# Flow-based Influence Graph Visual Summarization

Lei Shi
*SKLCS, Institute of Software*
*Chinese Academy of Sciences*
*Beijing, China*
*shil@ios.ac.cn*

Hanghang Tong
*Department of Computer Science*
*Arizona State University*
*Phoenix, USA*
*htong6@asu.edu*

Jie Tang and Chuang Lin
*Department of Computer Science and Technology*
*Tsinghua University*
*Beijing, China*
*{jietang, chlin}@tsinghua.edu.cn*

***Abstract*—Visually mining a large influence graph is appealing yet challenging. Existing summarization methods enhance the visualization with blocked views, but have adverse effect on the latent influence structure. How can we visually summarize a large graph to maximize influence flows? In particular, how can we illustrate the impact of an individual node through the summarization? Can we maintain the appealing graph metaphor while preserving both the overall influence pattern and fine readability?**

**To answer these questions, we first formally define the influence graph summarization problem. Second, we propose an end-to-end framework to solve the new problem. Last, we report our experiment results. Evidences demonstrate that our framework can effectively approximate the proposed influence graph summarization objective while outperforming previous methods in a typical scenario of visually mining academic citation networks.**

***Keywords*-influence graph; influence flow; visualization;**

## I. Introduction

Graphs are prevalent and have become a prevalent platform for the masses to interact and disseminate a variety of information (e.g., influence, memes, opinions, rumors, etc.). *How to make sense of an individual's influence in the context of such graphs?* This, which is referred as Influence Graph Summarization (IGS) problem, is the central problem we aim to address in this paper. For example, how does a highly-cited paper impact the research community to raise several topic threads; and consequentially, how do these topics interact with each other and lead to a new multi-disciplinary research direction?

Although closely related, IGS problem bears some subtle difference from the existing work. First (*influence maximization*), many elegant algorithms have been proposed for the so-called influence maximization problem [1]. While effective in identifying *who* are most influential in the graph, the question of *what makes them influential* largely remains open. Second (*graph summarization*), many interesting work has been done in the context of graph clustering and compression. These works typically look for homogeneous regions in graphs by optimizing a pre-defined loss function (e.g., minimizing the inter-cluster connection, maximizing the intra-cluster density, etc). Despite their own success, most, if not all, of the existing work on graph summarization tends to ignore the specific characteristics of influence graphs and how the end user would visually perceive and consume the summarization results.

To be specific, we outline two design objectives that differentiate our IGS problem from existing works.

- *D1. Flow Rate Maximization*. Quite different from extracting dense clusters on graph, the goal of IGS is to highlight the flow of influence not only within but also across clusters. By maximizing the overall flow rate, IGS-based summarization outlines the strongest interaction among groups of nodes on a graph. For example, Figure 1 depicts the influence of the famous power-law paper presented at SIGCOMM'99. The evolution of research topics is revealed, rather than the hot topics themselves.
- *D2. Localized Visualization*. While a large graph can span millions of nodes and prohibit any readable visual summarization, in IGS objective, we switch to summarize the influence of a single node on the graph (called the source node). This localized visualization problem is at least as important as the overall summarization problem. Consider a user navigating the citation graph of computer science papers, after an overview of the entire field, likely she will drill down to a few interested papers and examine their influence separately.

In this paper, we propose a unified framework to generate *flow-based*, *localized visual* summarization over large-scale influence graphs. The main contributions of the paper can be summarized as:

- *Problem Definition*, to fulfill the design objectives listed above for flow-based visual summarization of large influence graphs (Section II);
- *A Unified Framework and Implementation Details*, to solve the IGS problem (Section III and Section IV);
- *Performance Evaluations*, to demonstrate the effectiveness of the proposed framework (Section V).

## II. Problem Definition

Table I lists the notations used throughout the paper. The raw inputs are the influence graph $I$ and the source node $f$ either selected by the user or detected by any

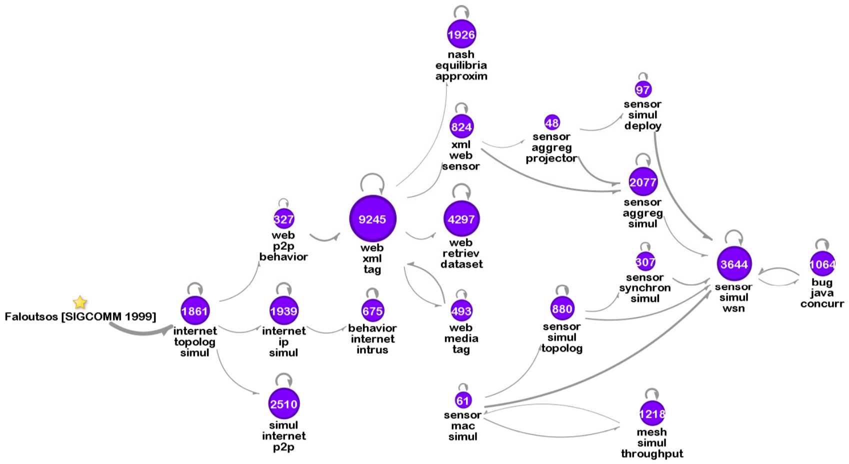


Figure 1. Influence graph summarization on [Faloutsos SIGCOMM'1999] (#Cluster = 20). Node label gives the cluster size and summary on paper title+abstract normalized by keyword frequency. Link thickness indicates the normalized flow rate.

Table I
NOTATIONS.

| SYMBOL | DESCRIPTION |
|---|---|
| $I$ | influence graph as input |
| $f$ | source node selected by user or algorithm |
| $G$ | maximal influence graph of $f$ in $I$ |
| $v_i$, $N(i)$, $n$ | nodes, neighbor set and # of nodes in $G$ |
| $A$, $a_{ij}$ | adjacency matrix of $G$ and its entries |
| $M^G$ | similarity matrix of $G$ |
| $S$ | graph summarization of $G$ |
| $\pi_c$, $\|\pi_c\|$, $k$ | clusters, cluster size and # of clusters in $S$ |
| $\xi_s$, $r(\xi_s)$, $l$ | flows, flow rate and # of flows in $S$ |
| $\pi_{c(s)}$, $\pi_{d(s)}$ | the source and target cluster of flow $\xi_s$ |

existing influence maximization algorithm. Without loss of generality, it is enough to consider a maximal influence graph $G$ of $f$ which is an induced subgraph of $I$ containing all the nodes reachable from $f$ in $I$ (including $f$). Though it is easy to extend the definition to a maximal origin graph by reversing all the links in $I$ or use the union of the two definitions, for relevancy to the IGS problem we stick to the maximal influence graph definition in this paper. Let $G$ have $n$ nodes, denoted as $\{v_i\}_{i=1}^n$. $G$ is represented by the adjacency matrix $A = \{a_{ij}\}_{i,j=1}^n$ in which $a_{ij}$ denotes the link weight. $a_{ij} > 0$ if there is a link from $v_i$ to $v_j$.

*Definition 1:* The **graph summarization** of $G$, denoted as $S$, is a super node-link graph of $G$. The node set of $S$ contains $k$ disjoint and exhaustive node clusters of $G$, denoted as $\{\pi_c\}_{c=1}^k$ where $|\pi_c|$ indicates the number of nodes in cluster $\pi_c$. The link set of $S$ contains $l$ flows between the nodes in $S$ (i.e., clusters in $G$), denoted as $\{\xi_s\}_{s=1}^l$. Each flow $\xi_s$ represents the collection of all the links in $G$ from nodes in cluster $\pi_{c(s)}$ to nodes in cluster $\pi_{d(s)}$. The flow rate of $\xi_s$ is defined by

$$r(\xi_s) = \frac{\sum_{v_i \in \pi_{c(s)}, v_j \in \pi_{d(s)}} a_{ij}}{|\pi_{c(s)}||\pi_{d(s)}|}$$

Note that $S$ can be a partial summarization of $G$, with fewer flows ($l < k^2$) than a full summarization ($l = k^2$). This is desirable for influence graph visualization where huge number of flows and edge crossings can cause unpleasant visual clutter.

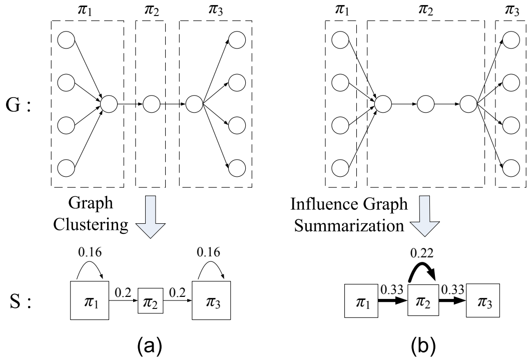


Figure 2. Difference between IGS problem and traditional graph clustering problem. Each dash box in the original graph $G$ becomes a square node in the graph summarization $S$. (a) Traditional graph clustering leading to higher intra-cluster flow rate; (b) Influence graph summarization exposing denser overall flows. In $S$, the flow rate is labeled above each link and is mapped to the link thickness visually. We assume a uniform link weight of 1 in the original graph $G$.

*Problem 1:* The **general IGS problem** is defined as finding a graph summarization $S$ with $k$ clusters and $l$ top flows of the maximal influence graph $G$ to optimize the objective function:

$$\max \quad \sum_{s=1}^{l} r(\xi_s)\sqrt{|\pi_{c(s)}||\pi_{d(s)}|} \qquad (1)$$

The general IGS problem defined in (1), although seemingly similar to, is different from the traditional graph clustering problems. Let us explain their difference using the classic ratio association graph clustering problem, whose objective function is shown below.

$$\max \quad \sum_{c=1}^{k} \sum_{i,j \in \pi_c} \frac{a_{ij}}{|\pi_c|} = \sum_{c=1}^{k} r(\xi_c)|\pi_c|$$

where $\xi_c$ denotes the intra-cluster flow from $\pi_c$ to itself.

The IGS objective function is designed to maximize the sum of $l$ selected flows between or within clusters, corresponding to $l$ arbitrary blocks in the adjacency matrix. On the other hand, the ratio association objective maximizes the sum of intra-cluster flows at all the $k$ diagonal matrix blocks. In other words, IGS finds dense flows through summarization which fits well the goal to highlight flows of influence across the graph. This is quite different from the traditional graph clustering objective that finds dense node clusters. An example is given in Figure 2 for visual comparison.

## III. Framework

### A. End-to-End Pipeline

We propose a unified framework to solve the IGS problem. The framework features an end-to-end pipeline, as shown in Figure 3, which decomposes the IGS problem into several building blocks. Initially, the maximal influence graph $G$ is computed from the input graph $I$ by a breadth-first or depth-first search starting from the source node $f$. Over the maximal influence graph $G$, three processing components work in parallel to generate three matrices on the graph: the topology similarity matrix, and the optional attribute and time matrices. The core of our framework is the decomposition of the topology similarity matrix to generate $k$ node clusters for the summarization. We carefully design the topology similarity matrix to ensure that the graph summarization approximates the flow rate maximization objective. The requirement of the $l$ flows in the summarization is handled by link pruning using either ranking-based filtering or the maximum spanning tree algorithm. The proposed pipeline is flexible and admits many existing graph mining algorithms for each of its building blocks. On the other hand, by itself, none of these existing algorithms is sufficient to solve the IGS problem.

### B. Node Summarization

Node summarization is the key building block of our proposed pipeline. First we compute the topology similarity matrix by the common neighbor heuristic:

$$M^G = \frac{AA^T + A^T A}{2} \tag{2}$$

where $A$ is the adjacency matrix of the maximal influence graph $G$.

We then propose a matrix decomposition based solution to generate $k$ node clusters from the similarity matrix $M^G$. The decomposition employs a Symmetric version of the Nonnegative Matrix Factorization (SymNMF [2]) which optimizes:

$$\min_{H \geq 0} ||M^G - HH^T||_F^2 \tag{3}$$

where $||\cdot||_F$ denotes the Frobenius norm of the matrix. $H = \{h_{ij}\}$ is a $n$ by $k$ matrix indicating the cluster membership assignment of nodes in $G$: $v_i$ will be clustered into $\pi_c$ if $h_{ic}$ is the largest entry in the $i$th row of $H$.

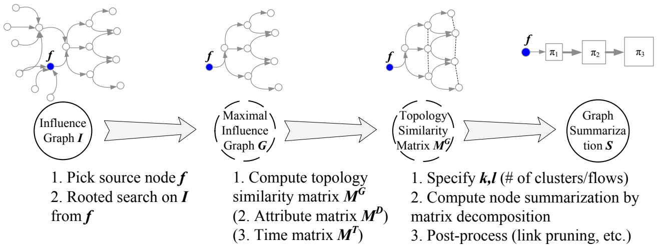

f
f
f
f
$\pi_1$
$\pi_2$
$\pi_3$
Influence Graph I
Maximal Influence Graph G
Topology Similarity Matrix $M^G$
Graph Summarization S
1. Pick source node f
2. Rooted search on I from f
1. Compute topology similarity matrix $M^G$
(2. Attribute matrix $M^D$)
(3. Time matrix $M^T$)
1. Specify k,l (# of clusters/flows)
2. Compute node summarization by matrix decomposition
3. Post-process (link pruning, etc.)

Figure 3. The framework to solve the IGS problem.

## IV. Implementation Details

In this section, we provide some additional implementation details. As shown in Figure 3, our framework involves four kinds of algorithm-driven building blocks. The rooted graph search follows the standard BFS/DFS implementation. Below we describe details for similarity matrix computation, node summarization and the link pruning for post-processing of the summarization.

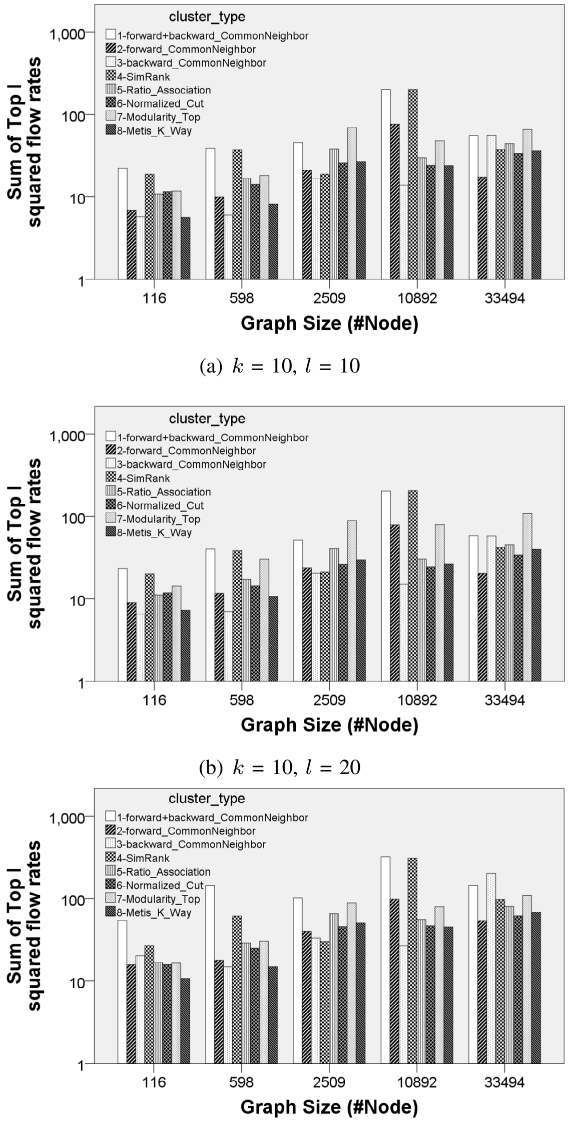
**Similarity Matrix Computation**. In Section III, we propose to use the heuristic of common neighbors to construct the similarity matrix (CommonNeighbor) to approximate the objective function of the IGS problem. This algorithm runs fast even for very large graphs due to a complexity of $O(md^2)$ where $m$ is the number of links in $G$ and $d$ is the average node degree. We have implemented three versions of the algorithm and it is shown that bidirectional CommonNeighbor is generally better than one-directional forward or backward CommonNeighbor in Section V.

**Node Summarization with SymNMF**. The node summarization is done by applying SymNMF on similarity matrix $M^G$, and using the factorized matrix $H$ for cluster membership assignment. In our implementation, we apply the iterative SymNMF solver with the multiplicative updating rule in [2] which guarantees convergence. In this iterative algorithm, the initialization of $H$ is critical to the final result. We introduce nonnegative eigenvalue decomposition similar to the method in [3] to compute a good initial factorization.

**Link Pruning.** The graph summarization by SymNMF needs further post-processing to select $l$ top flows for the final summarization $S$. Here we extract the top flows according to the rank of the normalized flow rate. The other flows are then filtered out. Notice that to recover critical links, we introduce a constraint to keep a connected graph in the summarization. It is achieved by adding back the most dense flow going to each node cluster. An alternative choice is to use the maximum spanning tree (MST) algorithm [4].

## V. Evaluation

In this section, we evaluate the proposed IGS framework and the CommonNeighbor algorithms by comparing with alternative graph summarization methods. Nine approaches



(a) $k$ = 10, $l$ = 10

(b) $k$ = 10, $l$ = 20

(c) $k$ = 20, $l$ = 20

Figure 4. The performance in maximizing the IGS objective on five sample graphs. The flow rate is summed from the top $l$ flows between $k$ clusters.

are considered: three using *CommonNeighbor* algorithms to compute the similarity matrix for SymNMF (i.e. forward+backward, forward, and backward settings), one using *SimRank* algorithm [5] to compute the similarity matrix for SymNMF, the classical graph clustering algorithm with *Ratio Association* and *Normalized Cut* objectives [6], agglomerative *Modularity*-based graph clustering [7], *Metis* K-way graph partition [8] and the Minimal Description Length (*MDL*) based graph summarization [9]. Note that modularity clustering is executed agglomeratively until all clusters stop merging at the top level or the number of clusters reaches $k$. The MDL algorithm can not specify the number of clusters, in fact, it generates 4,937 clusters on one medium-sized influence graph. To ensure fair comparison (a larger number of clusters will lead to a much higher overall flow rate), we exclude MDL from numeric comparisons, but present its visual summarization results. The experiment data are paper citation graphs collected from ArnetMiner [10]. The influence graphs are obtained by reversing citation links.

### *A. Flow Rate Maximization*

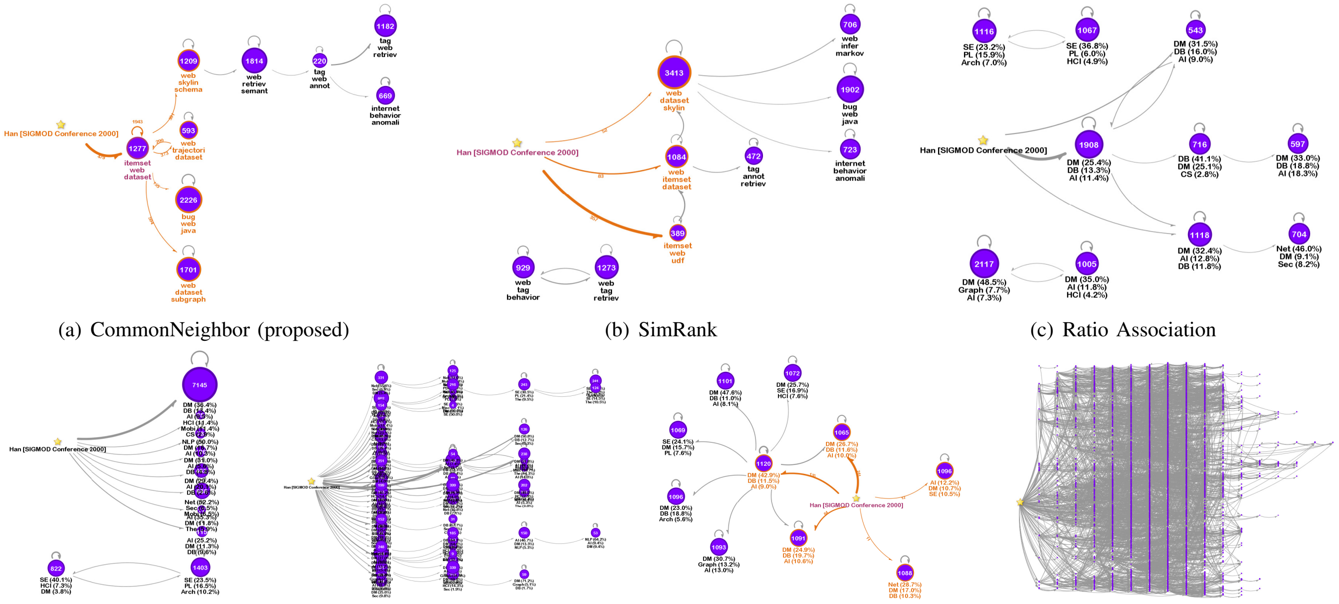
We first pick five source papers from the data set to generate maximal influence graphs. These influence graphs are summarized into $k$ clusters, between which the top $l$ flow rates are summed according to the IGS objective. Figure 4(a)$\sim$(c) present the comparisons among eight summarization methods on the numeric objective function.

The initial result in Figure 4(a) with a minimal graph summarization ($k = 10, l = 10$) suggests that among three CommonNeighbor algorithms, the bidirectional setting almost always achieves the best performance in maximizing the IGS objective (at least $> 100\%$ gain), except on the largest graph (#Node=33,494), the backward CommonNeighbor obtains a tiny advantage (1%). Further, comparing the bidirectional CommonNeighbor to traditional graph summarization methods, CommonNeighbor achieves much better performance than Ratio Association, Normalized Cut and Metis (at least $> 20\%$, in average $> 100\%$). In some cases, the performance of CommonNeighbor is matched by SimRank ($< 10\%$ gain) or outperformed by Modularity. This is because the Modularity algorithm generates more clusters than the initial setting of $k = 10$. For example, the sample graph with 33,494 nodes stops at 71 clusters in the top modularity level.

When we double the number of flows ($k = 10, l = 20$) in Figure 4(b), the sum of flow rates does not increase much on all algorithms (in average $< 15\%$) and the overall comparative patterns stay unchanged. This shows that the top $k$ flows already capture most of the flow rates on the graph summarization. We then increase the number of clusters ($k = 20, l = 20$). The results in Figure 4(c) reveal that the objection function increases much as the number of clusters increases (at least $> 30\%$, in average $> 90\%$, comparing Figure 4(c) with Figure 4(b)), except for Modularity, which remains unchanged because their number of clusters are already larger than $k$ and kept stable. On the comparative pattern, bidirectional CommonNeighbor regains performance advantage over SimRank and Modularity under a large number of clusters.

### *B. Visualization*

We evaluate the effectiveness of summarization methods also by comparing their visualization results: whether they produce a clean influence graph summarization with little visual clutter and whether the results are meaningful for users with domain knowledge. We first pick the famous frequent pattern mining paper by Prof. Jiawei Han et al. as the source to generate the maximal influence graph. Then we execute seven typical summarization methods and depict


Han [SIGMOD Conference 2000]
1209 web skylin schema
1814 web retriev semant
220 tag web annot
1182 tag web retriev
669 internet behavior anomali
593 web trajectori dataset
1277 itemset web dataset
2226 bug web java
1701 web dataset subgraph
Han [SIGMOD Conference 2000]
3413 web dataset skylin
706 web infer markov
1902 bug web java
723 internet behavior anomali
1084 web itemset dataset
472 tag annot retriev
389 itemset web udf
929 web tag behavior
1273 web tag retriev
1116 SE (23.2%) PL (15.9%) Arch (7.0%)
1067 SE (36.8%) PL (6.0%) HCI (4.9%)
543 DM (31.5%) DB (16.0%) AI (9.0%)
Han [SIGMOD Conference 2000]
1908 DM (25.4%) DB (13.3%) AI (11.4%)
716 DB (41.1%) DM (25.1%) CS (2.8%)
597 DM (33.0%) DB (18.8%) AI (18.3%)
1118 DM (32.4%) AI (12.8%) DB (11.8%)
704 Net (46.0%) DM (9.1%) Sec (8.2%)
2117 DM (48.5%) Graph (7.7%) AI (7.3%)
1005 DM (35.0%) AI (11.8%) HCI (4.2%)
7145
Han [SIGMOD Conference 2000]
422 SE (40.1%) HCI (7.3%) DM (3.8%)
1403 SE (23.5%) PL (16.5%) Arch (10.2%)
1101 DM (47.6%) DB (11.0%) AI (8.1%)
1072 DM (25.7%) SE (16.9%) HCI (7.6%)
1069 SE (24.1%) DM (15.7%) PL (7.6%)
1120 DM (42.9%) DB (11.3%) AI (9.0%)
1065 DM (26.7%) DB (11.6%) AI (10.7%)
1096 AI (12.2%) DM (10.7%) SE (10.5%)
Han [SIGMOD Conference 2000]
1096 DM (23.0%) DB (18.8%) Arch (5.6%)
1093 DM (30.7%) Graph (13.2%) AI (13.0%)
1091 DM (24.9%) DB (19.7%) AI (10.6%)
1088 Net (28.7%) DM (17.8%) DB (10.3%)

(a) CommonNeighbor (proposed)

(b) SimRank

(c) Ratio Association

(d) Normalized Cut

(e) Modularity

(f) Metis K-way

(g) MDL

Figure 5. Influence graph summarization results on [Han SIGMOD'2000] by different methods ($k$ = 10, $l$ = 20). Node label gives the number of papers in each cluster and their content summary by either title+abstract keywords in (a),(b) or the top 3 research fields in (c)∼(f). Link thickness indicates the normalized flow rate. Some part of the graph is highlighted to show the number of citations as edge labels. Note that the modularity algorithm stops at 62 clusters and can not merge any further. MDL produces 4,937 clusters, leaving a half of the visual complexity from the input graph.

their results in Figure 5(a)∼(g). At the first glance, the proposed bidirectional CommonNeighor method generates a connected tree-like influence graph summarization without edge crossing (Figure 5(a)). Compared to that, SimRank gets a similar visual form (Figure 5(b)) due to the comparable objective function result, but the generated graph is not connected. The Metis result is also clean (Figure 5(f)), but all the clusters have a similar number of nodes, making the summarized graph impractical for usage. Ratio Association and Normalized Cut look inferior due to the poor graph connectivity (Figure 5(c)) and the flat influence hierarchy (Figure 5(d)). Modularity and MDL are the worst because of the visual clutter generated from the large number of clusters remained in the summarization (Figure 5(e)(g)).

Taking a closer look at the visual summarizations, we find that by CommonNeighbor, most flows represent at least 300 citation links. While by SimRank, the critical flows linking the source node are fragmented, two of which only include 52 and 83 citations. The same deficiency is found in the result by Metis, where two highlighted flows only have 11 and 12 citations. We also invite a senior researcher from the database and data mining community to evaluate the summarization result. With our interactive tool, she can switch between the title+abstract summary and the research field summary. She can also access paper details in each node cluster with a sorted list by citation count. She mainly compares the visual summarization by CommonNeighbor and SimRank. In this case, she prefers the result by CommonNeighbor in Figure 5(a) because the influence evolutions make more sense: the initial paper quickly raises much attention on pattern mining research such as itemset and association rule mining, then the thread splits into four streams on general data management research (such as web and uncertainty skyline analysis), trajectory analysis, subgraph analysis and application in software engineering (e.g. bug analysis). The thread of web data analysis gradually moves to web retrieval and finally leads to tag analysis and anomaly behavior detection. Compared with CommonNeighbor, SimRank creates some false links, e.g. the direct flow from the frequent pattern mining paper to uncertainty data analysis.

Furthermore, we ask another invited researcher to study the influence of the well-known Internet power-law paper in SIGCOMM'1999. The maximal influence graph is summarized by the bidirectional CommonNeighbor algorithm into Figure 1 (in the second page). From the visual summarization, she learns that the SIGCOMM paper directly influences the research on Internet topology and simulation. Next, over the Internet topology topics, the P2P research becomes popular and after that the web-related research and XML. The most recent hot topic in this thread appears to be sensor network which corresponds well to his domain knowledge.

## VI. Related Work

First, *graph summarization*, constructing a smaller abstraction to represent the large graph has been a traditional research topic, e.g. using graph clustering algorithms. These algorithms usually optimize certain association or cut measure during the k-way graph partition. Several measures have been proposed, e.g. ratio association, ratio cut [11] and normalized cut [6]. The similar problem is also studied in the context of community detection by interdisciplinary researchers [12]. However, most of the clustering and community detection methods on graph target at maximizing intra-cluster connections while minimizing inter-cluster connections. This is fairly different from the IGS problem studied here. On the other hand, there are also plenty of works in compressing large graphs for efficient storage and representation. In [9], MDL-based compression was proposed to present the graph with an aggregated structure and an error correction list. On influence graphs which are sparse, it performs similarly to a structural equivalence based grouping [13], leaving huge visual clutters unsettled. Meanwhile, Shahaf et al. [14][15] studied the similar problem of summarizing large amount of information into user-friendly visual maps. On a quite different focus, our method is built on the graph with explicit linkage data while the textual content of each node can be absent or incomplete.

Second, considerable work has been conducted for studying the effects of *social influence*. For example, Bakshy et al. [16] conducted randomized controlled trials to identify the effect of social influence on consumer responses to advertising. Tang et al. [17] presented a Topical Affinity Propagation (TAP) approach to quantify the topic-level social influence in large networks. Kempe et al. [1] proposed to use a submodular function to formalize the influence maximization problem and develop a greedy algorithm to solve the problem with provable approximation guarantee. Most of these works focus on the existence of social influence or the nature of the information diffusion process and do not consider the summarization problem. Recently, Mehmood et al. proposed CSI [18], a model that generalizes the classical Independent Cascade model to the community level. CSI can produce similar visual forms to our result. However, the CSI model is designed for the social influence scenario, while our method is more focused on the visual summarization of large influence graphs in the objective of maximizing flows. We do not leverage the information propagation model and the associated log data in such scenarios.

## VII. Conclusions

In this paper, we propose the influence graph summarization problem and present a unified framework to solve it. The framework achieves our design objectives, including (1) flow rate maximization that highlights the evolution of influence; (2) localized visualization from the source node. The framework is comprehensive and flexible. We provide both the SymNMF based solution and implementation details. Through evaluations with real-world academic citation graphs, we demonstrate that our framework constantly outperforms classical methods, such as graph clustering and compression algorithms, in both quantitative performance and qualitative visual effects.

## Acknowledgment

This work is supported in part by China National 973 project 2014CB340301, NSFC grant 61379088, NSF grant IIS1017415, U.S. Army Research Laboratory Cooperative Agreement W911NF-09-2-0053 and DARPA project W911NF-12-C-0028. Jie Tang is supported by the National High-tech R&D Program 2014AA015103 and NSFC grant 61222212.

## References

[1] D. Kempe, J. Kleinberg, and E. Tardos, “Maximizing the spread of influence through a social network,” in *KDD*, 2003, pp. 137–146.

[2] C. Ding, X. He, and H. D. Simon, “On the equivalence of nonnegative matrix factorization and spectral clustering,” in *SDM*, 2005.

[3] C. Boutsidis and E. Gallopoulos, “SVD based initialization: A head start for nonnegative matrix factorization,” *Pattern Recognition*, vol. 41, no. 4, pp. 1350–1362, 2008.

[4] J. B. Kruskal, “On the shortest spanning subtree of a graph and the traveling salesman problem,” *Proceedings of the American Mathematical Society*, vol. 7, no. 1, pp. 48–50.

[5] G. Jeh and J. Widom, “Simrank: A measure of structural-context similarity,” in *KDD*, 2002, pp. 538–543.

[6] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, pp. 888–905, 1997.

[7] M. E. J. Newman, “Fast algorithm for detecting community structure in networks,” *Physical Review E*, vol. 69, no. 6, p. 066133, 2004.

[8] G. Karypis, V. Kumar, and V. Kumar, “Multilevel k-way partitioning scheme for irregular graphs,” *Journal of Parallel and Distributed Computing*, vol. 48, pp. 96–129, 1998.

[9] S. Navlakha, R. Rastogi, and N. Shrivastava, “Graph summarization with bounded error,” in *SIGMOD*, 2008, pp. 419–432.

[10] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, “Arnetminer: extraction and mining of academic social networks,” in *KDD*, 2008, pp. 990–998.

[11] P. K. Chan, M. D. F. Schlag, and J. Y. Zien, “Spectral k-way ratio-cut partitioning and clustering,” *IEEE Trans. on CAD of Integrated Circuits and Systems*, vol. 13, no. 9, pp. 1088–1096, 1994.

[12] S. Fortunato, “Community detection in graphs,” *Physics Reports*, vol. 486, no. 3-5, pp. 75–174, 2010.

[13] F. Lorrain and H. C. White, “Structural equivalence of individuals in social networks,” *The Journal of Mathematical Sociology*, vol. 1, no. 1, pp. 49–80, 1971.

[14] D. Shahaf, J. Yang, C. Suen, J. Jacobs, H. Wang, and J. Leskovec, “Information cartography: creating zoomable, large-scale maps of information,” in *KDD*, 2013, pp. 1097–1105.

[15] D. Shahaf and C. Guestrin, “Connecting the dots between news articles,” in *KDD*, 2010, pp. 623–632.

[16] E. Bakshy, D. Eckles, R. Yan, and I. Rosenn, “Social influence in social advertising: evidence from field experiments,” in *EC*, 2012, pp. 146–161.

[17] J. Tang, J. Sun, C. Wang, and Z. Yang, “Social influence analysis in large-scale networks,” in *KDD*, 2009, pp. 807–816.

[18] Y. Mehmood, N. Barbieri, F. Bonchi, and A. Ukkonen, “Csi: Community-level social influence analysis,” in *ECML/PKDD*, 2013, pp. 48–63.