Connected-Dense-Connected Subgraphs in Triple Networks

Dhara Shah\(^1\), Yubao Wu\(^1\), Sushil Prasad\(^1\), and Danial Aghajarian\(^2\)

\(^1\) Department of Computer Science, Georgia State University, Atlanta 30303, USA
\{dshah8, ywu28, sprasad\}@gsu.edu
\(^2\) daghajarian@cs.gsu.edu

Abstract. Finding meaningful communities - subnetworks of interest within a large scale network - is a problem with a variety of applications. Most existing work towards community detection focuses on a single network. However, many real-life applications naturally yield what we refer to as Triple Networks. Triple Networks are comprised of two networks, and the network of bipartite connections between their nodes. In this paper, we formulate and investigate the problem of finding Connected-Dense-Connected subgraph (CDC), a subnetwork which has the largest density in the bipartite network and whose sets of end points within each network induce connected subnetworks. These patterns represent communities based on the bipartite association between the networks. To our knowledge, such patterns cannot be detected by existing algorithms for a single network or heterogeneous networks. We show that finding CDC subgraphs is NP-hard and develop novel heuristics to obtain feasible solutions, the fastest of which is \(O(n\log n + m)\) with \(n\) nodes and \(m\) edges. We also study different variations of the CDC subgraphs. We perform experiments on a variety of real and synthetic Triple Networks to evaluate the effectiveness and efficiency of the developed methods. Employing these heuristics, we demonstrate how to identify communities of similar opinions and research interests, and factors influencing communities.

Keywords: Triple Networks · Unsupervised community detection · max-flow densest bipartite subgraph · NP-Hard · greedy node deletions · local search

1 Introduction

Community detection is a key primitive with a wide range of applications in real world \[7\]. Most existing work focuses on finding communities within a single network. In many real-life applications, we can often observe Triple Networks consisting of two networks and a third bipartite network representing the interaction between them. For example, in Twitter, users form a follower network, hashtags form a co-occurrence network, and the user-hashtag interactions form a bipartite network. The user-hashtag interactions represent a user’s posts or tweets containing a hashtag. Figure 4 shows a real Twitter Triple Network. The nodes on the left part represent users and those on the right represent hashtags.
The edges among the nodes on the left represent a user following other user. The edges among the nodes on the right represent two hashtags appearing in the same tweet. The edges in between represent a user interacting with tweets containing a hashtag. This Triple Network model can ideally represent many real world applications such as taxi pick-up-drop-off networks, Flixster user-movie networks, and author-paper citation networks.

In general, maximizing the density of connections in the Connected-Dense-Connected (CDC) subgraph of a triple network is an unsupervised method for approximating the communities affiliated with the attributes. In the twitter example the density is the number of connections between users and tweets and reflects the degree to which the users are engaged with those tweets. Therefore finding the CDC subgraph is likely to be a useful approach to understanding social and other networks. Given a Triple Network consisting of two graphs \( G_a(V_a, E_a) \) and \( G_b(V_b, E_b) \) and a bipartite graph \( G_c(V_a, V_b, E_c) \), the CDC consists of two subsets of nodes \( S \subseteq V_a \) and \( T \subseteq V_b \) such that the induced subgraphs \( G_a[S] \) and \( G_b[T] \) are both connected and the density of \( G_c[S,T] \) is maximized.

In the Twitter Triple Network in Figure 1, we observe two CDC subgraphs: the one at the top with \( S_1 = \{ \text{Tammie, Bill, Stacy, Vivien} \} \) and \( T_1 = \{ \text{Patriots, TomBrady, SuperbowlChamps, halftime, Maroon5} \} \), and the one at the bottom with \( S_2 = \{ \text{Mike, Daniel, Rob, Brent} \} \) and \( T_2 = \{ \text{Rams, toughgame, Maroon5} \} \). In either of the two CDCs, the left and right networks are connected and the middle one is dense. These CDCs are meaningful. The CDC at the top shows that Patriots’ fans are praising Tom Brady and are happy to be champions again. The CDC at the bottom shows that LA Rams’ fans are disappointed to lose the game.

Our problem is different from finding co-dense subgraphs\([11,14]\) or coherent dense subgraphs\([9,13]\), whose goal is to find the dense subgraphs preserved across multiple networks with the same types of nodes and edges. In our problem, the left and right networks contain different types of nodes and the edges in the three networks represent different meanings. Our problem is also different than the densest connected subgraphs in dual networks\([20]\). Dual networks consist of one set of nodes and two sets of edges. Triple Networks consist of two sets of nodes and three sets of edges. Triple Networks can degenerate to dual networks.
when the two sets of nodes are identical and the bipartite links connect each node to its replica.

2 Background and related work

The problem of finding a densest subgraph of a graph has been well studied by data mining community. At the core, this problem asks for finding subgraphs with the highest average degree. This problem has been solved in polynomial time using max-flow min-cut approach \cite{8}. Inspired by this approach, the problem of finding densest subgraph in a directed graph has also been solved in polynomial time \cite{12}. The prohibitive cost of these polynomial time algorithms has been addressed with 2-approximation algorithm \cite{6}. However, variations of densest subgraph problems, such as discovery of densest subgraph with \emph{k} nodes, have been shown to be NP-hard \cite{3}. On the other hand, the problem of finding densest subgraph with pre-selected seed nodes is solvable in polynomial time \cite{15}.

The solutions above are designed for homogeneous information network structure where the nodes and edges have just one type. Heterogeneous information networks \cite{18} – the networks with multiple node and edge types – have been a new development in the field of data mining. Heterogeneous network structure provides a model for graph infusion with rich semantics. The Triple Networks introduced in this paper are a type of heterogeneous network with node types \( V_a \) and \( V_b \), and edge types \( E_a, E_b \) and \( E_c \). Our work can be categorized as unsupervised clustering in heterogeneous network. Parallel to our work, Boden et al. discuss a density based clustering approach of \( k \)-partite graphs in heterogeneous information structure \cite{4}. In this work, two types of nodes \( V_a \) and \( V_b \) are considered. With node type specific hyper-parameters and the bipartite connections \( E_c \), the connections \( E_a \) and \( E_b \) are inferred. This method of clustering is different from our work where \( E_a \) and \( E_b \) are part of the network, and the definition of density is hyper-parameter free. Boden et al. detect communities by subspace clustering on nodes’ projection to attribute space. In contrast, our work of finding CDC subgraphs cannot be inferred as a subspace clustering technique. Though both works produce iterative refinement algorithms, the former concentrates on improving inference of \( E_a \) and \( E_b \) iteratively.

The closest network schema to our work is dual networks \cite{20}, discovered by Wu et al. A dual network is comprised of two networks having the same set of nodes but different types of edges. These two networks are inferred as physical and conceptual networks. Wu et al. provide 2-approximation algorithms for NP-hard problem of finding subgraphs that are densest in conceptual network, and are connected in physical network. Though the network architecture and subgraph patterns are different, our work is inspired by the pruning methods and variants proposed in this work. To the best of our knowledge there is no comparable algorithm for finding a densest CDC subgraph.
3 Triple network, CDC subgraphs and variants

In this section we define Triple Network, CDC subgraph and its variants. We prove that finding CDC subgraph and variants from a Triple Network is NP-hard.

Definition 1 (Triple network). Let \( G_a(V_a, E_a) \) and \( G_b(V_b, E_b) \) represent graphs of two networks. Let \( G_c(V_a, V_b, E_c) \) represent the bipartite graph between \( G_a \) and \( G_b \). \( G(V_a, V_b, E_a, E_b, E_c) \) is the Triple Network generated by \( G_a, G_b \) and \( G_c \).

We abbreviate a Triple Network as \( G \). An example of Triple Network is illustrated in figure 2(a).

The subgraphs induced by \( S_a \subset V_a \) and \( S_b \subset V_b \) in networks \( G_a, G_b \) and \( G_c \) are denoted by \( G_a[S_a], G_b[S_b] \) and \( G_c[S_a, S_b] \). For brevity, we denote this sub Triple Network, a set of three subgraphs, as \( G[S_a, S_b] \).

Definition 2 (Density of a Triple Network). Given a Triple Network \( G[S_a, S_b] \), its density is defined as \( \rho(S_a, S_b) = \frac{|E_c(S_a, S_b)|}{\sqrt{|S_a||S_b|}} \), where \( |E_c[S_a, S_b]| \) is the number of bipartite edges in subgraph \( G_c[S_a, S_b] \), \( |S_a| \) is the number of nodes in \( G_a[S_a] \) and \( |S_b| \) is the number of nodes in \( G_b[S_b] \).

For example, the density of sub Triple Network in figure 2(b) with \( S_a = \{1, 2, 3\} \) and \( S_b = \{5, 6, 7\} \) is \( \rho(S_a, S_b) = \frac{|E_c(S_a, S_b)|}{\sqrt{|S_a||S_b|}} = \frac{6}{\sqrt{3\times3}} = 2 \).

By definition of density, only the bipartite edges of a Triple Network contribute to the density. Hence, the density of a Triple Network \( G \) is same as the density of its bipartite subgraph \( G_c \).

3.1 Connected-Dense-Connected (CDC) subgraphs

Definition 3 (CDC subgraph). Given Triple Network \( G(V_a, V_b, E_a, E_b, E_c) \), a CDC subgraph is a sub Triple Network \( G[S_a, S_b] \) such that

1. \( G_a[S_a] \) and \( G_b[S_b] \) are connected subgraphs, and
2. the density \( \rho(S_a, S_b) \) is maximized.

For example, the density of each CDC subgraph in figure 2(b) is 2, higher than density of any other sub Triple Network of the Triple Network 2(a) that is connected in \( G_a \) and \( G_b \). A Triple Network can have multiple CDC subgraphs.

**Theorem 1.** Finding a CDC subgraph in a Triple Network is NP-Hard.

**Proof.** We prove that finding a CDC subgraph is a reduction of set-cover problem. Please refer to [17] for details.

### 3.2 Variants of CDC subgraph

CDC subgraphs stipulate connectedness of \( G_a(S_a) \) and \( G_b(S_b) \). Alleviating this connectivity constraint, we define OCD subgraphs for which exactly one of \( G_a(S_a) \) or \( G_b(S_b) \) is connected.

**Definition 4 (OCD subgraph).** Given a Triple Network \( G(V_a, V_b, E_a, E_b, E_c) \) a OCD subgraph is a sub Triple Network \( G[S_a, S_b] \) such that

1. Exactly one of \( G_a[S_a] \) or \( G_b[S_b] \) is connected, and
2. The density \( \rho(S_a, S_b) \) is maximized.

For example, the sub Triple Network \( G[\{1, 2, 3, 4, 5\}, \{6\}] \) with the highest density 2.23 in figure 2(c) is an OCD subgraph as \( G_a[\{5\}] \) is connected. A Triple Network can have multiple OCD subgraphs.

### Adding constraints to CDC and OCD subgraphs

We observe that CDC patterns are meaningful around pre-selected nodes in \( G_a(S_a) \) or \( G_b(S_b) \). We identify these pre-selected nodes as seeds. We introduce CDC and OCD subgraphs with seed constraints, where \( G_a(S_a) \) or \( G_b(S_b) \) should maintain their connectivity constraints while containing the seeds.

**Definition 5.** (CDC seeds). Given a Triple Network \( G(V_a, V_b, E_a, E_b, E_c) \) and sets of seed nodes \( V_1 \subset V_a \) and \( V_2 \subset V_b \), the CDC_seeds subgraph consists of sets of nodes \( S_a, S_b \) such that \( V_1 \subset S_a \), \( V_2 \subset S_b \), \( G_a[S_a] \) and \( G_b[S_b] \) are connected and density of \( G[S_a, S_b] \) is maximized.

**Definition 6.** (OCD seed). Given a Triple Network \( G(V_a, V_b, E_a, E_b, E_c) \) and a set of node \( V \) with \( V \subset S \), the OCD_seed consists of sets of nodes such that either \( G_a[S] \) or \( G_b[S] \) is connected and the density of \( G[S, E_c[S]] \) is maximized.

Finding OCD, CDC_seeds and OCD_seed subgraphs in a Triple Network is NP-hard. Similar set-cover arguments as in Theorem 1 could be used to prove it. Please refer to [17] for details.
4 Heuristic algorithms

Finding CDC subgraphs is NP-hard. Hence in this section, we propose heuristic algorithms for finding feasible solutions. We propose algorithms with following two approaches.

In the first approach, we first obtain the densest bipartite subgraph $G_c[\mathcal{S}_a, \mathcal{S}_b]$. We then find the connected components of $G_a[\mathcal{S}_a]$ and $G_b[\mathcal{S}_b]$ using BFS. Thus we obtain connected sub Triple Networks with bipartite edges in $G_c[\mathcal{S}_a, \mathcal{S}_b]$. We choose the highest density results as feasible CDC subgraphs. Since the time complexity of obtaining densest bipartite subgraph is higher than that of BFS, algorithms in sections 4.1 and 4.2 focus on improving the complexity of finding the densest bipartite subgraphs.

In the second approach, we obtain local CDC subgraphs with given seed nodes from $V_a$ and $V_b$ by adding highest bipartite degree nodes while maintaining the connectedness in $G_a$ and $G_b$. This Local Search algorithm is presented in section 4.3.

We observe that real-world Triple Networks are sparse in $E_c$. We also observe that a connected densest subgraph exists for a bipartite graph[16]. Exploiting these virtues, we divide the bipartite graph $G_c[\mathcal{V}_a, \mathcal{V}_b]$ into smaller connected bipartite subgraphs and apply the densest subgraph algorithms only for larger subgraphs. This optimization significantly reduces the running-times of our algorithms.

4.1 Maxflow Densest Subgraph (MDS)

MDS algorithm, formalized as Algorithm 1, finds a densest bipartite subgraph of a Triple Network in polynomial time using max-flow min-cut strategies discussed in [12] and [8]. We provide the details of our derivation including proofs, over all approach and examples in [16]. The density difference of any two subgraphs of a bipartite graph $G_c[\mathcal{V}_a, \mathcal{V}_b]$ is no less than $\frac{1}{|\mathcal{V}_a|^2|\mathcal{V}_b|^2}$. Hence, the binary search in MDS with step size $\frac{1}{|\mathcal{V}_a|^2|\mathcal{V}_b|^2}$ halts in $O(|\mathcal{V}_a|^{3/2}|\mathcal{V}_b|^{3/2})$ iterations. Within each iteration, the min cut is calculated in $O((|\mathcal{V}_a| + |\mathcal{V}_b|)^2(2(|\mathcal{V}_a| + |\mathcal{V}_b|) + |E_c|))$. Hence, the complexity of MDS is $O(|\mathcal{V}_a|^{4.5}|\mathcal{V}_b|^{4.5})$. Adding the cost of BFS for finding connected components in $G_a$ and $G_b$, the upper-bound still remains unchanged.

Though polynomial time, the prohibitive time complexity of MDS algorithm makes it impracticable to employ for large Triple Networks. By using MDS results on smaller bipartite graphs as a baseline, we develop heuristics in section 4.2.

4.2 Greedy Node Deletions

In this section, we present heuristics to obtain a dense bipartite subgraph with a reduced time complexity.

The first heuristic is to iteratively delete the nodes with the lowest bipartite degree and yield the densest subgraph obtained in the process. This algorithm of
Algorithm 1 Maxflow Densest Subgraph (MDS)

**Input:** Triple Network $G(V_a, V_b, E_a, E_b, E_c)$ with $V_a \neq \emptyset, V_b \neq \emptyset$

**Output:** A densest bi-partite subgraph $G_c[S_a, S_b]$ of $G$

1. possible_ratios = $\{ \frac{i}{j} \mid i \in [1, \cdots |V_a|], j \in [1, \cdots |V_b|] \}$
2. densest_subgraph = $\phi$, maximum_density = $\rho(V_a, V_b)$
3. for ratio_guess $r \in$ possible_ratios do
4.   low = $\rho(V_a, V_b)$, high = $\sqrt{|V_a||V_b|}$, $g = G_c[V_a, V_b]$
5.   while high - low $\geq \frac{|V_a||V_b|}{2}$ do
6.     mid = $\frac{\text{high} + \text{low}}{2}$
7.     construct a flow graph $G'$ as described in [16] and find the minimum s-t cut $S, T$
8.     if $g' \neq \phi$ then
9.       $g' = S \setminus \{$source node s$\}$
10.      low = $\max\{\text{mid}, \rho(g)\}$
11.     else high = mid
12.   if maximum_density $<$ low then
13.       maximum_density = low
14.       densest_subgraph = $g$

Greedy Node Deletion using degrees (GND) is formalized as Algorithm 2, where criterion in line 4 is node degree.

However, degree is not the best measure of a node’s impact on density. Figure 2(a) illustrates that GND deletes the nodes $\{3, 4, 5\}$ iteratively. This order of deletions leads to missing the densest bipartite subgraph $\{1, 2, 3, 4, 5\}, \{6\}$ in figure 2(c). Instead of accounting for the connections of a node, the percent of the possible connections of that node may serve as a better measure of the node’s impact on density. With this intuition, we define rank of a node.

**Definition 7 (Rank).** Let $G(V_a, V_b, E_a, E_b, E_c)$ be a Triple Network. For $v_a \in V_a$, rank($v_a$) = $\frac{d(v_a)}{|V_b|}$ and for $v_b \in V_b$, rank($v_b$) = $\frac{d(v_b)}{|V_a|}$.

Using the lowest rank as the deletion criterion, we modify Algorithm 2 to formulate Greedy Rank Deletion (GRD) algorithm where the criterion of deletion in line 4 is rank.

GND and GRD delete nodes sequentially. To expedite this process, we delete all the nodes satisfying the deletion criterion in bulk in each iteration instead. This idea is formulated as fast Rank Deletion (FRD) Algorithm in 3. These bulk deletions do not lower the time complexity upper-bound, but the number of iterations decreases exponentially. The deletion criterion of FRD could be tuned by choosing different $\epsilon$ values from $(-1, 1)$ with $\epsilon$ values from lower to higher resulting in less to more deletions per iteration.

By maintaining two $\{\text{degree}: \text{node}\}$ Fibonacci heaps and an index on the nodes, the time complexity of these greedy deletion algorithms is $O((V_a + V_b) log(V_a +$
Adding the cost of BFS for connected components in $G_\alpha$ and $G_\beta$, the total time complexity for obtaining CDC subgraphs is $O((V_\alpha + V_\beta)\log(V_\alpha + V_\beta) + E_c + E_\alpha + E_\beta)$.

**Algorithm 2** Greedy Node Deletions

**Input:** Triple Network $G(V_\alpha, V_\beta, E_\alpha, E_\beta, E_c)$, with $V_\alpha \neq \emptyset, V_\beta \neq \emptyset$.

**criterion** to delete nodes

**Output:** A densest subgraph $G_c[S_a, S_\beta]$ of $G$

1. $S_a = V_\alpha, S_\beta = V_\beta$
2. $\maximim\text{density} = \rho(V_\alpha, V_\beta)$
3. while $V_\alpha \neq \emptyset$ and $V_\beta \neq \emptyset$
4. $v$ = node with minimum $\rho(V_a, V_b)$
5. $V_a = V_a \setminus \{v\}, V_\beta = V_\beta \setminus \{v\}$
6. if $\maximim\text{density} < \rho(V_a, V_b)$ then
7. $S_a = V_a, S_\beta = V_\beta$
8. $E_c = E_c[V_a, V_\beta]$
9. return $G_c[S_a, S_\beta]$

**Algorithm 3** Fast Rank Deletion (FRD)

**Input:** Triple Network $G(V_\alpha, V_\beta, E_\alpha, E_\beta, E_c)$, with $V_\alpha \neq \emptyset, V_\beta \neq \emptyset$.

value of $\epsilon \in (-1, 1)$

**Output:** A densest bi-partite subgraph $G_c[S_a, S_\beta]$ of $G$

1. $S_a = V_\alpha, S_\beta = V_\beta$
2. $\maximim\text{density} = \rho(V_\alpha, V_\beta)$
3. while $V_\alpha \neq \emptyset$ and $V_\beta \neq \emptyset$
4. $\bar{r} = \text{average node rank in } G$
5. $V = \{v \in V_\alpha \cup V_\beta | \text{rank}(v) < (1 + \epsilon)\bar{r}\}$
6. $V_a = V_a \setminus V, V_\beta = V_\beta \setminus V$
7. if $\maximim\text{density} < \rho(V_a, V_b)$ then
8. $S_a = V_a, S_\beta = V_\beta$
9. $E_c = E_c[V_a, V_\beta]$
10. return $G_c[S_a, S_\beta]$

**Algorithm 4** Local Search (LS)

**Input:** $G(V_\alpha, V_\beta, E_\alpha, E_\beta, E_c)$, with $V_\alpha \neq \emptyset, V_\beta \neq \emptyset$.

$\text{seed}S_a$ = Set of seeds in $V_\alpha$.

$\text{seed}S_\beta$ = Set of seeds in $V_\beta$.

**Output:** A sub Triple Network $G[S_a, S_\beta]$ of $G$

1. $S_a = \text{Spanning tree of } \text{seed}S_a$ in $G_\alpha$
2. $S_\beta = \text{Spanning tree of } \text{seed}S_\beta$ in $G_\beta$
3. $\delta(S_a) = \{v \in S_a | S_a \text{ contains } v \text{'s neighbor in } G_\alpha\}$, Boundary of $S_a$ in $G_\alpha$
4. $\delta(S_\beta) = \{v \in S_\beta | S_\beta \text{ contains } v \text{'s neighbor in } G_\beta\}$, Boundary of $S_\beta$ in $G_\beta$
5. $\text{nbhd}$, the adjacency list of $V_\alpha$ in $G_\alpha$ and $V_\beta$ in $G_\beta$
6. $\maximim\text{density} = \rho(G[S_a, S_\beta])$
7. do
8. $v$ = node in $\delta(S_a) \cup \delta(S_\beta)$ with the highest bi-partite connections to $S_a \cup S_\beta$
9. $S_a = S_a \cup v$ if $v \in V_\alpha, S_\beta = S_\beta \cup v$ if $v \in V_\beta$
10. $\delta(S_a) \cup \delta(S_\beta) = \delta(S_a) \cup \delta(S_\beta) \cup \text{nbhd}(v) \setminus \{v\}$
11. $\maximim\text{density} = \max(\maximim\text{density}, \rho(G[S_a, S_\beta]))$
12. while $\rho(G_c[S_a, S_\beta]) \geq \maximim\text{density}$ and $\delta(S_a) \cup \delta(S_\beta) \neq \emptyset$
13. return $G[S_a, S_\beta]$
4.3 Local Search

In this section, we introduce Local Search (LS), a bottom-up approach for obtaining CDC subgraphs around seeds – pre-selected nodes. Let $S_a$ and $S_b$ be the spanning trees of desired seeds in $V_a$ and $V_b$. LS, outlined as Algorithm 4, iteratively includes previously un-included boundary node of $S_a \cup S_b$ with the maximum adjacency value to the set of included nodes. LS hence finds CDC subgraph by adding nodes that increase the density while maintaining connectedness of $S_a$ and $S_b$.

As illustrated in experiments, LS yields local patterns with good semantic value. In practice, the search stops in a few iterations and hence LS is empirically the fastest algorithm yet.

4.4 Algorithms for variants

We obtain OCD subgraphs as bi-products of mining CDC subgraphs. For MDS and Greedy Node Deletions, the resultant sub Triple Networks maintaining exactly one connectedness with the highest density are yielded as OCD subgraphs. We instantiate LS algorithm with either $S_a$ or $S_b$ to be empty and obtain CDC seed and OCD seed subgraphs.

5 Experiment results

In this section, we evaluate the effectiveness and efficiency of the proposed methods through comprehensive experiments on real and synthetic datasets. We demonstrate the effectiveness of CDC and OCD subgraphs by illustrating novelty of the information obtained from these subgraphs on real Triple Networks. We demonstrate the efficiency of our algorithms by measuring the running times of the algorithms and the density of the resultant CDC subgraphs. The programming language employed is Python 2.7 and the experiments were conducted on Intel Core i7 3.6Gz CPU with 32G memory.

5.1 Real Triple Networks

We employ Triple Networks constructed from Twitter, NYC taxi data, Flixter and ArnetMiner coauthor datasets. Table 1 describes the statistics of these real Triple Networks.

**NYC Taxi data** New York City (NYC) yellow cab taxi data is a dataset where each taxi trip’s pick-up and drop-off point is a geographic location in decimal degrees. We consider the trips from June 2016 to construct a Triple Network. The geographic location accuracy of this dataset is thresholded up to 5 decimal points, preserving granularity to different door-entrances. Hence $G_a$ and $G_b$ are the networks of pick-up and drop-off points. In these networks, edges connect the points within 50 meters of haversine distance. The taxi trips are represented as $E_c$. 
**Twitter network** Twitter is a social media for micro-blogging where users can follow each other for updates. To extract meaningful user-follower relationships, we choose popular news networks, namely CNN, Huffington Post and Fox News, and randomly extract a few thousand of their intersecting followers. We iteratively grow this network by including followers of existing nodes using Twitter’s REST API. At each iteration, we threshold users by number of recent tweets and number of followers. Thus, we construct a 5-hop users-followers network $G_a$, where two users are connected if one follows the other. We collect different hashtags from these users’ tweets with $E_c$ as users posting hashtags. We consider two hashtags connected if they appear in the same tweet, and thus construct hashtag co-occurrence network as $G_b$.

**ArnetMiner Coauthor data** ArnetMiner Coauthor dataset [19] is comprised of two types of networks: authors and their co-author relationships as $G_a$, and their research interests as $G_b$, with $E_c$ as relations of authors to their research interests. We consider two research interests linked if they co-occur in an other’s list of research interests.

**Flixter data** Flixter [10] is a social network of users and their movie ratings. We consider the users social network as $G_a$, the users’ rankings of movies as $E_c$, and movies as $V_b$. With no sufficient information, we consider $|E_b| = 0$.

### Table 1. Real triple-networks on NY Taxi data (TX), Twitter (TW), ArnetMiner (AM), and Flixter (FX) data

| Data | $|V_a|$ | $|E_a|$ | $|V_b|$ | $|E_b|$ | $|E_c|$ |
|------|--------|--------|--------|--------|--------|
| TX   | 735896 | 31513503 | 794085 | 13465065 | 2066569 |
| TW   | 61726  | 7008491 | 3679824 | 2896925 | 48269139 |
| AM   | 1712433 | 4258946 | 3901018 | 955490 | 12589981 |
| FX   | 786936 | 7058819 | 48594 | 0 | 8196077 |

### Table 2. Synthetic Random and R-MAT networks

| $|V_a|$ | $|E_a|$ | $|V_b|$ | $|E_b|$ | $|E_c|$ |
|--------|--------|--------|--------|--------|
| $2^{21}$ | $5 \times 10^7$ | $2^{20}$ | $5 \times 10^6$ | $10^4$ |
| $2^{20}$ | $10^7$ | $2^{20}$ | $10^7$ | $2 \times 10^4$ |
| $2^{22}$ | $2 \times 10^7$ | $2^{22}$ | $2 \times 10^7$ | $4 \times 10^4$ |
| $2^{22}$ | $4 \times 10^7$ | $2^{22}$ | $4 \times 10^7$ | $8 \times 10^4$ |

### 5.2 Synthetic Triple Networks

We generated random networks with synthetic $G_a, G_b$ and $G_c$ having random edges in order to evaluate efficiency of our algorithms. To approximate real world Triple Networks, we also generated R-MAT networks with $G_a$ and $G_b$ having R-MAT edges [52] and $G_c$ having random edges. We generated four different configurations for random and R-MAT networks (see Table 2). To the best of our knowledge, there are no algorithms to obtain CDC subgraphs. However, the MDS algorithm provides the densest bipartite subgraph, and hence is an upper-bound to the density of CDC. The high time complexity of MDS algorithm limits its applicability with real problems and thus we used synthetic benchmarks.

### 5.3 Effectiveness Evaluation on Real Networks

We illustrate the effectiveness of CDC subgraphs and variants by emphasizing the knowledge gain from these patterns obtained from real networks. These figures demonstrate that CDC subgraphs and variants are communities detected by the
Connected-Dense-Connected Subgraphs in Triple Networks

Table 3.

| $|V_a| = |V_b|$ | $|E_a| = |E_b|$ | $|E_c|$ | Random networks: MDS/GRD bipartite | RMAT networks: MDS/GRD bipartite |
|---|---|---|---|---|
| $2^{15}$ | $3.125 \times 10^5$ | $6.25 \times 10^4$ | 0.9897 | 1.1970 |
| $2^{16}$ | $6.25 \times 10^5$ | $1.25 \times 10^4$ | 0.9901 | 1.1898 |
| $2^{17}$ | $1.25 \times 10^5$ | $2.5 \times 10^4$ | 0.9865 | 1.2101 |
| $2^{18}$ | $2.5 \times 10^5$ | $5 \times 10^4$ | 1.0010 | 1.1985 |
| $2^{19}$ | $5 \times 10^5$ | $1 \times 10^4$ | 0.9753 | 1.2021 |

Fig. 3. CDC and OCD subgraphs from NY Taxi data. Triangles and circles represent pick-up and drop-off points respectively.

(a) CDC subgraph with pick-up locations on 6th Avenue between 18th and 27th street populated with food and shopping destinations, and drop-off locations on 8th Avenue. This CDC subgraph is generated by observing the 6:00-7:00 pm traffic on June 4, 2016. The drop-off points are clustered near 42nd street Port Authority bus terminals of city transit. This CDC subgraph gives a directional flow of human migration in a short distance during a specific time-frame. Figure 3(b) illustrates OCD subgraph with pick-up seeds.

NYC Taxi data

Fig. 4. CDC subgraphs from Twitter. Users-followers networks on the left and hashtag networks on the right.

strong associations to their attributes. These subgraphs identify similar opinions, research interests and factors influencing communities. They are also effective tools for hot-spot detection and fraud detection.

NYC Taxi data

Fig. 3 illustrates CDC and OCD subgraphs with pick-up and drop-off points as triangles and circles respectively.
near 5th Avenue and Central Park South. This subgraph is generated by observing 4:00-8:00 pm traffic on June 1, 2016. The pick-up points are scattered along Manhattan and the drop-off points are clustered around Pennsylvania Station, a public transit hub. Thus, OCD subgraphs could be equivalents to hot-spot detection.

**Twitter Network** Figure 4 represents CDC subgraphs obtained from Twitter Network. Left and right subgraphs represent users-followers and hashtag networks. We remove usernames to protect user privacy. These figures represent twitter users and their opinions about SuperBowl contenders, Patriots and LA Rams. Hence, CDC subgraphs can identify communities with contrasting opinions.

**ArnetMiner coauthor data** Figure 5 depicts CDC seeds and OCD seed subgraphs from ArnetMiner Triple Network. Left and right subgraphs represent author-coauthor and research-interest networks.

Figure 5(a) is a CDC seeds subgraph with randomly chosen author seed \{M.Kandimir\} and interest seeds \{power,energy\}. This pattern yields author seed’s associates working on related research topics of interest seeds. Figure 5(b)
is OCD seed subgraph with interest seeds chosen as \{algorithm, gpu, performance\}. This pattern yields 16 authors and their respective co-author networks with publications related to interests seeds. Thus, even with the given seeds, the CDC and OCD subgraphs are different from supervised community detection.

**Flixter data** Figure 6 depicts OCD subgraphs illustrating influence of movies on users. Left and right subgraphs represent the users’ social networks and the movies networks. The users networks are connected.

Figure 6(a) is an OCD seed subgraph with users seeds, chosen at random. The right network represents movies with 5 star rankings by the users on the left. This pattern hence finds the movies influencing the friend-circle of the seed users. An OCD subgraph in figure 6(b) depicts a suspicious ranking activity, where the 3 users on the left give a 5 star ranking to 144 movies on the right. CDC and OCD subgraphs hence illustrate the power of potential fraud detection.

### 5.4 Efficiency evaluation

We evaluate the efficiency of our heuristic algorithms by their running-time and the quality of the resulting CDC subgraphs from real and synthetic networks.

**Greedy node deletions** The running-times of MDS, GND, GRD, FRD algorithms on real, random and R-MAT networks are depicted in Figure 7. The x axis represents the number of nodes in $V_a \cup V_b$ and the y axis represents log scale of seconds. Each point represents running-time of the algorithm for given network. The running-time of MDS algorithm for larger networks is more than 24 hours, when we halted the algorithm computations. Running-times increase with network size, but vary a little for random and R-MAT graphs of the same size. FRD with $\epsilon = 0$ is the fastest algorithm.

We discover that GRD yields the densest bipartite subgraph among all algorithms. The densities of CDC subgraphs obtained by GND, GRD and FRD from random and R-MAT networks are presented in table 4 and 5. For each graph, DBP represents the density of the densest bipartite graph obtained by GRD, without being connected in $G_a$ or $G_b$. The ratio, DBP/CDC density, varies a little with the network size. This trend is observed across all network types and algorithms. GRD produces the best and FRD with $\epsilon = 0$ produces the least accurate results.

**Local Search (LS)** Given the seeds of $V_a$ and $V_b$, LS produces meaningful, locally dense CDC patterns. We evaluate the efficiency of LS algorithm by measuring its running-times with 2, 4 and 8 seeds. Figure 8 presents the running-times of LS. The x axis represents the number of nodes in $V_a \cup V_b$ and the y axis represents running-times in seconds. Each point represents running-time of FRD for given network and seed configuration. The seeds are chosen randomly in the same connected components. The boundaries $\delta(S_a)$ and $\delta(S_b)$ grow larger with increase in the number of seeds. Hence the running-time of LS increases with the number of seeds. We observe similar trends from real networks. In synthetic...
Fast Rank Deletion (FRD) The purpose of FRD is to obtain feasible CDC subgraphs faster. This is achieved by deleting all the nodes with degree less than \((1 + \epsilon) \times \text{average degree} \) at each pass. However, lower \(\epsilon\) values result in fewer deletions per pass, defying the purpose of FRD. Higher \(\epsilon\) values result in more deletions per pass, lowering the densities of the resulting CDC subgraphs. Hence the meaningful results are obtained with \(\epsilon\) values in the range of interval \([-0.4, 0.4]\).

Figures 9(a) and 9(b) represent the running-times of FRD. The x axis represents different \(\epsilon\) values and the y axis represents running-times in log scale of seconds. Each point represents running-time of FRD for given network and \(\epsilon\) configurations. Increase in \(\epsilon\) value causes higher amount of deletion per pass,
resulting in fewer passes. Hence, the running-times decrease with the increase of \( \epsilon \).

Figures 9(c) and 9(d) represent the density change of resultant CDC subgraphs for given \( \epsilon \) value, with respect to \( \epsilon = 0 \). The x axis represents different \( \epsilon \) values, and the y axis represents the ratio, Density of CDC for \( \epsilon = 0 \)/Density of CDC with given \( \epsilon \). Each point represents this density ratio obtained by FRD, for given network and \( \epsilon \) configurations. Higher \( \epsilon \) values result in more deletions per pass, lowering the densities of the resulting CDC subgraphs. Hence, the density ratio increases as the \( \epsilon \) value decreases. We observe similar trends from real networks. The densities of resultant CDC subgraphs obtained by FRD depend on network topologies. Hence, for the same type of synthetic networks with the same \( \epsilon \) value, the variance in the density ratio is low.

6 Conclusion

In this paper, we introduce Triple Network, its CDC subgraph problem and its variants. We provide heuristics to find feasible solutions to these patterns, otherwise NP-Hard to find. We conclude that CDC subgraphs yield communities with similar characteristics by illustrating the information gain of these patterns in NYC taxi, Twitter, ArnetMiner, and Flixter networks. We demonstrate the efficiency of our algorithms on large real and synthetic networks by observing running-time and density trends in real and synthetic networks.

References

1. NYC taxi & limousine commission - trip record data, [https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page](https://www1.nyc.gov/site/tlc/about/tlc-trip-record-data.page)
2. Bader, D.A., Madduri, K.: Gtgraph: A synthetic graph generator suite. Atlanta, GA, February (2006)
3. Bhaskara, A., Charikar, M., Chlamtac, E., Feige, U., Vijayaraghavan, A.: Detecting high log-densities: an \( o (n^{1/4}) \) approximation for densest k-subgraph. In: Proceedings of the forty-second ACM symposium on Theory of computing. pp. 201–210. ACM (2010)
4. Boden, B., Ester, M., Seidl, T.: Density-based subspace clustering in heterogeneous networks. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. pp. 149–164. Springer (2014)
5. Chakrabarti, D., Zhan, Y., Faloutsos, C.: R-mat: A recursive model for graph mining. In: Proceedings of the 2004 SIAM International Conference on Data Mining. pp. 442–446. SIAM (2004)
6. Charikar, M.: Greedy approximation algorithms for finding dense components in a graph. In: International Workshop on Approximation Algorithms for Combinatorial Optimization. pp. 84–95. Springer (2000)
7. Fortunato, S.: Community detection in graphs. Physics reports 486(3-5), 75–174 (2010)
8. Goldberg, A.V.: Finding a maximum density subgraph. University of California Berkeley, CA (1984)
9. Hu, H., Yan, X., Huang, Y., Han, J., Zhou, X.J.: Mining coherent dense subgraphs across massive biological networks for functional discovery. Bioinformatics 21(suppl_1), i213–i221 (2005)
10. Jamali, M., Ester, M.: A matrix factorization technique with trust propagation for recommendation in social networks. In: Proceedings of the fourth ACM conference on Recommender systems. pp. 135–142. ACM (2010)
11. Kelley, R., Ideker, T.: Systematic interpretation of genetic interactions using protein networks. Nature biotechnology 23(5), 561 (2005)
12. Khuller, S., Saha, B.: On finding dense subgraphs. In: International Colloquium on Automata, Languages, and Programming. pp. 597–608. Springer (2009)
13. Li, W., Hu, H., Huang, Y., Li, H., Mehan, M.R., Nunez-Iglesias, J., Xu, M., Yan, X., Zhou, X.J.: Pattern mining across many massive biological networks. In: Functional coherence of molecular networks in bioinformatics, pp. 137–170. Springer (2012)
14. Pei, J., Jiang, D., Zhang, A.: On mining cross-graph quasi-cliques. In: Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining. pp. 228–238. ACM (2005)
15. Saha, B., Hoch, A., Khuller, S., Raschid, L., Zhang, X.N.: Dense subgraphs with restrictions and applications to gene annotation graphs. In: Annual International Conference on Research in Computational Molecular Biology. pp. 456–472. Springer (2010)
16. Shah, D., Prasad, S., Aghajarian, D.: Finding densest subgraph in a bi-partite graph. Department of Computer Science, Georgia State University, Atlanta, GA (2019), https://scholarworks.gsu.edu/computer_science_technicalreports/17
17. Shah, D., Prasad, S., Wu, Y.: Finding Connected-Dense-Connected Subgraphs and variants is NP-Hard. Department of Computer Science, Georgia State University, Atlanta, GA (2019), https://scholarworks.gsu.edu/computer_science_technicalreports/2
18. Sun, Y., Yu, Y., Han, J.: Ranking-based clustering of heterogeneous information networks with star network schema. In: Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 797–806. ACM (2009)
19. Tang, J., Zhang, J., Yao, L., Li, J., Zhang, L., Su, Z.: Arnetminer: extraction and mining of academic social networks. In: Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining. pp. 990–998. ACM (2008)
20. Wu, Y., Jin, R., Zhu, X., Zhang, X.: Finding dense and connected subgraphs in dual networks. In: Data Engineering (ICDE), 2015 IEEE 31st International Conference on. pp. 915–926. IEEE (2015)