As Strong as the Weakest Link: Mining Diverse Cliques in Weighted Graphs

Petko Bogdanov[1], Ben Baumer[2], Prithwish Basu[3], Amotz Bar-Noy[4], and Ambuj K. Singh[1]

[1] University of California, Santa Barbara, CA 93106, USA, {petko,ambuj}@cs.ucsb.edu
[2] Smith College, Northampton, MA 01063, USA, bbaumer@smith.edu
[3] Raytheon BBN Technologies, 10 Moulton St., Cambridge, MA 02138, USA, pbasu@bbn.com
[4] The City University of New York, New York, NY 10016–4309, USA, amotz@sci.brooklyn.cuny.edu

Abstract. Mining for cliques in networks provides an essential tool for the discovery of strong associations among entities. Applications vary, from extracting core subgroups in team performance data arising in sports, entertainment, research and business; to the discovery of functional complexes in high-throughput gene interaction data. A challenge in all of these scenarios is the large size of real-world networks and the computational complexity associated with clique enumeration. Furthermore, when mining for multiple cliques within the same network, the results need to be diversified in order to extract meaningful information that is both comprehensive and representative of the whole dataset.
We formalize the problem of *weighted diverse clique mining (mDkC)* in large networks, incorporating both individual clique strength (measured by its weakest link) and diversity of the cliques in the result set. We show that the problem is NP-hard due to the diversity requirement. However, our formulation is sub-modular, and hence can be approximated within a constant factor from the optimal. We propose algorithms for *mDkC* that exploit the edge weight distribution in the input network and produce performance gains of more than *3* orders of magnitude compared to an exhaustive solution. One of our algorithms, *Diverse Cliques (*DiCliQ*)*, guarantees a constant factor approximation while the other, *Bottom Up Diverse Cliques (*BUDiC*)*, scales to large and dense networks without compromising the solution quality. We evaluate both algorithms on *5* real-world networks of different genres and demonstrate their utility for discovery of gene complexes and effective collaboration subgroups in sports and entertainment.

1 Introduction

It is often said that while the success of a sports or business team depends on good individual performances, it depends even more on how individuals *gel* as a team – thus resulting in the idiom "There is no "I" in T-E-A-M". While this

expression downplays the importance of individual performance, a team comprised of players with moderate individual talent but superior teamwork skills can outperform a dysfunctional team that emphasizes superlative individual performances. For example, the Detroit Pistons basketball team won the 2004 NBA championship with a collection of relatively unheralded players who were thought to collaborate so well together that "the whole was greater than the sum of its parts." Conversely, the team they defeated, the heavily-favored Los Angeles Lakers, featured four future Hall of Famers, none of whom appeared to collaborate particularly well together [14].[5] *Could the unexpected success of the Pistons or the demise of the Lakers have been predicted based on previous observations?*

While the importance of teamwork between elements in a group is easy to articulate, it is a non-trivial analytical task to isolate the core subgroups of entities that are responsible for the overall team performance. If the performance of the whole team can be measured, say, in terms of wins or losses, or revenue generated, the key problem is the discovery and identification of the *team cores* – subgroups within a team, whose inclusion results in higher-than-expected overall team performance, since their collaboration appears to motivate the success of the team as a whole. The discovery of core subgroups can illuminate distinctive individual characteristics, whose combination has a *super-additive* effect on the team [21]. This could provide important assistance to executives, who are ultimately judged by the success of the team, rather than the personal achievements of individual players. For example, sports executives could use team performance data from prior years to identify and acquire players exhibiting a combination of traits that lead to team success. Similarly, Hollywood studios can use data on prior collaborations among actors, directors, editors, cinematographers etc. while assembling a cast for an upcoming film, since successful past collaborations may portend similar success in the future.

"Teamwork" is not restricted to sports or business; it is also observed inside cells of living organisms – multiple proteins interact with each other to form a *multi-protein complex*, which is a cornerstone of many (if not most) biological processes, and together they form various types of molecular machinery that perform a vast array of biological functions [17]. The challenge here is to discover those complexes which are core groups of interacting genes within high-throughput pairwise interaction data [27]. The biological setting presents a distinctive challenge due to the difficulty in measuring the existence of a complex directly. The good news is that the strength of pairwise associations between genes can be tested efficiently via high-throughput methods employed to build functional interaction networks for analysis [28]. Hence, analytical techniques for mining strong gene subgroups can allow biologists to infer the existence of protein complexes that participate in the same cellular process and predict functional annotations for new gene sequences [11].

[5] Indeed, the open feud between Shaquille O'Neal and Kobe Bryant combusted after the season, resulting in the trade of O'Neal to Miami.

Our main goal is to extract an informative set of high-scoring[6] cliques that are representative of the entire network. To mitigate the presence of *free riders* (nodes attached by weak links to a strong clique), we define an intuitive score based on the *weakest link* in the clique. Alternative scoring schemes using sum or average are more forgiving to free riders since their link weights, albeit weak, would not drag the overall clique score down significantly. Similar scoring functions have been used in the Bioinformatics literature, namely Bandyopadhyay et al. [8] measure multi-way interaction strength as the minimum average weight of links adjacent to a node. A weakest link explanation of group success is also central to the *pooled interdependence theory* for business organizations as discussed by the classic text of Thompson [25].

Another challenge is to handle possible overlap among the best scoring cliques in order to represent all network locations of interest in the result set. Less overlap amounts to greater *diversity* among the reported cliques. Consideration of diversity is imperative in certain team sports such as ice hockey, in which a coach decides which *lines* (subgroups) of players play together on ice before being substituted by other *lines* Over the course of the season, the coach experiments with the makeup of these lines several times in order to figure out which players play well together. Diversity is also important in team formation for multiple tasks, where one aims to maximize the fitness of each team while simultaneously incorporating fairness by not overloading members with multiple tasks [6]. Thus, an analytical scheme that can mine a diverse set of subgroups is more useful than one which merely returns the top scoring ones without taking into account the overlap between them.

Our Contributions in this paper include the following:
Novelty: We formulate a novel *weighted diverse clique mining (mDkC)* problem that incorporates clique strength and diversity of result; and show that, although NP-hard, the formulation is sub-modular and allows accurate approximation schemes.
Scalability: We propose a $(1 - 1/e)$-approximation algorithm, DiCliQ, and a faster heuristic, BUDiC, for *mDkC*. Both achieve an improvement in running time of 3 orders of magnitude, when compared to an exhaustive search.
Quality: We demonstrate the utility of DiCliQ and BUDiC to identify team cores of significant performance and protein complexes in a gene interaction network.

2 Related Work

Clique mining work has focused on quasi (almost) cliques which allow a controlled number of missing edges [3, 15, 29]. While this relaxation is to accommodate possible missing links and noise, all of the above methods operate in the

[6] The edge weights between entities (player/protein/stock symbol) are indicative of the strength of their pairwise relation. Particularly well-performing subgroups (or tightly interacting proteins) manifest in the resulting graph as "strong" cliques associating nodes with heavy edge weights.

scenario of unweighted and labeled graphs and optimize the frequency of clique occurrence as opposed to scores (labels are not unique and hence a quasi-clique may occur multiple times in the same data graph or in a database of small graphs). In contrast, we operate in settings of dense and weighted interactions exhibiting variance in the link strength. Weighted clique mining was considered by Bandyopadhyay et al. [8] who proposed a heuristic for the largest cardinality clique with average node-connectivity weight exceeding a user-defined threshold. Instead of using the link quality as a *constraint*, we incorporate it in the solution score. In addition, we are interested in finding a diverse set of (multiple) high scoring cliques as opposed to the single largest one.

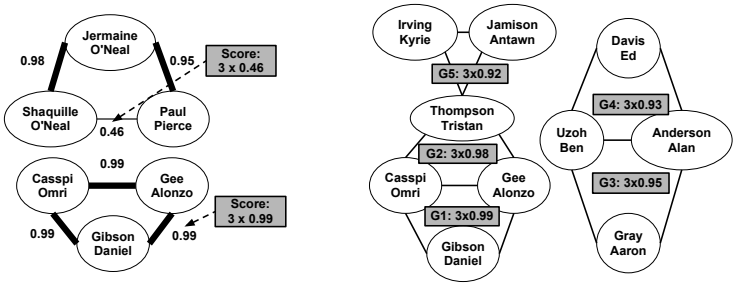
Different subgraph "goodness" criteria have also been considered in graph mining, including diameter and spanning tree cost [16], Steiner tree and bottleneck cost [18]. Communities and modules have also been defined based on clique percolation (highly overlapping cliques) in CFinder [22] and applied to biological and social networks in a series of follow-up work. Such formulations allow sparse structures including nodes that do not interact directly. Instead, our methods are targeted to the discovery of "flat–organization" teams and all-to-all interactions in the case of gene complexes. As we show experimentally, our method (and formulation) outperforms CFinder by 20% when employed in gene complex discovery.

Alternative definitions of diversity have also been considered. Lappas et al. [16] investigated diversity of the node roles within single cliques. This within-clique diversity definition is targeted to cases where multiple nodes may have the same role (label) and can be considered in conjunction with ours. Anagnastopolus et al. [6] considered structures with fair assignment of tasks within the team and in follow-up work [7, 18] combined communication cost with fair task assignment in online team formation. While these formulations target both overlap (task assignment) and structure, they incorporate one of the criteria (overlap or structure density) as a user-defined constraint and allow for sparse structures, i.e. they are suitable for hierarchical as opposed to "flat" teams/complexes.

3 Preliminaries

We model interaction strength between entities from sports, business, cinema and biology as a weighted undirected network $G = (V, E, w)$. Nodes V of the network graph correspond to agents or entities, while edge weights $w(u, v)$ between two nodes u and v reflect their connection strength: joint performance, interaction strength or similarity. For the rest of the presentation, without loss of generality, we assume edge weights are scaled to the interval $[0, 1]$.

Our goal is to extract a diverse set of groups, requiring that all internal pairwise connections are strong. In the graph setting, strong all-pair-connected groups map to cliques (complete graphs) of high weights on all edges. Finding the *Maximum Clique (*Max Clique*)* (the one of largest size in an unweighted graph) is NP-hard to solve and also approximate to within $n^{1-\epsilon}$ [13]. Introducing weights on edges preserves the same general complexity.



(a) Clique score (b) Small graph example

Fig. 1. Comparison of two cliques based on the weakest link as their score (a). Example of five high-scoring triples from NBA (b). A diverse 2-set of cliques of size 3 minimizes the overlap in the resulting set, i.e. prefers $\{G1, G3\}$ over $\{G1, G2\}$ (slightly lower total scores but increased diversity).

4 Problem Definition

In this paper, we model the strength of a subgroup in terms of the pairwise interactions between individuals in that subgroup. Ideally, techniques that go beyond graph theory, e.g., hypergraphs or abstract simplicial complexes [9, 19], are appropriate for modeling higher order interactions (triples, quadruples, etc.) directly. Such higher-order modeling approaches, however, are limited by both data scarcity and intractability. Enumerating and scoring all subgroups becomes computationally demanding for datasets of even hundreds of nodes. Moreover, in order to validate methods that score higher order groups directly, we need empirical data containing sufficiently many observations in which a particular group has interacted as a whole — a requirement that is hard to meet.

To address the above challenges, we require that all pairwise interactions (performances) are strong and relate the performance of a group to its weakest link, seeking to approximate the group behavior. Most professional sports teams, scientific collaborations, and protein networks are all examples of predominantly non-hierarchical ("flat") organizations that may collapse without strong pairwise links among all group members. This is true especially in a team sport such as basketball in which lack of good communication between any pair of players can easily threaten the success of the team (see [14]), especially since a highly competitive opponent is typically smart enough to exploit that vulnerability during the game. Thus, our clique scoring scheme is targeted to groups in which all pairs correlate/interact strongly, where the potential success of a group is limited by the worst pairwise connection therein.

We define the *score* of a group (network clique) in terms of its weakest link. That is, for any subset of nodes $C \subseteq V$, $s(C) = |C| \min_{u,v \in C} w(u, v)$. If C is not a clique (at least one missing edge), then the score is $s(C) = 0$. This scoring criterion prefers larger cliques, whose minimal edge score is high. It is also designed to eliminate the *free-rider* effect, i.e. inclusion of nodes that exhibit some weak links to others within the group.

An example from the NBA that illustrates our scoring method is shown in Fig. 1(a). Although both cliques feature high weight edges, a single weak link (between Shaquille O'Neal and Paul Pierce) results in a much lower score for the top triplet. Conversely, the {*Casspi,Gibson,Gee*} triplet is scored higher as all pairwise edges retain high score.

The clique score we adopt agrees with statistical measures of success for triples in our sports data sets with 90% of the top-scoring NBA triples having statistically significant performance (p-value ≤ 0.05). Furthermore, the highest scoring cliques of genes in our experimental gene network have homogeneous biological functions, and hence likely correspond to gene complexes. Details of the above observations are available in Sec. 6.

Based on our weakest link score definition, we formalize the problem of finding the best weighted fixed-size clique in a graph.

Definition 1. *The* maximum weighted k-clique *in a graph $G = (V, E, w)$ is the k-clique $C_k^*(G)$ of maximum score, i.e. $C_k^*(G) = \arg\max_{C \subseteq V, |C|=k} s(C)$. Given a weighted graph G,* Maximum Weighted k-Clique *(WkC) is the problem of finding the maximum weighted k-clique in G.*

WkC is NP-hard as it can be reduced from the Max Clique problem by restriction of the edge weights to 1. In our solutions, we will exploit the weight distribution of edges in a network in order to explore more promising cliques first and prune unpromising candidates for extension.

Beyond a single group, our goal is to report the best set of cliques, in order to represent all locations of interest in a large network. There are two main computational challenges to this end: (i) efficient discovery of good (high-scoring) cliques and (ii) ensuring informativeness of the result set by diversification.

To illustrate the intuition behind the importance of diversity in the resulting set, consider the NBA player example in Fig 1(b). There are five candidate cliques of size 3 and their scores are listed in shaded boxes in the corresponding triangles. Assuming that the top 2 highest scoring cliques are of interest, ignoring diversity amounts to reporting $\{G1, G2\}$. However, there is a lot of overlap (the duo *Casspi-Gee*) between $\{G1, G2\}$. Intuitively, reporting all super-cliques of *Casspi-Gee* is not representative of the overall network. Instead, we can diversify by pairing $G1$ with a slightly lower scoring clique of lower overlap such as $G3$. The idea of diversity has been considered in a number of other settings including information systems for web search [4, 10, 12], image retrieval [24, 26], cheminformatics [5] and other domains.

Next, we formalize a joint diversity-score formulation and a corresponding solution with a good approximation guarantee.

Definition 2. *For a set $\mathcal{A}$ of cliques each of size k, we define their diversity score as*

$$ds(\mathcal{A}) = \alpha \frac{\sum_{C \in \mathcal{A}} s(C)}{k} + (1 - \alpha) \frac{|\bigcup_{C \in \mathcal{A}} C|}{k},$$

where $\alpha \in [0, 1]$, $\bigcup_{C \in \mathcal{A}} C$ is the union of nodes in the cliques of $\mathcal{A}$ and $ds(\emptyset) = 0$. Then given parameters k and α, the m Diverse k-Clique *(mDkC) problem seeks the set $\mathcal{A}$ of size m that maximizes $ds(\mathcal{A})$.*

The above definition combines the average score of the answer set cliques and the diversity of their comprising nodes in a linear fashion. The trade-off between score and diversity can be controlled by the parameter α. Note that since both terms are bounded above by $|\mathcal{A}|$, $ds(\mathcal{A}) \in [0, |\mathcal{A}|]$.

If we set $m = \alpha = 1$, then $ds(\mathcal{A}) = s(C)$, and $mDkC$ is equivalent to WkC. Since, as we saw above, WkC is NP-hard, $mDkC$ is NP-hard. However, we prove the stronger statement that $mDkC$ is NP-hard even if an efficient heuristic for WkC (or an alternative high score structure) exists.[7] That is, we prove that the hardness of $mDkC$ comes not only from the hardness of WkC, but also from the difficulty of diversifying the result set.

Theorem 41 *For any scoring function $s()$ that maps a graph substructure to a non-negative real number, the decision problem corresponding to $mDkC$, namely: "Is there a set of m substructures $\mathcal{A}$, each of size k, such that $ds(\mathcal{A}) \geq B$ for some positive number B," is NP-complete.*

Proof. The proof is available in the Appendix [2].

While our focus is on cliques, the above theorem shows a more general result for arbitrary subsets of nodes in a graph given a scoring function for each subset. Hence, in different applications in which finding an optimal score substructure is computationally tractable, ensuring that the solution comprised of multiple substructures is diverse remains NP-hard.

Although the $mDkC$ problem is NP-hard (due to the NP-completeness of the decision version), we show that the diversity score function is monotonic and sub-modular. These properties allow a fixed-quality approximate solution based on a greedy scheme. Next, we formally show the monotonicity and sub-modularity of our diversity score formulation.

Theorem 42 *If k and α are fixed, the diversity score function $ds(\mathcal{A})$ is:*
- Monotonic, i.e. for any subset $\mathcal{A} \subseteq \mathcal{B}$, $ds(\mathcal{A}) \leq ds(\mathcal{B})$
- Sub-modular, i.e. for any sets $\mathcal{A}, \mathcal{B}$, $ds(\mathcal{A}) + ds(\mathcal{B}) \geq ds(\mathcal{A} \cup \mathcal{B}) + ds(\mathcal{A} \cap \mathcal{B})$.

Proof. The proof is available in the Appendix [2].

Due to the monotonicity and sub-modularity of the diversity score and based on the seminal result of Nemhauser et al [20], we can show the following corollary.

Corollary 41 *A* Greedy *procedure for $mDkC$ that adds cliques in decreasing order of their diversity score improvement always achieves a solution within $1-\frac{1}{e}$ from the optimal. Since $ds(\emptyset) = 0$, this the best possible approximation ratio for the problem.*

The $(1-1/e)$-approximation guarantee assumes that we can construct Greedy and hence solve WkC optimally (when $\alpha = 1$, the first clique to be added is the WkC solution). This is by itself a hard problem as we argued above, however, we exploit the edge weights to provide scalable solutions for real-world datasets.

[7] Recall that by [13], the general clique problem cannot be approximated within $n^{1-\epsilon}$ for any given ϵ.

5 Weighted Diverse Clique Mining

In this section we propose two algorithms for the *mDkC* problem: DiCliQ and BUDiC. Both adopt pruning of infeasible candidates based on partially explored cliques in order to reduce computation time. DiCliQ works by enumerating cliques within a thresholded version of the network: first high-scoring edges are considered and as the algorithm progresses lower-weight edges are included if needed. It provides a $(1 - 1/e)$-approximation guarantee as it implements a greedy strategy. For large and dense instances (exceeding $4,000$ nodes and $30,000$ edges) and for higher number of cliques and clique sizes, DiCliQ's running time worsens (requiring on the order of minutes to complete in our experimental datasets). To handle larger and denser instances, we develop a scalable heuristic BUDiC that achieves more than 90% of DiCliQ's diversity score (and at times even better scores than DiCliQ). BUDiC employs similar pruning, but avoids expensive enumeration of cliques by greedy expansion from a single edge.

5.1 Bounding the Diversity Score for Partial Cliques

We first show an upper bound for the contribution of a clique C when added to a set of cliques $\mathcal{A}$. If the newly added clique C is of the desired size k then its contribution to the overall score can be computed according to the definition of $ds()$. If, however, C is not a complete clique of the desired size $|C| < k$, one can bound the contribution of any of its super cliques (cliques that contains all nodes in C) of size k to the diversity score.

Theorem 51 *Let $C, |C| \leq k$ be a clique of size not exceeding k. The maximum improvement of ds score when adding any k super clique of C to a clique set $\mathcal{A}$ is bounded by:*

$$\delta(\mathcal{A}, C) = ds\left(\mathcal{A} \cup C\right) - ds(\mathcal{A}) = \alpha \min_{u,v \in C} w(u,v) + (1-\alpha)\frac{k - |(\cup_{B \in \mathcal{A}} B) \cap C|}{k},$$

where in the diversity part, the set $(\cup_{B \in \mathcal{A}} B) \cap C$ is the intersection of nodes included in $\mathcal{A}$ and nodes in C.

Proof. The proof is available in the Appendix [2].

The upper bound can be applied for incomplete cliques C of any size, even ones that are completely unobserved, i.e. $|C| = 0$. In the latter case, the score part increases by at most α (assuming the maximum possible edge weight is 1) and the diversity part increases by at most $(1 - \alpha)$. Equipped with the upper bound δ, we next define our edge thresholding algorithm DiCliQ that considers high-scoring cliques first and prunes infeasible candidates.

Algorithm 1 DiCliQ

Require: $G = (V, E, w), k, m, \alpha$, threshold schedule $T = \{T_i\}$
Ensure: A set of cliques $\mathcal{A} = \{C_i\}, |\mathcal{A}| = m, |C_i| = k$
1: $\mathcal{A} = \emptyset$, $l = 0$
2: **while** $|\mathcal{A}| < m$ AND $l < |T|$ **do**
3: Obtain $G_l(V, E_l, w), e \in E_l \iff w(e) \geq T_l$
4: Compute $\delta(\mathcal{A}, C_l^i), \forall |C_l^i| \leq k$ (incl. $C_l^i = \emptyset$)
5: **while** $\max_{|C_l^i|=k} \delta(\mathcal{A}, C_l^i) \geq \max_{|C_l^j|<k} \delta(\mathcal{A}, C_l^j)$ **do**
6: $\mathcal{A} = \mathcal{A} \bigcup argmax_{|C_l^i|=k} \delta(\mathcal{A}, C_l^i)$
7: break if $|\mathcal{A}| = m$
8: Update $\delta(\mathcal{A}, C_l^i)$ based on the new $\mathcal{A}$
9: **end while**
10: $l = l + 1$
11: **end while**
12: **return** $\mathcal{A}$

5.2 DiCliQ: Enumeration of Cliques with Thresholding

A naive *Baseline* heuristic for $mDkC$ (with the $(1-1/e)$-approximation) can (i) enumerate all possible cliques of the desired size k and then (ii) greedily (based on best $ds()$ improvement) compile an m-size result-set. While such *Baseline* might be feasible for small sparse networks (up to $|V| = 500$) and small values of k, the clique enumeration step quickly becomes a bottleneck as the input size increases due to its combinatorial nature. It fails to complete in less than 4 hours in all but our smallest network from the NBA.

Different from *Baseline*, we observe that in order to maximize the diversity score, we can first consider only edges of high weights. Then, as needed, we can consider lower score edges completing cliques of small overlap with the partial result set. Following this intuition, DiCliQ enumerates cliques in a thresholded subgraph induced by the highest-score edges and gradually includes more edges on demand. This process is based on a decreasing schedule $T = \{T_l\}$ of edge weight thresholds. The best-scoring cliques are discovered first within a much smaller instance of the graph. In addition, DiCliQ employs the upper bound on the improvement of the ds score for candidate cliques in order to filter out infeasible candidates and guarantee that cliques are added to the result set in a greedy order ensuring a $(1 - 1/e)$-approximation.

DiCliQ is presented in Alg. 1. Apart from the input graph and parameters k, m and α, the algorithm also takes as an input a schedule $\{T_l\}$ of descending edge value thresholds. The result set $\mathcal{A}$ is first initialized as empty and the threshold level l to 0 (i.e. highest edge values). While a set $\mathcal{A}$ of size m is not obtained and we have not reached the last level of thresholding, the algorithm (i) filters the graph based on T_l (Line 3), (ii) enumerates and upper-bounds all cliques of size up to k (Line 4) from the filtered graph and (iii) attempts to add cliques to the result set if they are the best next cliques to add (Lines 5-9). Note, that on Line 4 an upper bound $\delta(\mathcal{A}, \emptyset) = \alpha T_l + 1 - \alpha$ on all yet unobserved cliques is also computed.

If the maximum improvement δ of a size-k clique C_l^i exceeds the upper bound on any incomplete clique, we add C_l^i to the solution (Line 6) and update the

Algorithm 2 BUDiC

Require: $G = (V, E, w), k, m, \alpha$
Ensure: A set of cliques $\mathcal{A} = \{C_i\}, |\mathcal{A}| = m, |C_i| = k$
1: $\mathcal{A} = \emptyset$
2: **for** $i = 1 \rightarrow m$ **do**
3: **for all** Edges $e \in E$ **do**
4: $C_e = e$;
5: **repeat**
6: Grow C_e by $argmax_{v \in V} \delta(\mathcal{A}, C_e \bigcup v)$
7: **until** $(|C_e| = k) \vee (C_e$ cannot be extended)
8: **end for**
9: **if** no C_e of size k are found **then**
10: break
11: **end if**
12: $\mathcal{A} = \mathcal{A} \bigcup argmax_{|C_e|=k, e \in E} \delta(C_e)$
13: **end for**
14: **return** $\mathcal{A}$

improvements of the cliques based on the new $\mathcal{A}$ (Line 8). Additions of cliques are performed until no complete clique exceeds the upper bound of an incomplete one, or until $|\mathcal{A}| = m$. After all possible additions are exhausted, if the result set does not contain m cliques, we lower the edge weight threshold (Line 10) and repeat Lines 3-5 for the new thresholded graph. Since we add cliques to the results set only if their score improvement exceeds the upper bound of all possible candidates (line 5) we ensure that the cliques are added in a greedy (descending score) order and hence DiCliQ implements a greedy strategy and obtains a $(1 - 1/e)$-approximation.

The thresholding scheme of DiCliQ is effective when the result set of m best cliques is completed before reaching the lowest threshold level, i.e. enumerating cliques in the whole graph. An important means to this end is choosing an appropriate schedule that reflects the distribution of edges. We divide the set of all edge weights into equi-size bins and adjust the threshold to incorporate one more of these bins at every iteration. Other schedules (exponentially increasing subsets of edges) are also possible, but were not more favorable in our experiments.

5.3 BUDiC: Scalable *B*ottom-*U*p *Di*verse *C*lique Heuristic

The bottleneck in DiCliQ is the enumeration and bounding of all cliques at a given edge weight level (Line 4, Alg. 1). This step is in general exponential and the algorithm is efficient only when the results set is computed at the first several thresholding levels. To scale to larger and denser graphs, while avoiding exhaustive enumeration of cliques, we employ a greedy *Bottom-up* scheme BUDiC.

The intuition behind BUDiC (Alg. 2) is that one can get good candidates for the result set by starting from a good edge and growing a clique, while avoiding overlap according to the diversity α. Cliques are added one at time in the outer loop (Lines 2-13). Good local cliques are grown greedily by nodes of best improvement (Line 3-8). If no clique of the desired size is found the main loop is terminated and an incomplete set of cliques is returned (Line 9-11). The

best clique in each iteration is added to $\mathcal{A}$ (Line 12). Note, that BUDIC does not have the same approximation guarantee as DICLIQ because in the greedy expansion from an edge it does not consider all possible cliques.

The algorithm runs in polynomial time $O(k \cdot m \cdot n \cdot |E|)$ as every edge is grown to a clique of at most size k and this is repeated m times. The n term is due to the possibility of considering all graph nodes in Step 6 when the graph is complete. BUDIC is also suitable for parallel implementation, since Lines 3-8 can be executed on separate machines assuming the graph is partitioned with redundancy and distributed to all machines.

5.4 Discussion on Setting Parameters

While it is unlikely that a universally appropriate value of k exists, in certain applications there are domain-specific constraints that could be used. For example, in basketball, subgroups of sizes less than 5 are of interest, while in gene networks appropriate sizes are between 4 and 6 since many known yeast complexes contain 4.7 subunits on average [23]. When no prior domain knowledge is available, we envision varying k while tracking the relationship among consecutive result sets and concentrating on values for which the solution changes substantially (i.e. solution cliques of size k do not tend to include those of size $k-1$). A similar approach can be adopted to determine interesting values of the diversity weight α. We perform such analysis for α in the experimental section. The number of cliques in the resulting set (m) can be increased until the score contribution of adding additional cliques diminishes significantly relative to the average contributions of already included cliques.

6 Experimental Evaluation

We evaluate our algorithms on a variety of real world data from sports, cinema, biology and finance. Our goal in experimentation is to (i) assess the scalability of DICLIQ and BUDIC to large problem instances; (ii) demonstrate the quality of BUDIC compared to the $(1-1/e)$-approximation DICLIQ; and (iii) demonstrate the relevance of the mined diverse cliques to real world applications.

Data. We experiment with 5 publicly-available data sets including participation in teams sports (*NBA, MLB*), collaboration in movies (*IMDB*), a gene interaction network (*YeastNet*) and a correlation network of stock symbols (*Stocks*) (see Table 1). Edge strength in the sport/collaboration are based on the statistical significance of the performance of the pair of entities when in groups (sport team success or movie cast ratings). The edge weights in the gene network is based on strength of measured interaction of the genes, while the absolute Pearson's correlation serves as a weight in the stock network. The sizes of the datasets are listed in Table 1 (columns 2,3). We discuss in detail the sources and preprocessing of our datasets in the Appendix [2].

Scalability. All scalability measurements are on a Dell Desktop with $6GB$ RAM and Dual Core 4GHz processor. We measure the clock time of the exhaustive

Table 1. Summary of our datasets including time span, number of nodes V and edges E. The second part of the table lists the running time (in seconds) and quality (% of Baseline) on all datasets $\alpha = 0.5$, $m = 10, k = 5$. The quality of BUDiC and an iterative version of iMDV is measured as percentage of DiCliQ 's score. *Baseline* does not complete in $4h$ and its memory footprint exceeds $6GB$ causing an out-of-memory exception.

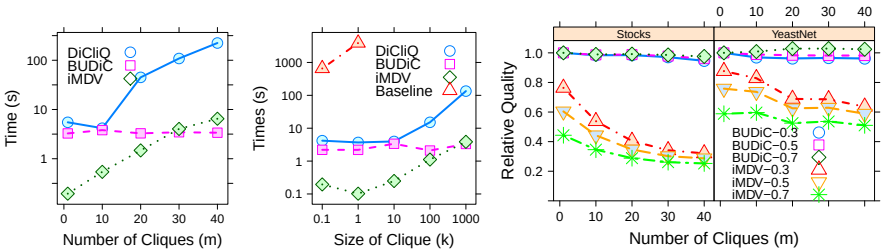
			Baseline	DiCliQ	iMDV		BUDiC	
Source	$\|V\|$	$\|E\|$	*Time*	*Time*	*Time*	*% sc.*	*Time*	*% sc.*
NBA	532	5,945	23.00s	0.48s	0.018s	57	0.13s	**98**
MLB	1,569	40,126	>4h	3.00s	0.015s	52	0.24s	**95**
IMDB	25,141	417,705	>4h	107.00s	0.087s	54	0.17s	**94**
YeastNet	4,450	30,5416	>4h	5.8s	0.463s	73	0.77s	**99**
Stocks	1,194	32,406	>4h	12s	0.022s	44	0.4s	**99**

Baseline, DiCliQ, BUDiC and the iterative extension of [8] called *iMDV*. Note that both DiCliQ and *Baseline* implement a greedy strategy and hence obtain a constant $1 - 1/e$ factor approximate solution. An optimal solution for the problem would further require considering all possible (exponential) subsets of cliques and is not be feasible even for our smallest datasets. Details of competing techniques are available in the Appendix [2].

The right part of Table 1 shows the performance of competing techniques in all datasets. *Baseline* was able to complete only on our smallest dataset NBA and it was 40 times slower than DiCliQ and 2 orders of magnitude slower than BUDiC. On the rest of the datasets ($\alpha = 0.5$, $m = 10$ and $k = 5$) *Baseline* does not complete in 4 hours and runs out of memory, due to the exponential number of candidate cliques that it has to consider for inclusion in the result set. DiCliQ, BUDiC and *iMDV* have comparable running time on small datasets, while in denser and larger networks DiCliQ is 10 to 100 times slower. In terms of diverse clique score, our fast heuristic BUDiC dominates *iMDV* by $30-50\%$.

We present the scalability behavior of our techniques for varying clique size and number of cliques in the results set within YeastNet in Fig. 2. *Baseline* does not complete in 4 hours for $k = 5$ and any value of m. The reason for this long running time is that *Baseline* consumes all allocated memory ($6GB$) while enumerating all possible cliques. For smaller clique sizes it is 3 to 4 orders of magnitude slower than DiCliQ and BUDiC. When increasing k and m, DiCliQ slows down due to the need to lower its edge weight threshold and enumerate more cliques in progressively larger graphs. BUDiC's performance does not change significantly for these experimental settings, making it a good scalable method for higher k and m.

Quality of BUDiC. BUDiC reduces the computational time by up to 2 orders of magnitude, as expected due to its polynomial complexity. However, an immediate question is: *What is its quality?* We showed that BUDiC's quality on all datasets is above 95% (diversity score as a fraction of *Baseline*'s score) for one setting of parameters in Table 1. Next, we explore the quality dependence on the number of cliques m in the result set and on the value of α.



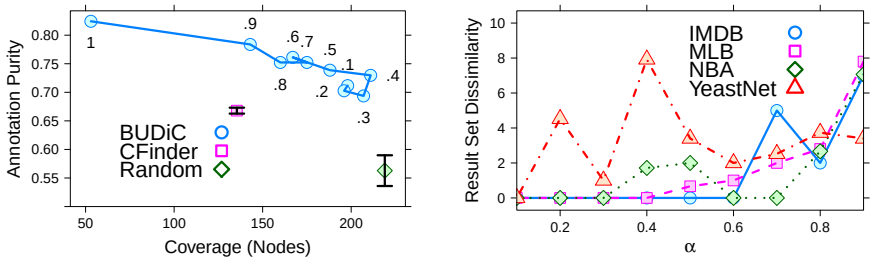
(a) Scalability with m (b) Scalability with k (c) Diversity score quality

Fig. 2. Scalability comparison of *Baseline*, DiCliQ and BUDiC for increasing number of cliques m in the result set for $k = 5, \alpha = 0.5$ (a) and increasing clique size k for $m = 5, \alpha = 0.5$ (b) in the *YeastNet* network. The Baseline approach does not complete in 4 hours for $k = 5, \alpha = 0.5$ and its memory footprint exceeds $6GB$. (c) Quality of BUDiC's diversity score as a fraction of the score obtained by a GREEDY heuristic (both *Baseline* and DiCliQ obtain the same score) in the *Stocks* (Left) and *YeastNet* (Right) networks($k = 5$).

Fig. 2(c) summarizes the quality of BUDiC in the YeastNet and Stocks networks. We show its diversity score as a fraction of a Greedy solution (obtained by either *Baseline* or DiCliQ.) For high values of α (i.e. when the clique score matters more than diversity), BUDiC is able to find even better score solutions than *Baseline*. On average, it behaves similar to the greedy alternatives with $(1-1/e)$-approximation. We observe similar behavior on the rest of the datasets as well. We also explored qualitatively the mined clique sets and found that for m up to 20 the intersection of the cliques obtained by BUDiC and *Baseline* (and DiCliQ) remains above 80% as well (i.e. only 2-3 cliques differ in the result sets). Hence, BUDiC achieves tremendous savings in time at almost no cost in quality in the data we analyzed.

Gene Complexes and Influential Sub-Groups. Next, we demonstrate the applicability of our formulation and methods for gene complex discovery and summarization of effective groups in sports. We label genes in YeastNet with known *process Gene Ontology (GO)*terms [1]. The GO labels are hierarchical with specificity increasing with the distance from the root. To account for varying specificity and hierarchy utilization, we only consider labels at level 4 and their descendants (i.e. 4 hops or more from the root). Annotations of higher specificity are mapped to their corresponding level 4 ancestors and the *YeastNet* network is filtered to include only genes that are annotated.

To evaluate the ability of BUDiC to identify meaningful gene complexes, we measure discovered groups' *purity* as the fraction of genes sharing the same label and compare to a recent overlapping community detection algorithm *CFinder* [22] and a *Random* subsets of genes as control (see details in the Appendix [2]). Figure 3(a) is a scatter plot of the average solution annotation purity versus coverage (the union size of nodes in the solution). Our diversity parameter α allows for control over the coverage/purity trade-off (labels of the BUDiC trace show the selected α). BUDiC's average group purity is 20% higher than that of



(a) Complex purity (b) Diversity effect

Fig. 3. (a) Average group *GO biological process* purity (fraction of genes sharing labels) versus coverage (number of covered nodes) in *YeastNet*. Comparison among BUDiC (varying α labeling each point), CFinder ($\delta = 0.6, w = 0.15$) and *Random* grouping of genes ($k = 6, m = 37$); (b) Dissimilarity of consecutive solutions when increasing α. Peaks correspond to drastic changes in the result set ($m = 10, k = 4$).

CFinder [22] (at coverage 136 nodes) and 30% higher than the average random purity (at coverage 220 nodes). This separation demonstrates that our minimum edge weight formulation allows for discovery of biologically more relevant complexes, while allowing for diversity (overlap) control within the result set.

In sports data, we compared the scores of DiCliQ cliques and the significance scores of the corresponding groups of players (in the form of p-values). In the NBA data set, DiCliQ retrieves 44% of the triples of lowest p-values (considering 1% of the lowest p-value triples). These high-performing triples are of paramount interest, since they represent the cliques that are likely driving team success. Furthermore, the average p-values of DiCliQ's top cliques are comparable with the reference set of lowest p-value triples (0.010 versus 0.026) with 90% of DiCliQ triples having p-value less than 0.05 (a common level of determining statistical significance in general scenarios). We discuss the mined subgroups and their relevance across the various datasets in the Appendix [2].

Effect of Diversity. By changing the value of the diversity parameter (α), we can alter the amount of overlap between the cliques returned by BUDiC. In Fig. 3(b), we show how the result sets change as a function of α. For any two consecutive values of α (e.g. 0.3 and 0.4), we obtain two result sets A and B. To measure their dissimilarity, we form the complete bipartite graph between the cliques in A and B, and assign weights to the edges based on the Jaccard similarity of the individuals cliques. Thus, for each clique $C_a \in A, C_b \in B$, the weight of the corresponding edge is given by $1 - Jaccard(C_a, C_b) = 1 - |C_a \cap C_b|/|C_a \cup C_b|$. The maximum weighted matching on this graph provides a dissimilarity score for A and B.

In Fig. 3(b), the dissimilarity between result sets in the NBA, for example, spikes at α between 0.3 and 0.6. The top 3 scoring quartets returned by BUDiC consist of only five distinct players, all playing for the Cleveland Cavaliers. However, by increasing diversity ($\alpha = 0.4$) we retain the first and third quartets only and bring in a different team quartet. Thus, α allows for application-specific

control of the amount of diversity desired in the result sets. When exploring a new data set, appropriate α values can be chosen based on the tipping points of the solutions (spikes in Fig. 3(b)).

7 Conclusion

Mining strong subgroups in networks is an important, yet challenging computational problem. In this paper, we proposed a novel and flexible formulation $mDkC$, in which a *diverse* set of *strong* cliques is identified. We show that $mDkC$ is NP-hard, but due to its submodularity, allows a constant factor approximation. We develop scalable approximation schemes: DiCliQ with $(1 - 1/e)$-approximation guarantee and BUDiC that scales to large and dense networks. Both algorithms are more than 3 orders of magnitude faster compared to exhaustive counterparts, and BUDiC achieves 2 times higher scores than previous clique-mining heuristics. We demonstrate the utility of our algorithms for identifying interesting sets of high-performance collaborators in sports and entertainment, and complexes of similar biological function (30% improvement over earlier approaches) in gene networks. The developed algorithms thus present a useful tool for mining influential core subgroups in large networks from diverse sources.

Acknowledgements. This research was supported by the Army Research Laboratory under cooperative agreement W911NF-09-2-0053 (NS-CTA). The content of the information does not necessarily reflect the position or the policy of the Government, and no official endorsement should be inferred. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notice herein.

References

[1] The gene ontology, `http://www.geneontology.org/`.
[2] Appendix, `http://cs.ucsb.edu/~dbl/papers/cliquesappendix.pdf`. 2013.
[3] C. C. Aggarwal, Y. Li, P. S. Yu, and R. Jin. On dense pattern mining in graph streams. *PVLDB*, 2010.
[4] R. Agrawal, S. Gollapudi, A. Halverson, and S. Ieong. Diversifying search results. In *WSDM*, 2009.
[5] L. Akella and D. DeCaprio. Cheminformatics approaches to analyze diversity in compound screening libraries. *Current Opinion in Chemical Biology*, 14(3):325–330, 2010.
[6] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi. Power in unity: forming teams in large-scale community systems. In *CIKM*, 2010.
[7] A. Anagnostopoulos, L. Becchetti, C. Castillo, A. Gionis, and S. Leonardi. Online team formation in social networks. In *WWW*, pages 839–848, 2012.
[8] S. Bandyopadhyay and M. Bhattacharyya. Mining the largest dense vertexlet in a weighted scale-free graph. *Fundam. Inform.*, 96(1-2):1–25, 2009.

[9] M. Brinkmeier, J. Werner, and S. Recknagel. Communities in graphs and hypergraphs. In *CIKM*, 2007.
[10] G. Capannini, F. M. Nardini, R. Perego, and F. Silvestri. Efficient diversification of search results using query logs. In *WWW*, pages 17–18, 2011.
[11] G. Cui, Y. Chen, D.-S. Huang, and K. Han. An Algorithm for Finding Functional Modules and Protein Complexes in Protein-Protein Interaction Networks. *J. of Biomedicine and Biotechnology*, 2008.
[12] A. Dubey, S. Chakrabarti, and C. Bhattacharyya. Diversity in ranking via resistive graph centers. In *KDD*, pages 78–86, 2011.
[13] J. Håstad. Clique is hard to approximate within $n^{1\text{-epsilon}}$. In *FOCS*, 1996.
[14] P. Jackson and M. Arkush. *The last season: a team in search of its soul.* Penguin, 2004.
[15] D. Jiang and J. Pei. Mining frequent cross-graph quasi-cliques. *TKDD*, 2(4), 2009.
[16] T. Lappas, K. Liu, and E. Terzi. Finding a team of experts in social networks. In *KDD*, 2009.
[17] X.-L. Li, S.-H. Tan, C.-S. Foo, and S.-K. Ng. Interaction Graph Mining for Protein Complexes Using Local Clique Merging. *Genome Informatics*, 16(2):260–269, 2005.
[18] A. Majumder, S. Datta, and K. Naidu. Capacitated team formation problem on social networks. In *KDD*, 2012.
[19] T. Moore, R. Drost, P. Basu, R. Ramanathan, and A. Swami. Analyzing collaboration networks using simplicial complexes: A case study. In *INFOCOM WKSHPS*, pages 238–243, 2012.
[20] G. Nemhauser, L. Wolsey, and M. Fisher. An analysis of approximations for maximizing submodular set functions–i. *Mathematical Programming*, 14(1):265–294, 1978.
[21] D. Oliver and M. Fienen. Importance of teammate fit: Frescoball example. *J. of Quantitative Analysis in Sports*, 5(1), January 2009.
[22] G. Palla, I. Derenyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 435:814–818, 2005.
[23] S. Pu, J. Wong, B. Turner, E. Cho, and S. J. Wodak. Up-to-date catalogues of yeast protein complexes. *Nucleic acids research*, 37(3):825, 2009.
[24] K. Song, Y. Tian, W. Gao, and T. Huang. Diversifying the image retrieval results. In *ACM Multimedia*, 2006.
[25] J. Thompson. *Organizations in Action: Social Science Bases of Administrative Theory.* Classics in Organization and Management Series. Trans. Publishers, 1967.
[26] R. H. van Leuken, L. G. Pueyo, X. Olivares, and R. van Zwol. Visual diversification of image search results. In *WWW*, pages 341–350, 2009.
[27] Q. Yang and S. Lonardi. A graph decomposition approach to the identification of network building modules. In *W. BIOKDD*, 2007.
[28] C. H. You, L. B. Holder, and D. J. Cook. Temporal and structural analysis of biological networks in combination with microarray data. In *CIBCB*, 2008.
[29] Z. Zeng, J. Wang, L. Zhou, and G. Karypis. Coherent closed quasi-clique discovery from large dense graph databases. In *KDD*, pages 797–802, 2006.