density of subgraph induced by $S \subseteq V$ can be defined as; $d(S) = |E(S)| / |S|$

Remember that, here 2d(S) is the actually average degreee of the subgraph induced by S. The Densest Subgraph problem can be defined as,

$DS(G) = \max\{d(S)\}$, where $S \subseteq V$.

The densest subgraph problems have received significant attention for detecting important substructers in very massive graphs in different social networks and web. In a web graph, hubs that is resource lists and authorities that is authoritative pages on a topic are charecterized by large number of links between them. Finding a densest subgraph also has been acted as a useful primitive for discovering communities in social networks and web, for compressed representation of a graph and for spam detection. Here we consider the greedy algorithm for the densest subgraphs in undirected graphs. Mining coherent dense subgraphs across massive biological networks for functional discovery.It's a useful subroutine for many application.

Extracting the densest subgraph that is finding the subgraph that maximizes the average degree, is specifically attractive as it can be solved exactly in polynomial time or approximated within a factor of 2 in linear time. Indeed it is a very popular choice in many applications. However, as we will see in detail next of this project, maximizing the average degree tends to favor large subgraphs with not very large edge density. A prototypical dense graph is the clique, but,finding the largest clique is inapproximable. Infact, the clique

definition is too strict in practice, as not even a single edge can be missed from an otherwise dense subgraph. This observation of portion leads to the definition of quasiclique, whose underlying intuition is the following,

Assumingly each edge in a subgraph G(S) exists with probability α, then the expected number of edges in G[S] is $\alpha.\binom{|s|}{2}$. Thus, the condition of α-quasi-clique expresses the fact that the subgraph G(S) has more edges than those expected by binomial model. Thus motivated by this definition, we can turn the quasi-clique condition into an objective function. In particular, we can define the density function f_α(S) = e[S] – α. $\binom{|s|}{2}$, which expresses the edge surplus of a set S over the expected number of edges under the random-graph model.

Some definitions :

Clique :

In graph theory clique is a subset of vertices of an undirected graph such that every two different vertices are adjacent to each other.

Maximal Clique :

A maximal clique is a clique which can not be extended by including one more adjacent vertex. A clique which does not exist exclusively within the vertex set of a larger clique.

Maximum clique :

A maximum clique of a graph G, is also a clique, so that there is no clique having more vertices.However, the clique number w(G) is the number of verticesin a maximum clique in G.

Densest K-Subgraph:

There are so many variations in the densest subgraph problem in graph theory. One of them is the densest subgraph problem, where the objective is to find the maximum density subgraph on exactly vertices

Densest atmost K-Subgraph:

Here the objective of the densest at most K problem is to find the maximum density subgraph on at most K vertices.

Densest atleast K-Subgraph:

The densest at least K problem is defined similarly to the densest at most subgraph problem. Here the objective is to find the maximum density subgraph on atleast K vertices.

K- Clique Densest subgraph :

This variation of the densest subgraph problem aims to maximize the average number of induced k cliques , where $d_k(s) = \frac{|c_k(s)|}{|v_s|}$, where $c_k(s)$is the set of k-cliques induced by S . Notice that the densest subgraph problem is obtained as a special case for k=2. This generalization provides an empirically successful poly-time approach for extracting large near-cliques from large-scale real-world networks.

Relative and Absolute Density

There are two main classes of density which are relative and absolute density. Relative density is defined by comparing the

density of the subgraph with the external components. Here, dense areas are defined by being separated by low-density areas. Relative density is often used for graph clustering and will not be further discussed in this paper.

Whereas, absolute density uses a specific formula to calculate the density and every graph has an explicit value. Examples of that are cliques and relaxations thereof. The average degree which will be used in this project is also an example of absolute density.

NP- hard :

NP-hardness(non-deterministic polynomial time hardness). In computational computation theory, is the defining property of a class of problems that are informally " atleast as hard as the hardest problems in NP". A simple example of an NP-hard problem is the subset sum problem.

Statement of the problem

In this section, I present a model of undirected graphs, define the density of an undirected graph, and give a formal statement of the densest subgraph problem on undirected graphs. Here, I also have introduced some helpful notation

PROPOSED METHODOLOGY :

How does the method actually work? The greedy algorithm at each step chooses a vertex of minimum degree, and then deletes it and preceeds for (n-1) steps where the number of vertices, |V| = n. At each and every step the density of the remaining subgraph is calculated and finally the the algorithm returns a graph with maximum density.

Densest subgraph without any size restriction

For undirected graphs, a flow based algorithm was developed by Goldberg, that finds a densest subgraph of a graph in polynomial time. However,no flow based algorithm was known for directed graphs. Here I consider the greedy algorithm for densest subgraph for undirected graphs proposed by M. Charikar. This improves the running time from O($|v|^3$ + $|v|^2$|E|) to O(|V | + |E|). Here I will also show that a very simple proof of 2-approximation for the greedy algorithm developed by Charikar to obtain a densest subgraph for undirected graphs

ALGORITHM

Here I implement M. Charikar's greedy algorithm for finding densest subgraph in an undirected graph.

Input: Given an undirected graph G = (V, E)

Output: S, a subgraph with maximum density.

n← IVI ,

H_n←G,

For j = n to 1

do

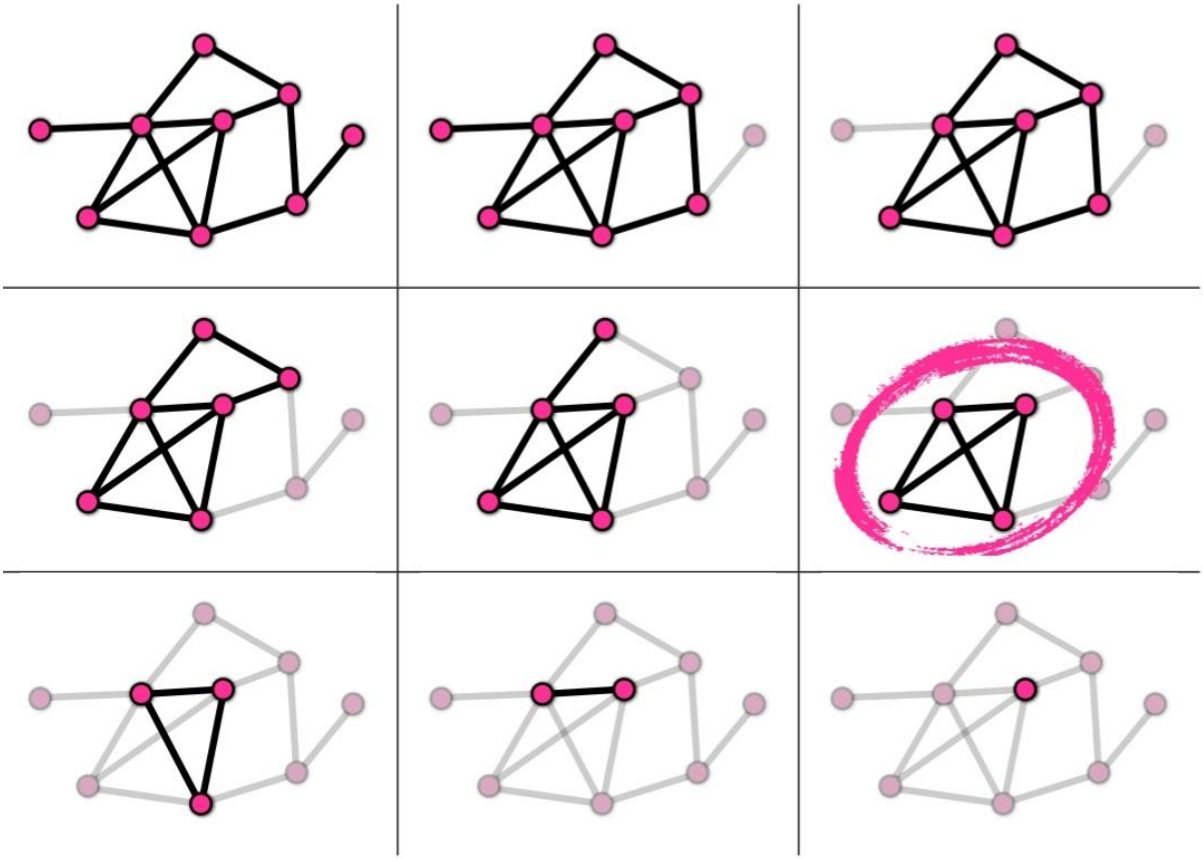
If v be a vertex in H_j of minimum degree

H_{j-1} ← H_j - {v}

return H_k

Here, H_k is having the maximum density among H_j's , j=1,2,....,n

Algorithm for densest subgraph-example



Greedy 2-approximation for d(G)

We want to produce a subgraph of G which has a large average degree. Intuitively, we should throw away those low degree vertices in order to produce such subgraph. It suggests us a fairly natural greedy algorithm. In fact, the performance of such algorithm has been analyzed by some others authors slightly different problem, that of obtaining a large average degree subgraph on a given number k of vertices from a graph. The algorithm clearly

maintains a subset S of vertices. Initially S ← V . In each iteration, the algorithm here identifies imin, the vertex of minimum degree in the subgraph induced by S. The algorithm eliminates imin from the set S and it moves on to the next iteration. The algorithm stops when the set S is empty. Of all the sets S constructed during the execution of the algorithm, the set S maximizing d(S) (that is, the set of maximum average degree) is returned as an output of the algorithm. We will prove that the algorithm produces a 2 approximation for d(G). There are various ways for proving this. This will set the stage for the algorithm for d(G) later. However, I believe the proof is interesting because it connects between the greedy algorithm and the dual of the LP formulation. In order to analyze the algorithm, we produce an upper bound on the optimal solution. The upper bound has following form: I assign each edge ij to the either i or to the j. For the vertex i, d(i) is the number of edges ij or ji assigned to i. Let, dmax = $max_i d(i)$. (There is an another way to view this matter is that we will orient the edges of the graph and dmax is the maximum number of edges oriented towards any vertex). This shows that f (S) is bounded by dmax .

$max_{S \subseteq V}$ {d(s)} ≤ dmax

Consider the set S that maximizes d (S). Now, each edge of E(S) must be assigned to a vertex in the set S. Thus |E(S)| ≤|S|. dmax

d(S) = |E(S)|/ |S| ≤ dmax

Hence, proved.

Now, the assignment of the edges to one of the end points is constructed as the algorithm executes. Initially, all the edges are

unassigned. When the least degree vertex is deleted from S, the vertex is assigned all the edges that go from the vertex to the rest of the vertices in the set S. It is maintaining that the invariant that all edges between two vertices in the current set S are unassigned; all the other edges are assigned. At the end of the execution of the algorithm, all edges will be assigned.

Let, dmax be defined as before for the specific assignment constructed corresponding to the execution of greedy algorithm. It relates the value of the solution constructed by the greedy algorithm to dmax .

If v be the maximum value of d (S) for all sets S obtained during the execution of the greedy algorithm. Then dmax $\leq$ 2v.

Consider, single iteration of the greedy algorithm. Since, i_{min} is selected as the minimum degree vertex in S, its degree will at most 2|E(S)|/|S|$\leq$ 2v. Note that a particular vertex gets assigned edges to it only at a point when it is deleted from S. This proves that dmax $\leq$ 2v.

The greedy algorithm gives a 2 approximation for d (G).

It is easy to observe that the greedy algorithm can be implemented to run in O(n^2) time for a graph with n vertices and m edges. We can maintain the degrees of the vertices in the subgraph induced by S. In each iteration involves identifying and removing the minimum degree vertex as well as it is updating the degrees of the remaining vertices and both of which can be done in O(n) time. Moreover, we can implement the algorithm to run in linear time. As, the degree of a vertex is an integer, we can maintain lists of vertices with the same

degrees, that is a list of vertices of degree 0; 1; 2 and so on. So, in each iteration, the minimum degree vertex is eliminated and the degrees of the neighboring vertices updated, requiring them to be moved to new lists. The total work done in these all updates is O(m). Here the minimum degree drops by at most 1 in each iteration. If the minimum degree in a specific iteration was d, the minimum degree vertex for the next iteration is obtained by scanning the lists for degrees d-1, d, d + 1 and it goes like that. So, the total work done in scanning these lists is O(n). Thus, the algorithm runs in time O(m + n).

Structure in LP The optimal densest subgraph for undirected graph can also be computed using the following LP.

maximize $\sum_{i,j} x_{i,j}$

$x_{i,j} \leq \min(y_i , y_j)$, $\forall$ (i, j) $\in$ E(G)

$\sum_i y_i = 1$, $\forall$ i $\in$ V (G)

$x_{i,j}, y_i \geq 0$, $\forall$ (i, j) $\in$ E(G), $\forall$ i $\in$ V (G)

Charikar showed that there exists an i $\in$ {1, .., |V (G)|} such that the density of the subgraph induced by the vertices with the value of y at least y_i is equal to the optimal density. Thus for considering each value of y_i , i = 1, . . . , |V (G)| and checking their density, the maximum density subgraph can be obtained. Moreover, here we can show that there exists an LP solution where all the y_i values are equal and thus the integrality gap of the above LP (2) is 1. We also show that for any LP optimal solution, picking all the given vertices with positive y values returns a maximum density subgraph.

Related Works :

In recent times, there have been significant efforts in devising efficient algorithms for finding densest subgraph. There are several definitions of density have been studied including cliques and quasi cliques, minimum degree density, α-β communities, k-cores and densest subgraph. Among these all, the average degree density stands outs as a natural definition of density. For an undirected graph, the average density is defined as a ratio between the number of edges and number of vertices for such a graph. Charikar devised a linear programming based approach as well as linear 2-approximation algorithm for such a problem. In a large input graph, finding densest subgraph has emerged as an important subroutine used to tackle a host of real world problems. Finding dense region in any web graph has received large importance recently. Densest subgraph computation is a very crucial subroutine in graph indexing and efficient reachability and distance queries as well as for graph compression. There are useful different applications for finding different definitions of dense components. Understanding the feature of various types of components are very much valuable for deciding which type of component to pursue. Now we can divide density definitions into two classes. One is absolute density and the other is relative density. An absolute density measures parameter values and establishes methods for what constitutes a dense component. For an example, we can say that we are interested in cliques, fully connected region of maximum density. It measures take the form of relaxations of the pure clique measure.

On the other hand, relative density measure has no preset level for what is sufficiently dense there. The density of one region to another with the goal of finding the densest region is compared by relative density. For establishing the boundaries of components, a metric

typically looks to maximize the difference between inter-component connectedness and intra-component connectedness. It is often but not necessarily, the relative density techniques looks for a user defined k densest regions. If we alert on concentrating, then we may have noticed that relative density discovery is closely related to clustering and in fact many features are shared with it.

Dense Subgraph Problem

The history for Dense Subgraph problem for static graphs has a short history rather, as the best exact solution was already proposed by Goldberg in 1984 and the best approximation algorithm were proposed so far by Charikar in 2000, Khuller and Saha in 2009 for undirected and directed graphs. But Goldberg's solution works for undirected graphs only. His idea was to interestingly transfer this dense subgraph problem into a min-cut problem by adding two vertices s and t, where both the vertices s and t are connected with all the vertices in graph G = (V, E). For each vertex $v_i \in$ V , edge (s, v_i) has edge weight that is the same as the degree of v_i and edge (v_i , t) has edge weight of a positive constant c. If all the edges in the original graph has an edge weight of 1, then buy performing a min-cut call that splits s and t into two subgraphs, among them one of the subgraphs would be the densest subgraph of G after removing s or t. Since the min-cut problem can be solved using the parametrix max-flow algotirhm, this algorithm has a O(|V ||E|) time complexity. Thus Goldberg's algorithm is not suitable for large graphs. Now faster approximation algorithms are more preferred in industry situations.

Application of dense subgraph discovery:

In present era, for financial and economic analysis, densest components represent entities that correlated highly. For an example, define a graph of market, here each vertex is a financial instrument and two vertices are connected if their behaviour (say, price change overtime) are highly correlated. A dense component indicates a set of instruments whose members are well correlated to one another. The information provided here are very valuable both for predicting the behaviour of individual instruments and understanding market dynamics. Density can indicate strength and robustness.

In the first part of the 21st century, the field perhaps had shown the greatest benefitted and interest the most from dense component analysis is in biology. Molecular and many systems biologists formulated many types of networks, like signal transduction and gene regulation networks, protein interaction networks, ecological network, metabolic networks, phylogenetic networks.

However, there is some organization among the proteins. Dense components in protein-protein interaction networks have been shown to correlate to the functional units

Gene expression faces some more similar challenges. Here Microarray experiments can be recorded of which of the thousands of genes in a genome are expressed under a set of test conditions and over time. After compiling the expression results from several trials and experiments, a network can be constructed there. Clustering the genes into dense groups can be used to identify not only healthy functional classes, but also for the expression pattern for genetic diseases. Proteins interact with genes by activating and regulating gene transcription and transcription. Density in a protein-

gene bipartite graph suggests which protein groups or complexes operate on which genes. Other biological systems are also being modeled as networks. Ecological networks, famous for food chains and food webs, are receiving new attention as more data becomes available for analysis here and as the effects of climate change become apparent more. Today, the natural sciences, the social sciences, and technological fields are all using network and graph analysis methods to understand complex systems better. Dense component discovery and analysis is one very much important aspect of network analysis. Therefore, the readers from many different backgrounds will certainly benefit from understanding more about the characteristics of dense components and some of the methods used to uncover them.

Fraud Detection

Data-driven approaches have been received great success in the field of fraud detection. Most of the methods indentify unexpected dense regions of the bipartite graph, as creating fake reviews or ratings unavoidably generates edges in the graph.

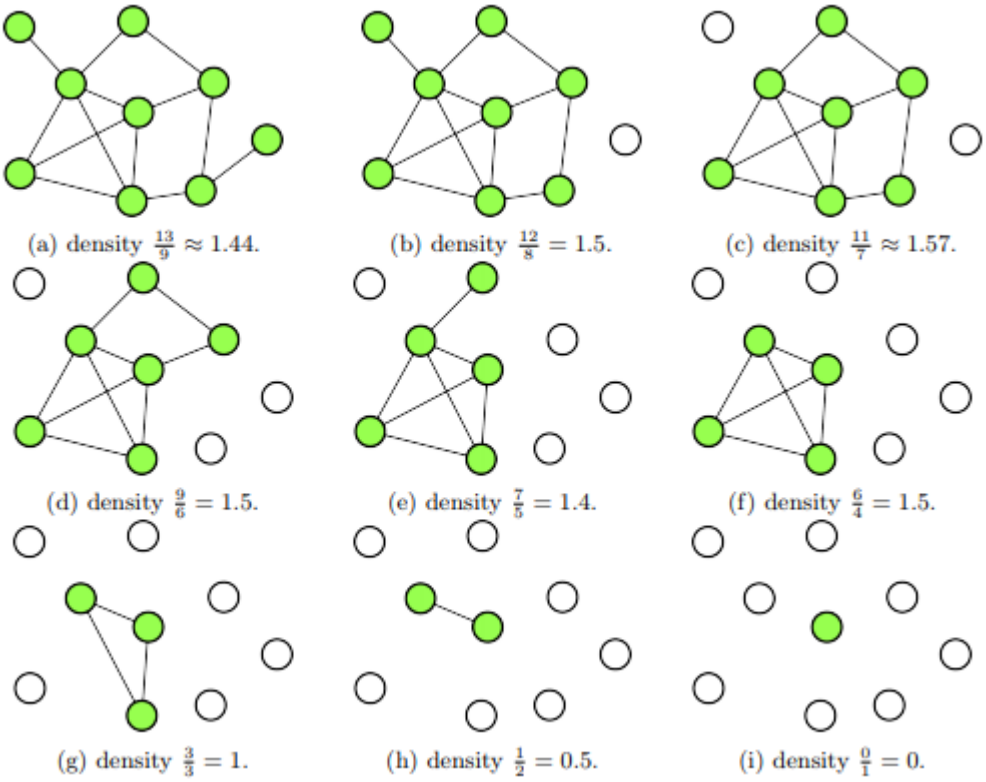
Global graph mining methods model the entire graph to find the fraud based on singular value decomposition (SVD), latent factor models, and belief propagation (BP). “spokes” was considered by SpokEn, the pattern produced by pairs of eigenvectors of graphs, and it was later generalized for fraud detection. fBox focuses on mini-scale attacks missed by spectral techniques. BP was used for fraud classification by eBay, link farming by Twitter and fake software review detection.

Finding dense subgraphs has been studied from a wide array of perspectives like mining frequent subgraph patterns, detecting communities, and finding quasi-cliques. Now, average degree of subgraph can be maximized with approximation guarantees. The density of adjacency matrix of subgraph can be optimized with quality guarantees. It adopts both the node degree and edge density to model suspiciousness of subgraph and further increases accuracy in binary adjacency matrix of a bipartite graph

Typically in time series there are two kinds of representations on density. One is dense subgraphs in evolving graphs. Local search heuristics is used by CopyCatch to find Δt-bipartite cores in which users consistently likes the same Facebook pages at the same short of time interval.

Experimental Evaluation :

Here all graphs were made simple and undirected by ignoring the edge direction. The experiments were performed in a single machine, with Intel(R) Core(TM) i3 4005U CPU @ 1.70 GHz, 4GB main memory, 64-bit windows 8.1 operating system. I have implemented this program in python, version 3.7.



(a) density $\frac{13}{9} \approx 1.44$. (b) density $\frac{12}{8} = 1.5$. (c) density $\frac{11}{7} \approx 1.57$.

(d) density $\frac{9}{6} = 1.5$. (e) density $\frac{7}{5} = 1.4$. (f) density $\frac{6}{4} = 1.5$.

(g) density $\frac{3}{3} = 1$. (h) density $\frac{1}{2} = 0.5$. (i) density $\frac{0}{1} = 0$.

The program returns a python list of the nodes that are in the densest subgraph and the density of the subgraph. Here each step removes the node with the smallest degree. (c) has the highest density so that is the solution.

Each data set is a collection of edge sets on roughly the same node set. These data sets were organized in a set of text files where each file represented an edge set. . Before we could use the graphs, we had to make sure all graphs were undirected. Here all the vertices are represented by integers.

The following matrix is the input for my algorithm,

	0	1	2	3	4	5	6	7	8	9
0	0	1	0	0	0	0	1	0	1	0
1	1	0	0	0	1	0	1	0	0	1
2	0	0	0	0	1	0	1	0	0	0
3	0	0	0	0	1	1	0	0	1	0
4	0	1	1	1	0	1	0	0	0	1
5	0	0	0	1	1	0	0	0	0	0
6	1	1	1	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0	1	1
8	1	0	0	1	0	0	0	1	0	0
9	0	1	0	0	1	0	0	1	0	0

Here the densest subgraph will be the piece of graph connecting the edges ;

0 : [1,6,8] , 1: [0,4,6,9] , 6 :[0,1], 8 : [0,3] , 4 :[1,3,9] , 9 :[1,4] , 3 :[4,8]

The corresponding density of the densest subgraph is 2.5714285714285716

Conclusion and future research :

In this project, I have presented about densest subgraph discovery. Here the problem has been studied in the classical literature in the context of graph partitioning. Moreover a technique has been designed for dense subgraph discovery. So many recent applications are designed for the context of the social, web, communication and biological networks. Those networks have some number of properties, in that they are dynamic and massive in nature. It will lead to a number of very interesting problems for future research:

In large scale applications, the data can be often disk resident. It leads to issues involving efficient processing of underlying network. It happens because it is impossible to perform random access of the edges in the disk resident networks.

In applications such as the web and social networks, the domain of underlying graph may be massive. In many web, telecommunication, biological and social networks, we will may have millions of nodes in the underlying graph and consequently the number of edges may range in the trillions. It leads to storage issues and this is because the number of different edges may not even be possible to store effectively on many desktop machine.

The number of recent applications may lead to streaming scenario in which the edges in the graph incrementally received overtime in a fast speed. Here this is the case in many large social networks and telecommunication. For such cases this may be extremely challenging to analyze the underlying graph within real time to determine dense patterns.

I have discussed about different variations of the densest subgraph problems without size constraints and also have considered hardness issues related to these problems and have developed fast algorithms for them for undirected graphs. All these problems can be generalized to weighted setting, with the same time-complexity or sometimes with only a log |V| increase in the running time. An interesting fact will be to design linear time algorithm with an approximation factor better than 2 for densest subgraph without any size constraint or to improve the approximation factor for DalkS problem. Now, obtaining faster algorithms for densest at least-k1, k2 subgraph problem, or removing the requirement of guessing a in it or in the flow graph construction of maximum density directed

subgraph will also be useful since it will improve the running time significantly

APENDIX

References

https://en.wikipedia.org/wiki/Dense_subgraph

https://ipfs.io/ipfs/QmXoypizjW3WknFiJnKLwHCnL72vedxjQkDDP1mXWo6uco/wiki/Dense_subgraph.html

https://dblp.uni-trier.de/pers/hd/c/Charikar:Moses