On Finding Dense Subgraphs [*]

Samir Khuller[†]
Department of Computer Science
University of Maryland, College Park
samir@cs.umd.edu

Barna Saha[‡]
Department of Computer Science
University of Massachusetts Amherst
barna@cs.umass.edu

Abstract

Given an undirected graph $G = (V, E)$, the density of a subgraph on 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 nodes in S. Finding subgraphs of maximum density is a very well studied problem. One can also generalize this notion to directed graphs. For a directed graph one notion of density given by Kannan and Vinay is as follows: given subsets S and T of vertices, the density of the subgraph is $d(S,T) = \frac{|E(S,T)|}{\sqrt{|S||T|}}$, where $E(S,T)$ is the set of edges going from S to T. Without any size constraints, a subgraph of maximum density can be found in polynomial time. When we require the subgraph to have a specified size, the problem of finding a maximum density subgraph becomes NP-hard. In this paper we focus on developing fast polynomial time algorithms for several variations of dense subgraph problems for both directed and undirected graphs. When there is no size bound, we extend the flow based technique for obtaining a densest subgraph in directed graphs and also give a linear time 2-approximation algorithm for it. When a size lower bound is specified for both directed and undirected cases, we show that the problem is NP-complete and give fast algorithms to find subgraphs within a factor 2 of the optimum density. We also show that solving the densest subgraph problem with an upper bound on size is as hard as solving the problem with an exact size constraint, within a constant factor.

1 Introduction

Given an undirected graph $G = (V, E)$, the density of a subgraph on 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. The problem of finding a densest subgraph of a given graph G can be solved optimally in polynomial time, despite the fact that there are exponentially many subgraphs to consider [20, 15]. In addition, Charikar [9] showed that we can find a 2 approximation to the densest subgraph problem in linear time using a very simple greedy algorithm (the greedy algorithm was previously studied by [5]). This result is interesting because in many applications of analyzing social networks, web graphs etc., the size of the graph involved could be very large and so having a fast algorithm for finding an approximately dense subgraph is extremely useful. However when there is a specified size constraint - namely find a densest subgraph of exactly k vertices (*DkS*), the densest k subgraph problem becomes NP-hard [11, 4]. When $k = \Theta(|V|)$, Asahiro et al. [5] gave a constant factor approximation algorithm for the *DkS* problem. However for general k, the algorithm developed by Feige, Kortsarz and Peleg [11] achieves an approximation guarantee of $O(n^a)$, where $a < \frac{1}{3}$. The best known bound for this problem is by Bhaskara et al. [7] where an algorithm running in $n^{\frac{1}{\epsilon}}$ time is developed and

[*] A preliminary version of this paper appeared in the proceedings of 36th International Colloquium on Automata, Languages and Programming (ICALP 2009).

[†] Research supported by NSF Awards CCF-0728839 and CCF-0937865.

[‡] Work done while the author was a Ph.D. student at University of Maryland at College Park.

provides an approximation factor to $n^{\frac{1}{4}+\epsilon}$ for any $\epsilon > 0$. However, the technique does not extend to obtain $n^{\frac{1}{4}}$-approximation in polynomial time. [17] showed that there does not exist any PTAS for the *DkS* problem under a reasonable complexity assumption. Recently, assuming stronger complexity assumptions, constant factor approximations have been ruled out [1]. Closing the gap between the approximation factor and the hardness guarantee for *DkS* remains as an outstanding open question.

Two interesting variations of the problem of finding a densest k subgraph was considered by Andersen [3]. The first problem, *the densest at-least-k-subgraph problem* (*DalkS*) asks for an induced subgraph of highest density among all subgraphs with at least k nodes. This relaxation makes *DalkS* significantly easier to approximate and Andersen et al. gave a fast algorithm based on Charikar's greedy algorithm that guarantees a 3 approximation for the *DalkS* problem. In addition, they showed that this problem has a polynomial time 2 approximation, albeit with significantly higher running time. However it was left open as to whether or not this problem is NP-complete. The second problem studied was *the densest at-most-k-subgraph problem* (*DamkS*), that asks for an induced subgraph of the highest density among all the subgraphs with at most k nodes. For the *DamkS* problem, Andersen et al. showed that if there exists an α approximation for *DamkS*, then there is a $\Theta(\alpha^2)$ approximation for the *DkS* problem, indicating that this problem is likely to be quite difficult as well.

For directed graphs, [16] defined a suitable notion of density to detect highly connected subgraphs and provided a $\Theta(\log n)$ approximation algorithm for finding such dense components. Let $G = (V, E)$ be a directed graph and S and T be two subsets of nodes of V. Density corresponding to S and T is defined as $d(S,T) = \frac{|E(S,T)|}{\sqrt{|S||T|}}$, where $E(S,T)$ consists of the edges going from S to T. Charikar showed that the problem can be solved in polynomial time by solving an LP using n^2 different values of a parameter [9]. However a max-flow based technique similar to the one developed by Goldberg [15] for the densest subgraph problem in undirected graphs was not known for directed graphs. It was mentioned as one of the open problems in [9]. In addition to providing a polynomial time solution for the densest subgraph problem in directed graphs, Charikar also gave a 2 approximation algorithm that runs in $O(|V|^3 + |V|^2|E|)$ time.

The densest subgraph problems have received significant attention for detecting important substructures in massive graphs like web and different social networks. In a web graph, hubs (resource lists) and authorities (authoritative pages) on a topic are characterized by large number of links between them [18]. Finding a dense subgraph also acts as a useful primitive for discovering communities in web and social networks, for compressed representation of a graph and for spam detection [10, 8, 14]. [14] provided effective heuristics based on two-level fingerprints for finding large dense subgraphs in massive graphs. Their aim was to incorporate this step into web search engine for link spam control. Dourisboure gave a scalable method for identifying small dense communities in web graph [10]. Buehrer showed how large dense subgraphs can be useful in web graph compression and sub-sampling a graph [8]. In all these applications the underlying graph is massive and thus fast scalable algorithms for detecting dense subgraphs are required to be effective.

Following our work, the *DalkS* problem has received significant attention. Gajewar and Das Sharma studied a generalization of *DalkS* problem where there are multiple groups of vertices and from each group a minimum number of nodes must be selected [12]. The *DkS* and *DalkS* problem, both are studied in the streaming and map reduce model by Bahmani, Kumar and Vassilvitskii [6].

One of the main new insights in this paper is to illustrate the power of the flow based methods [15, 20] to find dense subgraphs not only when there is no requirement on the size of the obtained subgraph, but also for cases when there is a constraint on the size of the obtained subgraph. Precisely our contributions are as follows:

1.1 Contributions

- For the densest subgraph problem without any size restrictions (Section 2):

  - We give a max-flow based polynomial time algorithm for solving the densest subgraph problem in directed graphs.
  - We give a linear time 2-approximation algorithm for the densest subgraph problem in directed graphs.
  - We show that a linear programming relaxation considered by [9] for the densest subgraph problem in undirected graphs has integrality gap 1, and a simpler algorithm exists to obtain the maximum density subgraph from the optimum LP solution.

- For the densest at least k subgraph problem (Section 3):
  - We show that the densest at least k subgraph problem is NP-Hard.
  - For undirected graphs, we give a flow-based and an LP based approximation algorithm, for the densest at least k subgraph problem. These run much faster than the polynomial time approximation algorithm of Andersen and deliver the same worst case approximation factor of 2.
  - We define the notion of the densest at least k_1, k_2 subgraph problem for directed graphs and give a 2-approximation algorithm for it.

- Densest at most k subgraph problem (Section 4):
  - We show that approximating the densest at most k subgraph problem is as hard as the densest k subgraph problem within a constant factor, specifically an α approximation for *DamkS*, implies a 4α approximation for *DkS*.

2 Densest subgraph without any size restriction

In this section, first we give a max-flow based algorithm for the densest subgraph problem in directed graphs. For undirected graphs, Goldberg developed a flow based algorithm that finds a densest subgraph in polynomial time [15]. However for directed graphs no flow based algorithm was known. Next we consider the greedy algorithm for the densest subgraph in undirected graphs proposed by Charikar [9] and develop an extension of this algorithm to give a 2 approximation algorithm for finding a densest subgraph in directed graphs. This improves the running time from $O(|V|^3 + |V|^2|E|)$ to $O(|V| + |E|)$. We also give a very simple proof of 2-approximation for the greedy algorithm developed by [9] to obtain a densest subgraph in undirected graphs.

2.1 Max-flow based algorithm for finding densest subgraphs in directed graphs

For a directed graph $G = (V, E)$, we wish to find two subsets of nodes S and T, such that $d(S,T) = \frac{|E(S,T)|}{\sqrt{|S||T|}}$ is maximized. Let us denote the optimum subsets of nodes by S^* and T^* respectively. To detect such subsets of nodes, we first guess the value of $\frac{|S^*|}{|T^*|}$ in the optimum solution. Since there are $|V|^2$ possible values, in $\Theta(|V|^2)$ time, it is possible to guess this ratio exactly[1]. Let this ratio be a. We create a bipartite graph $G' = (V_1, V_2, E)$, where $V_1 = V_2 = V$ and for every directed edge (i, j) in the original graph, we add an edge from vertex $i \in V_1$ to $j \in V_2$. We now wish to find $S \subseteq V_1$ and $T \subseteq V_2$, such that $\frac{|E(S,T)|}{\sqrt{|S||T|}}$ is maximized. We also know, $\frac{|S^*|}{|T^*|} = a$.

We add a source s and a sink t to $G' = (V_1, V_2, E)$. We guess the value of the optimal (maximum) density. Let the guessed value be g. Note that the optimal density involves computing the square-root. If

[1] If we want a $(1 + \epsilon)$ approximation, only $O(\frac{\log |V|}{\epsilon})$ guessed values suffice.

we round off the values, there will necessarily be a small error introduced in the computation. If we imagine ordering the (distinct) density values of all possible subgraphs, then the lower bound on the gap between two consecutive values is $\Omega(\frac{1}{n^2})$. This enables the binary search to run in polynomial time.

The following edges with weights are then inserted into $G' = (V_1, V_2, E)$:

- We add an edge of weight $2m$ from source s to each vertex of V_1 and V_2, where $m = |E|$.
- We add an edge of weight $(2m + \frac{g}{\sqrt{a}})$ from each vertex of V_1 to the sink t.
- We add an edge from each vertex j of V_2 to sink t of weight $2m + \sqrt{a}g - 2d_j$, where d_j is the in-degree of j.
- All the edges going from V_1 to V_2 are given weight 0. For each edge going from V_1 to V_2, a reverse edge of weight 2 is added.

Now consider a s-t min-cut in this weighted graph. Since the cut $\{s\}, \{t, V_1, V_2\}$ has weight $2m(|V_1| + |V_2|)$, the min-cut value is $\leq 2m(|V_1| + |V_2|)$. Now consider the cut $\{s, S \subseteq V_1, T \subseteq V_2\}$,$\{t, (V_1 \setminus S) \subseteq V_1, (V_2 \setminus T) \subseteq V_2\}$. The number of edges crossing the cut is,

$$\begin{aligned} & 2m(|V_1| - |S| + |V_2| - |T|) + (2m + \frac{g}{\sqrt{a}})|S| + \sum_{i \in T}(2m + \sqrt{a}g - 2d_i) + \sum_{\substack{i \in T, j \in V_1 \setminus S, \\ (j,i) \in E(G)}} 2 \\ = \; & 2m(|V_1| + |V_2|) + |S|\frac{g}{\sqrt{a}} + |T|\sqrt{a}g - 2|E(S,T)| \\ = \; & 2m(|V_1| + |V_2|) + \frac{|S|}{\sqrt{a}}\left(g - \frac{|E(S,T)|}{|S|/\sqrt{a}}\right) + |T|\sqrt{a}\left(g - \frac{|E(S,T)|}{|T|\sqrt{a}}\right) \end{aligned}$$

Let us denote the optimum density value by d_{OPT}. If $g < d_{OPT}$, then there exists S and T (corresponding to the optimum solution), such that both $\left(g - \frac{|E(S,T)|}{|S|/\sqrt{a}}\right)$ and $\left(g - \frac{E(S,T)}{|T|\sqrt{a}}\right)$ are negative. Therefore S and T are nonempty. If the guessed value $g \geq d_{OPT}$, let if possible S and T be non-empty. Let in this returned solution, $\frac{|S|}{|T|} = b$. We have,

$$\begin{aligned} & \frac{|S|}{\sqrt{a}}\left(g - \frac{|E(S,T)|}{|S|/\sqrt{a}}\right) + |T|\sqrt{a}\left(g - \frac{E(S,T)}{|T|\sqrt{a}}\right) \\ = \; & \sqrt{|S||T|}\frac{\sqrt{b}}{\sqrt{a}}\left(g - \frac{d(S,T)}{\sqrt{b}/\sqrt{a}}\right) + \sqrt{|S||T|}\frac{\sqrt{a}}{\sqrt{b}}\left(g - \frac{d(S,T)}{\sqrt{a}/\sqrt{b}}\right) \\ = \; & \sqrt{|S||T|}\left(\left(\frac{\sqrt{b}}{\sqrt{a}} + \frac{\sqrt{a}}{\sqrt{b}}\right)g - 2d(S,T)\right) \qquad (1) \end{aligned}$$

Now, $\left(\frac{\sqrt{b}}{\sqrt{a}} + \frac{\sqrt{a}}{\sqrt{b}}\right) \geq 2 \; \forall$ reals a, b and we have, $g > d_{OPT} \geq d(S,T)$. Hence the value of (1) is > 0. Thus if S and T are non-empty, then this cut has value $> 2m(|V_1| + |V_2|)$. Hence if $g > d_{OPT}$, min-cut is $(\{s\}, \{t, V_1, V_2\})$. If the guessed value $g = d_{OPT}$, then we get a cut of the same cost as the trivial min-cut, even by having S and T corresponding to S^* and T^* respectively. We can always ensure that we obtain a min-cut, which has the biggest size on the source side. Thus when the guessed value is correct, the optimum subsets S and T are obtained from the subsets of vertices of V_1 and V_2 that belong to the side of the cut that contains s. The algorithm detects the correct value of g using a binary search, similar to Goldberg's algorithm for finding a densest subgraph in undirected graphs [15]. Also it is easy to verify that, when the correct value of g is guessed, we have $b = a$. Using a parametric max-flow algorithm [13], the total time required is same as one flow computation within a constant factor.

2.2 2 approximation algorithm for the densest subgraph problem in undirected and directed graphs

We first consider the greedy algorithm for the densest subgraphs in undirected graphs (first discovered by Kortsarz and Peleg [19] and also considered by [9], [5]).

The greedy algorithm at each step chooses a vertex of minimum degree, deletes it and proceeds for $(n-1)$ steps, where $|V| = n.$ At every step the density of the remaining subgraph is calculated and finally the one with maximum density is returned.

Algorithm 2.1: DENSEST-SUBGRAPH($G = (V, E)$)

```
$n \leftarrow |V|, H_n \leftarrow G$
for $i = n$ to $2$
  do { Let $v$ be a vertex in $H_i$ of minimum degree
       $H_{i-1} \leftarrow H_i - \{v\}$
return ($H_j,$ which has the maximum density among $H_i^\prime s, i = 1, 2, .., n$)
```

The above greedy algorithm *Densest-Subgraph* achieves an approximation factor of 2 for undirected networks, which is kind of a folklore result. Here we give a simple proof of it, much simpler than the one provided in [9]. For directed graphs, Charikar developed a different greedy algorithm, that has a significantly high time-complexity of $O(|V|^3 + |V|^2|E|)$. We show that the algorithm *Densest-Subgraph-Directed* which is a generalization of the algorithm *Densest-Subgraph*, detects a subgraph with density within a factor of 2 of the optimum for directed graphs. This reduces the time complexity from $O(|V|^3+|V|^2|E|)$ to $O(|V|+|E|)$.

Theorem 2.1. *The greedy algorithm* **Densest-Subgraph** *achieves a* 2*-approximation for the densest subgraph problem in undirected networks.*

Proof. Let d_{OPT} denote the optimal density. Observe that in an optimal solution, every vertex has degree $\geq d_{OPT}$. Otherwise removing a vertex of degree $< d_{OPT}$, will get a subgraph with higher density. Consider the iteration of the greedy algorithm when the first vertex of the optimum solution is removed. At this stage all the vertices in the remaining subgraph have degree $\geq d_{OPT}$. If the number of vertices in the subgraph is s, then the total number of edges is $\geq d_{OPT}s/2$, and the density is $\geq d_{OPT}/2$. Since the greedy algorithm returns the subgraph with the highest density over all the iterations, it always returns a subgraph with density at least $\frac{1}{2}$ of the optimum. □

One can make examples showing that the bound of 2 is tight for Charikar's algorithm. Let G be the union of two graphs G_1 and G_2. G_1 is a complete bipartite graph having d and D nodes respectively in each partition. G_2 is a disjoint union of D cliques, each of size $d+1$. The density of G_1 is $\frac{dD}{d+D}$ and if we fix d and let D become very large, this approaches d. If we run Charikar's algorithm then the nodes in G_2 have degree d and the nodes in G_1 have degree D or d. If we delete all the noes of degree d in G_1, then the best subgraph we can return will have density $\frac{d}{2}$. However the optimal solution has density approaching d.

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

$$\begin{aligned} \text{maximize} \quad & \sum_{i,j} x_{i,j} \\ & x_{i,j} \le min(y_i, y_j) \ , \forall (i,j) \in E(G) \\ & \sum_{i} y_i = 1 \ , \forall i \in V(G) \\ & x_{i,j}, y_i \ge 0 \ , \forall (i,j) \in E(G), \forall i \in V(G) \end{aligned} \tag{2}$$

Charikar [9] showed that there exists an $i \in \{1, .., |V(G)|\}$ such that the density of the subgraph induced by the vertices with y value at least y_i is equal to the optimal density. Thus by considering each value of $y_i, i = 1, \ldots, |V(G)|$ and checking their density, we can obtain the maximum density subgraph. However here we show that there exists an LP solution where all the y_i values are equal and hence the integrality gap of the above LP (2) is 1. We also show that for any LP optimal solution, picking all the vertices with positive y values returns a maximum density subgraph.

Theorem 2.2. *There exists a solution of the LP (2) such that all the y_i variables are equal. Also for any solution of LP (2), the vertices with non-zero y_i constitute a densest subgraph.*

Proof. Consider an optimal solution of LP (2) that creates the minimum number of different values for y variables. Let these different values be $r_1 > r_2, \ldots > r_s > 0$, $s \ge 2$. Let the number of vertices with corresponding y value equaling to r_i be V_i. Thus, we have $\sum_i V_i r_i = 1$. Since $x_{i,j} = min(y_i, y_j)$, edge variables $x_{i,j}$ can also be classified into s groups. Let the number of edges with value r_i be E_i. These are the edges with end vertices having values at least r_i. We have the value of LP (2) to be $\sum_{i=1}^{s} E_i r_i$. Now modify the solution by setting each vertex with value r_s to 0 and updating the value of r_1 to $r_1^{new} = r_1 + \frac{V_s r_s}{V_1}$. Clearly, $r_1^{new} V_1 + \sum_{i=2}^{s-1} r_i V_i = 1$. Hence we have,

$$\sum_{i=1}^{s} r_i E_i \ge r_1^{new} E_1 + \sum_{i=2}^{s-1} r_i E_i$$

Thus we get,

$$\begin{aligned} & E_s r_s \ge \frac{E_1 V_s r_s}{V_1}, \\ & \text{or, } \frac{E_s}{V_s} \ge \frac{E_1}{V_1} \end{aligned} \tag{3}$$

Now consider another perturbation of the LP solution. Let $\gamma = \min((r_1 - r_2)V_1/V_s, r_{s-1} - r_s)$ and set all the vertices with value r_1 to $r_1^{new} = r_1 - \gamma V_s / V_1$ and update the value of r_s to $r_s^{new} = r_s + \gamma$. Clearly again, $r_1^{new} V_1 + \sum_{i=2}^{s-1} r_i V_i + r_s^{new} V_s = 1$. Thus this perturbed solution is feasible. Hence we have,

$$\sum_{i=1}^{s} r_i E_i \ge r_1^{new} E_1 + \sum_{i=2}^{s-1} r_i E_i + r_s^{new} E_s$$

Thus we get,

$$\begin{aligned} & (r_1 - r_1^{new}) E_1 \ge (r_s^{new} - r_s) E_s \\ & \text{or, } E_1 \gamma V_s / V_1 \ge E_s \gamma \\ & \text{or, } \frac{E_1}{V_1} \ge \frac{E_s}{V_s} \end{aligned} \tag{4}$$

Thus from Equations [(3), (4)], we get $\frac{E_s}{V_s} = \frac{E_1}{V_1}$. Using similar argument, by fixing all but the values of vertices with r_i and r_s, $i \in [2, s-1]$, we get $\frac{E_1}{V_1} = \frac{E_2}{V_2} = \ldots = \frac{E_s}{V_s}$. Let the density of the optimal densest subgraph in G be λ. Then $\frac{E_1}{V_1} \leq \lambda$. Let $\frac{E_1}{V_1} = \frac{E_2}{V_2} = \ldots = \frac{E_s}{V_s} = \lambda' \leq \lambda$. Therefore we have the LP optimal solution, $\sum_{i=1}^{s} E_i r_i = \lambda' \sum_{i=1}^{s} V_i r_i = \lambda' \leq \lambda$. Since we know any optimal solution of LP (2) is at least λ, it must be the case that $\frac{E_1}{V_1} = \frac{E_2}{V_2} = \ldots = \frac{E_s}{V_s} = \lambda$. Thus one possible optimal solution of LP 2 is to set all the vertices counted in V_1 to $\frac{1}{V_1}$ and rest to 0. In fact, for any $i \in [1, s]$, we can consider all the vertices counted in V_j, for $j \in [1, i]$ and set their values to $\frac{1}{\sum_{j=1}^{i} V_j}$. This is true for $j = s$, thus for any LP solution, we can consider all the vertices with nonzero y values and that gives the optimum solution. □

We now consider the case of directed graphs. In a directed graph, for each vertex we count its in-degree and out-degree separately. Let v_i be a vertex with minimum in-degree and v_o be a vertex with minimum out-degree. Then we say v_i has minimum degree, if the in-degree of v_i is at most the out-degree of v_o, else v_o is said to have the minimum degree. In the first case, the vertex with minimum degree belongs to category IN. In the second case, it belongs to category OUT. The greedy algorithm for directed graphs deletes the vertex with minimum degree and then depending on whether it is of category IN or OUT, either deletes all the incoming edges or all the outgoing edges incident on that vertex, respectively. If the vertex becomes a singleton, the vertex is deleted. To compute the density of the remaining graph after an iteration of *Densest-Subgraph-Directed*, any vertex that has nonzero out-degree is counted in the S side and all the vertices with non-zero in-degree are counted in the T side. Therefore the same vertex might appear both in S and T and will be counted once in S and once in T. We denote the optimum solution by (S^*, T^*).

Algorithm 2.2: DENSEST-SUBGRAPH-DIRECTED($G = (V, E)$)

```
$n \leftarrow |V|, H_{2n} \leftarrow G, i \leftarrow 2n$
while $H_i \neq \emptyset$
  do
    Let $v$ be a vertex in $H_i$ of minimum degree
    if category($v$) $= IN$
      then Delete all the incoming edges incident on $v$
      else Delete all the outgoing edges incident on $v$
    if $v$ has no edges incident on it then Delete $v$
    Call the new graph $H_{i-1}, i \leftarrow i - 1$
return ($H_j$ which has the maximum density among $H_i's$)
```

Define $\lambda_i = |E(S^*, T^*)| \left(1 - \sqrt{1 - \frac{1}{|T^*|}}\right)$ and $\lambda_o = |E(S^*, T^*)| \left(1 - \sqrt{1 - \frac{1}{|S^*|}}\right)$.

Lemma 2.3. *In an optimal solution, each vertex in S^*, has out-degree $\geq \lambda_o$ and each vertex in T^* has in-degree $\geq \lambda_i$.*

Proof. The proof is by simply observing that otherwise removing the vertex with minimum degree from either S or T, gives a higher density subgraph. Suppose if possible, $\exists v \in S^*$ with out-degree $< \lambda_o$. Remove v from S^*. The density of the remaining subgraph is $> \frac{E(S^*,T^*)-\lambda_o}{\sqrt{(|S^*|-1)|T^*|}} = d_{OPT}$, which is not possible. Here we get the last equality by plugging in the value of λ_o. Similarly, every vertex $v \in T^*$ has degree $\geq \lambda_i$. □

Theorem 2.4. *The greedy algorithm* **Densest-Subgraph-Directed** *achieves a 2 approximation for the densest subgraph problem in directed networks.*

Proof. Consider the iteration of the greedy algorithm, when the vertices in S have out-degree $\geq \lambda_o$ and the vertices in T have in-degree $\geq \lambda_i$. Let us call the set of vertices on the side of S and T by S'

and T' respectively. Then the number of edges, $E' \geq |S'|\lambda_o$, and also $E' \geq |T'|\lambda_i$. Hence, the density $d(S',T') \geq \sqrt{\frac{|S'|\lambda_o|T'|\lambda_i}{|S'||T'|}} = \sqrt{\lambda_o\lambda_i}$. Substituting the values of λ_o and λ_i from Lemma 2.3, we get $d(S',T')^2 \geq |E(S^*,T^*)|^2 \left(1 - \sqrt{1 - \frac{1}{|S^*|}}\right)\left(1 - \sqrt{1 - \frac{1}{|T^*|}}\right)$. Now putting $|S^*| = \frac{1}{\sin^2\theta}$ and $|T^*| = \frac{1}{\sin^2\alpha}$, we get $d(S',T') \geq \frac{|E(S^*,T^*)|}{\sqrt{|S^*||T^*|}} \frac{\sqrt{(1-cos\theta)(1-cos\alpha)}}{sin\theta sin\alpha} = \frac{d_{OPT}}{2cos\frac{\theta}{2}cos\frac{\alpha}{2}} \geq \frac{d_{OPT}}{2}$. □

3 Densest at least k subgraph problem

For undirected graphs, the *DalkS* algorithm tries to find a subgraph of highest density among all subgraphs, that have size at least k. We prove that the *DalkS* problem is NP-complete. and develop two algorithms; a combinatorial algorithm and one based on solving a linear programming formulation of the *DalkS* problem. Each algorithm achieves an approximation factor of 2. Finally we consider the *DalkS* problem in directed graphs, and give a combinatorial 2-approximation algorithm for the problem. We complement it by showing an LP based approximation for directed *DalkS*; but it achieves a worse approximation ratio of 3.

Theorem 3.1. DalkS *is NP-Hard.*

Proof. We reduce the densest k subgraph problem (this problem is NP-hard [11, 4]) to densest at least k subgraph problem. Suppose we are given a graph G and a value l and we want to know whether a subgraph of size l, with density $\geq \lambda$ exists. We construct a clique G' of size n^2, where $|V(G)| = n$. We then consider the graph formed by the union of G and G' and ask for a subgraph of size at least $n^2 + l$ of highest density. The following properties are satisfied by the solution S returned by DalkS on the union of G and G'.

- $G' \subset S$: Suppose not. Let $G' - S = T \neq \emptyset$. Observe that the density of S is $< (n^2-1)/2$, since the density of any proper subgraph of G' and any subgraph of G is strictly less than $(n^2-1)/2$. Let $|S| = r$ and the density of S be $\lambda(S)$. By adding T to S, we get additional E^T edges (edges incident on nodes in T). $E^T = T(n^2 - \frac{1}{2}(T-1))$. So the new density of $S \cup T$ becomes $\frac{\lambda(S)r+E^T}{r+|T|}$. We claim that this is $> \lambda(S)$. To prove this we need to show that $\lambda(S)T < E^T$. This is easy to verify by the earlier observation on $\lambda(S)$. We thus obtain $\frac{\lambda(S)r+E^T}{r+|T|} > \frac{\lambda(S)r+\lambda(S)|T|}{r+|T|} > \lambda(S)$. Therefore G' must be entirely contained in S.

- $|S \cap G| = l$: Let us denote by, $S_G = S \cap G$. Since $|S| \geq n^2 + l$, $|S_G| \geq l$. If possible let $|S_G| = l' > l$. Let the density of S_G be $\lambda(S_G)$. Therefore the density of S is

$$\begin{aligned} \frac{\lambda(S_G) * l' + E(G')}{l' + V(G')} &\leq \frac{l'(l'-1)/2 + E(G)}{l' + V(G')} \\ &\leq \frac{1 + E(G)}{l' - 1 + V(G')} \end{aligned} \tag{5}$$

  Hence the density of S is a decreasing function of $|S_G|$. Thus we must have $|S_G| = l$.

It follows from the above two properties that S_G is the densest subgraph of size l in G, since otherwise we can replace the size l subgraph of S_G with the densest l subgraph of G and get a better solution. Therefore if the solution returned by *DalkS* has density $\geq \frac{\binom{n^2}{2}+\lambda l}{n^2+l}$, then the answer is yes, otherwise the answer is no. □

We develop two algorithms for *DalkS* that both achieve an approximation factor of 2. We note that [2] proposed a 2 approximation algorithm, that requires n^3 max-flow computations. Even using the parametric flow computation [13] the running time is within a constant factor of n^2 flow computations. Whereas our first algorithm uses at most $max(1, (k - \gamma))$ flow computations using parametric flow algorithm and in general much less than that. Here γ is the size of the densest subgraph without any size constraint. The second algorithm is based on a linear programming formulation for *DalkS* and requires only a single solution of a LP.

3.1 Algorithm 1: Densest at least k subgraph

Let H^* denote the optimum subgraph and let d^* be the optimum density. The algorithm starts with the original graph G as G_0, and D_0 as $\emptyset$. In the ith iteration, the algorithm finds the densest subgraph H_i from G_{i-1} without any size constraint. If $|V(D_{i-1})| + |V(H_i)| \geq k$, the algorithm stops. Otherwise the algorithm adds H_i to D_{i-1} to obtain D_i. All the edges and the vertices of H_i are removed from G_{i-1}. For every vertex $v \in G_{i-1} \setminus H_i$, if v has l edges to the vertices in H_i, then in G_i a self loop of weight l is added to v. The algorithm then continues with G_i. When the algorithm stops, each subgraph D_i is padded with arbitrary vertices to make their size k. The algorithm then returns the D_j with maximum density.

Algorithm 3.1: DENSEST AT LEAST-K(G, k)

```
$D_0 \leftarrow \emptyset, G_0 \leftarrow G, i \leftarrow 1$
while $|V(D_i)| < k$
    do { $H_i \leftarrow$ maximum-density-subgraph($G_{i-1}$)
         $D_i \leftarrow D_{i-1} \cup H_i$
         $G_i =$ shrink($G_{i-1}, H_i$), $i \leftarrow i + 1$
for each $D_i$
    do Add an arbitrary set of $max(k - |V(D_i)|, 0)$ vertices to it to form $D'_i$
return ($D'_j$, which has the maximum density among the $D'_is$)
```

We prove that algorithm *Densest At least-k* achieves an approximation factor of 2.

Theorem 3.2. *The algorithm* **Densest At least-k** *achieves an approximation factor of* 2 *for the* DalkS *problem.*

Proof. If the number of iterations is 1, then H_1 is the maximum density subgraph of the original graph whose size is $\geq k$. Therefore $H^* = H_1$ and the algorithm returns it. Otherwise, say the algorithm iterates for $l \geq 2$ rounds. There can be two cases:

Case 1: *There exists a $l' < l$ such that $E(D_{l'-1}) \cap E(H^*) \leq \frac{E(H^*)}{2}$ and $E(D_{l'}) \cap E(H^*) \geq \frac{E(H^*)}{2}$.*

Case 2: *There exists no such $l' \leq l$.*

For case 2, we have for any $j \leq l - 1$, $E(D_j) \cap E(H^*) \leq \frac{E(H^*)}{2}$. Therefore, $E(G_j) \cap E(H^*) \geq \frac{E(H^*)}{2}$. Consider $V' = V(G_j) \cap V(H^*)$. The density of the subgraph induced by V' in G_j is $\geq \frac{E(G_j) \cap E(H^*)}{|V'|} \geq \frac{E(H^*)}{2V(H^*)} = d^*/2$. Hence the density of H_l must be $\geq d^*/2$. So in case 1, for each $j \leq l$, the density of H_j is $\geq d^*/2$. Therefore the total number of edges in the subgraph D_l is $\geq \frac{d^* \sum_{j=1}^{l'} |V(H_j)|}{2}$, or the density of $D_{l'}$ is $\geq d^*/2$ and it has $\geq k$ vertices.

For case 1, the subgraph $D_{l'}$ has at least $E(H^*)/2$ edges and since $V(D_{l'}) \leq k$, the density of $D'_{l'}$ is $\geq \frac{d^*}{2}$.

Since the algorithm returns the subgraph D'_j with maximum density among all the D'_is, the returned subgraph has density at least $d^*/2$. □

We now give an example where Algorithm 1 of *DalkS* achieves the worst case approximation ratio of 2. Let $G = H_1 \cup H_2 \cup H_3 \cup H_4$, where H_1, H_2, H_3, H_4 are disjoint. H_1 is a clique of size $\sqrt{2v}$, H_2 is a tree on

v vertices, H_3 is a cycle on v^2 vertices and H_4 are v disjoint vertices. We have $\sqrt{2v} + 2v + v^2 = n$. Density of H_1 is $\approx \sqrt{2v}/2$ and it is the densest subgraph of G. we have $E(H_1) \approx v$. We also have density of H_2 is $1 - 1/v$ and density of H_3 is 1. The optimum densest subgraph of size at least $v + \sqrt{2v}$ is $H_1 \cup H_2$. Its density is $\frac{E(H_1)+E(H_2)}{|H_1|+|H_2|} \approx 2v/(v + \sqrt{2v}) \approx 2$. The algorithm will find H_1 in the first iteration. In the second iteration it will find H_3. So the algorithm has the option of returning $H_1 \cup H_3$ or append arbitrary v vertices to H_1 to satisfy the size requirement. In the first case, the density is $\frac{E(H_1)+E(H_3)}{|H_1|+|H_3|} \approx v + v^2/(\sqrt{2v} + v^2) \approx 1$. In the second case if the arbitrary vertices chosen are from H_4, we get a density of $\frac{E(H_1)}{\sqrt{2v}+v} \approx 1$.

3.2 Algorithm 2: Densest at least k subgraph

Next we give a LP based solution for the *DalkS* problem. Define a variable $x_{i,j}$ for every edge $(i,j) \in E(G)$ and a variable y_i for every vertex $i \in V(G)$. Consider now the following LP:

$$\text{maximize} \quad \sum_{i,j} x_{i,j} \tag{6}$$
$$x_{i,j} \le y_i \ , \forall (i,j) \in E(G); \ \ x_{i,j} \le y_j \ , \forall (i,j) \in E(G)$$
$$\sum_i y_i = 1; \ \ y_i \le \frac{1}{l} \ , \forall i \in V(G); \ \ x_{i,j}, y_i \ge 0 \ , \forall (i,j) \in E(G), \forall i \in V(G)$$

Here $l \ge k$ is the size of the optimal solution of the *DalkS* problem. Since there can be $n - k + 1$ possible sizes of the optimal solution, we can guess this value, plugging in different values of l. In Section 3.3 we show that by first running the algorithm *Densest-Subgraph* and then solving one single LP, we can guarantee a 2-approximation.

Lemma 3.3. *The optimal solution of LP (6) is greater than or equal to the optimal value of* DalkS.

Proof. Let the optimal solution for *DalkS* be obtained for a subgraph H having $l \ge k$ vertices and density λ. Consider a solution for the above LP, where each of the variables y_i corresponding to the vertices of H have value $\frac{1}{l}$. All the variable $x_{i,j}$ corresponding to the induced edges of H have value $\frac{1}{l}$. The solution is feasible, since it satisfies all the constraints of LP (6). The value of the objective function of the LP is $\sum_{(i,j) \in H} x_{i,j} = \frac{E(H)}{l} = \frac{E(H)}{V(H)} = \lambda$. Therefore the optimal value of the LP is $\ge \lambda$ □

Lemma 3.4. *If the value of an optimal solution of LP (6) is λ, then a subgraph of size $\ge k$ with density $\ge \lambda/2$ can be constructed from that solution of LP (6).*

Proof. Define $S(r) = \{i | y_i \ge r\}$ and $E(r) = \{(i,j) | x_{i,j} \ge r\}$. As observed in [9], $E(r)$ is the set of edges induced by the subgraph $S(r)$. This follows from the fact that $x_{i,j} = min(y_i, y_j)$. Hence if $x_{i,j} \in E(r)$, then both y_i and y_j are in $S(r)$. On the other hand if y_i and y_j are in $S(r)$, then $x_{i,j} \in E(r)$. Now $\int_{r=0}^{1/l} |S(r)| dr = \sum_{i \in V(G)} y_i = 1$. Also $\int_{r=0}^{1/l} |E(r)| dr = \sum_{(i,j) \in E(G)} x_{i,j} \ge \lambda$. Consider $E(\delta)$, where δ is the smallest step by which $x_{i,j}$ increases. Let H^* denote the optimal solution for *DalkS*. Then it must hold that $|E(\delta)| \ge |E(H^*)|$. Otherwise since $E(r') \subseteq E(r)$, $\forall r' \ge r$, $|E(r)| \le |E(\delta)|$. Therefore we have, $\int_{r=0}^{1/l} |E(r)| dr \le |E(H^*)| \int_{r=0}^{1/l} dr = \frac{|E(H^*)|}{l} = \lambda$.

So, if $|S(\delta)| = l$ ($|S(\delta)|$ is always $\ge l$), then since $|E(\delta)| \ge |E(H^*)|$, we get the optimal solution by considering the induced subgraph $S(\delta)$.

Let r be the minimum index, such that $|S(r)| = l$. If any one of the following two cases holds, we get a 2 approximation:

Case 1: *For $r' < r$, $E(r')/S(r') \ge \frac{\lambda}{2}$.* It is obvious that in this case we get a 2 approximation.

Case 2: *For $r' \geq r$, $E(r') \geq E(H^*)/2$.* In this case we can add arbitrary vertices to the subgraph induced by the vertices in $S(r')$, to make its size l, since $|S(r')| \leq |S(r)| = l$.

Case 3: *Neither case 1 nor case 2 holds.* We show by contradiction, that case 3 cannot occur. If possible, let us assume that case 3 occurs. Then $|E(r)| < \frac{E(H^*)}{2}$. Since $E(r)$ is a decreasing function of r, $\forall r' \geq r, |E(r)| < \frac{|E(H^*)|}{2}$. Hence we have,

$$\int_{x=r}^{1/l} |E(x)|dx < \frac{|E(H^*)|}{2} \int_{x=r}^{1/l} dr = \frac{|E(H^*)|}{2} \left(\frac{1}{l} - r \right) \tag{7}$$

Also we have $\forall r' < r$, $\frac{|E(r')|}{|S(r')|} < \frac{\lambda}{2}$. Hence we get,

$$\int_{x=0}^{r-\delta} |E(x)|dx \quad \leq \quad \frac{\lambda}{2} \int_{x=0}^{r-\delta} |S(x)|dx \leq \frac{\lambda}{2} \int_{x=0}^{1/l} |S(x)|dx = \frac{\lambda}{2} = \frac{E(H^*)}{2l} \tag{8}$$

Therefore we get,

$$\int_{x=0}^{x=1/l} |E(x)|dx \quad = \quad \int_{x=0}^{r-\delta} |E(x)|dx + \int_{x=r}^{1/l} |E(x)|dx = \frac{|E(H^*)|}{l} - \frac{|E(H^*)|r}{2} < \lambda$$

Here we get the second equality by adding Equation (7) and (8) and the last inequality by noting $r > 0$. But we have $\int_{r=0}^{1/l} |E(r)|dr = \sum_{(i,j) \in E(G)} x_{i,j} \geq \lambda$. Hence we arrive at a contradiction. □

Theorem 3.5. *If the value of an optimal solution of LP (6) is λ, a subset S of vertices can be computed from that solution such that*

$$d(G(S)) \geq \frac{\lambda}{2} \text{ and } |S| \geq k$$

Proof. Consider every possible subgraph by setting $r = y_i$ for all distinct values of y_i. By Lemma 3.4, there exists a value of r such that $|S(r)| \geq k$ and $\frac{|E(r)|}{|S(r)|} \geq v/2$, where v is an optimal solution of the LP. By Lemma 3.3 , $v \geq \lambda$ and hence the proof. □

3.3 Reducing the number of LP solutions

To reduce the number of LP solutions, we first run the algorithm *Densest-Subgraph*, consider the solutions over all the iterations that have more than k vertices and obtain the one with maximum density. We call this modified algorithm *Densest-Subgraph*$_{>k}$. We compare the obtained subgraph from *Densest-Subgraph*$_{>k}$ with the solution returned by the LP based algorithm with $l = k$. The final solution is the one which has the higher density.

When the optimal solution for *DalkS* has exactly k vertices, Theorem 3.5 guarantees that we obtain a 2 approximation. Otherwise, the optimum subgraph has size $> k$. In this situation, the following lemma shows that the solution returned by *Densest-Subgraph*$_{>k}$ has density at least $\frac{1}{2}$ of the optimal solution of *DalkS*. Therefore using only a single solution of LP (6) along with the linear time algorithm *Densest-Subgraph*$_{>k}$, we can guarantee a solution for *DalkS*within a factor of 2 of the optimum.

Lemma 3.6. *If the optimum subgraph of* DalkS *problem has size $> k$, then* **Densest-Subgraph**$_{>k}$ *returns a* 2 *approximate solution.*

Proof. Let the optimal density be λ. Since the size of the optimal solution is $> k$, if there exists any vertex in the optimum solution with degree $< \lambda$, then removing that we would get higher density and the size of the subgraph still remains $\geq k$. Hence all the vertices in the optimum solution has degree $\geq \lambda$. Now from Theorem 2.1, we get the required claim. □

3.4 Integrality Gap of LP (6) for *DalkS* and the worst case example for Algorithm 2

Let a, b, c be three positive integers, values to be fixed later. Consider a graph G which is a union of a complete graph L and a random graph H. So $G = L \bigcup H$, L is a complete graph of size an and H is a graph on bn vertices with each edge existing with probability $p < 1$. Therefore the expected number of edges of H is $\frac{bn(bn-1)p}{2}$ edges. We pick a H which has exactly $\frac{bn(bn-1)p}{2}$ edges. Set $k = (a + c)n$ and let $a < c < b$. Let us now compute the optimum integral solution of *DalkS*in G.

We claim that the optimum integral solution is $L \cup H'$ where $H' \subset H$ with exactly cn vertices when $p \leq \frac{a^2(b-c)}{b^2(a+c)-c^2(a+b)}$ and $b \leq c + 2a$. Suppose not and let if possible the optimum consists of some dn vertices of H and $rn < an$ vertices of L. Then by removing any $(a - r)$ vertices of H and adding the remaining $(a - r)$ vertices of L, we get more edges. Therefore the optimum solution must contain the entire L. Now let if possible d be greater than c. Let H'' be the subgraph of H that appears in the optimum solution and has size dn. Consider the cn vertices of H that has the highest density and call it H_{cn}. We have density of $L \bigcup H_{cn}$ is $d_1 \approx \frac{a^2n+c^2np}{2(a+c)}$ and the density of $L \bigcup H''$ is $d_2 \leq \frac{a^2n+d^2np}{2(a+d)}$. Now plugging back the value of p and noting that $b \leq c + 2a$, we get $d_1 > d_2$ and hence a contradiction.

Now consider an LP feasible solution. Assign each vertex of L a value of $\frac{1}{(a+c)n}$. Therefore the total contribution of these vertices is $\frac{a}{a+c}$. Assign each vertex of H a value of $\frac{1-\frac{a}{a+c}}{bn} = \frac{c}{(a+c)bn}$. Hence the objective of the LP solution has a value of at least

$$\begin{aligned} &\approx \frac{1}{(a+c)n}\frac{a^2n^2}{2} + \frac{c}{(a+c)bn}\frac{b^2n^2p}{2} \\ &= \frac{a^2n + cbnp}{2(a+c)}. \end{aligned}$$

Thus the integrality gap is at least $\frac{a^2+cbp}{a^+c^2p}$, where $p = \frac{a^2(b-c)}{b^2(a+c)-c^2(a+b)}$ and $b \leq c + 2a$. Setting $a = 1, c = 1, b = 3$, we get $p = \frac{1}{7}$ and the integrality gap is at least $\frac{1+3/7}{1+1/7} = \frac{5}{4}$.

However using the same worst case example as in Algorithm 1 for *DalkS*, it can be shown that our specific algorithm cannot do better than $\frac{1}{2}$ approximation. We leave open the question of designing a new rounding technique to achieve a better approximation than 2 or improving the integrality gap of $5/4$ of LP (6).

3.5 Densest at least k subgraph problem for directed graphs

Given a directed graph $G = (V, E)$ and integers k_1, k_2, the densest at least k directed subgraph (*DaLkDS*) *problem finds two subsets of nodes S and T containing at least k_1 and k_2 vertices respectively for which $\frac{E(S,T)}{\sqrt{|S||T|}}$ is maximized.*

In this section we give a 2 approximation algorithm for the *DaLkDS* problem. Since there are two parameters, k_1, k_2; we refer to this problem by *densest at least-k_1, k_2 problem* from now on.

3.6 Densest at least k_1, k_2 directed subgraph problem

Let S^*, T^* represent the optimum solution of *DaLkDS* and d^* represent the value of the density corresponding to S^*, T^*. Let the ratio $\frac{|S^*|}{|T^*|}$ be a. Since the possible values of a can be $\frac{i}{j}$, where $i \geq k_1, j \geq k_2$ and $i, j \leq |V|$, we can guess the value of a. We run the max-flow based algorithm of Section 2.1 (maximum-directed-density-subgraph) with the chosen a to obtain the densest directed subgraph without any size constraints. Instead of shrinking and removing the vertices and the edges in the densest directed subgraph, as in algorithm *Densest At least-k* for *DalkS*, we only remove the edges and maintain the vertices. We continue this procedure for the same choice of a, until at some round both the sizes of S and T thus obtained exceed k_1 and k_2 respectively. Let S_i and T_i be the partial subsets of vertices obtained up to the ith round. We append arbitrary vertices A and B to S_i and T_i to form S'_i and T'_i respectively, such that $|S'_i| \geq k_1$ and $|T'_i| \geq k_2$. The algorithm returns S'_j, T'_j, such that the density $d(S'_j, T'_j)$ is maximum over all the iterations.

Algorithm 3.2: DENSEST AT LEAST-$k_1, k_2(G, k_1, k_2, a)$

```
$S_0 \leftarrow \emptyset, T_0 \leftarrow \emptyset, G_0 \leftarrow G, i \leftarrow 1$
while $|S_{i-1}| < k_1$ or $|T_{i-1}| < k_2$
    do { $H_i(S,T) \leftarrow$ maximum-directed-density-subgraph$(G_{i-1}, a)$
         $S_i \leftarrow S_{i-1} \cup H_i(S)$
         $T_i \leftarrow T_{i-1} \cup H_i(T)$
         $G_i =$ shrink$(G_{i-1}, H_i), i \leftarrow i + 1$
for each $S_i, T_i$
    do { Add arbitrary $max(k_1 - |S_i|, 0)$ vertices to $S_i$ to form $S'_i$
         Add arbitrary $max(k_2 - |T_i|, 0)$ vertices to $T_i$ to form $T'_i$
return $(S'_j, T'_j)$ which has maximum density among the $(S'_i, T'_i)s$
```

Theorem 3.7. *Algorithm* **Densest At least-**k_1, k_2 *achieves an approximation factor of* 2 *for the* DaLkDS *problem.*

Proof. For a chosen a, algorithm *Densest At least-*k_1, k_2 returns subsets $H_i(S)$ and $H_i(T)$ at iteration i, such that $\frac{|H_i(S)|}{|H_i(T)|} = a$. Suppose up to l_1th iteration, $|S_{l_1}| < k_1$ and $|T_{l_1}| < k_2$. Let $|S_{l_1+1}| \geq k_1$, but up to l_2th iteration, $|T_{l_2}| < k_2$. At iteration $l_2 + 1$, $|T_{l_2+1}| \geq k_2$. Now we consider the following cases,

Case 1: $|E(S_{l_1}, T_{l_1})| \geq |E(S^*, T^*)|/2$.

Case 2: $|E(S_{l_2}, T_{l_2}) \bigcap E(S^*, T^*)| \leq |E(S^*, T^*)|/2$.

Case 3: $\exists l', l_1 < l' \leq l_2$, *such that* $|E(S_{l'}, T_{l'})| > |E(S^*, T^*)|/2$ *and* $|E(S_{l'-1}, T_{l'-1})| \leq |E(S^*, T^*)|/2$.

These three cases are mutually exclusive and exhaustive. When case 1 occurs, we can append arbitrary vertices to S_{l_1} and T_{l_1} to make their sizes respectively k_1 and k_2. In that case $\frac{E(S'_{l_1}, T'_{l_1})}{\sqrt{|S'_{l_1}||T'_{l_1}|}} \geq \frac{E(S^*, T^*)}{2\sqrt{|S^*||T^*|}} = d^*/2$. When case 2 occurs, at iteration l_2 at least half of the edges of the optimum are still not covered. Since no vertices are ever deleted, choice of S^* and T^* maintains the ratio a and returns a density that is at least $\frac{d^*}{2}$. Then

$$\forall i = 1, 2, .., l_2, \frac{E(H_i(S), H_i(T))}{\sqrt{|H_i(S)||H_i(T)|}} \geq \frac{d^*}{2};$$

which implies

$$E(H_i(S), H_i(T)) \geq \sqrt{|H_i(S)||H_i(T)|}\frac{d^*}{2} = \sqrt{a}|H_i(T)|\frac{d^*}{2}.$$

Hence by summing over the iterations 1 to $l_2 + 1$ we get,

$$\begin{aligned} E(S_{l_2+1}, T_{l_2+1}) \quad & \geq \sqrt{a}\sum_{i=1}^{l_2+1}|H_i(T)|\frac{d^*}{2} \\ & \geq \frac{d^*}{2}\sqrt{a}|T_{l_2+1}| \\ = \quad & \frac{d^*}{2}\sqrt{|T_{l_2+1}||S_{l_2+1}|}. \end{aligned}$$

Hence we have,

$$\frac{E(S'_{l_2+1}, T'_{l_2+1})}{\sqrt{|S'_{l_2+1}||T'_{l_2+1}|}} = \frac{E(S_{l_2+1}, T_{l_2+1})}{\sqrt{|S_{l_2+1}||T_{l_2+1}|}} \geq \frac{d^*}{2}.$$

When case 3 occurs, again

$$\forall i = 1, 2, .., l', \frac{E(H_i(S), H_i(T))}{\sqrt{|H_i(S)||H_i(T)|}} \geq \frac{d^*}{2}.$$

Now following an analysis identical to case 2 we get,

$$\frac{E(S_{l'}, T_{l'})}{\sqrt{|S_{l'}||T_{l'}|}} \geq \frac{d^*}{2}.$$

Since $|S'_{l'}| = |S_{l'}|$ might be much larger than k_1, an analysis similar to case 1 cannot guarantee a 2-approximation. Let $X_1 = |\bigcup_{i=1}^{l'} H_i(S)| = |S_{l'}|$ and $X_2 = \sum_{i=1}^{l'} |H_i(S)|$. Similarly $Y_1 = |\bigcup_{i=1}^{l'} H_i(T)| = |T_{l'}|$ and $Y_2 = \sum_{i=1}^{l'} |H_i(T)|$. We have

$$Y_2 = \frac{X_2}{a} \geq \frac{X_1}{a} \geq \frac{k_1}{a} = k_2.$$

We also have,

$$\frac{E(X_1, Y_1)}{\sqrt{|X_2||Y_2|}} \geq \frac{d^*}{2}.$$

Therefore,

$$\frac{E(X_1, Y_1)}{\sqrt{|S'_{l'}|}} = \frac{E(X_1, Y_1)}{\sqrt{|X_1|}} \geq \frac{E(X_1, Y_1)}{\sqrt{|X_2|}} \geq \sqrt{Y_2}\frac{d^*}{2} \geq \sqrt{k_2}\frac{d^*}{2}.$$

We add arbitrary vertices to Y_1 to make its size equal to k_2. Hence,

$$\frac{E(X_1, Y_1)}{\sqrt{|S'_{l'}|}} \geq \sqrt{|T'_{l'}|}\frac{d^*}{2}.$$

Also

$$\frac{E(X_1, Y_1)}{\sqrt{|T'_{l'}|}} = \frac{E(X_1, Y_1)}{\sqrt{k_2}} \geq \frac{E(X_1, Y_1)}{\sqrt{|Y_2|}} \geq \sqrt{|X_2|}\frac{d^*}{2} \geq \sqrt{|S'_{l'}|}\frac{d^*}{2}.$$

Multiplying we get,

$$\frac{E(S'_{l'}, T'_{l'})^2}{\sqrt{|S'_{l'}||T'_{l'}|}} \geq \sqrt{|S'_{l'}||T'_{l'}|}\frac{d^{*2}}{4},$$

or we have,

$$\frac{E(S'_{l'}, T'_{l'})}{\sqrt{|S'_{l'}||T'_{l'}|}} \geq \frac{d^*}{2}$$

. □

For a particular value of a, using parametric max-flow algorithm *Densest At least*-k_1, k_2 requires time of a single flow computation within a constant factor. However there are $|V|^2$ possible choices of a. Improving the time complexity for this variant is open. Next we present an LP based algorithm for densest at least k_1, k_2 directed subgraph problem. It achieves an approximation factor of 3, worse than 2 that is obtained here. But we don't know whether the integrality gap is tight and thus an LP based approach is a viable one for future improvements.

3.7 Algorithm 2: Densest at least k_1, k_2 directed subgraph problem

Define a variable $x_{i,j}$ for every edge $(i,j) \in E(G)$, variables s_i and t_j for every vertex $i \in S = V(G)$ and $j \in T = V(G)$. We guess the value of $c = \frac{|S_{OPT}|}{|T_{OPT}|}$ and solve the following LP for each value of c.

$$
\begin{aligned}
\text{maximize} \quad & \sum_{i,j} x_{i,j} && (9)\\
& x_{i,j} \leq s_i \ , \forall (i,j) \in E(G)\\
& x_{i,j} \leq t_j \ , \forall (i,j) \in E(G)\\
& \sum_i s_i = \sqrt{c} \ , \forall i \in S\\
& \sum_j t_j = \frac{1}{\sqrt{c}} \ , \forall j \in T\\
& s_i \leq \frac{\sqrt{c}}{l_1} \ , \forall i \in S\\
& t_j \leq \frac{1}{l_2\sqrt{c}} \ , \forall j \in T\\
& x_{i,j}, s_i, t_j \geq 0 \ , \forall (i,j) \in E(G), \forall i \in S, j \in T
\end{aligned}
$$

Here $l_1 \geq k_1$ and $l_2 \geq k_2$ are the size of S_{OPT} and T_{OPT} respectively. Since there can be $n - k_1 + 1$ possible choices for S_{OPT} and similarly $n - k_2 + 1$ choices for T_{OPT}, we can guess these values as well. Denote by λ an optmal solution for LP (9) and by d_{OPT} an optimal value of *DaLkDS*.

Lemma 3.8. $\lambda_{OPT} \geq d_{OPT}$.

Proof. Let $\frac{|S_{OPT}|}{|T_{OPT}|} = \frac{l_1}{l_2} c$. Consider a solution for LP (9), where each of the variables s_i for $i \in S_{OPT}$ have value $\frac{\sqrt{c}}{l_1}$ and each of the variables t_j, for $j \in T_{OPT}$ have value $\frac{1}{l_2\sqrt{c}}$. Since, $l_1 = l_2 * c$, $s_i = \frac{\sqrt{c}}{l_1} = \frac{1}{\sqrt{l_1 l_2}} = \frac{1}{l_2\sqrt{c}} = t_j$ for all $i \in S_{OPT}$ and $j \in T_{OPT}$. Thus, we can set the variable $x_{i,j} = s_i = t_j$ for all edges $(i,j) \in S_{OPT} \times T_{OPT}$. The solution is feasible, since it satisfies all the constraints of LP (9). The value of the objective function is $\sum_{(i,j)\in S_{OPT}\times T_{OPT}} x_{i,j} = \frac{E(S_{OPT},T_{OPT})}{\sqrt{l_1 l_2}} = \frac{E(S_{OPT},T_{OPT})}{\sqrt{|S_{OPT}||T_{OPT}|}} = d_{OPT}$. Therefore the optimal value of the LP, $\lambda \geq d_{OPT}$ □

Theorem 3.9. *Subsets S', T' of vertices can be computed from an optimal solution of LP (9), such that*

$$d(S', T') \geq \frac{\lambda}{3};$$

$$|S'| \geq k_1 \textit{and} |T'| \geq k_2$$

Proof. Consider an optimal solution of LP (9) where the guessed values of c, l_1, l_2 are correct. Therefore, $c = \frac{l_1}{l_2}$ and each $s_i, t_j \leq \sqrt{l_1 l_2}$. Define $S(r) = \{i | s_i \geq r\}, T(r) = \{j | t_j \geq r\}$ and $E(r) = \{(i,j) | x_{i,j} \geq r\}$. It can be easily seen that $E(r)$ is the set of edges that goes from $S(r)$ to $T(r)$. Now

$$\int_{r=0}^{\sqrt{l_1 l_2}} |S(r)| dr = \sum_{i \in V(G)} s_i = \sqrt{c}$$

and

$$\int_{r=0}^{\sqrt{l_1 l_2}} |T(r)| dr = \sum_{j \in V(G)} t_j = 1/\sqrt{c}.$$

Hence by Cauchy-Schwarz inequality,

$$\int_{r=0}^{\sqrt{l_1 l_2}} \sqrt{|S(r)||T(r)|}dr \leq \sqrt{\int_{r=0}^{\sqrt{l_1 l_2}} |S(r)|dr \int_{r=0}^{\sqrt{l_1 l_2}} |T(r)|dr} \leq 1.$$

Also

$$\int_{r=0}^{\sqrt{l_1 l_2}} |E(r)|dr = \sum_{(i,j)\in E(S,T)} x_{i,j} \geq \lambda.$$

Consider $E(\delta)$, where δ is the smallest step by which $x_{i,j}$ increases. It must hold that $|E(\delta)| \geq |E(S_{OPT}, T_{OPT})|$. Otherwise since $E(r') \subseteq E(r), \forall r' \geq r$, we have $\forall r, |E(r)| \leq |E(\delta)|$. Therefore

$$\int_{r=0}^{\sqrt{l_1 l_2}} |E(r)|dr \leq |E(S_{OPT}, T_{OPT})| \int_{r=0}^{\sqrt{l_1 l_2}} dr = \frac{|E(H^*)|}{\sqrt{l_1 l_2}} = \lambda.$$

So, if $|S(\delta)| = l_1, |T(\delta)| = l_2$ ($|S(\delta)|$ is always $\geq l_1$ and similarly $|T(\delta)|$ is always $\geq l_2$), then since $|E(\delta)| \geq |E(S_{OPT}, T_{OPT})|$, we get an optimal solution by considering the induced subgraph $S(\delta), T(\delta)$.

Let r_1, r_2 be the minimum indices, such that $|S(r_1)| = l_1$ and $|T(r_2)| = l_2$.

First consider the case $r_1 \leq r_2$. The case with $r_2 < r_1$ is similar.

If any one of Case 1 or Case 2 holds, we get a 3 approximation:

Case 1: $\exists r' < r_1$, $E(r')/\sqrt{S(r')T(r')} \geq \frac{\lambda}{3}$. It is obvious that in this case we get a 3 approximation.

Case 2: $\exists r' \geq r_2$, $E(r') \geq E(S_{OPT}, T_{OPT})/3$. In this case we can add arbitrary vertices to the subgraph induced by the vertices in $S(r')$, to fulfill the size requirements.

Case 3: $\exists r_1 < r' < r_2, E(r')/\sqrt{l_1|T(r')|} \geq \frac{\lambda}{3}$. In this case, we can append arbitrary vertices to $S(r')$ to make its size exactly l_1 and thus get a 3 approximation for *DaLkDS*. Now we show that one of these three cases always occur. Suppose, if possible neither of these three cases occur. Since Case 1 does not occur, we have,

$$\begin{aligned} \int_0^{r_1} |E(r)|dr &< \frac{\lambda}{3}\int_0^{r_1} \sqrt{|S(r)||T(r)|}dr \\ &\leq \frac{\lambda}{3}\sqrt{\int_0^{r_1} |S(r)|dr \int_0^{r_1} |T(r)|dr} \\ &\leq \frac{\lambda}{3} \end{aligned} \tag{10}$$

Since Case 2 does not occur, we have,

$$\begin{aligned} \int_{r_2}^{1/\sqrt{l_1 l_2}} |E(r)|dr &< \frac{E(S_{OPT}, T_{OPT})}{3} \int_{r_2}^{1/\sqrt{l_1 l_2}} dr \\ &= \frac{E(S_{OPT}, T_{OPT})}{3} \left(\frac{1}{\sqrt{l_1 l_2}} - r_2\right) \\ &< \frac{\lambda}{3} \end{aligned} \tag{11}$$

Since Case 3 does not occur, we have,

$$\begin{aligned}
\int_{r_1}^{r_2} |E(r)|dr &< \frac{\lambda}{3}\sqrt{l_1}\int_{r_1}^{r_2} \sqrt{|T(r)|}dr \\
&= \frac{E(S_{OPT},T_{OPT})}{3\sqrt{l_2}}\int_{r_1}^{r_2} \sqrt{|T(r)|}dr \\
&< \frac{E(S_{OPT},T_{OPT})}{3\sqrt{l_1 l_2}}\int_{r_1}^{r_2} \sqrt{l_1|T(r)|}dr \\
&= \frac{E(S_{OPT},T_{OPT})}{3\sqrt{l_1 l_2}}\int_{r_1}^{r_2} \sqrt{cl_2|T(r)|}dr \\
&< \frac{E(S_{OPT},T_{OPT})}{3\sqrt{l_1 l_2}}\int_{r_1}^{r_2} \sqrt{c}|T(r)|dr \\
&\leq \frac{E(S_{OPT},T_{OPT})}{3\sqrt{l_1 l_2}} \\
&< \frac{\lambda}{3} \qquad (12)
\end{aligned}$$

Therefore from Equations (10), (11) and (12), we get;

$$\int_0^{1/\sqrt{l_1 l_2}} |E(r)|dr < \lambda$$

This gives a contradiction.

Therefore there exists a value of r, for which one of the three cases hold. We can try every possible subgraph $(S(r), T(r), E(r))$ by setting $r = s_i$ and t_j for all distinct values of s_i, t_j. □

4 Densest at most k subgraph problem

The densest at most k subgraph problem (*DamkS*) tries to find a subgraph of the highest density whose size is at most k. Andersen et al. [3] showed that an α approximation for *DamkS* implies a $\Theta(\alpha^2)$ approximation for the densest k subgraph problem. We prove that approximating *DamkS* is as hard as the *DkS* problem, within a constant factor. Precisely we prove the following theorem:

Theorem 4.1. *An α approximation algorithm for* DamkS *implies an 4α approximation algorithm for the densest k subgraph problem*

Proof. Let algorithm A be an α approximation algorithm for *DamkS* problem. We run algorithm A on graph G. If the returned subgraph H_1 has k vertices, then we get an α approximation for the densest k subgraph problem. Otherwise if the returned subgraph has less than k vertices, then we use the same shrinking procedure as in algorithm *Densest At least-k*. We shrink H_1 to a single vertex. For every vertex, $v \in G \setminus H_1$, if v has c edges to H_1, we add a self loop to v of weight c. The shrunk vertex is deleted. We run algorithm A but maintain the same size threshold of k to compute H_2. Call $D_2 = H_1 \cup H_2$. In general, $D_i = H_i \cup D_{i-1}$, where H_i is the subgraph obtained on running the ith iteration of algorithm A on the remaining graph. Note that we do not change the size threshold in any iteration. The procedure is repeated until at round l the size of D_l exceeds k. Let H^* be the optimum solution for the densest k subgraph problem and λ be the density of H^*. Since H^* is a feasible solution for *DamkS*, the optimum solution for *DamkS* has density $\geq \lambda$. The following two cases might occur:

Case 1: *At step j, $E(D_j) \cap E(H^*) \geq |E(H^*)|/2$ and $|D_j| <= k$.* Add arbitrary vertices to D_j to make its size k and call the new subgraph thus created, $H(D_j)$. Clearly $|H(D_j)| = k$ and density of $H(D_j) \geq \frac{|E(H^*)|}{2k} = \lambda/2$. Hence we get a 2 approximation for the densest k subgraph problem

Case 2: *The algorithm iterates for l rounds and there does not exist any $j \leq l$ such that $E(D_j) \cap E(H^*) \geq |E(H^*)|/2$.* Since at step j, $E(D_j) \cap E(H^*) \leq |E(H^*)|/2$, then following the same argument as in Theorem 3.2, the algorithm finds H_{j+1} whose density is at least $\frac{\lambda}{2\alpha}$. This relation holds till the l-th round. So up to $(l-1)$-th round we have $|D_{i-1}| < k$ and $d(D_{i-1}) >= \frac{\lambda}{2\alpha}$. We also have $|H_i| \leq k$ and density of $H_i \geq \frac{\lambda}{2\alpha}$. Now there can be two subcases:

Subcase 1: $|D_{i-1}| \geq k/2$. We add arbitrary r vertices to D_{i-1} to make its size k. $r < |D_{i-1}|$. So we have, the density of the subgraph thus formed $\geq \frac{\lambda}{4\alpha}$.

Subcase 2: $|D_{i-1}| \leq k/2$. Then we need to add $r = k - |D_{i-1}| \geq k/2$ vertices. Since $|D_{i-1}| + |H_i| \geq k$, $k \geq |H_i| \geq r$. We use the greedy algorithm of [11] to pick r vertices from H_i. Since $r \geq |H_i|/2$, by the greedy algorithm we get a subgraph H' of size r whose density is $\geq d(H_i)/2 \geq \frac{\lambda}{4\alpha}$. Since the density of D_{i-1} and H' are both $\geq \frac{\lambda}{4\alpha}$, we get a 4α approximation for the densest k subgraph problem. □

5 Conclusion

In this paper, we have discussed different variations of the densest subgraph problems with and without size constraints. We have considered hardness issues related to these problems and have developed fast algorithms for them for both undirected and directed networks. All these problems can be generalized to weighted setting, with same time-complexity or sometimes with only a $\log |V|$ increase in running time. An interesting open question 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. Obtaining faster algorithms for *densest at least-k_1, k_2 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.

References

[1] N. Alon, S. Arora, R. Manokaran, D. Moshkovitz, and O. Weinstein. Inapproximability of densest k-subgraph from average case hardness. *Manuscript*.

[2] R. Andersen. Finding large and small dense subgraphs. *CoRR*, abs/cs/0702032, 2007.

[3] R. Andersen and K. Chellapilla. Finding dense subgraphs with size bounds. In *WAW '09*, pages 25–36, 2009.

[4] Y. Asahiro, R. Hassin, and K Iwama. Complexity of finding dense subgraphs. *Discrete Appl. Math.*, 121(1-3):15–26, 2002.

[5] Y. Asahiro, K. Iwama, H. Tamaki, and T. Tokuyama. Greedily finding a dense subgraph. In *SWAT '96*, pages 136–148, 1996.

[6] B. Bahmani, R. Kumar, and S. Vassilvitskii. Densest subgraph in streaming and mapreduce. *PVLDB*, 5(5):454–465, 2012.

[7] A. Bhaskara, M. Charikar, E. Chlamtac, U. Feige, and A. Vijayaraghavan. Detecting high log-densities: an o (n $1/4$) approximation for densest k-subgraph. In *STOC'10*, pages 201–210. ACM, 2010.

[8] G. Buehrer and K. Chellapilla. A scalable pattern mining approach to web graph compression with communities. In *WSDM '08*, pages 95–106, 2008.

[9] M Charikar. Greedy approximation algorithms for finding dense components in a graph. In *APPROX*, pages 84–95, 2000.

[10] Y. Dourisboure, F. Geraci, and M. Pellegrini. Extraction and classification of dense communities in the web. In *WWW '07*, pages 461–470, 2007.

[11] U. Feige, G. Kortsarz, and D. Peleg. The dense k-subgraph problem. *Algorithmica*, 29:410–421, 1997.

[12] A. Gajewar and A. Das Sarma. Multi-skill collaborative teams based on densest subgraphs. In *SDM*, pages 165–176, 2012.

[13] G. Gallo, M. D. Grigoriadis, and R. E. Tarjan. A fast parametric maximum flow algorithm and applications. *SIAM J. Comput.*, 18(1):30–55, 1989.

[14] D. Gibson, R. Kumar, and A. Tomkins. Discovering large dense subgraphs in massive graphs. In *VLDB '05*, pages 721–732, 2005.

[15] A. V. Goldberg. Finding a maximum density subgraph. Technical report, 1984.

[16] R. Kannan and V. Vinay. Analyzing the structure of large graphs. Technical report, 1999.

[17] S. Khot. Ruling out ptas for graph min-bisection, dense k-subgraph, and bipartite clique. *SIAM J. Comput.*, 36(4):1025–1071, 2006.

[18] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, 1999.

[19] Guy Kortsarz and David Peleg. On choosing a dense subgraph. In *FOCS' 93*, pages 692–701. IEEE, 1993.

[20] E. Lawler. *Combinatorial optimization - networks and matroids*. Holt, Rinehart and Winston, New York, 1976.