FINDING DISJOINT DENSE CLUBS IN AN UNDIRECTED GRAPH

by

Peng Zou

A thesis submitted in partial fulfillment
of the requirements for the degree

of

Master of Science

in

Computer Science

MONTANA STATE UNIVERSITY
Bozeman, Montana

May, 2016

ACKNOWLEDGEMENTS

I would like to thank my advisor Dr. Binhai Zhu for his patient guidance and encouragement. Working with Dr. Zhu has been a great research experience.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

In a social network, the trust among its members usually cannot be carried over many hops. So it is important to find disjoint clusters with a small diameter and with a decent size, formally called *dense clubs* in this thesis. We focus on handling this NP-complete problem in this thesis. First, from the parameterized computational complexity point of view, we show that this problem does not admit a polynomial kernel (implying that it is unlikely to apply some reduction rules to obtain a practically small problem size). Then, we focus on the dual version of the problem, i.e., deleting d vertices to obtain some disjoint dense clubs. We show that this dual problem admits a simple FPT algorithm using a bounded search tree method (the running time is still too high for practical datasets). Finally, we combine a simple reduction rule together with some heuristic methods to obtain a practical solution (verified by extensive testing on practical datasets). Empirical results show that this heuristic algorithm is very sensitive to all parameters. This algorithm is suitable on graphs which have a mixture of dense and sparse regions. These graphs are very common in the real world.

1

INTRODUCTION

Background

For over a decade, software like Twitter, Facebook and WeChat have changed people's lives by creating social groups and networks easily. They give people a new convenient "world" where we can share everything that happens around us, and social networks have grown enormously in recent years. For example, at the beginning, the Tencent company developed WeChat which is a mobile text and voice messaging communication service in China. Nowadays, WeChat provides so many services, such as mobile text messaging, voice messaging, video games, moments, money transfer and so on. It has become an essential feature for smart phones in China. As of December 2015, WeChat has over a billion created accounts, 650 million active users (roughly half of the population) in China, and has 70 million users outside of China [38]. In essence, social networks are full of data and have become an indispensable part of our life.

Trust is an important feature of the relationship between two users in a social network. With the development of social networks, the trust among its members has become a big issue. In a social network, the trust among its members usually cannot be carried over many users. In the corresponding social network modeled as a graph, a user is denoted by a vertex and an edge between two vertices means that these two users communicate a lot above some threshold and they trust each other. An online social community is usually corresponding to a dense region in such a graph. A complex social network is usually composed of several groups/communities (the

regions with a lot of edges), and this characterization of community structure means the appearance of densely connected groups of vertices, with only sparse connections between groups [27], see Fig. 1.1. For analyzing the structure of social networks and the relationship between users, it is important to find disjoint groups/communities with a small diameter and with a decent size, formally called *dense clubs* in this thesis.

The disjoint clubs/groups, after successfully identified, have some potential applications. For instance, some target marketing strategy can be applied based on the groups. And, in some situation, the groups can even help to identify some terrorist groups [23].

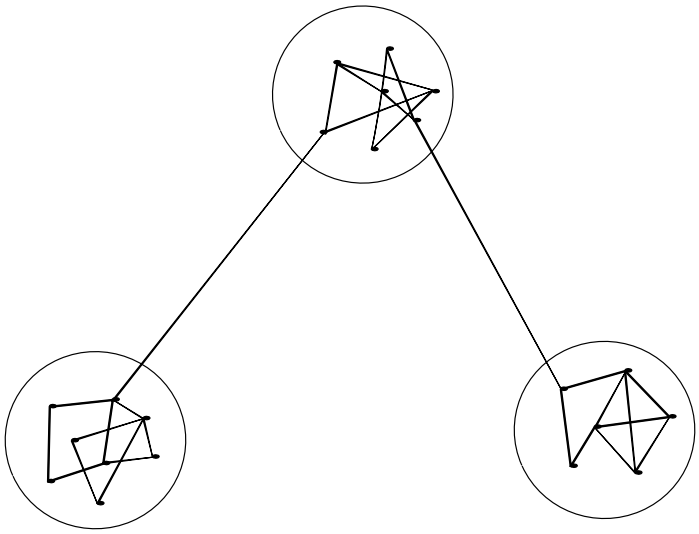

Figure 1.1: An example of the network structure. There are three groups denoted by the circles with many edges and only a small number of edges between groups.

Locating the groups/communications is the traditional clustering problem in computer science. Formally, the purpose of the clustering problem is to divide the input data set into several clusters which satisfy some specific requirement. Clustering is an important idea in data analysis. There is a huge amount of applications in data mining, social network analysis, web searching, pattern recognition, computer vision

and so on. For example, in the field of computer vision, object identification is a common task. The clustering can be used to identify the potential object in the image. In the bioinformatics field, clustering can be used to identify the potential functional modules in protein-protein interaction networks, which can further help to design new drugs. More details about clustering algorithms can be found in [21].

There are different ways to measure the quality of clustering based on the application requirements. In paper [34], Sinclair *et al.* proposed the conductance which is the ratio of the number of its outgoing edges to the sum of the degrees of its nodes for measuring the cluster. For example, the conductance for an isolated cluster which has no outgoing edge is 0, and the conductance for a cluster without any edge is 1. In paper [35], the authors developed a tool which contains two new local clustering algorithms which are Nibble and PageRank-Nibble. These two algorithms focus on quickly finding a good cluster which has the lowest conductance close to a fixed node. In paper [37], the authors proposed a novel measurement for the global clustering optimization which is LCP^2 (low two-hop conductance sets). The authors proposed a spectral approximation algorithm $SLCP^2$ and a greedy strategy algorithm $GLCP^2$. All operations of the algorithm are matrix computations. The LCP^2 measurement can group the vertices which have similar connections in one cluster, so these two algorithms can not only find the dense areas in the graph, but also locate the sparse areas in the graph. In paper [36], the authors mapped the original graph to an image graph. Then, they tried to minimize the error function which is used to compare the similarity between the input graph and the image graph. They wanted to make sure that the nodes in the same cluster have similar connection patterns. They designed their algorithm based on simulated annealing with path relinking. Compared to the traditional simulated annealing, their simulated annealing with path relinking method reduced the computational running time significantly.

In this thesis, we focus on diameter-based measurement of a cluster. Actually, this is a more restrictive measurement. Given the diameter of a cluster, this measurement potentially ensures that the nodes in the cluster have the same connection pattern. A clique is commonly used to describe dense subgroup. In fact, the maximum clique problem is one of the most widely studied NP-complete problems [22]. The requirements of a clique are too restrictive in many situations, thus various relaxed cohesive subgroup structures, based on the clique, have been proposed, such as s-club, s-clique, and s-plex, etc [24].

Given an undirected graph $G = (V, E)$, a clique is a subset of vertices such that any pair of vertices in this subset form an edge in E. The maximum clique problem is to compute a clique with the maximum size. Many algorithms for this NP-complete problem are available in the literature [8, 30, 40]. Motivated by practical applications in social and biological networks, s-club is a diameter-based graph-theoretic generalization of clique, which was first introduced as an alternative approach to model a cohesive subgroup in the social network area [2, 26]. The s-club is a subset of vertices $V' \subseteq V$, such that the diameter of the induced subgraph $G[V']$ is at most s.

An s-clique is a subset of vertices $S \subseteq V$ if $d_G(u, v) \leq s$ for all $u, v \in S$. It is pretty obvious that an s-club is also an s-clique, but the converse is not true in general [33]. And for $s = 1$, an s-club is simply a clique. The examples of these two different structures are shown in Fig. 1.2. In this graph, a subset of vertices $\{a, b, c, d\}$ is a 2-clique but not a 2-club, since the distance between any 2 points is 2 but the point e is not in the subset. And the subset of vertices $\{a, b, c, d, e\}$ is a 2-club and also it is a 2-clique. It is known that the maximum clique problem is a classical NP-complete problem [8, 30], which is also hard to approximate [20]. The maximum s-club problem [5, 10] is NP-complete for any fixed s, even when restricted

to graphs of fixed diameter $s+1$ [5]. In fact, testing whether an s-club is maximal is also NP-complete for any fixed integer s [29].

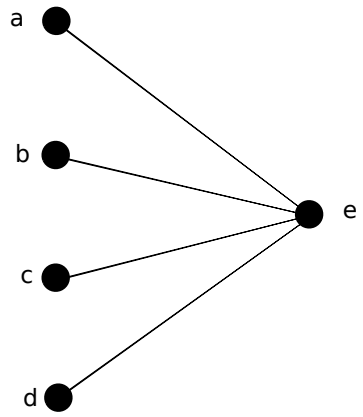

Figure 1.2: The difference between s-clique and s-club, $\{a, b, c, d\}$ is a 2-clique but not a 2-club.

The s-plex is a clique relaxation cohesive subgroup which is introduced in [32]. An s-plex is a subset of vertices $S \subset V$, such that, for $\forall v \in G[S]$, the degree of vertices v is at least $|S| - k$. And for $k = 1$, a k-plex is simply a clique. The examples of k-plex structures for $k = 1, 2, 3$ are in Fig 1.3. The subsets of vertices $\{a, b, c\}$ and $\{b, c, e\}$ are 1-plex, and these are cliques. The subset of vertices $\{a, b, c, e\}$ is a 2-plex. The subset of vertices $\{a, b, c, d, e\}$ is a 3-plex. A maximal k-plex is one that is not in any other k-plex. The maximum k-plex problem is to find the maximal k-plex of maximum size in a given graph, which is NP-complete for any constant positive integer k [4]. In paper [18], Guo *et al.* focused on the s-plex editing problem. This problem is to delete minimum edges to transform the given graph into disjoint s-plexes. They proposed some kernelization and search tree algorithms for this problem. In this thesis, we focus on s-club structure and will not discuss k-plex further.

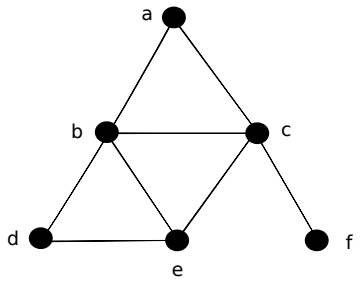

Figure 1.3: k-plex for $k = 1, 2, 3$, $\{a, b, c, d, e\}$ is a k-plex for $k = 3$.

Existing Methods

Heuristic Methods

In reference [9], Bourjolly *et al.* posed three heuristic methods which are DROP, CONSTELLATION and s-CLIQUE-DROP for the maximum s-club problem. A variant neighborhood search (VNS) meta-heuristic algorithm was proposed by Shahinpour *et al.* [33]. In fact, they used the VNS heuristic method as the lower bound to develop an exact algorithm for the maximum s-club problem [33]. Recently, a new heuristic algorithm called IDROP for the largest s-club problem was given in the paper [13].

Parameterized Methods

From the parameterized computational complexity point of view, two fixed-parameter tractable (FPT) algorithms for the maximum s-club problem were proposed [31]. The paper [19] extends the previous parameterized complexity study for 2-club and provides polynomial-size kernels for 2-club parameterized by "cluster editing set size of G" and "size of a cluster editing set of G". For the 2-club-editing problem, the paper [25] shows that the disjoint dense club problem is NP-complete for any $s \leq 2$ and proposes an improved search tree algorithm with running time $O^*(3.31^k)$ based on two new branching cases, improving the trivial $O^*(4^k)$ bound.

Other Methods

From the viewpoint of approximation algorithms, the maximum k-club problem for any fixed $k \geq 2$ is NP-hard to approximate within a factor of $n^{(1/2-\epsilon)}$ for any $\epsilon > 0$ in general graphs [3]. A simple polynomial-time algorithm which has an approximation ratio of $n^{1/2}$ for even $k \geq 2$, and an approximation ratio of $n^{2/3}$ for any odd $k \leq 3$ is also proposed in this paper. The integer linear programming formulation is presented in [5, 10]. In [39], two satisfiability-based formulations of the maximum k-club problem and the exact method are proposed.

In this thesis, we first study the disjoint dense club problem. Specifically, we show that the problem does not have a polynomial kernel (unless the polynomial hierarchy collapses to the third level). This implies that it is unlikely to obtain any efficient FPT algorithm for the problem (and the related ones). Then we consider the dual problem of editing a graph (by deleting vertices) into disjoint s-clubs. Since the trivial bounded-degree search method takes $O^*((s+2)^d)$ time, which is not efficient for most real datasets, we propose three rules for building an efficient heuristic method. We implement our algorithm and test this method on five real datasets.

Organization

The rest of this thesis is organized as follows.

1. In the Preliminaries chapter, we introduce some necessary definitions and notations about our practical algorithm.

2. In the Parameterized Results chapter, We describe the definitions of FPT algorithms, the kernelization concept, the bounded search tree algorithms, and the polynomial parameter transformation. Then, the theoretical results are reported.

3. All computational results are shown in the Computational Results chapter. We tested our algorithm on 5 graphs. The first two graphs are purely for testing purpose. The rest of the graphs are real social networks.

4. We conclude the thesis in the Conclusion chapter. We also propose some interesting open problems.

PRELIMINARIES

In this section, we present the relevant definitions and review some useful notions.

FPT Algorithms and Kernels FPT (Fixed-Parameter-Tractable) algorithms are used to study the computational complexity of NP-hard problems [16, 17, 28]. Beside the input size n, we also consider a parameter k (or several parameters). An FPT algorithm is one which solves a parameterized problem L in $O(f(k)n^c) = O^*(f(k))$ time (i.e., decide whether $(x, k) \in L$, where $n = |x|$), where $f(-)$ is any computable function and c is a constant not related to n and k.

A parameterized problem L admits a problem *kernel* if there is a polynomial-time transformation of any instance (I, k) to an instance (I', k') such that (1) $(I, k) \in L$ iff $(I', k') \in L$; (2) $|I'| \leq g(k)$; and (3) $k' \leq k$. L has a polynomial kernel if $g(k)$ is a polynomial function. It is known that L admits an FPT algorithm iff it has a kernel (not necessarily polynomial). But if L has a polynomial kernel then usually it means L is relatively easier to solve [16, 17, 28].

The main idea of kernelization is to obtain the relatively smaller problem instance which is equivalent to the original instance and the size of the instance is bounded by the function of some parameter k. Sometimes, we can apply some simple reduction rules on the input to dramatically reduce the size of the input. This is a very useful technique for designing efficient FPT algorithms, because every problem in FPT has a kernel [12]. The algorithm for the k-vertex cover problem which is described by Buss is the best example [11]. This algorithm is based on the rule that a node whose degree is bigger than k must be in the resulting solution set. This node must be in the resulting solution set, otherwise all its neighbors cannot be covered with k nodes. Then, we remove this node and update the graph and set $k' = k - 1$. This process is repeated until no such node can be found. Suppose this process is repeated p times,

the remaining instance is (G', k') where $k' = k - p$ and the maximum degree is k' in G'. For these k' vertices, they cover at most $k \cdot k'$ vertices. So the remaining graph G' has at most $k' \cdot k \leq k^2$ vertices, if G has a vertex cover of size at most k; moreover, G has a vertex cover of size k iff G' has a vertex cover of size k'. The most recently achievement obtains the linear kernel for the vertex cover problem [1]. In the paper, they proposed two ways to obtain a linear kernel for the vertex cover problem. The first result is that they use the linear programming method to obtain the $2k$ kernel for the vertex cover problem. They also use the crown reduction method to obtain a kernel bounded by $3k$ for the vertex cover problem.

Bounded Search Tree This is a basic and obvious technique which is used to design FPT algorithms. The classical and simple Bounded Search Tree algorithm was used to solve the vertex cover problem which is a classical NP-hard problem [22]. In the vertex cover problem, the input is a graph $G = (V, E)$. We need to find a minimum subset $S \subset V$ such that at least one endpoint of each edge is in the set S. The idea of bounded search tree algorithm is, for each $(v_1, v_2) \in E$, one of these two nodes must be in the resulting solution set. We set the parameter k as the size of resulting solution set S. We pick an edge $(v_1, v_2) \in E$ arbitrarily, then put one of these two points into the two branching set S and delete all edges connects to that point, then update the graph and recurse this process until the size of resulting solution set is k. After k steps, we have a binary tree with k levels. We can check whether we get the desired vertex cover. This is an exhaustive search algorithm. The depth of this search tree is k. Theoretically, the running time of this bounded search tree algorithm is $O^*(2^k)$, so this is an FPT algorithm for vertex cover. In paper [15], Chen *et al.* proposed the "advantageous" structure and designed an efficient branching algorithm for the vertex cover problem. This algorithm runs in $O(1.2738^k + kn)$

time and polynomial space which also improved their previous $O(1.286^k + kn)$-time polynomial-space algorithm [14] for the vertex cover problem.

Polynomial Parameter Transformations A polynomial parameter reduction is used to reduce a problem known to be without a polynomial kernel to another problem B [6, 7]. It is different from the traditional FPT-reductions.

Definition 1. Let P, Q be parameterized problem. P is polynomial parameter reducible to Q, written as $P \leq_{pp} Q$, if there exists a polynomial time computable function $f : \Sigma^* \times N \to \Sigma^* \times N$ and a polynomial p, such that for all $(x, k) \in \Sigma^* \times N$, $(1)(x, k) \in P$ if and only if $(x', k') = f(x, k) \in Q$, and (2) $|x'| \leq p(k)$. The function f is called a polynomial parameter transformation.

In [7], the following proposition was proved.

Proposition 1. Let P and Q be the parameterized problems and $\widetilde{P}$ and $\widetilde{Q}$ be the unparameterized versions of P and Q respectively. Suppose that $\widetilde{Q}$ is NP-complete and $\widetilde{P}$ is in NP. Furthermore, if there is a polynomial parameter transformation from P to Q, then if Q has a polynomial kernel, then P also has a polynomial kernel.

The above proposition can be used to prove kernelization lower bounds.

Graphs, Clubs, and Neighborhoods We consider simple undirected graphs in this thesis. Given a graph $G = (V, E)$, the distance $\delta(u, v)$ between two vertices $(u, v) \in V$ is the length of the shortest path between u and v. The *diameter* of G is the maximum of all minimum distances between pairs of nodes in V. The (open) i-neighborhood $N_i(v) = \{x|\delta(x, v) \leq i\}$ of v is the set of vertices that has distance at most i to v. The closed i-neighborhood of v is the set $N_i[v] = N_i(v) \cup \{v\}$. The exact i-neighborhood of v is the set $N_i^e(v) = \{x|\delta(x, v) = i\}$, which is the set of vertices have distance exactly i to v. For a vertex set T, $G[T]$ denotes the subgraph of G induced by T having edge set $E_T = \{(u, v) \in E|u, v \in T\}$.

A graph H is an s-club if the diameter of H is at most s. (A subset $S \subseteq V$ is an s-club if $G[S]$ is an s-club.) And an s-club H is a (t, s)-club if the number of vertices in V is at least t, i.e., $V(H) \geq t$. Notice that when $s = 1$, an s-club is a clique. Throughout this thesis, we assume that s is a small constant with $s \geq 2$. We next define the Maximum Club problem.

Problem (1): Maximum Club problem.

Input: An undirected graph G and constant integer $s \geq 2$, and integer t.

Question: Is there a (t, s)-club in G?

The Maximum Club problem is NP-complete [10]; in fact, it is NP-complete even in graphs of diameter $s + 1$ [5]. The parameterized version of the problem (with t being a parameter) is shown to be FPT but does not admit a polynomial kernel unless the polynomial hierarchy collapses to its third level (i.e., NP $\subseteq$ coNP/Poly) [31].

In a complex social network, finding a single club is probably not too interesting, so we define the following problems. As they are both NP-complete, we focus on the parameterized versions of the problems in this thesis.

Problem (2): Disjoint Dense Clubs (DDC) Problem.

Input: An undirected graph G and constant integer $s \geq 2$, and integers t, k.

Question: Are there k disjoint (t, s)-club in G?

Problem (3): Maximum Disjoint Dense Clubs (MDDC) problem.

Input: An undirected graph G and constant integer $s \geq 2$, and integer ℓ.

Question: Is there a set of disjoint s-clubs whose total size is at least ℓ in G?

13

PARAMETERIZED RESULTS

In this section, we first present some theoretical results on the parameterized complexity for DDC and MDCC. Based on some of these results, we try to design a practical algorithm for them.

Theorem 1. *The disjoint dense clubs problem, parameterized by k and t, does not admit a polynomial kernel unless NP $\subseteq$ coNP/Poly.*

Proof. We reduce the Maximum Club problem of finding a maximum s-club (with parameter p being its size) to the parameterized version of the DDC problem (parameterized by k and t), with a polynomial parameter reduction. Take the input $\langle G, s, p\rangle$, we construct p copies of G, i.e., $G_1, \ldots, G_p$, as the new graph $\mathcal{G}$. For $\mathcal{G}$ we set $k = p$ and $t = p$. Then G has an s-club of size p iff $\mathcal{G}$ has $k = p$ disjoint (p, s)-clubs.

"$\rightarrow$": If G has an s-club of size p, then each G_i has an s-club of size p (or, each G_i has a (p, s)-club). Obviously, $\mathcal{G}$ has p disjoint (p, s)-clubs.

"$\leftarrow$": If $\mathcal{G}$ has $k = p$ disjoint (p, s)-club, then each G_i must have a (p, s)-club. As G is known to be isomorphic to G_i, G must also have a (p, s)-club.

As deciding whether G has a (p, s)-club, parameterized by p, has no polynomial kernel unless NP $\subseteq$ coNP/Poly, the DDC problem (parameterized by k, t) also has no polynomial kernel unless NP $\subseteq$ coNP/Poly. □

Theorem 2. *The maximum disjoint dense clubs problem, parameterized by ℓ, does not admit a polynomial kernel unless NP $\subseteq$ coNP/Poly.*

Proof. The reduction is that same as before. Just set the parameter $\ell = p^2$. □

The above two results, while simple, imply that it is unlikely to obtain efficient FPT algorithms to solve the DCC and MDCC problems. Next we look at the dual version of these problems.

Theorem 3. *The disjoint dense clubs problem, parameterized by $d = |V| - kt$, does admit and FPT algorithm running in $O^*((s+2)^d)$ time.*

Proof. If between all pair of vertices u, v we have $\delta(u, v) \leq s$, then the whole graph G is an s-club. Then we are done. So to obtain disjoint s-club, we just need to branch over a pair of vertices u, v with $\delta(u, v) = s + 1$. Between and inclusive of u and v, there are $s + 2$ vertices. One of these $s + 2$ vertices must be deleted. Therefore, we have $s+2$ choices at level-1. Then we repeat the above process for d rounds. Among the $(\leq (s + 2)^d)$ leaf nodes, if there are k (t, s)-clubs left, return YES; otherwise, return NO. □

Theorem 4. *The maximum disjoint dense clubs problem, parameterized by $d = |V| - \ell$, does admit an FPT algorithm running in $O^*((s+2)^d)$ time.*

Proof. The branching algorithm is the same as before. When we have a bound search tree of depth d, there are at most $\leq (s+2)^d$ leaf nodes. We eliminate all those which are not an s-club, then check whether the (remaining) leaf nodes have a total size which is at least ℓ. The running time is $O^*((s+2)^d)$. □

Notice that the running time for the bounded search tree algorithm is too high for practical datasets. In fact, even if we could reduce the running time to roughly $O^*(3^d)$, similar to [25] for $s = 2$, it is still too high for practical datasets. So we need some practical method. We first present the following reduction rule.

Lemma 1. *(Reduction Rule I:) If $|N_s(v)| < t-1$, then v cannot be in any (t, s)-club.*

Proof. For any two nodes u, v in the same club, we must have $|N_s[v] \cap N_s[u]| \geq t$. Hence we have $N_s(v) \geq t-1$. All the nodes in the club must satisfy this property, otherwise this node cannot be in any (t,s)-club. □

Note that we can repeatedly run this reduction rule. And when it is not possible to apply it, we have the following lemmas which do not necessarily help us reduce the problem size, but could help us reduce the solution search space.

Lemma 2. *(Branching Rule II:) Let $d(u,v) = 1$. If for all $w \in N_s(u) \cap N_s(v)$ we have $|N_s(w)| < t-1$, then u, v cannot be in the same (t,s)-club.*

Proof. As $d(u,v) = 1$, if u, v are in the same (t,s)-club, then there must exist a $w \in N_s(u) \cap N_s(v)$ which is in the same (t,s)-club. However, by assumption, for all $w \in N_s(u) \cap N_s(v)$ we have $|N_s(w)| < t-1$, i.e., there is not such a club containing u, v and w which is dense enough. Then we have a contradiction. Therefore, u, v cannot be in the same (t,s)-club. □

Lemma 3. *(Branching Rule III:) Let $d(u,v) = 1$. If for all $w \in N_s(u) \cap N_s(v)$, we have $|N_s[w] \cap N_s[u] \cap N_s[v]| < t$, then u, v cannot in the same (t,s)-club.*

Proof. As $d(u,v) = 1$, if u, v are in the same (t,s)-club, then there must exist a node $x \in N_s(u) \cap N_s(v)$ which is in the same (t,s)-club. (Here, $|N_s[x]|$could be larger than t.) By assumption, for all $w \in N_s(u) \cap N_s(v)$ we have $|N_s[w] \cap N_s[u] \cap N_s[v]| < t$, i.e., there is no w connecting a club which contains both u and v, is completely in $N_s[u] \cap N_s[v]$ and is dense enough. (Intuitively, some nodes in $N_s[w]$ are more than s edges away from u or v.) Hence, we have a contradiction. Therefore, u, v cannot be in the same (t,s)-club. □

Based on Lemmas 1-3, we design a new practical bounded search tree algorithm for DCC. (The algorithm also works for the MDCC problem.) The first step is to

apply Lemma 1 on the input graph G. Lemma 1 can delete all nodes unlikely to be in any club. Then, we set $d(u, v) = 1$ to apply Lemma 2 and Lemma 3. If we can find two nodes satisfying Lemma 2 or Lemma 3, one of these two nodes must be deleted. The running time of this algorithm is $O^*(2^d)$.

Algorithm 3.1 A new practical bounded search tree algorithm

procedure: Branching(G, s, t, k)
1: **if** $k == 0$ or the G is already a s-club cluster **then**
2: the algorithm is finished, exit
3: **end if**
4: **while** there is a vertex v, where $|N_s(v)| \leq t$ **do**
5: delete all the nodes which satisfy the condition of Lemma 1
6: update the graph
7: **end while**
8: **if** there is an edge satisfying the Lemma 2 or Lemma 3 **then**
9: delete one vertex of this edge, form G' and call Branching$(G', s, t, k-1)$
10: delete the other vertex of this edge and form G'' and call Branching$(G'', s, t, k-1)$
11: **else**
12: no edge can be found, exit
13: **end if**

We would not be able to claim any theoretical result on the algorithm yet, as there are instances that the algorithm fails to handle (e.g., when we have two intersecting (t, s)-clubs). More investigation is needed to make the algorithm a true FPT algorithm for the dual version of DCC and MDCC with a running time of $O^*(2^d)$.

In the next chapter, we show some computational results based on this algorithm. The motivation is to avoid the $O^*((s+2)^d)$ time FPT algorithm (Theorems 3-4), which is too high even for $s = 2, 3$.

COMPUTATIONAL RESULTS

We implemented the new bounded-search tree algorithm based on Lemmas 1-3. We focused on $s = 2, 3$ for all of our empirical results. The algorithm was run on a laptop with Intel Core(TM) i7-3770 CPU, 3.40GHz, 16GB RAM.

Data Description

We tested our algorithm on five graphs. The first graph is the US Power Grid with 4941 vertices and 6594 edges where a vertex is either a generator, a transformator or a substation and an edge represents a power supply line. The diameter of the US Power Grid graph is 46. We downloaded it from the website: `http://konect.uni-koblenz.de/networks/opsahl-powergrid`. The density of this graph is lowest among these five graphs. The second graph is "Autonomous System AS-733" with 6474 vertices and 12572 edges. The diameter of this graph is 9. We downloaded the graph from the website: `http://snap.stanford.edu/data/as.html`. It was shown on the webpage that the graph has 6474 vertices and 13895 edges where a vertex is a user and an edge means that two users have some traffic flow. But there are self-loop edges which are meaningless to us. We preprocessed the graph and deleted all self-loop edges. These two graphs do not fall into the category of social networks and they were used purely for testing purpose.

We next used three graphs from social networks. The third graph is the General Relativity and Quantum Cosmology collaboration network with 5242 vertices and 14496 edges where a vertex is an author and an edge represents that the author co-authored a paper with another author. Actually this is a directed graph, we treated this graph as the undirected graph. We preprocessed the graph as follows. We ignored the direction of all edges and deleted all self-loop edges. After preprocessing,

this graph has 5242 vertices and 14484 edges. The diameter of this graph is 18. We downloaded it from the website: `http://snap.stanford.edu/data/ca-GrQc.html`. The forth graph is the trust network of Advogato. We ignored the direction of all edges and deleted all self-loop edges of the Advogato dataset where a vertex is a user and an edge represents the trust relationship between two users. After preprocessing, the original Advogato dataset has 2913 vertices and 17807 edges. The diameter of the trust network of Advogato is 9. We downloaded it from the website: `http://www.trustlet.org/datasets/`. The last graph is the Erdos Collaborations network which is similar as the third one. A vertex represents an author and an edge means these two authors coauthored one paper. This graphs has 6927 vertices and 11850 edges. The diameter of this network is 5. We downloaded it from the website: `http://vlado.fmf.uni-lj.si/pub/networks/data/Erdos/Erdos02.net`.

Empirical Result

As we are using a bounded-search tree algorithm, we cannot expect to delete too many vertices from the graph (i.e., d should not be too large). The key contribution of Lemma 1 is to delete nodes unlikely to be a member of any t-s-club, i.e., before running the bounded-search step, we could already reduce the problem size by Lemma 1. We tested Lemma 1 on the US Power Grid and the results are put in Table 4.1. From Table 4.1, we could find that Lemma 1 can delete more nodes with the increase of the size of clubs (e.g., t) when the diameter of clubs (e.g., s) is fixed. Moreover, Lemma 1 can delete more nodes with the decrease of the diameter of clubs (e.g., s) when the size of clubs (e.g., t) is fixed. In general, Lemma 1 is very sensitive to s and t.

We reported the empirical results for the US Power Grid graph in Table 4.2. In the last column, 'No' means a solution has not been found within d steps and 'Yes'

Table 4.1: The performance of Lemma 1 on the US Power Grid graph.

index	diameter (s)	size (t)	nodes deleted	time(ms)
1	2	10	4200	950
2	2	15	4857	935
3	2	20	4921	928
4	3	10	1236	595
5	3	15	3049	784
6	3	20	4240	936
7	3	25	4759	1011
8	3	30	4911	826

means a solution has been found. As the graph has a large diameter, there are in fact many small clubs.

We reported the empirical results on the Autonomous System AS-733 graph in Table 4.3, where 'Yes/No' carry the same meaning as in Table 4.2. As the graph has a diameter smaller than that of the US Power Grid graph, it is reasonable to assume that there are not many clubs.

We reported the empirical results on the General Relativity and Quantum Cosmology collaboration network in Table 4.4, where 'Yes/No' carry the same meaning as in Table 4.2. As the graph has a diameter smaller than that of the US power Grid graph, again, it is reasonable to assume that there are not many clubs.

The empirical results on the trust network of Advogato were shown in Table 4.5, where 'Yes/No' carry the same meaning as in Table 4.2. As the graph has a diameter smaller than that of the US power Grid graph, again, it is reasonable to assume that there are not many clubs. In the table, the running time for the instance with diameter 2 is the average over 5 tries.

The empirical results on the Erdos Collaborations network were shown in Table 4.6, where 'Yes/No' carry the same meaning as in Table 4.2. As the graph has a

diameter smaller than that of the US power Grid graph, again, it is reasonable to assume that there are not many clubs. In the table, the running time for the instance with diameter 2 is the average over 5 tries.

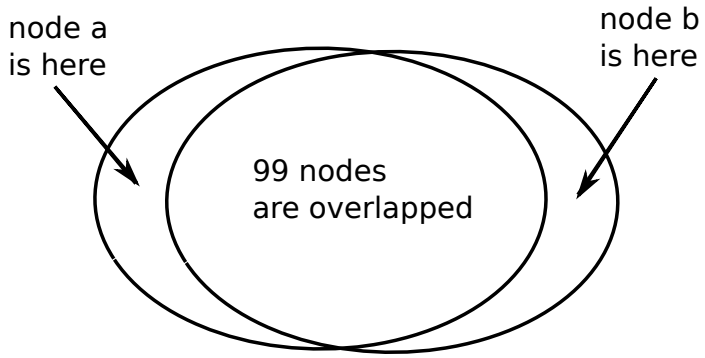

Figure 4.1: 2 s-clubs overlap almost completely.

As this bounded search tree algorithm is a heuristic algorithm, sometimes this algorithm cannot find any result. This situation is shown in Fig. 4.1. This algorithm focuses on finding the edges which satisfy Lemma 2 or Lemma 3. In Fig. 4.1, the distance of node a and node b is $s + 1$ and each circle is an s-club. When we set $t = 100$, the node a and node b can not satisfy Lemma 2 or Lemma 3. But one of these two nodes must be deleted to obtain the result. Our algorithm cannot delete node a or node b. In this situation, this club cannot be found.

For some cases, the running time of the algorithm does not necessarily exponentially increase with the number of steps. As the example in Table 4.3, the running time of the sixth result should be larger than the running time of the fifth result, because the number of steps is large. But when the algorithm can not find the edge which satisfy the Lemma 2 or Lemma 3, the algorithm will stop this branch. The density of the US Power Grid graph is lower than the density of the other two graphs, and the diameter of the US Power Grid is much larger than that of the other two graphs. So the size of the clubs in the US Power Grid graph are smaller than

Table 4.2: Empirical results for the US Power Grid graph.

index	diameter (s)	size (t)	steps (d)	running time(ms)	result
1	2	10	10	24258	No
2	2	10	11	37874	No
3	2	10	12	65843	No
4	2	10	13	121148	No
5	2	10	14	232313	No
6	2	11	15	420401	No
7	2	14	15	521941	Yes
8	2	15	15	18543	Yes
9	3	11	10	81634	No
10	3	11	11	158064	No
11	3	12	11	142547	No
12	3	13	11	129704	No
13	3	14	11	119538	No
14	3	15	11	107597	No
15	3	16	12	179937	No
16	3	17	12	169046	No
17	3	18	12	178398	No
18	3	19	12	183689	No
19	3	20	12	140499	No
20	3	21	12	120817	No
21	3	22	13	235501	No
22	3	23	13	221456	No
23	3	24	13	157454	No
24	3	25	13	149456	No
25	3	26	14	266087	No
26	3	27	14	87690	Yes

Table 4.3: Empirical results for the Autonomous System AS-733 graph.

index	diameter (s)	size (t)	steps (d)	running time(ms)	result
1	2	1300	1	67848	Yes
2	2	700	5	30197324	No
3	2	750	4	10993184	No
4	2	760	5	1188221	No
5	2	770	5	1183132	No
6	2	770	8	1129920	No
7	2	780	8	1161247	No
8	2	790	8	114362	Yes
9	2	750	8	193337876	No
10	3	2800	1	8640385	No
11	3	2850	3	37265	Yes

those of the clubs in the other two graphs. The running time of algorithm is faster when the input is the US Power Grid graph. For the same instance, the running time of this algorithm grows with the diameter of clubs and the size of clubs — which is reasonable.

Table 4.4: Empirical results for the General Relativity and Quantum Cosmology collaboration network.

index	diameter (s)	size (t)	steps (d)	running time(ms)	result
1	2	70	5	69149	No
2	2	70	10	1657369	No
3	2	70	13	15282446	No
4	2	75	10	1339576	No
5	2	75	11	2281832	No
6	2	75	12	4347167	No
7	2	75	15	27037095	No
8	2	78	15	27147968	No
9	2	80	10	630728	No
10	2	80	15	12453691	No
11	2	81	15	11549	No
12	2	83	15	11386	No
13	2	85	10	11485	No
14	3	180	5	179324	No
15	3	180	10	7531381	No
16	3	185	10	6723176	No
17	3	190	5	117225	No
18	3	190	10	619036	No
19	3	195	5	32239	No
20	3	200	5	112119	No
21	3	201	5	112119	No
22	3	205	5	91121	Yes
23	3	210	5	14938	Yes

Table 4.5: Empirical results for the trust network of Advogato. The running time for the instance with diameter 2 is the average over 5 tries.

index	diameter (s)	size (t)	steps (d)	running time(ms)	result
1	2	390	10	9391143	No
2	2	400	10	2334312	No
3	2	400	7	319544	No
4	2	400	5	85190	No
5	2	410	5	672516	Yes
6	2	420	5	202362	No
7	2	430	5	202651	No
8	2	440	5	8125	Yes
9	2	450	5	8070	Yes
10	2	450	5	7915	Yes
11	3	1300	1	951300	No
12	3	1380	1	770100	No
13	3	1380	5	44858616	No
14	3	1382	5	48658598	No

Table 4.6: Empirical results for the Erdos Collaborations network . The running time for the instance with diameter 2 is the average over 5 tries.

index	diameter (s)	size (t)	steps (d)	running time(ms)	result
1	2	280	5	469571	No
2	2	280	7	4264802	No
3	2	290	5	472412	No
4	2	290	7	4278567	No
5	2	295	5	474666	No
6	2	295	7	3945094	No
7	2	298	7	3805685	No
8	2	298	10	36987526	Yes
9	2	299	7	22339	Yes
10	2	300	5	22410	Yes
11	3	1000	1	1190032	No
12	3	1050	1	1389788	No
13	3	1050	5	14933664	No
14	3	1120	1	2346727	No
15	3	1130	5	24811	Yes
16	3	1150	5	27883	Yes

CONCLUSION

In this thesis, we considered the Disjoint Dense Clubs problem which originates from social networks. A social network is usually composed of dense areas and sparse areas. The club is a natural way to define the cohesive substructures. This structure is very important for analyzing social networks.

We proved some theoretical results in this thesis, which are summarized as follows.

1. The disjoint dense clubs problem is theoretically hard, i.e., this problem parameterized by k and t (where k is the number of dense clubs, and t is the minimum size of the club), does not admit a polynomial kernel unless NP $\subseteq$ coNP/Poly. This means it is unlikely that there is an efficient FPT algorithm for this problem.

2. The maximum disjoint dense clubs problem parameterized by l (where l is the total size of all clubs), does not admit a polynomial kernel unless NP $\subseteq$ coNP/Poly.

3. The disjoint dense club problem and the maximum disjoint dense clubs problem have a trivial bounded-degree search algorithm which takes $O^*((s+2)^d)$ time, where d is the number of nodes deleted from the graph.

4. We designed a heuristic bounded-degree search tree algorithm for the dual version of the disjoint dense club problem based on three lemmas. This heuristic algorithm takes $O^*(2^k)$ time.

5. The limitation of our practical bounded-degree search tree algorithm was discussed, i.e., there are cases that our algorithm cannot deal with.

We implemented our practical bounded-degree search tree algorithm for its dual version (based on three rules) and presented the empirical results in the Computational Results chapter. The first part of our algorithm is Lemma 1 which can delete some vertices that cannot be in any (s,t)-club. The second part of our algorithm is to find some edges that satisfy either Lemma 2 or Lemma 3. One of the two endpoints of these edges must be deleted, because these two endpoints cannot be in the same (s,t)-club. Some empirical results, which are shown in the Computational Results chapter, are summarized as follows.

1. We tested our algorithm on five real graphs of 4941, 6474, 5242, 2913 and 6927 vertices, respectively. The last three graphs are real social networks. The first two graphs were used purely for testing purpose.

2. Lemma 1 is very sensitive to the parameters, such as t and s (where t is the minimum size of the club, and s is the diameter of the club).

3. Our algorithm worked very well on the US Power Grid graph, but for the rest of the graphs with relatively shorter diameters, the algorithm sometimes cannot find a desired solution due to the scenario described in Fig. 4.1.

When we were testing this algorithm, we expected it would work well on graphs containing many clubs. The US Power Grid graph has the lowest density and the longest diameter among these five graphs. This graph has more small dense clubs compared to the other three graphs. Our algorithm worked very well when we tested our algorithm on this graph. The diameters of all the social networks we used are relatively small, and the density is very high, so the running time of our practical algorithm is relatively high.

The disjoint dense club problem is an interesting problem although it is a hard problem. Aside from the progress we made in this thesis, there are still some problems left and we pose some relevant open problems as follows.

1. It would be interesting to add some extra rules to obtain an FPT algorithm, roughly running in $O^*(2^d)$ time, for the dual version of the disjoint dense club problem.

2. There is a similar version of the problem, i.e., the edge deletion version of disjoint dense clubs problem. The question is to delete edges to obtain the disjoint dense clubs. This is also an interesting problem. It also has the trivial bounded-degree search method which runs in $O^*((s+1)^k)$, where k is the number of edges deleted. And an improved bounded search tree algorithm for $s = 2$ which runs in $O^*(2.74^k)$ is known [25], which is not efficient for most real datasets. It would be interesting to design a practical algorithm for this problem.

3. It would be interesting to discuss how to apply the s-club idea to directed graphs. In many social network applications, the potential graphs could be directed.

REFERENCES

[1] F. N. Abu-Khzam, M. R. Fellows, M. A. Langston, and W. H. Suters. Crown structures for vertex cover kernelization. *Theory of Computing Systems*, 41(3):411–430, 2007.

[2] R. D. Alba. A graph-theoretic definition of a sociometric clique. *Journal of Mathematical Sociology*, 3(1):113–126, 1973.

[3] Y. Asahiro, E. Miyano, and K. Samizo. Approximating maximum diameter-bounded subgraphs. In *LATIN 2010: Theoretical Informatics*, pages 615–626. Springer, 2010.

[4] B. Balasundaram, S. Butenko, and I. V. Hicks. Clique relaxations in social network analysis: The maximum k-plex problem. *Operations Research*, 59(1):133–142, 2011.

[5] B. Balasundaram, S. Butenko, and S. Trukhanov. Novel approaches for analyzing biological networks. *Journal of Combinatorial Optimization*, 10(1):23–39, 2005.

[6] H. L. Bodlaender, R. G. Downey, M. R. Fellows, and D. Hermelin. On problems without polynomial kernels. *Journal of Computer and System Sciences*, 75(8):423–434, 2009.

[7] H. L. Bodlaender, S. Thomassé, and A. Yeo. Analysis of data reduction: Transformations give evidence for non-existence of polynomial kernels. Technical report, Technical Report UU-CS-2008-030, Department of Information and Computing Sciences, Utrecht University, 2008.

[8] I. M. Bomze, M. Budinich, P. M. Pardalos, and M. Pelillo. The maximum clique problem. In *Handbook of combinatorial optimization*, pages 1–74. Springer, 1999.

[9] J.-M. Bourjolly, G. Laporte, and G. Pesant. Heuristics for finding k-clubs in an undirected graph. *Computers & Operations Research*, 27(6):559–569, 2000.

[10] J.-M. Bourjolly, G. Laporte, and G. Pesant. An exact algorithm for the maximum k-club problem in an undirected graph. *European Journal of Operational Research*, 138(1):21–28, 2002.

[11] J. F. Buss and J. Goldsmith. Nondeterminism within P. *SIAM Journal on Computing*, 22(3):560–572, 1993.

[12] L. Cai, J. Chen, R. G. Downey, and M. R. Fellows. Advice classes of parameterized tractability. *Annals of pure and applied logic*, 84(1):119–138, 1997.

[13] M. Chang, L. Hung, C. Lin, and P. Su. Finding large k-clubs in undirected graphs. *Computing*, 95(9):739–758, 2013.

[14] J. Chen, I. A. Kanj, and W. Jia. Vertex cover: further observations and further improvements. *Journal of Algorithms*, 41(2):280–301, 2001.

[15] J. Chen, I. A. Kanj, and G. Xia. Improved upper bounds for vertex cover. *Theoretical Computer Science*, 411(40):3736–3756, 2010.

[16] R. G. Downey and M. R. Fellows. *Parameterized complexity*. Springer-Verlag, New York, 1999.

[17] J. Flum and M. Grohe. *Parameterized Complexity Theory, volume XIV of Texts in Theoretical Computer Science. An EATCS Series*. Springer Verlag, Berlin, 2006.

[18] J. Guo, C. Komusiewicz, R. Niedermeier, and J. Uhlmann. A more relaxed model for graph-based data clustering: s-plex editing. In *Algorithmic Aspects in Information and Management*, pages 226–239. Springer, 2009.

[19] S. Hartung, C. Komusiewicz, and A. Nichterlein. Parameterized algorithmics and computational experiments for finding 2-clubs. In *Parameterized and Exact Computation*, pages 231–241. Springer, 2012.

[20] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. In *Foundations of Computer Science, 1996. Proceedings., 37th Annual Symposium on*, pages 627–636. IEEE, 1996.

[21] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: a review. *ACM computing surveys (CSUR)*, 31(3):264–323, 1999.

[22] R. M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations, R.E. Miller, J.W. Thatcher, Eds., New York: Plenum Press*, pages 85–103, 1972.

[23] V. Krebs. Uncloaking terrorist networks. *First Monday*, 7(4), 2002.

[24] V. E. Lee, N. Ruan, R. Jin, and C. Aggarwal. A survey of algorithms for dense subgraph discovery. In *Managing and Mining Graph Data*, pages 303–336. Springer, 2010.

[25] H. Liu, P. Zhang, and D. Zhu. On editing graphs into 2-club clusters. In *Frontiers in Algorithmics and Algorithmic Aspects in Information and Management*, pages 235–246. Springer, 2012.

[26] R. J. Mokken. Cliques, clubs and clans. *Quality & Quantity*, 13(2):161–173, 1979.

[27] M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2):026113, 2004.

[28] R. Niedermeier. Invitation to fixed-parameter algorithms. 2006.

[29] F. M. Pajouh and B. Balasundaram. On inclusionwise maximal and maximum cardinality k-clubs in graphs. *Discrete Optimization*, 9(2):84–97, 2012.

[30] P. M. Pardalos and J. Xue. The maximum clique problem. *Journal of global Optimization*, 4(3):301–328, 1994.

[31] A. Schäfer, C. Komusiewicz, H. Moser, and R. Niedermeier. Parameterized computational complexity of finding small-diameter subgraphs. *Optimization Letters*, 6(5):883–891, 2012.

[32] S. B. Seidman and B. L. Foster. A graph-theoretic generalization of the clique concept. *Journal of Mathematical Sociology*, 6(1):139–154, 1978.

[33] S. Shahinpour and S. Butenko. Distance-based clique relaxations in networks: s-clique and s-club. In *Models, algorithms, and technologies for network analysis*, pages 149–174. Springer, 2013.

[34] A. Sinclair and M. Jerrum. Approximate counting, uniform generation and rapidly mixing markov chains. *Information and Computation*, 82(1):93–133, 1989.

[35] K. Voevodski, S.-H. Teng, and Y. Xia. Finding local communities in protein networks. *BMC bioinformatics*, 10(1):297, 2009.

[36] Y. Wang and X. Qian. Functional module identification by block modeling using simulated annealing with path relinking. In *Proceedings of the ACM Conference on Bioinformatics, Computational Biology and Biomedicine*, pages 123–130. ACM, 2012.

[37] Y. Wang and X. Qian. Functional module identification in protein interaction networks by interaction patterns. *Bioinformatics*, 30(1):81–93, 2014.

[38] Wikipedia. Wechat — wikipedia, the free encyclopedia. Available from: `https://en.wikipedia.org/w/index.php?title=WeChat&oldid=716734952`, 2016.

[39] A. Wotzlaw. On solving the maximum k-club problem. *arXiv preprint arXiv:1403.5111*, 2014.

[40] Q. Wu and J. Hao. A review on algorithms for maximum clique problems. *European Journal of Operational Research*, 242(3):693–709, 2015.