

Understanding Importance of Collaborations in Co-authorship Networks: A Supportiveness Analysis Approach*

Yi Han[†] Bin Zhou[‡] Jian Pei[‡] Yan Jia[†]
[†]National University of Defense Technology, China
yihan@nudt.edu.cn, jiayanjy@vip.sina.com
[‡]Simon Fraser University, Canada
{bzhou, jpei}@cs.sfu.ca

Abstract

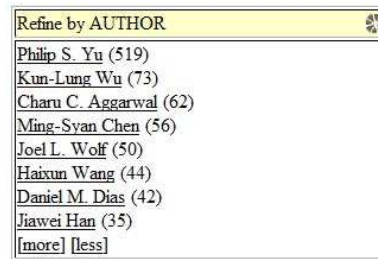
Co-authorship networks, an important type of social networks, have been studied extensively from various angles such as degree distribution analysis, social community extraction and social entity ranking. Most of the previous studies consider the co-authorship relation between two authors as a collaboration. In this paper, we introduce a novel and interesting “supportiveness” measure on co-authorship relation. The fact that two authors co-author one paper can be regarded as one author supports the other’s scientific work. We propose several supportiveness measures, and exploit a supportiveness-based author ranking scheme. Several efficient algorithms are developed to compute the top- n most supportive authors. Moreover, we extend the supportiveness analysis to community extraction, and develop feasible solutions to identify the most supportive groups of authors. The empirical study conducted on a large real data set indicates that the supportiveness measures are interesting and meaningful, and our methods are effective and efficient in practice.

1 Introduction

Co-authorship networks have been studied extensively from various angles such as degree distribution analysis [15, 10, 2, 1], social community extraction [18, 29, 26, 7], social entity ranking [16, 6, 36, 14, 24], and social link prediction [27, 5]. For example, the co-authorship relation has been analyzed in the context of mathematical sciences (e.g., the Erdős Number Project, <http://www.oakland.edu/enp>) and information retrieval [23]. Co-authorship analysis has also been extended to analyzing co-starring in movies, also known as *the Oracle Of Bacon* (<http://oracleofbacon.org/index.html>).

In a co-authorship network, each author is repre-

*The research of Bin Zhou and Jian Pei was supported in part by an NSERC Discovery grant and an NSERC Discovery Accelerator Supplements grant. The research of Yi Han and Yan Jia was supported in part by China National High-tech R&D Program (863 Program) 2006AA01Z451, 2007AA01Z474 and 2007AA010502. All opinions, findings, conclusions and recommendations in this paper are those of the authors and do not necessarily reflect the views of the funding agencies.



Refine by AUTHOR	
Philip S. Yu	(519)
Kun-Lung Wu	(73)
Charu C. Aggarwal	(62)
Ming-Syan Chen	(56)
Joel L. Wolf	(50)
Haixun Wang	(44)
Daniel M. Dias	(42)
Jiawei Han	(35)
[more]	[less]

Figure 1: The ranked co-author list of “Philip S. Yu” on DBLP.

sented as one vertex, and an edge represents the papers co-authored by two authors. Patterns mined from a co-authorship network have a few important applications, such as academic author ranking [17, 36, 16] and expert recommendation [34, 28].

Many existing studies on co-authorship networks model the co-authorship relation between two authors by an unweighted edge. However, such a method does not take into account the closeness of the relation.

To fully understand the co-authorship relation, an essential but remaining open question is *how important the collaboration between two authors is to each author and to the community*. Answering this question is critical for co-authorship network analysis. For example, it is interesting to rank the co-authors of a specific author based on the importance of the collaborations.

As a concrete example, recently, DBLP (<http://www.informatik.uni-trier.de/~ley/db/>) has provided an author a refined ranking function. For example, by searching “Philip S. Yu” on DBLP, the complete publication records are shown on the main column. Moreover, on the right upper corner, a ranked list of Philip’s co-authors in the number of co-authored papers is shown as well. Figure 1 shows a screen shot.

Kun-Lung Wu is ranked in the first place, who co-authored 37 papers with Philip. What does the

Paper-id	Authors
p_1, p_2	Ada, Bob, Deborah
p_3	Ada, Cathy
p_4, p_5	Ada, Deborah
p_6, p_7	Cathy, Deborah
p_8, p_9	Ada
$p_{10}, p_{11}, p_{12}, p_{13}, p_{14}, p_{15}$	Cathy

Table 1: A set of papers and the authors.

weight 37 tell us about the collaboration between Philip and Kun-Lung? How important is the collaboration to Philip, to Kun-Lung, and also to the communities that Philip and Kun-Lung belong to?

To answer the above questions, it is essential to understand the importance of collaborations. For a specific author, the collaborations between different co-authors may have different importance. Moreover, a collaboration may have different importance with respect to different participating authors.

To the best of our knowledge, there are no previous studies on importance of collaborations taking into account the weights of collaborations for co-authorship relation analysis. To tackle the problem, in this paper, we introduce a novel and interesting ‘‘supportiveness’’ measure on co-authorship relation. The fact that two authors co-author one paper can be regarded as one author supports the other’s scientific work. The supportiveness can be measured by analyzing the tightness of the collaboration. We make following contributions.

First, we propose the supportiveness measure in co-authorship networks. For an author a , the supportiveness from author b to a is used to measure how close the collaborations from b to a . We model supportiveness in a novel way, and show that supportiveness analysis is meaningful in some applications.

Second, we develop efficient methods to extract top- n most supportive authors in co-authorship networks. We model the supportiveness ranking problem as a reverse k nearest neighbor (k -RNN for short) searching problem on graphs. In this paper, we consider the number of k -RNNs of an entity as a measure of importance, and we propose several efficient methods to generate the competitive candidates and extract the top- n most supportive authors from them. Our methods utilize the graph structure and exploit some interesting patterns of the co-authorship graph. Some powerful pruning strategies are proposed to speed up the search process.

Third, we extend the supportiveness analysis from two authors to a group of authors. Specifically, we call two authors a mutual pair if the two authors regard each other as one of its k -RNN. The definition of mutual pair

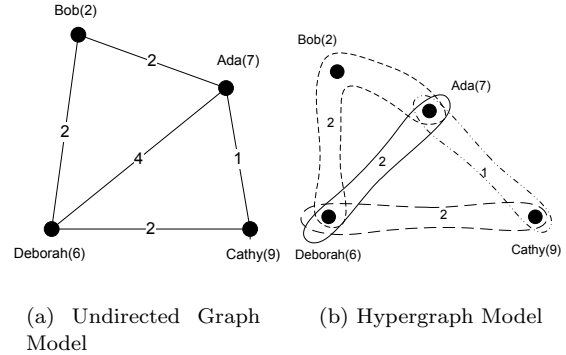


Figure 2: Graph models of co-authorship networks.

can be extended to a group of authors such that any two authors in the group form a mutual pair. We propose efficient algorithms to extract those groups of authors.

Last, we conduct systematic experiments on the DBLP data set, which contains more than 1 million publications and more than 600 thousand authors. The experimental results indicate that our supportiveness analysis is useful and interesting, and our methods are effective and efficient.

The rest of the paper is organized as follows. We formulate the supportiveness measure in Section 2, and review the related work in Section 3. We discuss the extraction of top- n most supportive authors in Section 4. We further propose mutual pairs and mutual cliques, and develop a feasible algorithm to extract mutual cliques in Section 5. A systematic empirical study conducted on the DBLP data set is reported in Section 6. Section 7 concludes the paper.

2 Ranking Authors Based on Supportiveness

Let us start with an example. Table 1 shows a synthesized set of 15 papers and the authors. A co-authorship network can be built as shown in Figure 2(a), which is an undirected graph. In the graph, each vertex represents an author. An author is also labeled by the total number of papers she/he participates. An edge connects two authors if they ever co-authored a paper. The weight of the edge is the total number of papers they co-author.

One may think that the weights on edges show the closeness among authors – the larger the weight, the closer the two participating authors. However, using the weights directly may be misleading. A paper with n co-authors leads to a clique of $\binom{n}{2}$ edges in the graph. The paper is counted $\binom{n}{2}$ times in weighting. The number of papers co-authored by a selected group cannot be derived by the weights.

As an example, consider the edge (Cathy, Deborah)

in Figure 2(a). It is unclear from the graph where the weight 2 comes from. Is there a paper co-authored by Ada, Cathy and Deborah altogether? Using edges weighted by the number of papers co-authored by two authors causes ambiguity.

To better model the co-authorship relation, we use hypergraphs in this paper.

2.1 Co-authorship Networks as Hypergraphs A co-authorship network can be modeled as a hypergraph $G = (V, E, \mathcal{L}, \mathcal{W})$, where V is the set of authors in the network, a hyperedge $e = \{v_1, \dots, v_n\} \in E$ is a subset of V , that is, $e \subseteq V$, representing a paper co-authored by v_1, \dots, v_n .

The vertices and hyperedges in G are labeled. \mathcal{L} is a labeling function on vertices and \mathcal{W} is a labeling function on hyperedges. Each vertex $v \in V$ is labeled by $\mathcal{L}(v)$, the number of papers that author v publishes. Each edge $e = \{v_i, \dots, v_j\}$ is labeled by $\mathcal{W}(e)$, the number of papers exclusively co-authored by authors v_i, \dots, v_j . Figure 2(b) shows the hypergraph of the co-authorship network in Table 1. Immediately, we have the following property.

PROPERTY 2.1. (LABELS) For any $e = \{v_1, \dots, v_n\}$,

$$1 \leq \mathcal{W}(e) \leq \min_{i=1}^n \{\mathcal{L}(v_i)\}$$

Moreover, for any $v \in V$, let $E(v)$ be the set of edges that v participates in, that is, $E(v) = \{e \in E | v \in e\}$. Then,

$$\sum_{v \in e, e \in E_v} \mathcal{W}(e) \leq \mathcal{L}(v).$$

Using hypergraphs to model co-authorship networks has some advantages. First, a collaboration among more than 2 authors cannot be captured properly in a simple graph model. To the contrary, such information is straightforwardly captured by a hyperedge. Second, as analyzed before, the weights on the edges in a simple graph cannot reveal the precise closeness among a set of authors. In a hypergraph model, each paper only contributes to one hyperedge. Thus, the total number of papers co-authored by a selected group can be derived by summing up the weights on the hyperedges among those vertices.

2.2 A Harmonic Distance Measure As discussed before, the weights on edges cannot be directly used to measure the closeness between authors properly. To model closeness precisely, we introduce a contribution-based closeness measure.

We consider the collaboration between two authors as the support to each other. In Figure 2(b), Ada and

Name	Ada	Bob	Cathy	Deborah	RNN
Ada	N/A	2.25	8	1.625	1
Bob	2.25	N/A	∞	2	0
Cathy	8	∞	N/A	3.75	0
Deborah	1.625	2	3.75	N/A	3

Table 2: The distance matrix of 4 authors.

Bob co-author 2 papers. For Ada, the 2 papers counts $\frac{2}{8} = 28.6\%$ of her papers, while the 2 papers counts $\frac{2}{2} = 100\%$ of Bob's. Therefore, the support from Ada to Bob is stronger than the other direction.

DEFINITION 1. (CONTRIBUTION) For authors u and v ($u \neq v$), the **contribution** from v to u , denoted by $cont(u \leftarrow v)$, is

$$cont(u \leftarrow v) = \frac{\sum_{u,v \in e} \mathcal{W}(e)}{\mathcal{L}(u)}.$$

Contribution $cont(u \leftarrow v)$ measures how much the collaboration between u and v contributes to the total work of u . Apparently, for $u, v \in V$, $0 \leq cont(u \leftarrow v) \leq 1$.

DEFINITION 2. (CLOSENESS AND DISTANCE) For authors u and v , the **closeness** is the harmonic mean of $cont(u \leftarrow v)$ and $cont(v \leftarrow u)$, that is,

$$\begin{aligned} closeness(u, v) &= \frac{1}{\frac{1}{2}(cont(u \leftarrow v) + cont(v \leftarrow u))} \\ &= \frac{2cont(u \leftarrow v)cont(v \leftarrow u)}{cont(u \leftarrow v) + cont(v \leftarrow u)}. \end{aligned}$$

If u and v co-author at least one paper, the **distance** between u and v is

$$\begin{aligned} dist(u, v) &= \frac{1}{closeness(u, v)} \\ &= \frac{cont(u \leftarrow v) + cont(v \leftarrow u)}{2cont(u \leftarrow v)cont(v \leftarrow u)}. \end{aligned}$$

A collaboration is bi-directional. A collaboration between u and v may contribute differently to u and v . It is natural to use the mean rather than the contribution from one to the other to measure the closeness between the two authors. Several Pythagorean mean measures are available, including the arithmetic mean, the geometric mean, and the harmonic mean. We define the closeness measure in harmonic mean instead of arithmetic mean or geometric mean because harmonic mean biases on the smaller number. This is desirable in measuring importance of contributions. For example, for two authors u, v such that $cont(u \leftarrow v) = 1$ and $cont(v \leftarrow u)$ is very small, the overall importance of the collaboration should be small instead of the average which is still larger than 0.5.

Please note that the distance measure defined as such is not metric. For example, in the co-authorship

network in Figure 2(b), the distances between the authors are shown in Table 2, we have $dist(Ada, Cathy) \geq dist(Ada, Deborah) + dist(Deborah, Cathy)$.

2.3 Supportiveness Based on k -NN and k -RNN

Let us consider authors' nearest neighbors in a co-authorship network.

EXAMPLE 1. (k -NN AND k -RNN) In the co-authorship network in Figure 2(b) whose distance matrix is in Table 2, Bob has 2 co-authors, Ada and Deborah. $dist(Ada, Bob) = 2.25$ and $dist(Deborah, Bob) = 2$. Deborah is Bob's nearest neighbor. It can be easily verified that Deborah is Ada's nearest neighbor and Cathy's nearest neighbor as well. Interestingly, Ada is Deborah's nearest neighbor.

Ada, Bob and Cathy all consider Deborah as their nearest neighbor, but only Deborah considers Ada as her nearest neighbor. In other words, Deborah supports more authors than Ada, thus Deborah is more supportive.

The way to measure supportiveness in Example 1 shares the similar philosophy of reverse nearest neighbors (RNN for short).

DEFINITION 3. (NN AND RNN) In a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, for vertex $u \in V$, the set of **nearest neighbors** of u is

$$NN(u) = \{v \in V \mid \nexists v' \in V : dist(u, v) > dist(u, v')\}.$$

The set of **reverse nearest neighbors** of u is

$$RNN(u) = \{v \in V \mid u \in NN(v)\}.$$

Following the idea in Example 1, we define the supportiveness measure. The RNN-based supportiveness score mimics a voting process. Every author has one vote. If $\|NN(u)\| = 1$, u votes for its nearest neighbor. If $\|NN(u)\| > 1$, u splits its vote evenly for its nearest neighbors. The supportiveness of an author is the votes she or he receives.

DEFINITION 4. (SUPPORTIVENESS) In a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, the **supportiveness** of author $u \in V$ is

$$sup(u) = \sum_{v \in RNN(u)} \frac{1}{\|NN(v)\|}.$$

PROPERTY 2.2. In a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, for any $u \in V$, $sup(u) \leq \mathcal{L}(u)$.

EXAMPLE 2. Table 2 shows the distances between the authors in Figure 2(b). We have $RNN(Ada) =$

$\{Deborah\}$, $RNN(Bob) = RNN(Cathy) = \emptyset$, and $RNN(Deborah) = \{Ada, Bob, Cathy\}$. As a result, we have $sup(Ada) = 1$, $sup(Bob) = 0$, $sup(Cathy) = 0$, and $sup(Deborah) = 3$.

The RNN-based supportiveness measure can be easily extended to the case of k -RNN.

DEFINITION 5. (k -NN AND k -RNN) In a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, for $u, v \in V$, v is a **k -NN vertex** of u if there do not exist k other vertices $v' \in V$ such that $dist(u, v) > dist(u, v')$. The set of k -NN vertices of u is denoted by $kNN(u)$.

The set of **k -RNN vertices** of u is $kRNN(u) = \{v \mid u \in kNN(v)\}$.

The **k -RNN supportiveness** of author u is

$$sup_{kRNN}(u) = \sum_{v \in kRNN(u)} \frac{k}{\|kNN(v)\|}.$$

3 Related Work

Our work is highly related to the previous studies on co-authorship network analysis, RNN search and maximum clique discovery. In this section, we review some representative work briefly.

3.1 Co-authorship Network Analysis Co-authorship network analysis has been conducted in many aspects, such as pattern and degree analysis [1, 16, 4], social community extraction [18, 29], frequent substructure identification [7, 31, 26], and centrality discovery [25, 5].

Some previous studies focused on analyzing static and dynamic properties of various co-authorship networks. Nascimento *et al.* [16] measured the activity of authors as the average distances to the others, while the distance is calculated using the length of a path in the graph. Newman [17] applied modern network analysis techniques to study the static network properties of several co-authorship networks. The last several years have seen the development of systematic algorithmic approaches to dynamic network analysis [9, 10, 15]. Recently, Huang *et al.* [6] conducted experimental studies on the CiteSeer data set to explore the importance of authors using some inherent properties of the network, such as structure, centrality and connectivity. All of the above studies considered that the linkages between authors are bi-directionally equivalent, and some hidden properties, such as the support to each other and the closeness between peers as defined in Section 2, have not been mined.

Several well known link-based ranking algorithms such as PageRank [20] and HITS [11] have been introduced for ranking entities in social networks. For

each vertex, the amount of ranking contribution from a neighbor is decided by the ranking score and the out-degree of the neighbor. Such methods can be applied to measure the importance of authors in citation network as well [3, 36]. However, since those link-based methods are mainly designed for directed graphs, they cannot be applied on co-authorship networks directly.

Some variations of link-based ranking algorithms have been proposed. For example, Zhou *et al.* [36] extended the traditional PageRank algorithm and proposed a new link analysis ranking approach by co-ranking authors and documents in the co-authorship and citation networks. Using the ideas of PageRank, the method is based on coupling two random walks into a combined one, presumably exploiting the mutually reinforcing relationship between documents and their authors: good documents are written by reputable authors and vice versa. However, this method is hard to extend to a group of entities. Furthermore, in co-authorship networks, the authors are grouped by the co-authored papers, and in each group, authors are highly knitted. Such highly connected graphs can be regarded as page farms [35], which make some individuals benefit from the members in the groups and get higher score than their due. Our methods in this paper focus on mining the sociality of the authors, and can be easily extended to a group of entities.

3.2 RNN Search in Metric Space and Graphs

Many studies have been conducted on calculating the RNN (reverse nearest neighbors) in Euclidean space [12, 30, 32, 13]. Pre-computation and vertex pruning are widely used in the previous studies. However, all those searching techniques are only capable to RNN search in metric spaces and cannot be adopted on graphs for several reasons. First, most of the previous methods are based on some indexes, such as R-tree. However, those indices cannot be applied on graphs directly. Second, in the Euclidean space, the number of RNNs is restricted by the dimensionality of the space. However, such a restriction does not hold on graphs.

Recently, Yiu *et al.* [33] conducted the first work that deals with RNN queries in large networks. They considered undirected weighted graphs where each edge is weighted using some distance measure. The network distance between two vertices is defined as the minimum of the sums of weights of all paths between them. Yiu *et al.* [33] showed that the network distance requires specific techniques for RNN processing, since the existing methods for RNN search in the Euclidean space are inapplicable on networks. The problem in [33] and the ones in this paper are different. We do not consider the network distance on paths. Due to the essential differ-

Algorithm 1 Top- n supportive author extraction: a straightforward algorithm

Input: a graph $G = (V, E, \mathcal{L}, \mathcal{W})$, parameters n and k ;
Output: the top- n vertices with the largest k -RNN supportiveness scores;

- 1: **for** each vertex $v \in V$ **do**
- 2: scan all neighbors of v and generate $kNN(v)$;
- 3: for each $u \in kNN(v)$, give u a vote with weight $\frac{k}{\|kNN(v)\|}$;
- 4: **end for**
- 5: **for** each vertex $v \in V$ **do**
- 6: calculate the total vote that v receives;
- 7: **end for**
- 8: output the top- n vertices with the largest votes;

ence in problem settings, the pruning strategies in [33] cannot be applied to solve our problems.

3.3 Clique discovery in Graphs In computational complexity theory, the problem of determining whether a graph contains a clique of a given size is one of Richard Karp’s original 21 problems shown NP-complete [8]. The maximum clique problem of finding the largest clique in a graph is also proved to be NP-complete [8]. In Section 5, we advocate to find mutual cliques as communities in co-authorship networks where authors in the clique support each other strongly. We do not aim at finding the largest cliques. Instead, we focus on finding the cliques with high supportiveness. We develop heuristics to speed up the search. Recently, some studies [21, 19] explore mining quasi-cliques on one or multiple graphs, which cannot be applied to our problems here directly.

4 Mining Top- n Most Supportive Authors

In this section, we study the problem of mining supportive authors. Given a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, we want to compute the top- n authors with the highest k -RNN supportiveness scores.

4.1 A Straightforward Method To calculate the k -RNN supportiveness score for a vertex u , we can scan the set of vertices once. For each vertex v , we can identify its k nearest neighbors $kNN(v)$. Then, we can give the votes from v to $u \in kNN(v)$ with weight $\frac{1}{\|kNN(v)\|}$. Finally, we calculate the vote received by each vertex and output the top- n vertices with the largest scores. Algorithm 1 gives the pseudocode.

4.2 Pruning Strategies

Apparently, the straightforward algorithm has to

scan the whole graph. This can be costly when the graph is large. Can we prune the search space and speed up the mining process?

Heuristically, a vertex with a large degree is less likely to have a small k -RNN supportiveness score. Therefore, at the beginning of the search, we can pick n “seed” vertices in G which have high degrees. We can use these seed vertices to prune other vertices.

We can extend Property 2.2 to k -RNN supportiveness. For a vertex u , let $d(u)$ be the number of co-authors of u , that is, the degree of u in the co-authorship network. It is easy to show the following.

PROPERTY 4.1. *For a vertex v in a graph $G = (V, E, \mathcal{L}, \mathcal{W})$, $\text{sup}_{k\text{RNN}}(v) \leq d(v)$.*

When mining the top- n k -RNN supportive authors, if we already have n vertices and their k -RNN supportiveness scores, let δ be the smallest k -RNN supportiveness score. Using Property 4.1, those vertices in $V_{\text{pruned}} = \{v \in V | d(v) < \delta\}$ cannot be ranked in the top- n list, and thus can be pruned.

The degrees of vertices in a large social network often follow the power law distribution which has a heavy tailed distribution [4]. Such degree distributions have been identified in various social networks including Internet, biological networks, and co-authorship networks. As a result, the number of vertices with large degrees is often very small, while the majority of vertices in the network only have very small degrees. According to property 4.1, if the smallest k -RNN supportiveness score δ is relatively large, many vertices in the network can be pruned.

Using this simple pruning rule, it is not necessary to compute the k -NN for each vertex. Moreover, for a vertex $v \in V$, to calculate the $k\text{RNN}(v)$, we only need to examine all the neighbors of v . In other words, by extracting a 2-neighborhood graph of vertex v (which is an induced subgraph containing v ’s 1-neighbors and v ’s 2-neighbors), we can calculate the k -RNN supportiveness score of v .

Algorithm 2 implements the above pruning techniques.

5 Mining Mutual Cliques

In a co-authorship network, a group of authors strongly supporting each other are interesting and significant since they should be inherently regarded as a collaborative community. In this section, we consider the problem of mining such communities. We first model the communities using mutual cliques. Then, we discuss how to mine significant mutual cliques.

Algorithm 2 Efficient top- n supportive author extraction

Input: same as Algorithm 1;

Output: same as Algorithm 1;

```

1: let  $Cand$  contain top- $n$  vertices with the largest degrees;
2: for each vertex  $v \in Cand$  do
3:   extract its 2-neighborhood graph, and calculate its  $k$ -RNN supportiveness score;
4: end for
5: let  $\delta$  be the smallest  $k$ -RNN supportiveness score for the vertices in  $Cand$ ;
6: let  $S = V - Cand$ ;
7: let  $V_{\text{prune}} = \{v \in S | d(v) < \delta\}$ ;
8:  $S = S - V_{\text{prune}}$ ;
9: while  $S \neq \emptyset$  do
10:  let  $u$  be the vertex in  $S$  with the largest degree;
11:  if  $\text{sup}_{k\text{RNN}}(u) \geq \delta$  then
12:    calculate  $\text{sup}_{k\text{RNN}}(u)$ ;
13:    update  $Cand$  and  $\delta$ ;
14:  else
15:    remove  $u$  from  $S$ ;
16:  end if
17: end while
18: return  $Cand$ ;

```

5.1 Mutual Cliques Intuitively, two vertices u and v support each other strongly if u is one of the k nearest neighbors of v , and vice versa. The smaller the value of k , the stronger their mutual support.

DEFINITION 6. (k -NN MUTUAL PAIR) *In a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, for two vertices $u, v \in V$, u and v form a k -NN mutual pair if $v \in k\text{NN}(u)$ and $u \in k\text{NN}(v)$. Moreover, the closeness between u and v is k , denoted by $\text{closeness}(u, v) = k$, if (u, v) is a k -NN mutual pair but not a $(k - 1)$ -NN mutual pair.*

EXAMPLE 3. (k -NN MUTUAL PAIRS) *In Table 2, (Ada, Deborah) form a 1NN-based mutual pair, and (Ada, Bob) form a 2NN-based mutual pair.*

There are 3 2-NN mutual pairs: (Ada, Bob), (Bob, Deborah), and (Ada, Deborah). $\text{closeness}(\text{Ada}, \text{Bob}) = \text{closeness}(\text{Bob}, \text{Deborah}) = 2$. $\text{closeness}(\text{Ada}, \text{Deborah}) = 1$.

Closeness measures the strength of the support between two co-authors to each other. The smaller the closeness value, the stronger the mutual support.

Can we extend the mutual pair relation to a group of authors, that is, a set of vertices in a co-authorship network which strongly support each other in the group?

DEFINITION 7. (*k*-NN MUTUAL CLIQUE) In a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, a set of vertices $S \subseteq V$ is a *k*-NN clique if for any two vertices $u, v \in S$, (u, v) is a *k*-NN mutual pair.

The **closeness** of S , denoted by $\text{closeness}(S)$, is

$$\text{closeness}(S) = \max_{u, v \in S} \{\text{closeness}(u, v)\}.$$

Apparently, a *k*-NN mutual pair S is a special case of a *k*-NN clique of size 2.

Given a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, we can derive a *mutual graph* $G_m = (V, E_m)$ where $(u, v) \in E_m$ if $\text{closeness}(u, v) \neq |V|$. Edge (u, v) is labeled by $\text{closeness}(u, v)$. Clearly, a *k*-NN clique S is the set of vertices of a complete subgraph in G_m such that every edge in the subgraph has a label at most k .

While supporting each other strongly is a desirable feature of communities in a co-authorship network, productivity in terms of the number of papers generated by a community is another important measure. We define the productivity measure formally as follows.

DEFINITION 8. (PRODUCTIVITY) For a vertex v in a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, the **productivity** of v is

$$\text{prod}(v) = \sum_{e \in E(v)} \frac{\mathcal{W}(e)}{\|e\|}.$$

The **productivity of a set of vertices** $S \subseteq V$ ($\|S\| \geq 2$) is

$$\text{prod}(S) = \sum_{e \in E, \|S \cap e\| \geq 2} \frac{\|e \cap S\|}{\|e\|} \cdot \mathcal{W}(e).$$

For a *k*-NN clique, the productivity $\text{prod}(S)$ is the weighted count of papers co-authored by at least 2 members in the clique. The reason we count only the papers co-authored by at least 2 members in the clique is that we want to measure the papers produced by collaboration in the clique. If there are $m \geq 2$ members participating in a paper of n -authors, the weight of this paper in $\text{prod}(S)$ is counted by $\frac{m}{n}$.

The productivity sums up the papers generated by collaborations in a group. However, it does not tell how the contributions are distributed on members in the group.

EXAMPLE 4. (BALANCED CLIQUE) Figure 3 shows a 4-NN clique of size 5 in a mutual graph. The edges are labeled by the number of papers co-authored by pairs of authors. The thickness of an edge represents the productivity contribution from the corresponding pair of authors.

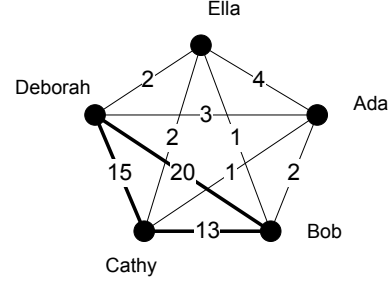


Figure 3: An example of unbalanced clique.

The productivity in this clique mainly comes from the collaboration among Bob, Cathy and Deborah. The sub-group $\{\text{Bob, Cathy, Deborah}\}$ do not contribute to the productivity at the same level in strength.

We define the balance factor to measure how evenly productivity is distributed in a clique.

DEFINITION 9. (BALANCE FACTOR) For a *k*-NN clique $S = \{v_1, \dots, v_n\}$ ($n \geq 2$), the **productivity vector**

$$\vec{S} = \underbrace{\langle \text{prod}(v_1, v_2), \text{prod}(v_1, v_3), \dots, \text{prod}(v_{n-1}, v_n) \rangle}_{\binom{n}{2}}.$$

The **balance factor** of S , denoted by $\text{bf}(S)$, is measured by $\cos \theta$, where θ is the angle between the perfectly balanced vector $\vec{1} = \langle 1, \dots, 1 \rangle$ and \vec{S} . Formally,

$$\text{bf}(S) = \frac{\sum_{u, v \in S} \text{prod}(u, v)}{\sqrt{\binom{\|S\|}{2} \sum_{u, v \in S} \text{prod}(u, v)^2}}$$

For any S , $0 < \text{bf}(S) \leq 1$. When $\|S\| = 2$, $\text{bf}(S) = 1$. The larger the balance factor, the more evenly the distribution of productivity contributions in the clique. When $\text{bf}(S) = 1$, every pair of co-authors in the clique contributes the same number of papers.

5.2 Mining Top- n Mutual Cliques When measuring the quality of cliques, both the productivity and the balance factor should be taken into account. Therefore, we use the *productivity and balance measure* (*PB* for short) $\text{PB}(S) = \text{prod}(S)\text{bf}(S)$ as the quality measure for cliques with at least 3 members.

Given a co-authorship network, how can we mine the top- n mutual cliques in *PB* measure? The problem of extracting *k*-NN cliques from a hypergraph is a general case of extracting cliques from undirected graphs. As discussed in Section 3, the problem of determining whether a graph contains a clique of a certain size is

NP-complete. So is the maximum clique problem which finds the largest clique in a graph.

In this subsection, we provide an exact algorithm to mine top- n k -NN cliques in a mutual graph. In the worst case, the algorithm takes exponential time. However, as shown in our experiments, the algorithm works well in practice. Particularly, we develop several pruning techniques to speed up the search.

The first pruning technique is to prune edges. The following result follows with the definition of k -NN cliques immediately.

PROPOSITION 5.1. (EDGE PRUNING) *In a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, if u and v are in a k -NN clique, then (u, v) is a k -NN mutual pair.*

Using the proposition, we can remove all edges (u, v) in the mutual graph such that (u, v) is not a k -NN mutual pair.

Computing the PB measure and the productivity of a group is not cheap. However, the PB measure and the productivity of a group can be bounded as follows.

THEOREM 5.1. (PRODUCTIVITY) *In a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, for any vertex v and any k -NN clique S containing v ,*

$$PB(S) \leq prod(S) \leq prod(v) + \sum_{u \in kNN(v) \cap kRNN(v)} prod(u)$$

Proof. For any members u, v in a k -NN clique S , (u, v) is a k -NN mutual pair (Definition 6). Therefore, the size of a clique containing v cannot be greater than $\|kNN(v) \cap kRNN(v)\| + 1$. Thus, we have $S \subseteq \{v\} \cup (kNN(v) \cap kRNN(v))$. Since $PB(S) = prod(S)bf(S)$ and $0 < bf(S) \leq 1$, we have the inequality in the theorem.

Using Theorem 5.1, we can prune vertices in a mutual graph. Suppose we already find n mutual cliques. For any vertex u , we calculate the upper bound of the PB measure using Theorem 5.1. If the bound is smaller than the PB measures of the top- n mutual cliques found so far, vertex u as well as the edges in the mutual graph using u as an end point can be pruned. This pruning can be applied iteratively to reduce the graph until no vertex or edge can be removed. If a vertex u is pruned, the upper bounds of the PB measures of all neighbors of u are also reduced.

Using Theorem 5.1, we can further have the following result.

PROPOSITION 5.2. *For vertices u, v, w , if $u, w \in kNN(v) \cap kRNN(v)$, but $u \notin kNN(w) \cap kRNN(w)$,*

Algorithm 3 Mining top- n k -NN cliques.

Input: A graph $G = (V, E, \mathcal{L}, \mathcal{W})$, parameter k and n ;

Output: Top- n k -NN cliques with largest PB measure;

Initialization:
// construct an adjacent list and only k -NN neighbors will be kept;
1: prepare an adjacent list and add all the vertices;
2: the threshold of top- n $\delta = 0$;
3: add the k -NN-based edges using Proposition 5.1;
4: remove these vertices which have no any neighbor;
Processing:
//traverse the vertices, when a vertex is visited, it will be removed
5: **while** there are some vertices left **do**
6: pick one vertex v as a seed vertex and calculate the upper bound of v with Theorem 5.1;
7: **if** $up(v) \geq \delta$ **then**
8: **while** there are unvisited combinations left **do**
9: get next next combination C ;
10: **if** the C is a clique **then**
11: calculate the $PB(v + C)$;
12: **if** $PB(v + C) \geq \delta$ **then**
13: update the top- n and δ
14: **end if**
15: **end if**
16: **end while**
17: **end if**
18: remove v ;
19: **end while**
20: return top- n ;

then for any clique S containing v

$$prod(S) \leq prod(v) + \sum_{u \in kNN(v) \cap kRNN(v)} prod(u) - \min\{prod(u), prod(w)\}$$

Using Proposition 5.2, for a vertex v , if $prod(v) + \sum_{u \in kNN(v) \cap kRNN(v)} prod(u)$ is larger than the smallest PB measure in the current top- n cliques, the top-2 most productive neighbors of v should be checked. If those two neighbors are not a mutual pair, the maximum possible productivity of v can be reduced.

The search algorithm is given in Algorithm 3. In the initialization step, the co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$ is organized as an adjacent list, where each entry represents a vertex in G , and only mutual neighbors are kept for each vertex. The vertices without any neighbors are deleted.

Algorithm 3 exploits some useful properties in co-authorship networks to speed up the searching process.

For any vertex v , if the neighbors of v are sorted in the productivity descending order, all subsets of the

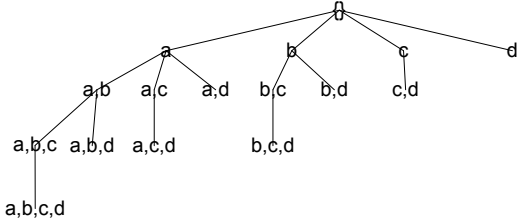


Figure 4: The neighbor enumeration tree of v , $prod(a) \geq prod(b) \geq prod(c) \geq prod(d)$.

neighbors of v can be enumerated systematically using a set enumeration tree [22], where each node represents a subset of neighbors. Figure 4 shows an example. Each subset combining with v can be regarded as a candidate set. The following result can be obtained easily using the related definitions and the property of the set enumeration tree.

LEMMA 5.1. *In the set enumeration tree of the neighbors of vertex v , for any node S in the tree, $prod(S) \geq prod(S')$ where S' is the parent node of S . Moreover, if S_1 is a left sibling of S_2 in the set enumeration tree, then $prod(S_1) \geq prod(S_2)$.*

For each vertex u , we construct a set enumeration tree of the neighbors of u to explore the possible cliques containing u . We conduct a depth-first search of the tree. Using Lemma 5.1, if a node in the tree fails the productivity requirement, then the node as well as its ancestors and the right siblings can be pruned immediately.

5.3 A Skyline of Mutual Cliques In the k -NN clique analysis, the cliques with high PB measure and large size can be regarded as the “successful” groups. However, the PB measure and the clique size are often two conflicting goals. When the clique size becomes larger, the average productivity and the balance factor often decrease. Thus, we want to find the cliques that present the tradeoffs among the three factors: size, PB measure, and closeness.

DEFINITION 10. (DOMINATION AND SKYLINE) *For two cliques S_1 and S_2 in a mutual graph, S_1 dominates S_2 , denoted by $S_1 \prec S_2$, if $\|S_1\| \geq \|S_2\|$, $PB(S_1) \geq PB(S_2)$, $closeness(S_1) \leq closeness(S_2)$, and not all the three equalities hold.*

*In a co-authorship network $G = (V, E, \mathcal{L}, \mathcal{W})$, a clique S in the mutual graph is a **skyline clique** if there exists no other clique S' in the mutual graph such that $S' \prec S$.*

To discover all skyline cliques, we first slice the search space (that is, all possible cliques in the mutual graph) on the dimension of size. The reason we slice on this dimension is that once the skyline cliques of size m are found, all vertices having degree $m - 1$ or less can be pruned.

For each slice, that is, the possible cliques of size m , we calculate the skyline on closeness factor and PB measure. The algorithm discussed in Section 5.2 can be applied with a minor revision: instead of maintaining the top- n cliques, we maintain the set of cliques which are not dominated by any others. Limited by space, we omit the details here.

6 Experimental Results

In this section, we report a systematic experimental study on a large real data set. All programs were implemented in C++ using Microsoft Visual Studio 2008.Net. We conducted the experiments on a laptop computer with a Pentium 4 3.0GB CPU and 2GB main memory, running on a 32-bit Windows XP system.

We use two data sets in our experiments. We use a data set containing 10,307 authors and 10,372 papers published in 9 database conferences (SIGMOD, VLDB, PODS, ICDE, ICDT, DOOD, EDBT, SSD, and CIKM) from January 2000 to August 2008. This data set is extracted from DBLP Computer Science Bibliography Server (<http://www.informatik.uni-trier.de/~ley/db/index.html>). We call this data set the **DB data set**. We use this data set to analyze some interesting mining results using our methods.

To test the efficiency of our methods, we use a snapshot of the DBLP data set in August 2008, which contains more than 1 million publications and more than 600 thousand authors. We call the data set the **DBLP data set**.

6.1 Ranking Authors Table 3 lists the top-10 supportive authors in the DB data set with different k values. The supportiveness values are also given.

One observation is that, as the value of k increases, the order of the top-10 authors tends to be stable. When k is very small, only the closest collaborators are considered in the computation of the supportiveness. When k is not too small, the major collaborators are counted. The most impactful authors can be captured.

Section 4.2 indicates that the supportiveness score of an author is relevant to the number of his/her co-authors. In Figure 5, for the top-250 authors in supportiveness, we plot the ranks of each author in terms of the supportiveness and the number of co-authors. When k is small, the correlation is weak. However, when k increases, the correlation is strengthened dramatically.

$k = 1$	$k = 3$	$k = 5$	$k = 7$	$k = 9$
Amr El Abbadi: 11.0	Amr El Abbadi: 30.0	Jiawei Han: 54.18	Jiawei Han: 63.65	Jiawei Han: 75.51
Christos Faloutsos: 10.0	Christos Faloutsos: 29.0	Christos Faloutsos: 51.0	Christos Faloutsos: 62.0	Christos Faloutsos: 69.0
Z. Meral Özsoyoglu: 9.0	Jiawei Han: 26.0	Beng Chin Ooi: 41.83	Elke A. Rundensteiner: 49.58	Elke A. Rundensteiner: 61.75
Gultekin Özsoyoglu: 9.0	Kian-Lee Tan: 25.0	Amr El Abbadi: 38.0	Beng Chin Ooi: 47.77	Michael Stonebraker: 57.0
Jayant R. Haritsa: 8.0	Divyakant Agrawal: 24.0	Jian Pei: 36.03	Michael Stonebraker: 44.0	Kian-Lee Tan: 54.64
Raymond T. Ng: 7.0	Beng Chin Ooi: 23.0	Elke A. Rundensteiner: 34.83	Kian-Lee Tan: 43.77	Beng Chin Ooi: 51.64
Shunsuke Uemura: 7.0	Philip S. Yu: 22.0	Kian-Lee Tan: 33.0	Donald Kossmann: 42.65	Philip S. Yu: 51.49
Roger King: 6.5	Hector Garcia-Molina: 22.0	Hector Garcia-Molina: 33.0	Divyakant Agrawal: 41.7	Hector Garcia-Molina: 49.82
Arie Segev: 6.0	Elisa Bertino: 22.0	Divyakant Agrawal: 33.0	Jian Pei: 41.44	Michael J. Carey: 48.82
Guido Moerkotte: 6.0	Anthony K. H. Tung: 20.0	Philip S. Yu: 32.62	Philip S. Yu: 40.11	Clement T. Yu: 48.00
Peter Triantafyllou: 6.0	Elke A. Rundensteiner: 20.0			
Ada Wai-Chee Fu: 6.0	Jayant R. Haritsa: 20.0			
Sharma Chakravorthy: 6.0				
Vram Kouramajian: 6.0				

Table 3: An example of top-10 “supportive” authors in DB dataset, the numbers after the names represent the supportiveness score.

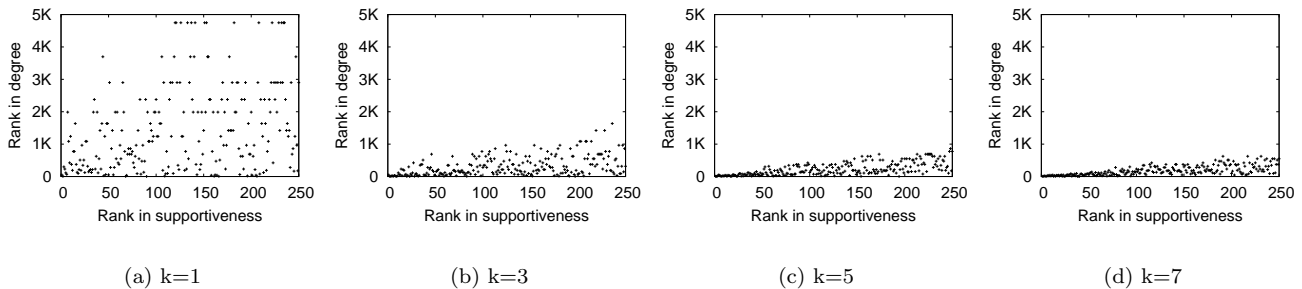


Figure 5: The relationship between the supportiveness and the number of co-authors.

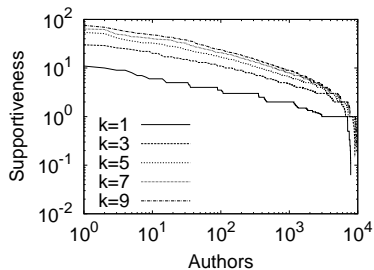


Figure 6: The supportiveness distribution in the DB data set.

When $k = 5$, all of the top-250 most supportive authors are among the top-1000 authors in the number of co-authors.

In Figure 6, we show the distribution of the supportiveness measure on the authors in the DB data set with respect to different values of k . Clearly, the supportiveness measure follows a long tail distribution.

6.2 Ranking Cliques Figure 7 shows the size of the top-100 cliques in PB score. The sizes of the top cliques are small, most of the time in the range between 3 and 5. When the value of k increases, the size of some cliques also increases.

Figure 8 shows the productivity of the top-100

cliques in PB measure. When k is small (e.g., $k = 5$), the cliques are small. Thus, the PB measure is dominated by the productivity. The productivity decreases as the ranks of the cliques increase. When k increases, the balance factor starts to play a role. Thus, the sorted list of top-100 cliques is not in the productivity descending order anymore.

Figure 9 shows the corresponding balance factor distribution. When k is small, the clique sizes are very small and thus the cliques are well balanced. When k increases, some larger cliques may be ranked high due to their high productivity. The balance factor in those cliques may not be high. Thus, the range of balance factors becomes larger.

6.3 Efficiency We run our algorithms on different subsets of the DBLP data set to test the scalability of our methods. The different subsets have various numbers of authors. To preserve the connectivity among authors, we do not sample the authors directly. Instead, we randomly pick papers from the DBLP data set until the required number of authors are obtained. The results are shown in Figure 10.

Our methods have linear scalability in mining top- n supportive authors and cliques, and thus can be applied on large co-authorship networks. Our method is insensitive to k when mining top- n supportive authors.

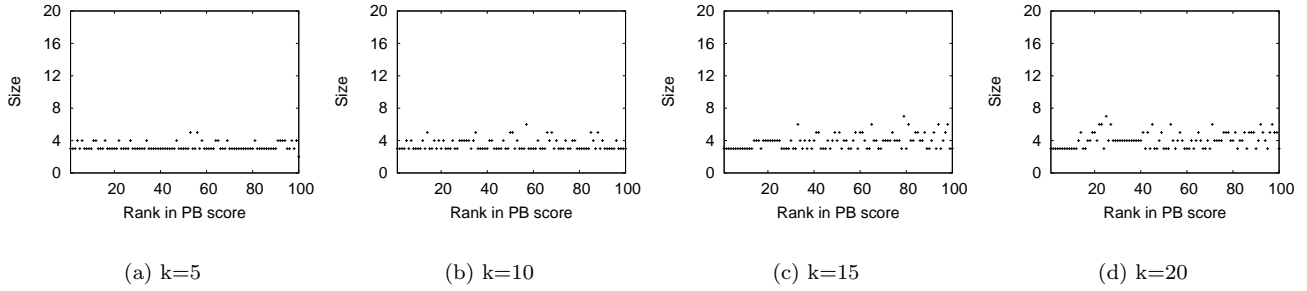


Figure 7: The size of the top-100 cliques in PB measure.

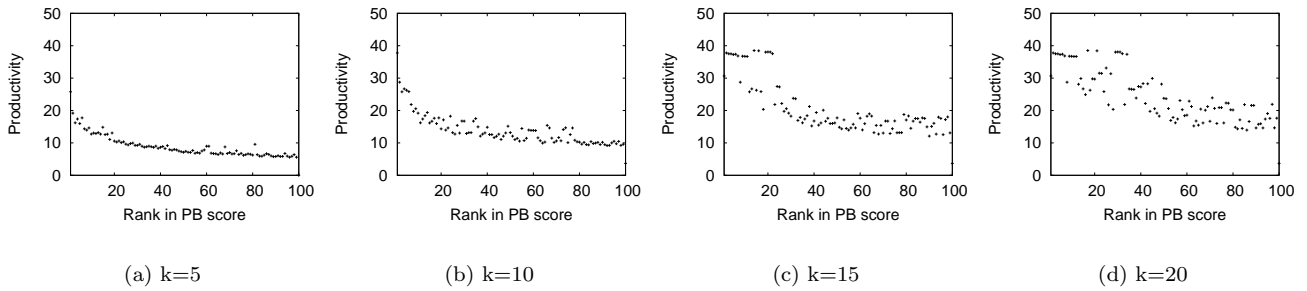


Figure 8: The productivity of top 100 cliques with highest PB measure.

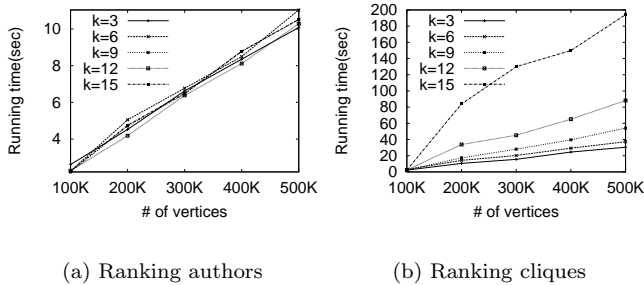


Figure 10: Scalability: mining top-100 authors/cliques.

When mining top- n cliques, k plays an important role. The larger the value of k , the more neighbors we need to search, and thus the longer the runtime.

7 Conclusions

In this paper, we proposed a novel and interesting supportiveness measure on co-authorship networks. We modeled the collaboration between two authors as a supporting activity between them. We discussed contribution-based distance and a k -RNN supportiveness measure. An efficient algorithm was developed

to mine top- n most supportive authors in a network. Moreover, we extended the supportiveness analysis from a pair of vertices to a group of vertices, and proposed the notion of k -NN cliques. The experimental results conducted on a large real co-authorship network indicates that our supportiveness measures are meaningful, and our methods are efficient in practice.

There are some interesting future directions. For example, we only considered the clique structure in this paper. In general social networks, some other structures like stars may also be meaningful patterns. Moreover, it is interesting to analyze the supportiveness contribution in a PageRank-like style.

References

- [1] R. Albert *et al.* The diameter of the world wide web. *Nature*, pages 130–131, 1999.
- [2] A.-L. Barabasi. The origin of bursts and heavy tails in human dynamics. *Nature*, 435:207, 2005.
- [3] P. Chen *et al.* Finding scientific gems with google’s pagerank algorithm. *Journal of Informetrics*, 1:8–15, 2007.
- [4] M. Faloutsos *et al.* On power-law relationships of the internet topology. In *SIGCOMM’99*.

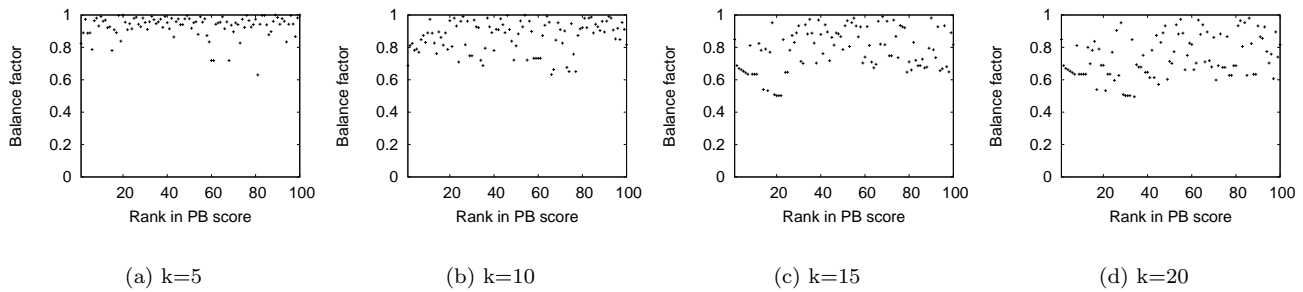


Figure 9: The balance factor of the top-100 cliques in PB measure.

- [5] L. Getoor and C. P. Diehl. Link mining: a survey. *ACM SIGKDD Explorations Newsletter*, 7:3–12, 2005.
- [6] J. Huang *et al.* Collaboration over time: characterizing and modeling network evolution. In *WSDM'08*.
- [7] A. Inokuchi *et al.* An apriori-based algorithm for mining frequent substructures from graph data. In *PKDD'00*.
- [8] R. M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, 1972.
- [9] D. Kempe *et al.* Maximizing the spread of influence through a social network. In *KDD'03*.
- [10] J. Kleinberg. Small-world phenomena and the dynamics of information. In *NIPS'01*.
- [11] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In *SODA'98*.
- [12] F. Korn and S. Muthukrishnan. Influence sets based on reverse nearest neighbor queries. In *SIGMOD'00*.
- [13] F. Korn *et al.* Reverse nearest neighbor aggregates over data streams. In *VLDB'02*.
- [14] H. Kretschmer. Author productivity and geodesic distance in bibliographic co-authorship networks, and visibility on the web. *Scientometrics*, 60(3):409–420, 2004.
- [15] J. Leskovec *et al.* Graphs over time: densification laws, shrinking diameters and possible explanations. In *KDD'05*.
- [16] M. A. Nascimben *et al.* Analysis of sigmod's co-authorship graph. *SIGMOD Record*, 32(3):8–10, 2003.
- [17] M. E. J. Newman. Who is the best connected scientist? a study of scientific coauthorship networks. *Physical Review E*, 64:016132, 2001.
- [18] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 69:26113, 2004.
- [19] B.-W. On *et al.* Improving grouped-entity resolution using quasi-cliques. In *ICDM'06*.
- [20] L. Page *et al.* The pagerank citation ranking: Bringing order to the web. Technical report, Stanford University, 1998.
- [21] J. Pei *et al.* On mining cross-graph quasi-cliques. In *KDD'05*.
- [22] R. Ryman. Search through systematic set enumeration. pages 539–550, 1992.
- [23] A. F. Smeaton *et al.* Analysis of papers from twenty-five years of sigir conferences: what have we been doing for the last quarter of a century? *SIGIR Forum*, 37(1):49–53, 2003.
- [24] H. Tong *et al.* Proximity tracking on time-evolving bipartite graphs. In *SDM'08*.
- [25] J. R. Tyler *et al.* Email as spectroscopy: Automated discovery of community structure within organizations. *Communities and Technologies: Proceedings of the First International Conference on Communities and Technologies, C&T 2003*, 2003.
- [26] J. Wang *et al.* Clan: An algorithm for mining closed cliques from large dense graph databases. In *ICDE'06*.
- [27] T. Washio and H. Motoda. State of the art of graph-based data mining. *ACM SIGKDD Explorations Newsletter*, 5:59–68, 2003.
- [28] S. M. Weiss *et al.* Knowledge-based data mining. In *KDD'03*.
- [29] F. Wu and B. A. Huberman. Finding communities in linear time: a physics approach. *The European Physical Journal B-Condensed Matter*, 38:331–338, 2004.
- [30] W. Wu *et al.* Finch: Evaluating reverse k-nearest-neighbor queries on location data. In *VLDB'08*.
- [31] X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *ICDM'02*.
- [32] C. Yang and K. I. Lin. An index structure for efficient reverse nearest neighbor queries. *ICDE'01*.
- [33] M. L. Yiu *et al.* Reverse nearest neighbors in large graphs. *IEEE Trans. Knowl. Data Eng.*, 18(4):540–553, 2006.
- [34] T. Yukawa *et al.* An expert recommendation system using concept-based relevance discernment. In *IC-TAI'01*.
- [35] B. Zhou and J. Pei. Sketching landscapes of page farms. In *SDM'07*.
- [36] D. Zhou *et al.* Co-ranking authors and documents in a heterogeneous network. In *ICDM'07*.