Efficient and Effective Anchored Densest Subgraph Search: A Convex-programming based Approach

Xiaowei Ye
Beijing Institute of Technology
Beijing, China
yexiaowei@bit.edu.cn

Rong-Hua Li
Key Laboratory of Intelligent Supply Chain Technology, Longgang District, Shenzhen; Beijing Institute of Technology, Beijing, China
lironghuabit@126.com

Lei Liang
Ant Group
Hangzhou, China
leywar.liang@antgroup.com

Zhizhen Liu
Ant Group
Hangzhou, China
zhizhen.lzz@antgroup.com

Longlong Lin
Southwest University
Chongqing, China
longlonglin@swu.edu.cn

Guoren Wang
Beijing Institute of Technology
Beijing, China
wanggrbit@gmail.com

ABSTRACT

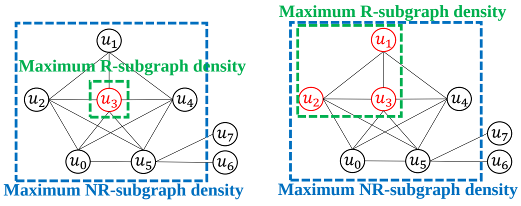
The quest to identify local dense communities closely connected to predetermined seed nodes is vital across numerous applications. Given the seed nodes R, the R-subgraph density of a subgraph S is defined as traditional graph density of S with penalties on the nodes in $S \setminus R$. The state-of-the-art (SOTA) anchored densest subgraph model, which is based on R-subgraph density, is designed to address the community search problem. However, it often struggles to efficiently uncover truly dense communities. To eliminate this issue, we propose a novel NR-subgraph density metric, a nuanced measure that identifies communities intimately linked to seed nodes and also exhibiting overall high graph density. We redefine the anchored densest subgraph search problem through the lens of NR-subgraph density and cast it as a Linear Programming (LP) problem. This allows us to transition into a dual problem, tapping into the efficiency and effectiveness of convex programming-based iterative algorithm. To solve this redefined problem, we propose two algorithms: FDP, an iterative method that swiftly attains near-optimal solutions, and FDPE, an exact approach that ensures full convergence. We perform extensive experiments on 12 real-world networks. The results show that our proposed algorithms not only outperform the SOTA methods by 3.6~14.1 times in terms of running time, but also produce subgraphs with superior internal quality.

KEYWORDS

Densest subgraph, Community search, Graph mining

1 INTRODUCTION

The global community search problem is a fundamental network analysis task, which aims at uncovering *all* communities with strong internal cohesiveness [10, 18, 32, 38, 41]. A notable variant of this problem is the local community search, which focuses on discovering *a specific* community that is highly correlated with the user-initiated seed nodes. The significance of the local community search is underscored by its diverse practical applications, such as recommender systems [22, 50], detecting spammers [47], inferring user attributes [35], and identifying protein complexes [40]. Therefore, the body of research dedicated to the local community search spans over a decade and encompasses numerous studies [3, 8, 15, 16, 19, 26, 29, 39, 42, 44–46, 49].


Maximum R-subgraph density
Maximum NR-subgraph density
u_0
u_1
u_2
u_3
u_4
u_5
u_6
u_7
Maximum R-subgraph density
Maximum NR-subgraph density
u_0
u_1
u_2
u_3
u_4
u_5
u_6
u_7

(a) $R = \{u_3\}$, $A = \{u_3\}$ (b) $R = \{u_1, u_2, u_3\}$, $A = \{u_3\}$

Figure 1: Our motivation: The red nodes are the seed nodes R. The traditional R-subgraph density metric makes it hard for the subgraphs (in green boxes) to contain nodes with large degrees even though these nodes are very much related to R. The proposed metric NR-subgraph density has no this limitation, and can identify the more reasonable and interpretive community around R (in blue boxes).

Global community search is concerned with optimizing certain global properties to ensure high-quality target communities. For instance, the densest subgraph search is a representative model within this domain. Given a graph $G(V, E)$, a subgraph S is considered the densest when it exhibits the maximum graph density ($\rho(S) = \frac{|E(S)|}{|S|}$), signifying an exceptionally interconnected community. In contrast to the global community search problem, local community search problems incorporate the principle of *locality*, which evaluates the degree of overlap with the predetermined seed nodes. To illustrate, we assume that R is the seed set, the *local conductance* [39, 46] of a community S is assessed based on the overlap score between S and R [46]. The state-of-the-art (SOTA) model is the *Anchored Densest Subgraph* [16], which optimizes R-subgraph density metric (Definition 2.1) and forces the nodes in $A \subseteq R$ (A is a given parameter) are included in the result. Unlike traditional density measures, R-subgraph density imposes a penalty on nodes that fall outside R, reinforcing the significance of the seed nodes in determining the community structure.

KDD '24, August 25–29, 2024, Barcelona, Spain

ACM ISBN 979-8-4007-0490-1/24/08…$15.00
https://doi.org/10.1145/3637528.3671727

Although R-subgraph density accounts for the relationship between the identified local community and the seed set, it often suffers from the dilemma that the identified communities do not retain favorable global properties well. For instance, as illustrated in Figure 1(b) with $R = \{u_1, u_2, u_3\}$, the subgraph that maximizes the R-subgraph density is the seed set R itself. This situation exemplifies a limitation that the subgraph identified with the largest R-subgraph density may fail to be a cohesive community (i.e., poor graph density). The reason why R-subgraph density has this shortcoming is that each node u in $S \setminus R$ has a penalty of the degree of u. Thus, the nodes with large degrees are not expected to be included in the subgraph with large R-subgraph density. The absence of large degrees nodes makes the subgraphs with large R-subgraph density sparse. In Figure 1 (a) and Figure 1(b), the subgraphs with the maximum largest R-subgraph density fail to identify $\{u_0, u_1, ..., u_5\}$ because the nodes out of R have large degrees. To address the limitations of existing local density metrics such as R-subgraph density, we propose an innovative metric named NR-subgraph density (Section 3.1). In NR-subgraph density, the penalty of each node u in $S \setminus R$ is the number of neighbors in the community S instead of the whole graph like R-subgraph density. Thus, for a community S with high NR-subgraph density, the nodes are naturally related to R. Further, NR-subgraph density is a closer lower bound to the traditional density when compared to R-subgraph density (Theorem 3.3), making subgraphs that achieve high NR-subgraph density demonstrate high traditional density. As a consequence, this correlation confirms that NR-subgraph density is valid in identifying densely connected communities.

While NR-subgraph density offers a compelling advantage in community search, the search for the subgraph with the maximum NR-subgraph density and anchored nodes A raises significant challenges. Firstly, incorporating the requirement that certain nodes $A \subseteq R$ must be included in the resulting subgraph introduces a layer of complexity not encountered in traditional global optimization problems. This requirement complicates the problem formulation as one that is amenable to convex optimization approaches. Secondly, the well-known Frank-Wolfe algorithm [28], which is adept at solving the densest subgraph problem in a global context, does not readily translate to the local domain where NR-subgraph density operates. To overcome these challenges, in this paper, we formulated the anchored densest subgraph search problem based on NR-subgraph density as a convex programming problem. In particular, this transformation allows us to leverage the powerful Frank-Wolfe optimization method, ultimately leading to the development of the FDP algorithm (Algorithm 5.2). Impressively, FDP can reach a nearly optimal solution within just a few iterations. On top of that, we introduce an exact algorithm FDPE (Algorithm 2), which continues to run FDP until convergence is achieved. We develop a maximum-flow algorithm to verify whether FDP has converged to the optimum. FDPE necessitates the execution of the maximum-flow algorithm a small number of times. Both FDP and FDPE exhibit local time and space complexity, rendering them highly scalable and exceedingly efficient for processing large-scale datasets. We highlight our main contributions as follows:

One novel NR-subgraph density metric. We present a new NR-subgraph density metric, a significant advancement that refines the community search problem by incorporating the concept of locality while still capturing essential global cohesive properties within network structures. This metric adds a nuanced perspective to the identification of communities in relation to a set of seed nodes, offering a more targeted approach to community detection.

Two Efficient algorithms. We formulate the anchored densest subgraph problem based on NR-subgraph density as a Linear Programming (LP) problem. This formulation leads to the transformation of the LP problem into its dual form, allowing us to develop two novel and efficient algorithms: the Frank-Wolfe based FDP algorithm, which iterates to near-optimal solutions, and FDPE, an exact algorithm that operates until true convergence is achieved. This contribution is crucial as it presents a methodology that overcomes the computational limitations encountered in prior models and offers a more efficient optimization framework.

Extensive experiments. To validate the effectiveness and efficiency of our proposed metrics and algorithms, we have conducted extensive experiments on 12 real-world networks. The results demonstrate that: (1) FDP outperforms the SOTA methods by at least one order of magnitude, and (2) FDPE can also be several times faster than the SOTA methods. Furthermore, the subgraphs identified with the largest NR-subgraph density exhibit high quality, reinforcing the effectiveness of our novel NR-subgraph density metric. These experimental results decisively demonstrate that FDP and FDPE are not only efficient but also effective. For reproducibility purpose, the source code of our work is available at https://github.com/LightWant/nrdensity.

2 PRELIMINARIES

We consider an unweighted and undirected graph $G(V, E)$, where V is the set of nodes and E is the set of edges. The graph comprises $n = |V|$ nodes and $m = |E|$ edges. Each edge $e(u, v)$ is a connection between u and v. The neighbors of u in graph $G(V, E)$ is $N(u, V)$. Given a node set S, we denote $G(S, E(S))$ as the induced subgraph of S, where $E(S) = \{e(u, v) | e \in E, u \in S, v \in S\}$. We abuse S to represent $G(S, E(S))$ if the context is clear. The density of a subgraph S is defined as $\rho(S) = \frac{|E(S)|}{|S|}$. Below, we introduce the definition of R-subgraph density [16], which was used for detecting the anchored densest subgraph.

Definition 2.1 (R-subgraph density [16]). Given a graph $G(V, E)$ and a set of nodes $R \subseteq V$, the R-subgraph density of a node set $S \subseteq V$ is

$$\rho_R(S) = \frac{2|E(S)| - \sum_{u \in S \setminus R} |N(u, V)|}{|S|}. \tag{1}$$

R-subgraph density incorporates the seed set R to evaluate the localized density of a subgraph. With the penalty item $\sum_{u \in S \setminus R} |N(u, V)|$, the nodes with large $|N(u, V)|$ tends not to be included in the subgraphs with large R-subgraph density. For example, as shown in Figure 1(a), the subgraph with the maximum R-subgraph density is simply $S = \{u_3\}$, yielding a R-subgraph density value of $\rho_R(S) = 0$. The subgraphs that include nodes with a larger number of neighbors have a negative R-subgraph density value. For instance, if S is expanded to include u_1, u_2, u_3, and u_4, the R-subgraph density drops to $\rho_R(S) = \frac{2 \times 5 - (3+4+4)}{4} = -\frac{1}{4}$. Figure 1(b) has a similar trend.

Anchored Densest Subgraph Search (ADSS [16]). Given a graph $G(V, E)$ and two node sets A, R with $A \subseteq R \subseteq V$, ADSS is to find the subgraph S^*_{ADSS} containing A, satisfying S^*_{ADSS} has the maximum R-subgraph density. Formally, we have

$$S^*_{\text{ADSS}} = \arg \max_{S : A \subseteq S \subseteq V} \rho_R(S) \tag{2}$$

Frank-Wolfe method. The Frank-Wolfe method is widely used for constrained primal-dual convergence problems [28]. Several previous works [11, 33, 34, 43] have employed the Frank-Wolfe method for the densest subgraph problem, which seeks a subgraph with the highest density, as denoted by $\arg\max_{S\subseteq V}\rho(S)$. This problem is modeled as a convex optimization problem in these studies, with the Frank-Wolfe algorithm being utilized to locate the optimal solution. Algorithms based on the Frank-Wolfe method are both efficient and effective in identifying the densest subgraph, which are recognized as the current SOTA methods for the densest subgraph problem.

3 A NEW LOCAL DENSITY METRIC

The subgraph with the maximum R-subgraph density often omits nodes that have a large number of neighbors, leading to the subgraph with the maximum R-subgraph density being a sparse community. Further, the ADSS problem is very hard to model as a convex programming problem due to the penalty item of R-subgraph density and anchored nodes, resulting in the efficient Frank-Wolfe method not being applied. To overcome these issues, we propose a new local density metric, called *NR-subgraph density* in Definition 3.1. The new metric does not repel high-degree nodes and the ADSS problem based on the NR-subgraph density is a convex programming problem. Moreover, we show that the subgraph identified based on the NR-subgraph density is denser than that based on the R-subgraph density in our empirical results.

Definition 3.1 (NR-subgraph density). Given a graph $G(V,E)$ and a node set $R \subseteq V$, the NR-subgraph density of a node set $S \subseteq V$ is

$$\rho_R^+(S) = \frac{2|E(S)| - \sum_{u\in S\setminus R}|N(u,S)|}{|S|} \tag{3}$$

Compared to R-subgraph density, NR-subgraph density modifies the neighborhood function by shifting from $N(u,V)$ to $N(u,S)$, placing a stronger emphasis on the internal structure of the subgraph, thus enhancing the measure of density.

Example 3.2. In Figure 1, based on the new NR-subgraph density metric, we can identify the subgraph $S^* = \{u_0, u_1, ..., u_5\}$ as having the maximum NR-subgraph density for both seed sets $R = \{u_3\}$ and $R = \{u_1, u_2, u_3\}$. Specifically, in Figure 1(a), the NR-subgraph density value for S^* when $R = \{u_3\}$ is calculated as $\rho_R^+(S^*) = \frac{2\times 12-(4+3+4+4+4)}{6} = \frac{5}{6}$. Similarly, in Figure 1(b), when $R = \{u_1, u_2, u_3\}$, the NR-subgraph density for S^* is $\rho_R^+(S^*) = \frac{2\times 12-(4+4+4)}{6} = 2$. In Figure 1(a), the community with the highest R-subgraph density has a conductance of 5, whereas that of NR-subgraph density is 1. It is similar in Figure 1(b). These examples clearly illustrate the superiority of our new metric.

Anchored Densest Subgraph Search based on NR-subgraph density (ADSS+). Given a graph $G(V,E)$ and two node sets A and R with $A \subseteq R \subseteq V$, ADSS+ is to find the subgraph $S^*_{\text{ADSS+}}$ containing A, satisfying $S^*_{\text{ADSS+}}$ has the maximum NR-subgraph density. Formally, we have

$$S^*_{\text{ADSS+}} = \arg\max_{S:A\subseteq S\subseteq V}\rho_R^+(S). \tag{4}$$

For any given subgraph S, we can easily derive that $2\rho(S) \geq \rho_R^+(S) \geq \rho_R(S)$, as detailed in Theorem 3.3. This indicates that NR-subgraph density serves as a more precise lower bound of the graph density in comparison to R-subgraph density, tightening the correlation between the local and overall density metrics. Due to the space limits, all the missing proofs can be found in the Appendix.

THEOREM 3.3. *For any subgraph S, $2\rho(S) \geq \rho_R^+(S) \geq \rho_R(S)$ holds.*

Remark. It is worth remarking that if there are nodes in the set difference $S \setminus R$ that are connected to nodes in $V \setminus S$, we can deduce that $\rho_R^+(S) > \rho_R(S)$. This is because there will be at least one node satisfying $|N(u,S)| < |N(u,V)|$. Clearly, such a condition is easy to meet in practice, making our metric is strictly better than the NR-subgraph density. Empirical results from our experiments demonstrate that $\rho(S^*_{\text{ADSS+}})$ is significantly larger than $\rho(S^*_{\text{ADSS}})$.

4 AN LP FORMULATION

In this section, we show that the ADSS+ can be formulated as a linear program problem, denoted by LP. The optimal solution of LP can be directly correlated with the optimal subgraph S^*. Specifically, we establish the following LP for the ADSS+ problem based on the NR-subgraph density metric.

$$\begin{array}{lcl}
[\mathsf{LP}] & \max \sum_{e\in E} w_e x_e & s.t. \\
C_0: & w_e = \sum_{u\in e} 1[u\in R] & \forall e \in E \\
C_1: & x_e = \min_{u\in e} y_u, & \forall e \in E \\
C_2: & \sum_{u\in V} y_u = 1 & \\
C_3: & y_u = \max_{v\in V} y_v & \forall u \in A \\
C_4: & x_e \geq 0, y_u \geq 0 & \forall e \in E, \forall u \in V
\end{array} \tag{5}$$

Below, we explain the relationship between LP and ADSS+.

LEMMA 4.1. *Given a subgraph S, we have $\rho_R^+(S)\cdot|S| = \sum_{e\in E(S)} w_e$, where w_e is defined in C_0 of* LP.

Lemma 4.1 gives the meaning of w_e in LP. w_e is the weight of edge e in the definition of NR-subgraph density. NR-subgraph density is the total weight of edges divided by the number of nodes, which is a classic form of weighted density.

LEMMA 4.2. *Let y be a feasible solution of* LP. *Let x be the vector produced by y, i.e. $x_e = \min_{u\in e} y_u$ (C_1 of* LP*). If $S = \{u|y_u > 0\}$, we have $E(S) = \{e|x_e > 0\}$.*

Lemma 4.2 establishes that any feasible solution of the linear program LP corresponds to a subgraph. More precisely, the nodes of the subgraph are represented by the positive entries in the solution vector y. This correlation also elucidates the purpose of constraint C_3 within LP. C_3 forces the nodes in the set of anchored nodes A having positive entries, ensuring that A is included in S.

LEMMA 4.3. *Given a subgraph S that $A \subseteq S$, there exists a feasible solution of* LP *to make the value of the objective function be $\rho_R^+(S)$.*

Lemma 4.3 gives a map from a subgraph S to a feasible solution LP. The value of the objective function with respect to (w.r.t.) the mapped feasible solution is exactly the NR-subgraph density of S.

LEMMA 4.4. *Given a feasible solution y with the objective function value v_y, there exists at least a subgraph $S \subseteq V$ such that $\rho_R^+(S) \geq v_y$.*

Leveraging Lemma 4.3 and Lemma 4.4, we can establish that there exists a one-to-one correspondence (i.e., bijection) between the optimal solution of the linear program LP and the optimal subgraph S^*, as shown in Theorem 4.5. This bijection ensures that each optimal solution to LP uniquely maps to an optimal subgraph S^*, and vice versa.

Theorem 4.5. *Assuming that y^* is the optimal solution of* LP*, we can derive that $y^* \Leftrightarrow S^*_{\text{ADSS+}}$.*

Designing an efficient algorithm to solve LP directly is not straightforward. To circumvent this challenge, we have developed a Frank-Wolfe based approach to converge on the optimal solution y^*. This is achieved by applying the Frank-Wolfe method to the dual problem of LP, allowing us to compute the solution indirectly but efficiently.

5 THE FRANK-WOLFE BASED ALGORITHM

In this section, we propose a Frank-Wolfe based approximation algorithm FDP. FDP is tailored to find an approximate solution of LP from the dual perspective. A key advantage of FDP is that its time and space complexity are linear w.r.t. the size of the subgraph R' which is such that each node in R' has at least one neighbor in R. On top of that, we also develop an exact algorithm to solve the ADSS+ problem by integrating FDP with a novel maximum-flow based technique, named FDPE. The time and space complexity of FDPE is also local to the size of R'.

5.1 DP: The Dual of LP

Here, we derive the dual of LP that was previously formulated in Equation 5.

Theorem 5.1. *Let* LP *be the linear program in Equation 5, the dual of* LP *is formalized as follows.*

$$\begin{array}{llll}
[\text{DP}] & \min \|r+\beta\| & s.t. & \\
D_0: & r_u = \sum_{e:u\in e} \alpha^e_u & \forall u \in V & \\
D_1: & \alpha^e_u \geq 0 & \forall u \in e \in E & \\
D_2: & \sum_{u\in e} \alpha^e_u = w_e & \forall e \in E & (6)\\
D_3: & \beta_u = 0 & \forall u \notin A & \\
D_4: & r_u + \beta_u = \max_{u\in V}(r_u+\beta_u) & \forall u \in A &
\end{array}$$

Below, we provide some properties of DP. Let α^* and β^* be the optimal solution of DP. Let r^* be the vector computed from α^* according to D_0 of DP. Let S^* represent $S^*_{\text{ADSS+}}$ for short.

Lemma 5.2. *For each $u \notin A$, we have $\beta^*_u = 0$.*

Lemma 5.2 is easy to prove by D_3. Thus, the nodes in $S^* \setminus A$ are all zeros in β^*.

Theorem 5.3. *Let r be a feasible solution of* DP*, we can derive that $\max_{u\in V} r_u$ is the upper bound of $\rho^+_R(S^*)$.*

Theorem 5.3 can help to bound the error of the proposed FDP algorithm, which will be explained in Section 5.2.

Theorem 5.4. *The optimal solution of* DP *maps to S^*.*

Combining Theorem 5.3 and 5.4, we obtain a clear understanding of $r^* + \beta^*$. (1) The nodes in the anchored set A have the maximum value in $r^* + \beta^*$ (according to D_4 and the proof of Theorem 5.4 in appendix). (2) For the nodes in $S^* \setminus A$, they also achieve the maximum value in $r^* + \beta^*$, with $\beta^*_u = 0$. (3) Nodes that are not part of the optimal subgraph S^* have a smaller value in $r^* + \beta^*$.

Compared to LP, DP approaches the ADSS+ problem from a different mathematical angle. In DP, α^e_u represents the allocation of weight w_e to node u. β_u serves as an adjustment to ensure inclusion of nodes from the anchored set A in the solution. β can be understood as a mechanism that adjusts the "attractiveness" or "penalty" for the anchored nodes.

Algorithm 1: Frank-Wolfe Based Algorithm (FDP)

```
   Input: The graph $G(V,E)$, $R$, $A$, number of iterations $T$
   Output: $\hat{S^*}$: the approximation solution of $S^*$; $\epsilon$: the error bound
           such that $\rho^+_R(S^*)/\rho^+_R(\hat{S^*}) \leq 1+\epsilon$
1  $r(u) \leftarrow 0, \forall u \in V$;
2  $\beta(u) \leftarrow 0, \forall u \in V$;
3  for $t \leftarrow 1, 2, ..., T$ do
4      foreach $e \in E$ with $w_e > 0$ do
5          $U_{min} \leftarrow \arg\min_{u\in e} r(u) + \beta(u)$;
6          $r(u) \leftarrow r(u) + w_e/|U_{min}|, \forall u \in U_{min}$;
7      $v_{max} \leftarrow \max_{u\in V}(r(u) + \beta(u))$;
8      for $u \in A$ do
9          $\beta(u) \leftarrow v_{max} - r(u)$;
10 Sort $V$ by $r(u) + \beta(u)$;
11 Denote by $V_i$ the previous $i$ nodes of $V$;
12 $\hat{S^*} = \arg\max_{V_i} \rho^+_R(V_i)$; /*implemented in linear time.*/
13 $\epsilon = \frac{\max_{u\in V} r(u)/T}{\rho^+_R(\hat{S^*})} - 1$;
14 return $\hat{S^*}, \epsilon$;
```

5.2 FDP: Frank-Wolfe Based Algorithm

The key idea of our algorithm lies in the fact that minimizing the norm $\min \|r+\beta\|$ is equivalent to minimizing the square of the norm $\min \|r+\beta\|^2 = \sum_u (r_u+\beta_u)^2$. It is a well-understood principle that the more evenly distributed the values of $r+\beta$ are, the smaller the sum of their squares will be. Therefore, our goal is to evenly distribute the weight of edges among the nodes (by computing α and r) while adjust the weight for the anchored set A (by computing β). We strive to balance the total assigned weight across all nodes ($r_u + \beta_u$) as evenly as possible. This evenly distributed weight assignment leads to the desired minimization of $\min \|r+\beta\|^2$.

With the above observations, we design Algorithm 1 to show how Frank-Wolfe based algorithm FDP works. Algorithm 1 inputs a graph $G(V,E)$, an integer T, and outputs an $(1+\epsilon)$-approximation of the subgraph with the maximum NR-subgraph density. Algorithm 1 maintains two vectors r and β (line 1 and line 2). Notably, the algorithm opts for vector r in place of α. To make the distribution of $r+\beta$ as even as possible, Algorithm 1 tries to assign the weight to the nodes with the smallest value of $r_u+\beta_u$. According to D_2 of DP, the weight of edge e can only be assigned to the nodes in e. As a result, the weight of each edge e is averagely assigned to the nodes in e with the smallest $r(u)+\beta(u)$ (lines 4-6). To make sure the nodes in A are contained in the result, update them to the maximum value. It is also an implementation of D_4 of DP (lines 7-9). Then, the nodes with the larger weight are ranked higher (line 10). The nodes with larger rank has larger priority to be in the result (line 12). Note that $\rho^+_R(V_i)$ can be computed in constant time from $\rho^+_R(V_{i-1})$, thus the time complexity of line 12 is linear. At last, the largest entry of r/T is treated as an upper bound of $\rho^+_R(S^*)$, which we will also describe in Lemma 5.6 (line 13). Next, we give the technical analysis of FDP.

Correctness. We analyzed the correctness of FDP as follows.

Lemma 5.5. *$\hat{S^*}$ is the node set returned by Algorithm 1, $A \subseteq \hat{S^*}$.*

Lemma 5.6. *ϵ is the decimal returned by Algorithm 1, we have that ϵ is a correct error bound.*

Lemma 5.7. *Algorithm 1 is an implementation of the Frank-Wolfe method for* DP.

Theorem 5.8. *Algorithm 1 is a correct solver of* DP.

Proof sketch: By Lemmas 5.5, 5.6 and 5.7, we can prove the theorem.

Local complexity. The time and space complexity of Algorithm 1 is local, i.e. it is independent of the whole graph.

Theorem 5.9. *Let E' be the set of edges with at least one node in R, and let $m' = |E'|$. Let n' be the number of nodes in the subgraph induced by E'. The time complexity of Algorithm 1 is $O(Tm')$. The space complexity of Algorithm 1 is $O(n' + m')$.*

Theorem 5.9 shows that the time complexity of the Algorithm 1 is upon R. Consequently, Algorithm 1 does not necessitate the loading of the complete network in implementation, which demonstrates efficiency in both time and space.

Guarantees on convergence in sub-linear iterations. We analyze the expected number of iterations to reach a required qualified solution. The result is that FDP converges fast because the fact that $w_e \leq 2$, by which we can bound the curvature constant (details see Appendix B, Theorem B.1) of the objective function.

Lemma 5.10. *Let m' be the count of edges with at lest one side in R. The* ***curvature constant*** *of the objective function of* DP *is $C_f = \sum_e {w_e}^2 \leq 4m'$.*

Theorem 5.11. *Let f be the objective function of* DP. *Let α^* and β^* be the optimal solution. Let t be the number of iterations. In* FDP, *it has $f(\alpha^{(t)}, \beta^{(t)}) - f(\alpha^*, \beta^*) \leq \frac{C_f}{t} \leq \frac{4m'}{t}$.*

Theorem 5.11 indicates that FDP has a nice converge rate. Indeed, as shown in our experiments (Section 6), FDP can achieve a near-optimal solution in only one iteration.

In practice, the performance of our proposed algorithms depends on the quality of the seed set, which is a common problem for the local community search algorithms. The seeds serve as the starting nodes for the detection and expansion into full communities. The effectiveness of these algorithms is affected by how well the seed nodes represent the communities of interest.

5.3 FDPE: FDP Based Exact Algorithm

In this subsection, we propose a FDP based exact algorithm, named FDPE (Algorithm 2). As shown in Theorem 5.8, FDP can reach the exact result when T is sufficiently large. Therefore, FDPE embodies an iterative process that continues to execute FDP until convergence is attained, yielding an accurate result.

To verify the optimality of the subgraph $\hat{S}^*$ identified by FDP, FDPE employs the "stable subset" technique [17] (the details of the stable subset are stated in Appendix C) along with a newly proposed technique based on maximum flow. The role of the **stable subset** technique is to ensure that the true optimal subgraph S^* is contained within the candidate subgraph $\hat{S}^*$. Following this containment check, FDPE applies the maximum-flow based technique to ascertain whether there is any subgraph S' strictly contained within $\hat{S}^*$ that has a higher NR-subgraph density, i.e., $\rho^+_R(S') > \rho^+_R(\hat{S}^*)$. If no such subgraph S' exists and $\hat{S}^*$ is confirmed to be a stable subset, it can be concluded that $\hat{S}^*$ is exactly S^*.

Maximum-flow based checker. We design a network whose minimum cut can judge whether $\hat{S}^*$ has denser subgraphs. Let E' be the set of edges with $w_e > 0$ and $e \subset \hat{S}^*$. Given $\hat{S}^*$, the flow network has $|\hat{S}^*| + 2$ nodes including the source node s, the sink node t and $\hat{S}^*$. Note that $A \subseteq \hat{S}^*$ by Lemma 5.5. The edges are built by the following way.

- From s to each $u \in A$, link an edge with infinity capacity.
- From s to each $u \in \hat{S}^* \setminus A$, link an edge with $|N(u, \hat{S}^* \cap R)|$.
- For each edge $e(u, v) \in E'$, link an edge from u to v with capacity w_e if $v \in R$, and link an edge from v to u with capacity w_e if $u \in R$.
- From each $u \in \hat{S}^*$ to t, link an edge with capacity $\rho^+_R(\hat{S}^*)$.

Algorithm 2: FDP Based Exact Algorithm (FDPE)

```
  Input: The graph $G(V, E)$, $R$, $A$, number of iterations $T$
  Output: $S^*$: the subgraph with the maximum $\rho^+_R(S)$
1 $E' \leftarrow \{e \in E | w_e > 0\}$;
2 repeat
3     $\hat{S}^*, \epsilon \leftarrow$ FDP$(G, R, A, T)$;
4 until isStableSubset($\hat{S}^*$) and
    maxflow$(\hat{S}^*, R, A, E') = \sum_{u \in \hat{S}^*} |N(u, R \cap \hat{S}^*)|$;
5 return $\hat{S}^*$;
```

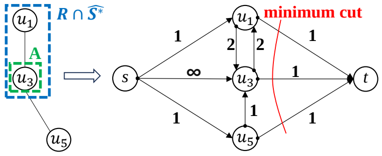


Figure 2: An example of the flow network

Theorem 5.12. *Let $f(\hat{S}^*)$ be the minimum cut of the proposed network. We can derive that $\hat{S}^*$ has no subgraph with larger NR-subgraph density if $f(\hat{S}^*) = \sum_{u \in \hat{S}^*} |N(u, \hat{S}^* \cap R)|$.*

Example 5.13. Figure 2 is an example of the flow network for the graph in Figure 1 when $\hat{S}^* = \{u_1, u_3, u_5\}$, $R = \{u_1, u_2, u_3\}$ and $A = \{u_3\}$. From s to u_3, the edge capacity is infinity since $u_3 \in A$. Both u_1 and u_5 have one neighbor in $\hat{S}^* \cap R$, so the capacity of (s, u_1) and (s, u_5) is 1. For the edge $e(u_1, u_3)$ with $w_e = 2$, link an edge from u_1 to u_3 and an edges from u_3 to u_1 because both u_1 and u_3 are in R. Since the NR-subgraph density of $\hat{S}^*$ is 1, each node link an edge to t with capacity 1. We observe that $f(\hat{S}^*) = \sum_{u \in \hat{S}^*} |N(u, \hat{S}^* \cap R)| = 3$, and thus there is no denser subgraph in $\hat{S}^*$.

The maximum-flow is well known to compute the minimum cut. We utilize the push-and-relabel maximum-flow algorithm in implementation. When the flow network has n nodes, the time complexity of the push-and-relabel algorithm is $O(n^3)$. Thus, the time complexity of the maxflow in line 4 of Algorithm 2 is $O(|\hat{S}^*|^3)$.

Discussion. In practice, the number of iterations within lines 2-4 of Algorithm 2 tends to be a small constant. This efficiency stems from the fact that nodes within the optimal subgraph S^* quickly achieve the highest ranks (as the nodes with the largest ranks are selected to form $\hat{S}^*$ in line 12 of FDP). Consequently, FDPE typically requires only a limited number of executions of the maximum flow algorithm to reach convergence. The experimental results also show that FDP converges fast and thus FDPE only requires a small number of call of maximum flow algorithm.

Table 1: The statistical information of the datasets

Networks	n	m	Description
WikiV	7,115	100,762	Social network
Epinion	75,879	405,740	Social network
Gowalla	196,591	950,327	Social network
Amazon	403,394	2,443,408	Co-purchase
DBLP	425,957	1,049,866	Collaboration
Berkstan	685,230	6,649,470	Web graph
Youtube	1,157,827	2,987,625	Social network
Pokec	1,632,803	22,301,964	Social network
Skitter	1,696,415	11,095,298	Internet topology
Orkut	3,072,627	117,185,083	Social network
ComLiveJ	4,036,538	34,681,189	Social network
Friendster	65,608,366	1,806,067,135	Social network

6 EXPERIMENTS

We conduct extensive experiments to answer the following Research Questions. **RQ1:** Can we identify local communities with higher quality than the start-of-the-art metrics by our proposed NR-subgraph density (Definition 3.1)? **RQ2:** How much our proposed FDP (Algorithm 1) and FDPE (Algorithm 2) improve in terms of running time or quality compared to other existing algorithms? **RQ3:** Whether our algorithms are sensitive to the parameters. **RQ4:** How fast does FDP converge?

6.1 Experimental Setup

Datasets. We evaluate our solutions on 12 real-world networks (Table 1) sourced from the Stanford Network Analysis Project (SNAP) [31], spanning various domains (e.g., social networks and collaboration networks). Note that the largest dataset Friendster with over 65 million nodes and 1.8 billion edges.

Generations of A and R. To create the seed sets A and R, we employ the same strategy introduced in prior work [16]. This process commences by randomly selecting a node u from the vertex set V. The set A is then constituted by randomly choosing nodes from the immediate and secondary neighbors of u, typically fixing the size of A at 8. The set R is constructed through several random walks from A. The number and length of walks are determined by drawing random integers within the range of $[3, 10]$. For each network in our study, we generate 500 distinct (A, R) pairs, with the average cardinality of R across all networks being 254. Unless otherwise stated, the results presented are derived from the mean metrics computed over these 500 pairs.

Algorithms. We implemented FDP (Algorithm 1) and FDPE (Algorithm 2) in C++. For comparative purposes, we also incorporate the LA [16] and FS [46] algorithms into our analysis. Each of these algorithms accepts a graph along with subsets A and R, where $A \subseteq R \subseteq V$, to output a community inclusive of A and confined to R. The LA seeks to optimize the R-subgraph density metric, while FS targets the improvement of local conductance, defined as $\frac{|E(S,\bar{S})|}{\sum_{u\in R\cap S} d(u)-\epsilon\sum_{u\in \bar{R}\cap S} d(u)-\sum_{u\in R\setminus S} p_u d(u)}$, with $d(u)$ representing the degree of node u. For consistency with previous studies [16], we standardize parameters by setting $p_u = 0$ and $\epsilon = 1$ to ensure a balanced comparison. Both LA and FS are based on the maximum flow algorithm. Since LA and FS are originally available in Julia [16, 46], we reimplemented them in C++ to ensures comparing all algorithms under the same programming language. Our C++ implementations consistently outperform their Julia counterparts.

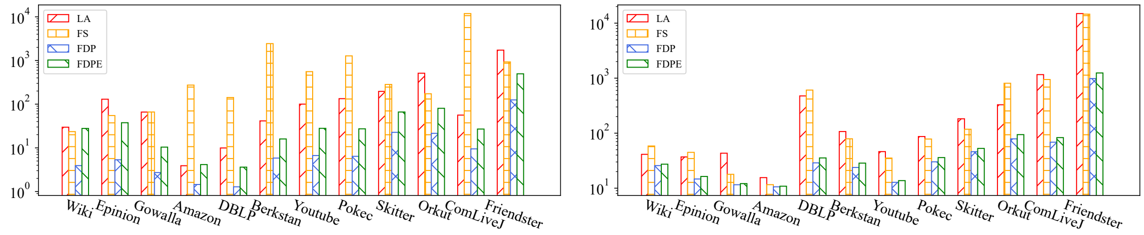
6.2 Experimental Results

Exp-1: Running Time and Memory Cost. The running time of LA, FS, FDP, and FDPE across 12 real-world networks is shown in Figure 3(a). The results indicate that FDP consistently outperforms the other algorithms in terms of speed, being on average 14.1×, 85.1×, and 3.9× faster than LA, FS, and FDPE, respectively. The efficiency of FDP can be attributed to its localized edge scanning around the set R, circumventing the computationally intensive maximum-flow algorithm required by LA, FS, and FDPE. While FDPE is not as rapid as FDP, it still significantly surpasses LA and FS in speed on 10 of the networks, exhibiting comparable performance on the remaining two, WikiV and Amazon. This is credited to the number of iterations of FDPE is typically low (as per lines 2-4 of Algorithm 2). Overall, these findings highlight the superior performance of our proposed algorithms in terms of running time.

Figure 3(b) shows the memory consumption of the evaluated algorithms, measured using the "massif" tool within Valgrind [37]. Our observations indicate that LA and FS exhibit comparable memory overheads. This similarity can be attributed to their analogous search strategies, which incrementally expand the search space from R to its neighboring nodes [16, 46]. On the other hand, FDP demonstrates a marginally lower memory usage compared to FDPE, the difference that stems from FDPE's additional requirement to maintain the weight assignment vector α as discussed in Section 5.3. Notably, both FDP and FDPE realize up to an order of magnitude reduction in memory consumption when contrasted with LA and FS. These results confirm the memory efficiency of our algorithms.

Exp-2: Community Quality. The quality analysis of the identified community is shown in Figure 4. Four distinct effectiveness metrics are stated as follows.

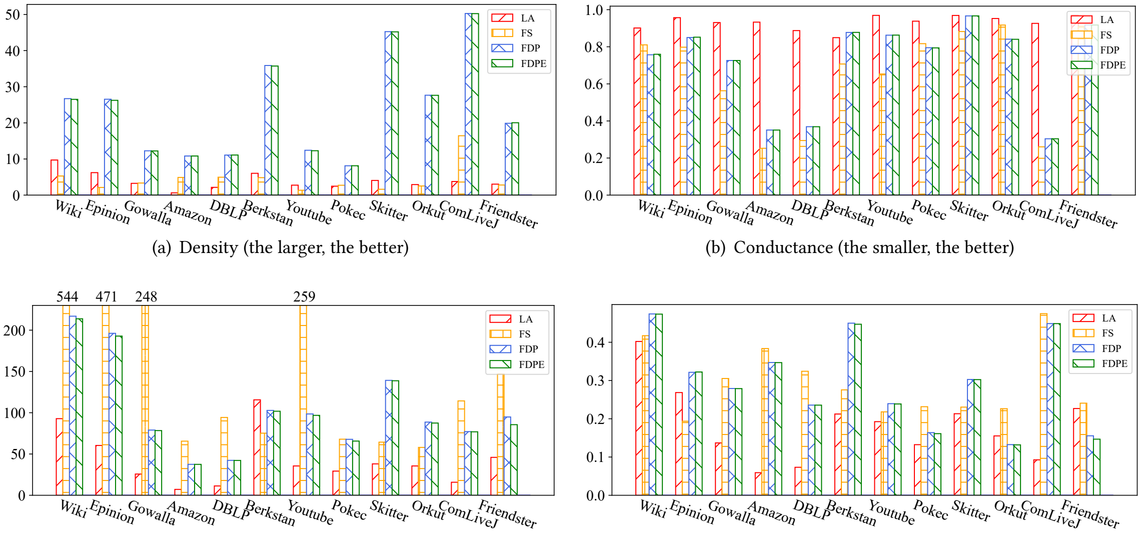
- **Density:** Recall that the density of a subgraph S is $\rho(S) = \frac{|E(S)|}{|S|}$. Figure 4(a) reveals that FDP and FDPE consistently report subgraphs with the highest density, averaging 6.1× and 5.4× greater than those identified by LA and FS, respectively. These results demonstrate the high effectiveness of our NR-subgraph density metric.
- **Conductance:** A subgraph S with lower conductance, defined as $\frac{\sum_{u\in S}|N(u,V\setminus S)|}{\min\{\sum_{u\in S}|N(u,V)|,\sum_{v\in V\setminus S}|N(v,V)|\}}$ [32], is indicative of a superior community. As shown in Figure 4(b), FS has the lowest conductance, which is in expectation because FS find the subgraph which optimizes the local conductance [46]. Both FDP and FDPE surpass LA, with conductance measures that are 23% lower.
- **Size:** Similar to [16], we also compare the sizes of the communities identified by different algorithms. The average sizes of the communities found by the algorithms are depicted in Figure 4(c). Despite FS identifying some exceptionally large subgraphs, their sparsity is evident from Figure 4(a). When compared to LA, the subgraphs discovered by FDP and FDPE are, on average, 2.4× larger. We also note that all the communities identified by various algorithms are small compared to the original graph, indicating the locality of all the compared local community search algorithms.
- **Locality to R:** The locality metric, $\frac{|S\cap R|}{|S\cup R|}$, assesses how local a subgraph S is to the set R. In Figure 4(d), both FDP and FDPE demonstrate 1.6× higher locality compared to LA and a marginal 0.8% improvement over FS. Since FS is designed to optimize the local conductance, it can outperform our method in terms of conductance. However, the conductance achieved by FS is only slightly smaller



(a) Time (ms)

(b) Memory (MB)

Figure 3: Running time (ms) and memory cost comparison (MB)



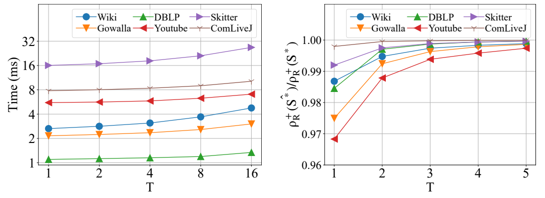
(a) Density (the larger, the better)

(b) Conductance (the smaller, the better)

(c) Size

(d) Locality to R (the larger, the better)

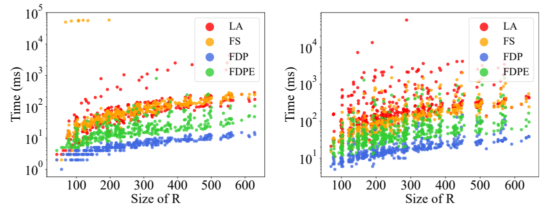
Figure 4: Community quality comparison



(a) running time

(b) the convergence rate $(\frac{\rho_R^+(\hat{S}^*)}{\rho_R^+(S^*)})$

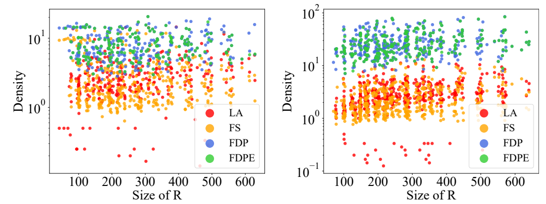
Figure 5: Sensitivity analysis with varying T



(a) Pokec

(b) Orkut

Figure 6: Running time analysis for different $|R|$ sizes



(a) Pokec

(b) Orkut

Figure 7: Community quality analysis for different $|R|$ Sizes

by 0.063 on average. Our solution's subgraph density can exceed FS's by up to 5 times. These results indicate that our solution can found much denser subgraphs than FS, with conductance slightly increased.

In short, these findings validate the rationality and superiority of the proposed NR-subgraph density metric. Communities with the high NR-subgraph density values correlate with high-quality communities, as evidenced by our comprehensive analysis.

Exp-3: Sensitivity Analysis with Varying T. The sensitivity of the running time of FDP is depicted in Figure 5(a). This figure illustrates results for 6 representative networks, with analogous patterns observed across the others. Remarkably, FDP exhibits robust performance against different T. The running time increases only marginally, even as T is doubled. This negligible sensitivity can largely be attributed to the locality of the time complexity of FDP that it only scans the edges with one side in R (lines 3-9 of Algorithm 1). Moreover, Figure 5(b) shows the convergence behavior of FDP with varying T, where convergence is quantified using the ratio $\frac{\rho_R^+(\hat{S}^*)}{\rho_R^+(S^*)}$. The figure presents results from 6 networks, with

Table 2: The value of the upper bound of error ϵ

Networks	T 1	2	4	8	16
WikiV	0.24	0.13	0.07	0.04	0.03
Gowalla	0.43	0.26	0.18	0.14	0.12
DBLP	0.44	0.31	0.25	0.22	0.20
Youtube	0.56	0.32	0.20	0.15	0.12
Skitter	0.18	0.11	0.07	0.05	0.04
ComLiveJ	0.16	0.11	0.09	0.08	0.08

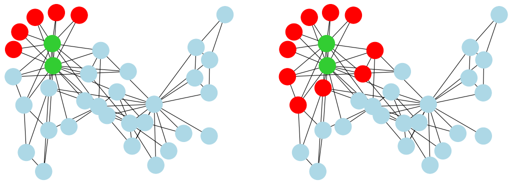
comparable trends across the remaining datasets. As depicted, FDP achieves over 96% approximation of the optimal solution by the first iteration. This rapid convergence is credited to the efficient initial weight distribution, where during the first iteration, weights are allocated to S^* (lines 4-6 of Algorithm 1), and nodes outside S^* typically receive lower weights. Hence, FDP can approach an near-optimal approximation within a small number of iterations.

Exp-4: Running Time Analysis for Different $|R|$ Sizes. The impact of varying the seed set size ($|R|$) on the running time is illustrated in Figure 6. We use datasets Pokec and Orkut as examples, with similar patterns observed across other datasets. Each data point in Figure 6 correlates to a unique (A, R) pair. As expected, an increase in $|R|$ leads to a corresponding rise in running time for all algorithms. Notably, FDP consistently exhibits the fastest running time. Moreover, the running time of FDP remains remarkably stable across different $|R|$ sizes. For instance, on the Pokec network with $|R|$ ranging between 290 and 310, the running time of FDP fluctuates narrowly between $6ms$ and $9ms$. This stability can be attributed to the time complexity of FDP being primarily dependent on the local edges surrounding R. While FDPE operates at a slower pace compared to FDP, it still outperforms LA and FS in 96% of the cases on Pokec and 95% on Orkut, demonstrating its relative scalability and efficiency. Hence, these results give preliminary evidence that both FDP and FDPE scale well when varying seed set sizes.

Exp-5: Community Quality Analysis for Different $|R|$ Sizes. The influence of seed set size ($|R|$) on community quality is depicted in Figure 7 for Pokec and Orkut networks, with consistent behavior noted across other datasets. In the figure, each point corresponds to an execution instance with a specific pair of A and R. The results reveals that communities identified by FDP and FDPE exhibit greater density than those found by LA and FS in over 99% of the cases evaluated. This finding demonstrates that both FDP and FDPE can consistently uncover denser communities than other baselines.

Exp-6: Evaluating the Error Bound ϵ. FDP provides an ϵ value, serving as the error upper bound (line 14 of Algorithm 1). Due to space limits, we detail the ϵ outcomes for 6 networks in Table 2. Remarkably, even with a single iteration ($T = 1$), ϵ remains below 1, suggesting that FDP achieves at least 2-approximation. As excepted, ϵ diminishes with increasing T. For instance, on the WikiV network with $T = 16$, ϵ reaches as low as 0.03, indicating an upper bound of approximation ratio of $\frac{\rho_R^+(S^*)}{\rho_R^+(\hat{S}^*)} \leq 1.03$. The empirical results confirm that ϵ is a reliable indicator of the accuracy of FDP in practice.

Exp-7: Case Studies. To further show the effectiveness of our FDP, we compare the performance of LA and FDP on the well-known and widely-used Zachary karate club network [1]. The details of the datasets are in Appendix E. LA identifies a community comprising only 5 members (red in Figure 8(a)) closely connected to the seed nodes (depicted in green). These members tend to have lower degrees, which aligns with the tendency of R-subgraph density to exclude nodes with higher degrees. In contrast, FDP is capable of discovering a broader community that includes all 10 members in the vicinity of the seed nodes (Figure 8(b)), which aligns with the known group assignments present in the publicly available datasets within the PyTorch framework (see source code for torch_geometric.datasets.karate, community 0). Indeed, it appears more reasonable to include these 10 members into a community, as all of them are densely connected to the seeds. The conductance of the subgraph with the maximum R-subgraph density and NR-subgraph density is 0.44 and 0.27, demonstrating the superiority of our NR-subgraph density. This result exemplifies the potential of FDP for effective local community detection and related applications.



(a) R-subgraph density by LA (b) NR-subgraph density by FDP

Figure 8: Case studies. Local communities (red nodes) are computed by LA and FDP on the Zachary karate club network, where R is composed of the green nodes.

7 CONCLUSION

This work presents a step forward in detecting local dense communities within networks. The proposed NR-subgraph density metric considers both the local connectivity to the seed set and the global graph density. We formulate the problem of the anchored densest subgraph search based on NR-subgraph density as a Linear Programming problem, which can be converted into its dual to facilitate the use of convex programming techniques. The proposed FDP and FDPE algorithms harness the efficiency and convergence properties of the Frank-Wolfe optimization method. The extensive experiments conducted on 12 real-world networks have demonstrated the high performance and utility of our algorithms. Both FDP and FDPE are not only substantially faster than existing methods but also uncover subgraphs with exceptional internal quality. These results stand as a testament to the practicality and effectiveness of our approach for local community detection.

8 ACKNOWLEDGMENT

This work was partially supported by (I) the National Key Research and Development Program of China 2021YFB3301301, (II) NSFC Grants U2241211 and 62072034. Ring-Hua Li is the corresponding author of this paper.

REFERENCES

[1] 2017. Zachary karate club network dataset – KONECT. http://konect.cc/networks/ucidata-zachary

[2] Esra Akbas and Peixiang Zhao. 2017. Truss-based Community Search: a Truss-equivalence Based Indexing Approach. *Proc. VLDB Endow.* 10, 11 (2017), 1298–1309.

[3] Morteza Alamgir and Ulrike von Luxburg. 2010. Multi-agent Random Walks for Local Clustering on Graphs. In *ICDM 2010*. IEEE Computer Society, 18–27.

[4] Aris Anagnostopoulos, Luca Becchetti, Adriano Fazzone, Cristina Menghini, and Chris Schwiegelshohn. 2020. Spectral Relaxations and Fair Densest Subgraphs. In *CIKM*. 35–44.

[5] Reid Andersen and Kumar Chellapilla. 2009. Finding Dense Subgraphs with Size Bounds. In *WAW*, Vol. 5427. Springer, 25–37.

[6] Reid Andersen and Kevin J. Lang. 2006. Communities from seed sets. In *WWW 2006*. ACM, 223–232.

[7] Nicola Barbieri, Francesco Bonchi, Edoardo Galimberti, and Francesco Gullo. 2015. Efficient and effective community search. *Data Min. Knowl. Discov.* 29, 5 (2015), 1406–1433.

[8] Yuchen Bian, Yaowei Yan, Wei Cheng, Wei Wang, Dongsheng Luo, and Xiang Zhang. 2018. On Multi-query Local Community Detection. In *ICDM 2018*. IEEE Computer Society, 9–18.

[9] Digvijay Boob, Yu Gao, Richard Peng, Saurabh Sawlani, Charalampos E. Tsourakakis, Di Wang, and Junxing Wang. 2020. Flowless: Extracting Densest Subgraphs Without Flow Computations. In *WWW*. 573–583.

[10] Lijun Chang and Lu Qin. 2019. Cohesive Subgraph Computation Over Large Sparse Graphs. In *ICDE*. 2068–2071.

[11] Moses Charikar. 2000. Greedy approximation algorithms for finding dense components in a graph. In *APPROX 2000*, Vol. 1913. Springer, 84–95.

[12] Chandra Chekuri, Kent Quanrud, and Manuel R. Torres. 2022. Densest Subgraph: Supermodularity, Iterative Peeling, and Flow. In *SODA*. SIAM, 1531–1555.

[13] Jiyang Chen, Osmar R. Zaïane, and Randy Goebel. 2009. Local Community Identification in Social Networks. In *ASONAM 2009*. IEEE Computer Society, 237–242.

[14] Wanyun Cui, Yanghua Xiao, Haixun Wang, Yiqi Lu, and Wei Wang. 2013. Online search of overlapping communities. In *SIGMOD*. ACM, 277–288.

[15] Wanyun Cui, Yanghua Xiao, Haixun Wang, and Wei Wang. 2014. Local search of communities in large graphs. In *SIGMOD*. ACM, 991–1002.

[16] Yizhou Dai, Miao Qiao, and Lijun Chang. 2022. Anchored Densest Subgraph. In *SIGMOD '22*. ACM, 1200–1213.

[17] Maximilien Danisch, T.-H. Hubert Chan, and Mauro Sozio. 2017. Large Scale Density-friendly Graph Decomposition via Convex Programming. In *WWW 2017*. ACM, 233–242.

[18] Luca Donetti and Miguel A Munoz. 2004. Detecting network communities: a new systematic and efficient algorithm. *Journal of Statistical Mechanics: Theory and Experiment* 2004, 10 (2004), P10012.

[19] Yixiang Fang, Xin Huang, Lu Qin, Ying Zhang, Wenjie Zhang, Reynold Cheng, and Xuemin Lin. 2020. A survey of community search over big graphs. *VLDB J.* 29, 1 (2020), 353–392.

[20] Yixiang Fang, Kaiqiang Yu, Reynold Cheng, Laks V. S. Lakshmanan, and Xuemin Lin. 2019. Efficient Algorithms for Densest Subgraph Discovery. *Proc. VLDB Endow.* 12, 11 (2019), 1719–1732.

[21] Uriel Feige, Guy Kortsarz, and David Peleg. 2001. The Dense *k*-Subgraph Problem. *Algorithmica* 29, 3 (2001), 410–421.

[22] Fabio Gasparetti, Giuseppe Sansonetti, and Alessandro Micarelli. 2021. Community detection in social recommender systems: a survey. *Appl. Intell.* 51, 6 (2021), 3975–3995.

[23] Andrew V Goldberg. 1984. Finding a maximum density subgraph. (1984).

[24] Kun Guo, Xintong Huang, Ling Wu, and Yuzhong Chen. 2022. Local community detection algorithm based on local modularity density. *Applied Intelligence* 52, 2 (2022), 1238–1253.

[25] Elfarouk Harb, Kent Quanrud, and Chandra Chekuri. 2022. Faster and Scalable Algorithms for Densest Subgraph and Decomposition. In *NeurIPS*.

[26] Kun He, Yiwei Sun, David Bindel, John E. Hopcroft, and Yixuan Li. 2015. Detecting Overlapping Communities from Local Spectral Subspaces. In *ICDM 2015*. IEEE Computer Society, 769–774.

[27] Xin Huang, Laks V. S. Lakshmanan, Jeffrey Xu Yu, and Hong Cheng. 2015. Approximate Closest Community Search in Networks. *Proc. VLDB Endow.* 9, 4 (2015), 276–287.

[28] Martin Jaggi. 2013. Revisiting Frank-Wolfe: Projection-Free Sparse Convex Optimization. In *ICML 2013*, Vol. 28. 427–435.

[29] Isabel M. Kloumann and Jon M. Kleinberg. 2014. Community membership identification from small seed sets. In *SIGKDD 2014*. ACM, 1366–1375.

[30] Tommaso Lanciano, Atsushi Miyauchi, Adriano Fazzone, and Francesco Bonchi. 2023. A Survey on the Densest Subgraph Problem and its Variants. *CoRR* abs/2303.14467 (2023).

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

[32] Longlong Lin, Ronghua Li, and Tao Jia. 2023. Scalable and Effective Conductance-Based Graph Clustering. In *AAAI*. 4471–4478.

[33] Chenhao Ma, Reynold Cheng, Laks V. S. Lakshmanan, and Xiaolin Han. 2022. Finding Locally Densest Subgraphs: A Convex Programming Approach. *Proc. VLDB Endow.* 15, 11 (2022), 2719–2732.

[34] Chenhao Ma, Yixiang Fang, Reynold Cheng, Laks V. S. Lakshmanan, and Xiaolin Han. 2022. A Convex-Programming Approach for Efficient Directed Densest Subgraph Discovery. In *SIGMOD '22*. ACM, 845–859.

[35] Alan Mislove, Bimal Viswanath, P. Krishna Gummadi, and Peter Druschel. 2010. You are who you know: inferring user profiles in online social networks. In *WSDM*. ACM, 251–260.

[36] Atsushi Miyauchi, Tianyi Chen, Konstantinos Sotiropoulos, and Charalampos E. Tsourakakis. 2023. Densest Diverse Subgraphs: How to Plan a Successful Cocktail Party with Diversity. In *SIGKDD*. 1710–1721.

[37] Nicholas Nethercote and Julian Seward. 2007. Valgrind: a framework for heavyweight dynamic binary instrumentation. In *PLDI*. ACM, 89–100.

[38] Mark EJ Newman. 2004. Fast algorithm for detecting community structure in networks. *Physical review E* 69, 6 (2004), 066133.

[39] Lorenzo Orecchia and Zeyuan Allen Zhu. 2014. Flow-Based Algorithms for Local Graph Clustering. In *SODA 2014*. SIAM, 1267–1286.

[40] Palla, Gergely, Derényi, Imre, Farkas, Illés, Vicsek, and Tamás. 2005. Uncovering the overlapping community structure of complex networks in nature and society. *Nature* 435, 7043 (2005), 814–818.

[41] Lior Rokach and Oded Maimon. 2005. Clustering methods. In *Data mining and knowledge discovery handbook*. 321–352.

[42] Mauro Sozio and Aristides Gionis. 2010. The community-search problem and how to plan a successful cocktail party. In *SIGKDD*. ACM, 939–948.

[43] Bintao Sun, Maximilien Danisch, T.-H. Hubert Chan, and Mauro Sozio. 2020. KClist++: A Simple Algorithm for k-Clique Densest Subgraphs in Large Graphs. *Proc. VLDB Endow.* 13, 10 (2020), 1628–1640.

[44] Hanghang Tong and Christos Faloutsos. 2006. Center-piece subgraphs: problem definition and fast solutions. In *SIGKDD*. ACM, 404–413.

[45] Twan van Laarhoven and Elena Marchiori. 2016. Local Network Community Detection with Continuous Optimization of Conductance and Weighted Kernel K-Means. *J. Mach. Learn. Res.* 17 (2016), 147:1–147:28.

[46] Nate Veldt, Christine Klymko, and David F. Gleich. 2019. Flow-Based Local Graph Clustering with Better Seed Set Inclusion. In *SDM*. SIAM, 378–386.

[47] Baoning Wu and Kumar Chellapilla. 2007. Extracting Link Spam using Biased Random Walks from Spam Seed Sets. In *AIRWeb*, Vol. 215. 37–44.

[48] Yubao Wu, Ruoming Jin, Jing Li, and Xiang Zhang. 2015. Robust Local Community Detection: On Free Rider Effect and Its Elimination. *Proc. VLDB Endow.* 8, 7 (2015), 798–809.

[49] Yubao Wu, Xiang Zhang, Yuchen Bian, Zhipeng Cai, Xiang Lian, Xueting Liao, and Fengpan Zhao. 2018. Second-order random walk-based proximity measures in graph analysis: formulations and algorithms. *VLDB J.* 27, 1 (2018), 127–152.

[50] Xiwang Yang, Harald Steck, and Yong Liu. 2012. Circle-based recommendation in online social networks. In *SIGKDD*. ACM, 1267–1275.

Algorithm 3: The Frank-Wolfe method [28]

```
Input: A convex programming problem with objective function $f$ and domain $\mathcal{D}$;
       Number of iterations $T$.
Output: A feasible solution $\hat{\alpha^*} \in \mathcal{D}$ with $f(\hat{\alpha^*}) - f(\alpha^*) \leq O(\frac{1}{T})$
1 Let $\alpha^{(0)} \in \mathcal{D}$;
2 for $t \leftarrow 1, 2, ..., T$ do
3     Compute $\hat{\alpha} := \arg\min_{\alpha \in \mathcal{D}} \langle \alpha, \triangledown f(\alpha^{(t-1)}) \rangle$;
4     $\alpha^{(t)} := (1 - \gamma_t) \cdot \alpha^{(t-1)} + \gamma_t \cdot \hat{\alpha}$
5 return $\alpha^{(T)}$;
```

A RELATED WORK

Seeds based local community search. Given a seed set or reference set R, the seeds based local community search problem (or the seed expansion problem) is to detect a community that is highly correlated to R [6, 13, 29, 45]. There are three categories of related methods, metric optimization [16, 39, 45, 46], random walk [3, 8, 26, 29, 44, 49] and cohesive subgraph search (see survey [19]).

Metric optimization based algorithms are designed to identify communities that score highly on a specific quality metric. These metrics are broadly classified into two types: conductance and density. Additional metrics, such as local modularity [24], are not be suitable for application to large-scale networks due to computational constraints or scalability issues. As a result, they are not addressed in this discussion. Conductance-based metrics evaluate the community by its cut and volume. The "cut" represents the number of outbound edges and the volume reflects the aggregate degree of all community members. An example is the local conductance metric, which assesses community quality by considering the cut alongside a volume that incorporates local penalties [39, 46]. Density-based metrics, on the other hand, measure the community by the ratio of the "sum of weights," influenced by the seed set R, to the size of the community [16, 48]. The state-of-the-art metric in seed-based local community search, R-subgraph density [16], exemplifies this approach. Similarly, our NR-subgraph density belongs to the density-based category, offering a nuanced perspective on community connectedness and relevance to the seeds.

Random walk-based algorithms initiate random walks from the seed nodes set R. They can discover multiple communities when the seeds are not confined to a single community [3, 8]. Despite this strength, random walk-based methods have faced criticism for identifying communities with tenuous links to the seed set R [16]. Various efforts have been made to refine these algorithms and enhance the relevance of the detected communities to the seeds [8, 49]. These improvements aim to balance the exploratory nature of random walks with the need for communities that are meaningfully connected to the initial set of interest.

Cohesive subgraph search [19] use the cohesive structures as the cohesiveness and connectivity metrics of community, such as k-core [7, 42], k-truss [2, 27] and k-clique [14]. However, it has been noted [16] that communities identified using cohesive subgraph search often contain a significant number of nodes that are not relevant to the seed set R. This tendency to include extraneous nodes limits the effectiveness of these methods for certain applications. Consequently, cohesive subgraph search has not been selected as a baseline in our work or in previous studies [16, 46] within the context of seed-based local community search.

Densest subgraph. The densest subgraph problem and its variations have been widely studied for over five decades (see survey [30]). Densest subgraph search is to find a subgraph that maximize a measure of density. The traditional measure is the edge density. There are also a lot of variants of the density measure. Densest at-least(most)-k-subgraph is the densest subgraph with size at least(most) k [5]; Densest k-subgraph is the densest subgraph with size k [21]; Fair densest subgraph is the densest subgraphs that has equal colors [4, 36]; Higher-order density is defined as the ratio of the count of a given motif and the subgraph size [20, 43]. Our NR-subgraph density is a kind of density by weighted edges, which is based on the seeds set R.

Existing methods for the densest subgraph and its variants are mainly in two kinds: binary-search and maximum-flow based algorithms [16, 23] and programming-based algorithms [9, 11, 12, 17, 25, 43]. Binary-search and maximum-flow based algorithms use binary-search to find a possible density and use maximum-flow to check whether the possible density is optimal [16, 23]. Programming-based algorithms model the densest subgraph problem into programming problem and utilize programming solver (such as Frank-Wolfe method [17, 43]) or peeling-based algorithms [9, 11, 12] to find the exact or approximate optimal solution.

B THE FRANK-WOLFE METHOD

The Frank-Wolfe method is widely recognized for its application to constrained primal-dual convergence problems, with its efficacy underpinned by the provision of duality gap certificates [28]. A notable advantage of the Frank-Wolfe algorithm is its projection-free nature that there is no need to verify the feasibility of interim solutions at each iteration.

The essence of the Frank-Wolfe method is to iteratively approximate the optimal solution by solving a linear optimization problem over the feasible domain $\mathcal{D}$, as depicted in line 3 of Algorithm 3. Each solution $\hat{\alpha}$ obtained from this process is inherently feasible, thereby obviating the necessity for any additional projection onto $\mathcal{D}$. The algorithm then updates the current solution by taking a step in the direction of $\hat{\alpha}$ with a step size γ_t (line 4).

By unifying a collection of greedy approximations, the Frank-Wolfe algorithm successively narrows the gap to the optimal solution, as proven in [28]. The details of the Frank-Wolfe framework are meticulously outlined in Algorithm 3, where each iteration refines the answer, converging towards optimality while avoiding the computational burden of projection steps.

The convergence rate of the Frank-Wolfe method is determined by the curvature constant, which is defined based on the Bergman divergence.

Theorem B.1 (curvature constant [28]). *The curvature constant of f over $\mathcal{D}$ is*

$$C_f = \sup_{\substack{\alpha_1, \alpha_3 \in \mathcal{D}, \\ \gamma \in [0,1], \\ \alpha_2 = \alpha_1 + \gamma(\alpha_3 - \alpha_1)}} \frac{2}{\gamma^2}(f(\alpha_2) - f(\alpha_1) - \langle \alpha_2 - \alpha_1, \nabla f(\alpha_1) \rangle).$$

Theorem B.2 (convergence rate [28]). *The Frank-Wolfe method with step size $\gamma_t = \frac{1}{t}, t = 1, 2, 3, ...$ satisfies*

$$f(\alpha^{(t)}) - f(\alpha^*) \leq \frac{C_f}{t}.$$

C STABLE SUBSET [17].

We present the concept of stable subset in Definition C.1.

Definition C.1 (stable subset [17]). A subset S is a stable subset with respect to a feasible solution (α, β) with $r = \alpha + \beta$, if (a) For all $u \in S$ and $v \notin S$, $r(u) > r(v)$; (b) For all e that intersects both S and $V \setminus S$, $\alpha_u^e = 0, \forall u \in e \cap S$.

As shown in Definition C.1, a subset S is stable when the nodes in S have the largest ranking in r and receive weights only from $E(S)$. And the remaining nodes $V \setminus S$ receive weights from $E \setminus E(S)$. By theorem 5.3, we know that the maximum rank is an upper bound of NR-subgraph density. Similarly, we can derive that the minimum rank is an lower bound. Since the minimum rank of S is larger than the maximum rank of $V \setminus S$, we can derive that $S^* \subseteq S$.

Theorem C.2. *When $\hat{S^*}$ is a stable subset, $S^* \subseteq \hat{S^*}$.*

In implementing FDPE, we handle the vector α where each α_u^e represents the weight node u receives from edge e. For the edges that intersect $\hat{S^*}$ and $V \backslash \hat{S^*}$, our approach assigns their weights entirely to $V \backslash \hat{S^*}$ to refresh α . This reassignment yields an updated rank vector r', derived from the refreshed α values. Should $\hat{S^*}$ prove a stable subset relative to r', it implies that the true optimal subgraph S^* is contained within $\hat{S^*}$. The stability check, denoted isStableSubset in line 4 of Algorithm 2, therefore runs in time proportional to $O(|E'|)$, with E' specified in line 1 of Algorithm 2. This ensures the time complexity of verifying stability is tied directly to the size of the edge set E'.

D MISSING PROOFS

The proof of **Theorem 3.3**.

Proof. To prove the lemma, we need to prove that $2|E(S)| \geq 2|E(S)| - \sum_{u \in S\setminus R} |N(u,V)| \geq 2|E(S)| - \sum_{u\in S \setminus R}|N(u,S)|$. Since $S \subseteq V$, we have $|N(u,S)| \leq |N(u,V)|$. Thus, we can complete the proof. □

The proof of **Lemma 4.1**.

Proof. From the definition of $\rho_R^+(S)$, we have

$$\begin{aligned} \rho_R^+(S) \cdot |S| &= 2|E(S)| - \sum_{u \in S\setminus R} |N(u,S)| \\ &= \sum_{u\in S} |N(u,S)| - \sum_{u\in S\setminus R}|N(u,S)| \\ &= \sum_{u \in S\cap R}|N(u,S)| = \sum_{e\in E(S)} w_e \end{aligned} \tag{7}$$

□

The proof of **Lemma 4.3**.

Proof. We construct a vector y that $y_u = \frac{1}{|S|}$ if $u \in S$ and $y_u = 0$ if $u \notin S$. It is easy to derive that y is a feasible solution of LP. Then we have $x_e = \frac{1}{|S|}$ for $e \in E(S)$ and $x_e = 0$ for $e \notin E(S)$. Thus, the objective function has a value of $\frac{\sum_{e\in E(S)} w_e}{|S|}$. Combined with Lemma 4.1, we can complete the proof. □

The proof of **Lemma 4.4**.

Proof. Suppose $\rho_R^+(S) < v_y$ for all subgraph $S \subseteq V$, and we have $\sum_{e\in E(S)} w_e < v_y|S|$ according to Lemma 4.1, for all subgraph $S \subseteq V$.

$$v_y = \sum_{e\in E} w_e \cdot x_e < v_y \int_0^1 \sum_{e:x_e = z} z \cdot |S(z)| dz \leq v_y,$$

which is a contradiction. The last inequality comes from C_2 of LP, which restricts $|S(z)| \leq \lfloor \frac{1}{z} \rfloor$ for all $z \in (0,1]$. Let z^+ be continues value larger than z. $S(z)$ is defined as the set of nodes with $z = y_u < z^+$. □

The proof of **Theorem 4.5**.

Proof. Let v^* be the objection function value of y^*. By Lemma 4.3, we have $\rho_R^+(S^*_{\text{ADSS+}}) \leq v^*$. Furthermore, according to Lemma 4.4, we have $\rho_R^+(S^*_{\text{ADSS+}}) \geq v^*$. As a result, $v^* = \rho_R^+(S^*_{\text{ADSS+}})$. □

The proof of **Theorem 5.1**.

Proof. The condition C_0 defines w_e, which are constants. The condition C_1 can be rewrite into $2|E|$ inequations $w_e \leq y_u, \forall u \in e \in E$. For each inequation, we introduce a variable $\alpha_u^e \geq 0$ that $\alpha_u^e \cdot x_e \leq \alpha_u^e \cdot y_u, \forall u \in e \in E$. Similarly, we introduce n variables β according to C_3 of LP, and we get $0 < \beta_u y_u, \forall u \in A$ and $0 = \beta_u y_u, \forall u \notin A$. Sum them together and we get

$$\sum_{e\in E}\left(\sum_{u\in e}\alpha_u^e\right)x_e \leq \sum_{u\in V}\left(\beta_u + \sum_{e:u\in e}\alpha_u^e\right)y_u.$$

Introducing conditions D_0 and D_2, the ineuqation becomes

$$\sum_{e\in E} w_e x_e \leq \sum_{u\in V}(r_u + \beta_u)y_u.$$

The left part of the inequation is the objective function of LP. We need to find the minimum value of the right part and make the minimum value be as large as possible due to the duality.

According to C_2 of LP, we have $\|y\| \leq 1$. As a consequence, we get the objective function of DP

$$\sum_{u\in V}(r_u + \beta_u)y_u = \langle r + \beta, y\rangle \leq \|r + \beta\| \cdot \|y\| \leq \|r+\beta\|.$$

According to C_3 of LP, the nodes in A has the minimum positive value of y. Then, we can deduce D_4 because the right part is the form of summary of product of $r + \beta$ and y. □

The proof of **Theorem 5.3**.

Proof. Let $r_{max} = \max_{u\in V} r_u$ and $x_{max} = \max_{u\in V}(r_u + \beta_u)$. We prove that $r_{max} = x_{max}$. For the nodes $u \notin A$ with $r_u = r_{max}$, we have $\beta_u = 0$ because only the nodes in A may have a positive β, and we can derive that $r_{max} = x_{max}$. When $\forall u \notin A, r_u < r_{max}$, we can infer that $\beta_u = 0$ for the node in A with $r_u = r_{max}$. So we have $r_{max} = x_{max}$.

At last, we can prove the theorem by linear programming duality

$$r_{max} = x_{max} \geq \|r^* + \beta^*\| \geq \rho_R^+(S^*)$$

. □

The proof of **Theorem 5.4**.

Proof. Let (α^*, β^*) be the optimal solution of DP. Let r^* be the vector computed by α^*. We prove that the nodes in S^* have the largest entry in $r^* + \beta^*$. By Lemma 4.1, we know that the total weight in S^* is $\sum_{e\in E(S^*)} w_e$. According to D_0 and D_2, we have $\sum_{u\in S^*} r^*(u) + \beta^*(u) \geq \sum_{e\in E(S^*)} w_e$. Since the objective function is to minimize $\|r^* + \beta^*\|$, we infer that the nodes in S^* has the same value of $r^* + \beta^*$. Otherwise, we can re-assign the weight to reach a more even distribution of $r^* + \beta^*$ and a smaller value of the objective function. Thus, the nodes in S^* are the maximum in $r^* + \beta^*$ and the nodes in $V \setminus S^*$ have smaller value in $r^* + \beta^*$. At last, we can find S^* from $r^* + \beta^*$ by extracting the entries with the largest value. □

The proof of **Lemma 5.6**.

Proof. Let r be the vector computed by Algorithm 1. We prove that $\max_{u \in S^*} \frac{r(u)}{T} \geq \rho_R^+(S^*)$. According to Lemma 4.1, we know that the total weight in S^* is $\sum_{e \in E(S^*)} w_e$. In Algorithm 1, the weight in $E(S^*)$ can only be assigned to the nodes in S^*. Thus, we have

$$\rho_R^+(S^*) = \frac{\sum_{e \in E(S^*)} w_e}{|S^*|} \leq \frac{\sum_{u \in S^*} r(u)}{T|S^*|} \leq \max_{u \in S^*} \frac{r(u)}{T}.$$

At last, we can prove that

$$\frac{\rho_R^+(S^*)}{\rho_R^+(\hat{S^*})} \leq \frac{\max_{u \in S^*} r(u)/T}{\rho_R^+(\hat{S^*})} \leq \frac{\max_{u \in V} r(u)/T}{\rho_R^+(\hat{S^*})} = 1 + \epsilon$$

. □

The proof of **Lemma 5.7**.

Proof. Let $r^{(t)}$ and $\beta^{(t)}$ be the vector in the t_{th} iteration of Algorithm 1 (line 3). Let $x^{(t)} = r^{(t)} + \beta^{(t)}$. Let $s^{(t)} = x^{(t)} - x^{(t-1)}$, i.e. the updated value during the t_{th} iteration. We have that

$$\frac{x^{(t)}}{t} = \frac{x^{(t-1)} + s^{(t)}}{t} = (1 - \frac{1}{t})\frac{x^{(t-1)}}{t-1} + \frac{1}{t}s^{(t-1)}.$$

In the t_{th} iteration of Algorithm 1 (lines 4-9), r/t and β/t is a feasible solution of DP with $\gamma_t = \frac{1}{t}$.

Let (α, β) be a feasible solution of DP. Let r be the vector computed by α by D_0 of DP. The objective function of DP can be seen as $f(\alpha, \beta) = \sum_{u \in V} (r_u + \beta_u)^2$. Further, the derivation is

$$\nabla f(\alpha, \beta) = 2 \sum_{u \in V} (r_u + \beta_u).$$

According to line 3 of Algorithm 3, we need to compute the subproblem

$$\arg \min_{(\alpha, \beta) \in \mathcal{D}} \langle r + \beta, \nabla f(\frac{\alpha^{(t-1)}}{t-1}, \frac{\beta^{(t-1)}}{t-1}) \rangle.$$

This is exactly what FDP do in lines 4-9 of Algorithm 1. FDP solves this subproblem by the basic *B&B* method. It divides the subproblem into problems of making the distribution be as even as possible on the edges.

At last, we can conclude that lines 1-9 of Algorithm 1 is an implementation of the Frank-Wolfe method. □

The proof of **Lemma 5.10**.

Proof. Since $w_e \leq 2$, we have $\sum_e {w_e}^2 \leq 4m'$. □

The proof of **Theorem 5.12**.

Proof. Let S and T be a cut of the flow network that $s \in S$ and $t \in T$. Let $S' = S \setminus \{s\}$ and $T' = T \setminus \{t\}$. We have $\hat{S^*} = S' \cup T'$. The cut is composed of three types of edge from S to T. The first type of edge is from s to T'. The summary of the first type of edge is $\sum_{u \in T'} |N(u, \hat{S^*} \cap R)|$. The second type of edge links from S' to T', with summary of capacity $\sum_{e(u,v):u \in S', v \in T'} w_e$. The third type edges links from S' to t, with summary of capacity $|S'|\rho_R^+(\hat{S^*})$. At last, we can derive that

$$f(\hat{S^*}) = \sum_{u \in T'} |N(u, \hat{S^*} \cap R)| + \sum_{e(u,v):u \in S', v \in T'} w_e + |S'|\rho_R^+(\hat{S^*}).$$

Then, we have

$$\begin{aligned} & f(\hat{S^*}) - \sum_{u \in \hat{S^*}} |N(u, \hat{S^*} \cap R)| \\ &= - \sum_{u \in S'} |N(u, \hat{S^*} \cap R)| + \sum_{e(u,v):u \in S', v \in T'} w_e + |S'|\rho_R^+(\hat{S^*}) \\ &= \sum_{u \in S'} \Big(|N(u, T' \cap R)| - |N(u, \hat{S^*} \cap R)|\Big) + |S'|\rho_R^+(\hat{S^*}) \\ &= - \sum_{u \in S'} |N(u, S' \cap R)| + |S'|\rho_R^+(\hat{S^*}) \\ &= -(2|E(S')| - \sum_{u \in S' \setminus R} |N(u, S')|) + |S'|\rho_R^+(\hat{S^*}) \end{aligned} \tag{8}$$

When $f(\hat{S^*}) - \sum_{u \in \hat{S^*}} |N(u, \hat{S^*} \cap R)| < 0$, we can state that there is a S' denser than $\hat{S^*}$ because $\rho_R^+(S') = \frac{2|E(S')| - \sum_{u \in S' \setminus R} |N(u,S')|}{|S'|} > \rho_R^+(\hat{S^*})$. When $f(\hat{S^*}) = \sum_{u \in \hat{S^*}} |N(u, \hat{S^*} \cap R)|$, we have $S' = \hat{S^*}$. Thus, we can complete the proof. □

E DETAILS OF CASE STUDY

The network is made up of a university karate club, where each node represents a member of the club, and each edge represents a tie between two members of the club. The network represents a conflict that arose within a karate club at a US university, which eventually led to the club splitting into two factions. The dataset is often used to test community detection algorithms, which aim to identify groups within the network that are more densely connected with each other than with the rest of the network. The initial club was led by the individual labeled "Mr. Hi", but a disagreement led to the formation of a rival faction under the leadership of a student named "John A". The case study focuses on the subgroup led by "John A". "John A" and one of his group members serve as the seeds to identify individuals affiliated with them.

F SUPPLEMENTAL EXPERIMENTS

On the image network FLICKER ($n = 105938$, $m = 2316948$), LA/FDP/FDPE takes 125.48/3.67/10.54 ms respectively, and reports subgraphs with average density 4.95/52.28/50.88 and average conductance of 0.98/0.85/0.85 respectively.

On the hyperlink network Notre-Dame ($n = 325729$, $m = 1117563$), LA/FDP/FDPE takes 5.83/0.46/1.69 ms respectively, and reports subgraphs with average density 3.51/8.22/8.22 and average conductance of 0.88/0.88/ 0.88 respectively.

On the road network USAROAD ($n = 23947347$, $m = 28854312$), LA/FDP/FDPE takes 16.81/14.38/ 24.73 ms respectively, and reports subgraphs with average density 0.34/1.07/1.07 and average conductance of 0.75/0.20/0.20 respectively.