An Algorithm for Densest Subgraphs of Vertex-weighted Graphs

Zhongling Liu
School of Computer Science and Educational Software, Guangzhou University
Guangzhou, 510006, China
E-mail: 240922247@qq.com

Wenbin Chen
Department of Computer Science and Guangdong Provincial Engineering Technology Research Center for Mathematical Educational Software, Guangzhou University
Guangzhou, 510006, China
E-mail: cwb2011@gzhu.edu.cn

Fufang Li; Ke Qi; Jianxiong Wang
School of Computer Science and Educational Software, Guangzhou University
Guangzhou, 510006, China
E-mail: lif@gzhu.edu.cn; 526903128@qq.com; xiong.ong@gmail.com

It's a significant problem to find the densest subgraph in many research areas. Now, there are so many groups in the WeChat, QQ and other online chat softwares. In order to find the closely connected subgraphs with the maximum average density for the vertex-weighted graph, we introduce the concept of the densest subgraph, and then an exactly algorithm is presented as an extension of the Goldberg’s algorithm. By theoretical analysis, we prove that our proposed algorithm is correct and it runs in polynomial time. Thus, our proposed algorithm can be used to find the closely connected subgroups with the maximum average density in fact.
1. Introduction

In 1980s, the problem of finding densest subgraph (DS) was presented. Then scientists have proposed some algorithms for solution because it was highly significant in research of network analysis, web communities inferred, hypertext analysis and the function of complex biological networks discovery[9,10,12,15,17]. In 1984, Goldberg firstly proposed the definition of the densest subgraph (DS): \( \hat{G}(V, E) \) is an undirected graph, \( |V| = n, |E| = m \). The density of its subgraph \( \hat{G}(V', E') \) is \( m/n \). DS in \( G \) is a subgraph that its density is the maximum value[11]. So the problem of looking for the DS is to find \( \hat{G}(V', E') \) with the largest density. In this paper, Goldberg gave an algorithm with polynomial time for this problem. It worked as follows. At first, they constructed a network, used the min-cut algorithms to divide the network and finally found the densest subgraph by binary method[11]. In 1989, Gallo et al. gave an algorithm by using max-flow techniques[8]. Then some approximation algorithms were given. For instance, in 1999, Kannan and Vinay proposed an approximation algorithm in directed graphs. In this paper, Kannan and Vinay introduced the density concept for the web graphs[13]. Soon, Charikar gave two algorithms to optimize the finding densest subgraph according to Kannan and Vinay's definition. Charikar firstly gave a linear programming (LP)-based exactly algorithm, then they developed a greedy algorithm with 2-approximation, the algorithm removed the minimum degree vertex by iterated[3].

From the early 1990s, an extensive research of densest subgraph (DS) problem has achieved wide attentions. The problem was to find the densest \( k \)-subgraph (DkS), implying an induced subgraph with the maximum density which has \( k \) vertices. Some approximation algorithms have been given by researchers. For the directed graph, Kortsarz and Peleg designed an algorithm for this problem in 1993. The algorithm has a \( O(n^{0.3885}) \)-ratio approximation[16]. In 1997, an semi-definite programming (SDP) based approximation algorithm was proposed by Feige and Seltser, it had \( n/k \)-radio[6]. Soon after Srivastav and Wolf presented an approximation algorithm based on SDP with the same radio[18]. Four years later, Feige developed another approximation algorithm with \( O(n^{0.7}) \) ratio, when \( \delta < 1/3 \). In 2010, an \( O(n^{1/4}) \)-approximation algorithm was presented by Bhaskara. This algorithm worked in the streaming model, and it was suitable for distributed models[2].

In 2009, in order to find the densest subgraph of size constraint undirected graphs, Andersen and Chellapilla introduced the notions of two problems: the densest at-least-\( k \)-subgraph (DalkS) and the densest at-most-\( k \)-subgraph (DamkS)[1]. For DalkS, they also gave two approximation algorithms, the 3-approximation which extended the algorithm, as proposed by Charikar and 2-approximation on the basis of Gallo’s maximum flow algorithm[8]. In the same year, Khuller and Saha also gave a max-flow algorithm based algorithm with 2-approximation and a LP-based algorithm which had the same radio. They also proved that DalkS was a NP-hard problem[14]. Chen et al. designed two approximation algorithms for DamkS. Both were \( (n−1)/(k−1) \) ratio and \( O(n^\delta) \) time for some \( \delta < 1/3 \) respectively. Chen et al. still gave a polynomial algorithm when \( k \) was constrained by the constant[4]. An improved algorithm was proposed with better time complexity in Chen et al.’s another paper. A greedy approximation algorithm was also proposed to get the densest subgraph of the specified subset[5].

However, for the vertex-weighted graph, how can we get the densest subgraph (DS)? To our knowledge, no one studied this problem. We began to study it in the paper. We firstly introduced a notion of DS of vertex-weighted graphs, and then we proposed an algorithm. And we have proved that this algorithm is polynomial. Our algorithm is an extension of Goldberg's
An Algorithm for Densest Subgraphs of Vertex-weighted Graphs

Zhongling Liu

algorithm[11]. Firstly, we assume a vertex-weighted graph \( \tilde{G} \), \( G' \) is its subgraph. Let \( D \) be the density of \( G' \). We make a estimated value \( g \) of \( D \). Secondly, we try to construct a new graph whose edges' weights relate to \( g \) and divide the graph by using the min-cut algorithms. At last, we find the dsg of \( \tilde{G} \) by a binary search.

2. Definitions

Some definitions are given as follows.

Definition 1: the densest subgraph of undirected graph.

For an undirected graph (UDG) \( \tilde{G}(V',E') \), \( V_x \in V' \). \( \tilde{G}(V,E) \) is an induced subgraph from the vertex \( V_x \). \( D(V) \) is the density of \( \tilde{G}(V,E) \) and defined as \( \frac{|E|}{|V|} \), i.e.

\[
D(V) = \frac{|E|}{|V|} \tag{2.1}
\]

For an undirected graph, its DS problem is a computational problem to find the maximum density as an induced subgraph from \( G \).

Definition 2: the densest subgraph of vertex-weighted UDG.

Give \( \tilde{G} (V', E', \omega') \) is a vertex-weighted UDG, in which, \( \omega' \) is a weighted function, \( \omega': V' \rightarrow N \), \( N \) is a natural number set, \( V_x \in V' \). For the induced subgraph \( \tilde{G}(V,E,\omega) \) from \( V_x \), the density \( D(V'') \) is \( \frac{|E|}{\sum_{\chi \in V} \omega(\chi)} \), i.e.

\[
D(V'') = \frac{|E|}{\sum_{\chi \in V} \omega(\chi)} \tag{2.2}
\]

If all vertices' weight is 1, \( D(V'') \) is the result as the undirected Definition 1.

For a vertex-weighted UDG, the DS problem is equal to find the maximum density induced subgraph.

Definition 3; the minimum cut.

For an UDG \( G(V,E) \), \( \{V_{\lambda} | \lambda \in [1,n]\} \), and \( \{E_{\kappa} | \kappa \in 1,2,...m\} \), in which, \( E_{\kappa} \) has a capacity \( c(E_{\kappa}) \).

1) A cut refers to splitting \( V \) into two sets \( V_1 \) and \( V_2 \). \( O \) is an edge set, \( O = \{(\kappa_1,\kappa_2) \in E, \kappa_1 \in V_1, \kappa_2 \in V_2\} \). \( c(V_1,V_2) \) is the capacity of \( (V_1,V_2) \) equal to adding the edge's capacity in this cut, i.e.

\[
c(V_1,V_2) = \sum_{(\kappa_1,\kappa_2) \in E} c(\kappa_1,\kappa_2), \kappa_1 \in V_1, \kappa_2 \in V_2 \tag{2.3}
\]

2) Given the two vertices \( s,t \in V \), divide \( V \) into two sets. \( s \) and \( t \) belong to two different sets \( M \) and \( N \), so the cut is a s-t cut. i.e. \( \{(M,N) | s \in M, t \in N\} \). Hence,

\[
c(M,N) = \sum_{(\kappa_1,\kappa_2) \in E} c(\kappa_1,\kappa_2), \kappa_1 \in M, \kappa_2 \in N \tag{2.4}
\]

So the min-cut is a s-t cut with the minimum capacity.

3. Algorithm for G

In this section, by extending the Goldberg’s algorithm[11], we propose an algorithm for the DS in a vertex-weighted UDG as follows. At first, we give graph \( G \), then let \( D \) be the maximum density of all subgraphs. We make an estimated value \( g \) of \( D \). Secondly we try to construct a new graph whose edges' weights relate to \( g \) and divide the graph by using the min-cut algorithms. At last, we find the densest subgraph of \( G \) by a binary search.
3.1 Undirected Graph Construction by A Guess Value

Assume that G (V, E, ω) be a vertex-weighted UDG, \( V = \{v_1, \ldots, v_n\} \), in which, ω is a weighted function, ω: V→N, N is a natural number set. \( d(\kappa) \) denotes the vertex \( \kappa \) ’s degree and the weight of \( \kappa \) is \( \omega(\kappa) \). Let D be the density of the induced subgraphs. We make a guess value \( g \) of D, then we construct the graph \( G_g = (V_g, E_g, \omega_g) \) related to \( g \). This is an extension of the Goldberg’s algorithm [11].

We add two vertex r, h of E. For any edge \((\kappa, \varphi)\), the capacity \( c(\kappa, \varphi) \) is 1. The capacity of edge \((r, \kappa)\) is \( m \). The capacity of edge \((\kappa, h)\) is \( m + 2g*\omega(\kappa) - d(\kappa) \).

\[
\begin{align*}
V_g &= V \cup \{r, h\} \\
E_g &= \{(\kappa, \varphi) : \kappa, \varphi \in V \} \cup \{(r, \kappa), \kappa \in V \} \cup \{(\kappa, h), \kappa \in V \} \\
c(\kappa, \varphi) &= \begin{cases} 
1, & \text{if } \kappa, \varphi \in V \\
m + 2g*\omega(\kappa) - d(\kappa), & \text{if } \kappa \in V, \varphi \in \{r, h\} \\
m, & \text{if } \kappa \in \{r, h\}, \varphi \in V \\
0, & \text{otherwise}
\end{cases} \\
c(r, \kappa) &= m, \kappa \in V \\
c(\kappa, h) &= m + 2g*\omega(\kappa) - d(\kappa), \kappa \in V
\end{align*}
\]  

(3.1)

3.2 Relationship Establishment with g and the Minimum Cut

We split the \( V_g \) into two sets M and N. If \( r \in M, h \in N \), cut (M, N) as a s-t cut. Let \( V_1 = M \setminus \{r\}, V_2 = N \setminus \{h\} \). If \( |V_1| = 0 \), \( c(M, N) = m|V_1| \). If \( |V_1| \neq 0 \), \( c(M, N) \) is \( \sum c(\kappa, \varphi), \kappa \in M, \varphi \in N \). Let \( D_1 \) be the density of the induced vertex-weighted subgraph from \( V_1 \), so we have the following conclusion:

\[ c(M, N) = m|V| + 2|V_1|(|g - D_1|) \]  

(3.2)

The reason is as follows:

\[
\begin{align*}
c(M, N) &= \sum_{\kappa \in V_1} c(\kappa, \varphi) \\
&= \sum_{\varphi \in V_1} c(r, \varphi) + \sum_{\kappa \in V_1} c(\kappa, h) + \sum_{\kappa \in V_1, \varphi \in V_2} c(\kappa, \varphi) \\
&= m|V_1| + 2g\sum_{\kappa \in V_1} \omega(\kappa) - \sum_{\kappa \in V_1} d(\kappa) + \sum_{\kappa \in V_1, \varphi \in V_2} 1 \\
&= m|V| + |V_1|2\left(1 - \frac{\sum_{\kappa \in V_1} d(\kappa) - \sum_{\kappa \in V_1, \varphi \in V_2} 1}{2\sum_{\kappa \in V_1} \omega(\kappa)}\right)
\end{align*}
\]

Because \( \frac{1}{2}\left(\sum_{\kappa \in V_1} d(\kappa) - \sum_{\kappa \in V_1, \varphi \in V_2} 1\right) \) is the edge number of G(V_1), so:

\[
D_1 = \frac{\sum_{\kappa \in V_1} d(\kappa) - \sum_{\kappa \in V_1, \varphi \in V_2} 1}{2\sum_{\kappa \in V_1} \omega(\kappa)}
\]  

(3.3)

Hence: \( c(M, N) = m|V| + 2|V_1|(|g - D_1|) \). According to similar argument of Goldberg’s paper, we get the result as follows[11].

Lemma 1: let cut (M, N) be the min-cut. If \( |V_1| \neq 0 \), then \( g \leq D \); if \( V_1 = 0 \), (i.e. M=\{\kappa\}), then \( g \geq D \).

3.3 Algorithm Description

Before describing the algorithm, let's prove the results below.
Lemma 2: given the vertex-weighted UDG $G(V, E, \omega)$, if $G(V', E', \omega')$ is one of the subgraphs, the density of $G(V', E', \omega')$ is $D'$ and the other's is $D''$, if $D' \leq D' + \frac{1}{N(N+1)}$, $N$ is $\sum_{i=1}^{n} \omega(i)$. Then $G(V', E', \omega')$ is the DS of the vertex-weighted UDG.

Proof: let the maximum density of $G(V', E', \omega')$ be $\eta$. $m'$ is edge number, $n'$ is $\omega(\mu)$. $H$ is $m'$ divided by $n'$, $H \in \left[\frac{m'}{n'}, 0 \leq m' \leq m, \min(\omega(\kappa)) \leq n' \leq N, N = \sum_{k=1}^{n} \omega(\kappa)\right]$ (3.4)

$D$ is between 0 and $\frac{m}{\min(\omega(\kappa))}$. If the difference between the two values of $D$ is $\Delta$,

$$\Delta = \frac{m_1 - m_2}{n_1} = \frac{n_2 - n_1}{n_1n_2}$$ (3.5)

If $n_1 = n_2$, then $\Delta \geq \frac{1}{N}$. Otherwise, $n_1n_2 \leq N(N-1)$, then $|\Delta| \geq \frac{1}{N(N-1)}$.

So the minimum difference between the two values of $D$ is $\frac{1}{N(N-1)}$.

Our algorithm is given below.

```plaintext
x ← 0; y ← \frac{m}{\min(\omega(\mu))}; V_1 ← \text{null};
while (y - x \geq \frac{1}{N(N-1)}) {
    g ← \frac{x + y}{2};
    construct G_g = (V_g, E_g, \omega_g);
    find mincut (S, T);
    if (S = \{f\}) y ← g;
    else {
        x ← g;
        V_1 ← S - \{f\};
    }
}
return (denest subgraph of $G$ induced by $V_1$)
```

Let $V_1 \subseteq V'$, $x$ is an estimated lowest value for $D'$, and $y$ is the highest one, so $D' \in [x, y]$; $g$ is an estimated maximum value of $D'$. Lemma 2 tells us that if $y \geq x + \frac{1}{N(N-1)}$, then $G'$ is the densest subgraph. So it is the loop condition for the while loop. In the while loop, we firstly assign the value $\frac{x + y}{2}$ to $g$, then construct an undirected graph $N$, divide $N$ and output the minimum cut for $(M,N)$, then we get the value of $g$ by using Lemma 1. At last, the return is the DS of $G$.

Theorem 3. The time complexity of algorithm is $O(n(n+m)\log^2 n)$.

Proof: for the algorithm, while loop is $O(\log n)$ time. And $K(\varsigma, \eta)$ denotes the time to find the min-cut. The number of $V_g$ is $n+2$ and $E_g$ is $2(m+n)$. In the time complexity, we can ignore

5

---

This text describes an algorithm for finding the densest subgraphs of vertex-weighted undirected graphs. It includes a lemma that defines the conditions under which a subgraph is densest, and a proof using the maximum density and minimum difference to determine the densest subgraph. The algorithm is outlined with a pseudocode example, and a theorem is stated with its proof, indicating the time complexity of the algorithm.
constants to find the min-cut in $O(K(n, n+m)\log n)$ time. Hence, our algorithm time is $O(K(n, n+m)\log n)$. However, the time of the minimum-cut problem is $O(|V||E|\log |V|)$ as introduced by Sleator and Tarjan's algorithm [19]. So our algorithm is polynomial.

4. Conclusion

In this paper, we propose the concept of DS, and then an exactly algorithm is presented as an extension of the Goldberg’s algorithm. By theoretical analysis, we’ve proved that our proposed algorithm is correct and it runs in polynomial time. Thus, our proposed algorithm can be used to find the closely connected subgroups with the maximum average density in fact.

References

An Algorithm for Densest Subgraphs of Vertex-weighted Graphs

Zhongling Liu


