

TANGENT: A Novel, “Surprise-me”, Recommendation Algorithm

Kensuke Onuma
Sony Corporation
1-7-1 Konan, Minato-ku
Tokyo, Japan
Kensuke.Oonuma@
jp.sony.com

Hanghang Tong
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA USA
htong@cs.cmu.edu

Christos Faloutsos
Carnegie Mellon University
5000 Forbes Avenue
Pittsburgh, PA USA
christos@cs.cmu.edu

ABSTRACT

Most of recommender systems try to find items that are most relevant to the older choices of a given user. Here we focus on the “surprise me” query: A user may be bored with his/her usual genre of items (e.g., books, movies, hobbies), and may want a recommendation that is related, but off the beaten path, possibly leading to a new genre of books/movies/hobbies.

How would we define, as well as automate, this seemingly self-contradicting request? We introduce TANGENT, a novel recommendation algorithm to solve this problem. The main idea behind TANGENT is to envision the problem as node selection on a graph, giving high scores to nodes that are well connected to the older choices, and at the same time well connected to unrelated choices. The method is carefully designed to be (a) parameter-free (b) effective and (c) fast. We illustrate the benefits of TANGENT with experiments on both synthetic and real data sets. We show that TANGENT makes reasonable, yet surprising, horizon-broadening recommendations. Moreover, it is fast and scalable, since it can easily use existing fast algorithms on graph node proximity.

Categories and Subject Descriptors

H.2.8 [Database Management]: Database applications—*Data mining*; H.3.3 [Information Storage and Retrieval]: Information search and retrieval—*Clustering, search process*

General Terms

Algorithms, Experimentation

1. INTRODUCTION

Recommender systems are vital for e-commerce sites, with most striking examples being Amazon, Netflix, Pandora, Strands, etc. Recommender systems are doubly useful: On one hand, they help users filter through enormous numbers of available items, and focus on the few ones that match their preferences; on the other hand, recommender systems help enterprises increase their sales (e.g. movies, books, songs), on the long tail. Numerous algorithms for collaborative filtering [28] have been studied for recom-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

KDD'09, June 28–July 1, 2009, Paris, France.

Copyright 2009 ACM 978-1-60558-495-9/09/06 ...\$5.00.

mender systems; whereas, graph-based algorithms have been attracting great interest among researchers recently.

Most of the recommendation algorithms focus on the precision in the proximity to user preferences. However, this strategy tends to suggest items only on the center of user preferences and thus narrows down the users’ horizons. According to the research about the quality of recommender systems [20], broadening users’ horizons is one of important qualities for recommender systems. Such systems provide a win-win situation: users may find more interesting items, and e-commerce enterprises increase their sales and their user satisfaction.

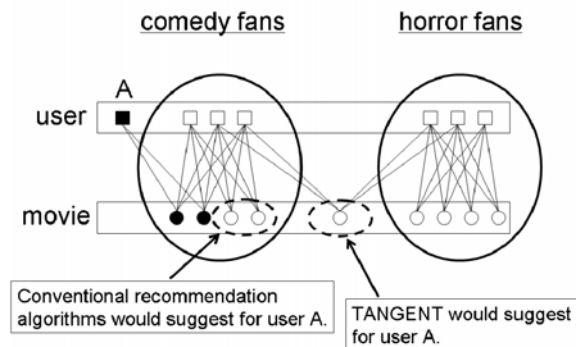


Figure 1: The difference between conventional recommendation algorithms and TANGENT. Square nodes represent users and circular nodes denote movies. Users are connected to movies which they like. Black nodes are a target user and his/her favorite movies.

In order to solve this seemingly self-contradicting problem, we proposed TANGENT, a recommendation algorithm that takes into account the connectivity to other groups in order to broaden users’ horizons. It is based on the graph mining technique of computing similarity between nodes and can be applied to any dataset that can be represented as a graph. Figure 1 illustrates the difference of results between conventional recommendation algorithms and TANGENT. We assume that this synthetic bipartite graph records rating information by users to movies. There are two groups in this graph; fans for comedy movies and those for horror movies. If “user A” rates two movies in the comedy group as favorite movies, conventional recommendation algorithms would suggest movies in the same group. In contrast, our proposed TANGENT method suggests a movie *between* two groups, to “surprise” the user and to gently broaden his/her horizons.

The rest of the paper is organized as follows: we first review the related work in Section 2; the proposed algorithm is presented in

Section 3; the experimental results are presented in Section 4; and we conclude the paper in Section 5.

2. RELATED WORK

In this section, we briefly review related work, which can be categorized into three parts: graph mining, recommender system and ranking and proximity on graphs.

Graph Mining. The main idea behind TANGENT is to envision the problem as node selection on a graph. Graph mining itself is a hot research topic in the recent years. For static graphs, representative work includes pattern and law mining [1, 6, 22], frequent substructure discovery [34, 14], collective classification [4], fraud and anomaly detection [21, 24], community mining and graph partition [11, 12, 16, 19], etc. More recently, there has been an increasing interest in mining time-evolving graphs, such as densification laws and shrinking diameters [18], community evolution and dynamic [2], etc.

Recommender System. To our knowledge, there is little work in recommender systems which focuses on broadening users' interest. The work of Kamahara et al. [15], who explore methods for locating unexpected items from similar clusters to user's cluster, is close to our problem. However, parameters such as the number of clusters and thresholds, which are required in [15], are not easy to be tuned. McNee et al. [20] raise a question about the current accuracy metrics and propose some aspects which should give to evaluation of recommender system.

Recommender systems have been attracting considerable research interest, with recent emphasis on graph-based such systems. Fous et al. propose in [9] to apply Euclidean commute time distance, which is one of random-walk-based methods of computing similarities between nodes, to a collaborative recommendation. Wang et al. [33] apply a node similarity algorithm to item-based recommendation. Bell et al. [3] propose a low rank matrix approximation method for collaborative filtering and successfully apply it to the Netflix competition. Although different in the specific methods, the basic idea behind all these methods is to find items that are most relevant to the older choices of the given user.

Ranking and Proximity on Graphs. In the literature, there are several methods for calculating relevance (a.k.a similarity) between nodes on graphs. Because the shortest path, which is a conventional property for representing relevance, fails to capture the multiple faceted relationship between nodes on the graph, random walk based techniques such as PageRank [25], personalized PageRank [13], Random walk with restart [30, 32], Euclidean commute time distance [9], escape probability [8, 17] have been widely applied recently. In this paper, we use random walk with restart; however, the idea behind TANGENT can be naturally extended to other proximity measurement.

The upcoming "bridging score" in our proposed TANGENT is also related to the node betweenness: The betweenness of a node is high if the node lies on the paths between many other pairs of nodes. Intuitively, such a node is a good "bridge", and deleting such a node may often disconnect the graph. Freeman [10] proposes a method for computing betweenness based on shortest paths, and Brandes [5] studies a fast algorithm for computing centrality. On the other hand, Newman [23] also proposes fast algorithms to compute, by using a random walk on the graph.

The degree of connectivity in a graph (which is the core idea behind the bridging score in the proposed TANGENT) is also related to the degree of anomaly in a graph because the nodes with considerable links to various groups are considered to be anomalies. Sun et al. [30] introduce the normality score of a node as a measure of how homogeneous (i.e., interconnected) its neighbors are.

Although potentially useful for TANGENT, none of these methods have been used in a recommender system.

3. THE TANGENT ALGORITHM

In this section, we describe details of the proposed TANGENT algorithm.

3.1 Notations and Problem Definitions

Symbol	Description
$\mathbf{A} = (a_{ij})$	the weighted adjacency matrix
\mathbf{A}	the normalized matrix of \mathbf{A}
$\vec{b} = (b_i)$	the vector of bridging score
$\mathbf{D} = (d_{ij})$	$d_{ii} = \sum_j a_{ij}$ and $d_{ij} = 0$ for $i \neq j$
E	the set of edges in G
\vec{e}_i	$n \times 1$ unit vector (all zero except one at row i)
G	the weighted, undirected graph
k	the number of query nodes
l_i	the number of edges connected to node i
m	the number of edges in G
n	the number of nodes in G
\hat{r}_{S_i}	the average of all non-diagonal elements in \mathbf{R}_i
$Q = (q_i)$	the set of query nodes ($1 \leq i \leq k$)
\mathbf{R}	the matrix of relevance score
\mathbf{R}_i	the similarity matrix among nodes $\in S_i$
$\vec{r}_i = (r_{i,j})$	the vector of relevance score to node i
$\vec{r}_Q = (r_{Q,j})$	the vector of relevance score to query nodes Q
$S_i = \{s_{i,j}\}$	the set of nodes which have links to node i
$\vec{t}_Q = (t_{Q,i})$	the vector of TANGENT score
V	the set of nodes in G
w_{ij}	weight on each in $(i, j) \in E$

Table 1: Symbols

Table 1 gives the main symbols used in this paper. We assume that we are given a weighted, undirected graph $G = (V, E)$, where V denotes the set of nodes and E represents the set of edges. n is the number of nodes and m is the number of edges. This graph has weights $w_{ij} > 0 ((i, j) \in E)$ on the edges. Since the graph is undirected, the weights are symmetric (i.e. $w_{ij} = w_{ji}$). The weight w_{ij} should indicate the strength of the relation between node i and node j . Let $A = (a_{ij})_{i,j \in V}$ be the adjacency matrix of the graph with $a_{ij} = w_{ij}$ if $(i, j) \in E$ and $a_{ij} = 0$ otherwise. A set of query nodes can be represented by $Q = (q_i)_{1 \leq i \leq k}$.

Let us consider a movie-rating data set. Basically this database consists of three kinds of information; demographic information about the users, information about the movies, and information about rating which users assigned to movies they have watched. Each user and movie corresponds to a node, and each user is connected to the corresponding movies by edges weighted according to the degree of rating. This graph expresses the relation between users and movies (a user-movie graph), and is constructed as a bipartite graph. In this graph, nodes can be divided into two disjoint sets such as $V = \{V_1, V_2\}$. A query node corresponds to a user to whom the recommender system want to give recommendations (i.e. $k = 1$).

Now we define the TANGENT problem as follows:

PROBLEM 1 (TANGENT).

Given: an edge-weighted undirected graph G with adjacency matrix \mathbf{A} , the set of query nodes $Q = (q_i)_{1 \leq i \leq k}$.

Find: a node that (1) is close enough to Q , and (2) has high potential to reach other nodes.

3.2 Alternatives and Subtle Problems

Here we quickly review algorithms for computing ranking and proximity on graphs and introduce alternatives we have explored to solve our problem but find to be inapplicable because of subtle problems.

Relevance Score. As described in Section 2, there are numerous methods to measure similarity $r_{i,j}$ between a pair of nodes (i, j) : escape probability, random walk with restarts, electricity-based

[26], maximum flow, to name a few. Although most of them give reasonable scores, none of them takes into account the connectivity to other groups. Therefore, they can fulfill only the first of our requirements.

It is an additional, subtle question on how to measure group proximities, that is, how close is node i to the *set* of query nodes Q . Averaging, or just adding the individual scores $\sum_{q \in Q} r_{q,i}$ is a reasonable, but not necessarily the best choice. Another choice would be to take the maximum such score, roughly corresponding to an ‘OR’: a node gets high score if it is close to at least one of the query nodes. The list continues: the product is also a reasonable choice (corresponding to an ‘AND’), and the CePS algorithm [31] gives additional, more sophisticated choices.

Negation of Relevance Score. One of our first approaches to solve the “surprise me” query problem was to consider negation. Specifically: “Find nodes close to one query node, but far away from all other query nodes.” That is, we want a candidate node v that has high proximity to only one of the query nodes, and as far as possible from the rest. We assume that a relevance score corresponds to probability, such as steady-state probability [32]. Since multiplication of scores corresponds to conjunction, 1-complement would correspond to negation, and thus the score of a node v should be

$$\text{negation}(v, Q) = \max_{1 \leq j \leq k} \{r_{q_j, v} \times \prod_{i \neq j} (1 - r_{q_i, v})\} \quad (1)$$

However, although this equation has solid foundation in logic, when used on large, real graphs, it gives almost the same arithmetic scores as $\max_{1 \leq j \leq k} (r_{q_j, v})$. This is because most proximity/relevance scores are extremely low ($\ll 10^{-2}$) and thus the product of the $(1 - r_{q_i, v})$ terms is practically equal to 1.

Entropy. Among the methods we tried but eventually did not work, is an entropy-based idea, inspired by maximum marginal gains: a candidate node v would be good, if it would make maximum difference in the existing distribution of proximities. This would mean that it opens horizons. That is, let $r_{Q,i}$ be the score of node i wrt Q ; let $\hat{r}_{Q,i}$ be the score if we add node v to the query set Q . We want the node v that maximize the difference of the entropy (i.e., $H(\hat{r}_{Q,i}) - H(r_{Q,i}) = -\sum_{i \in V} \{\hat{r}_{Q,i} \times \log(\hat{r}_{Q,i})\} + \sum_{i \in V} \{r_{Q,i} \times \log(r_{Q,i})\}$).

However, despite the fact that maximum marginal gain should lead to “surprising” choices, they are too far away from query nodes because they are the best choices to make uniform distribution of proximities, which maximize the entropy.

Our main point of this detour is that, despite the wealth of ideas on node proximity, there are subtle issues that need to be carefully thought out.

3.3 Framework of TANGENT Algorithm

In this subsection, we illustrate the framework of our proposed algorithm.

The function of TANGENT can be described as $\vec{t}_Q = \text{tangent}(A, Q)$ where $\vec{t}_Q = (t_{Q,i})_{i \in V}$ is a vector of degree of recommendation, to which we refer as TANGENT score in this paper. We would like to have a recommendation algorithm for suggesting items which are not only relevant to user preferences but also have large connectivity to other groups. In order to fulfill these requirements, TANGENT algorithm consists of three parts as follows.

1. Calculate relevance score (RS) of each node : \vec{r}_Q .
2. Calculate bridging score (BRS) of each node : \vec{b} .
3. Compute the TANGENT score (\vec{t}_Q) by somehow merging two criteria above.

Moreover, the algorithm for computing bridging score is designed to use relevance score, which re-uses the result of relevance

scores calculated at the first part and also prevents the last merging part from being complicated.

The rest of this section describes details of each part.

3.4 Relevance Score (RS)

First we would like to compute the relevance score of a single node j , for a single query node q_i . There are several methods for computing the relevance score between nodes, as described in Section 2. Although we can use any algorithm to compute relevance score, we propose to use random walk with restart [30, 32] in this paper.

According to [30, 32], the relevance score \vec{r}_{q_i} is computed by the following formula:

$$\begin{aligned} \vec{r}_{q_i} &= c\tilde{\mathbf{A}}^T \vec{r}_{q_i} + (1-c)\vec{e}_{q_i} \\ &= c\tilde{\mathbf{A}}\vec{r}_{q_i} + (1-c)\vec{e}_{q_i} \end{aligned} \quad (2)$$

where $\tilde{\mathbf{A}}$ is the normalized adjacency matrix of \mathbf{A} by normalized graph Laplacian ($\tilde{\mathbf{A}} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, where \mathbf{D} is the degree matrix of \mathbf{A} .) and $(1-c)$ denotes the fly-out probability.¹ Note that \mathbf{A} is symmetric because $a_{ij} = w_{ij} = w_{ji} = a_{ji}$. \vec{r}_{q_i} is determined by

$$\begin{aligned} \vec{r}_{q_i} &= (1-c)(\mathbf{I} - c\tilde{\mathbf{A}})^{-1}\vec{e}_{q_i} \\ &= \mathbf{R} \cdot \vec{e}_{q_i} \end{aligned} \quad (3)$$

from the derivation of equation (2).

One of solutions to obtain \vec{r}_{q_i} from equation (2) is the iterative method, iterating equation (2) until it converges; however, we propose to pre-compute \mathbf{R} because we can obtain the solution of \vec{r}_{q_i} for all nodes from equation (3) with less online computation costs. Details are described in subsection 3.7. Note that it can be proved by equation (3) that

$$\mathbf{R} = [\vec{r}_1 \vec{r}_2 \dots \vec{r}_n] \quad (4)$$

If there are multiple query nodes, we compute a relevance score from all query nodes by taking Boolean ‘OR’ of relevance scores to query nodes by

$$r_{Q,j} = 1 - \prod_{i=1}^k (1 - r_{q_i,j}) \quad (5)$$

3.5 Bridging Score (BRS)

Bridging score means the degree of connectivity to other groups on a graph. Although betweenness [5, 10, 23] might be one of the criteria for it, their algorithms are completely different and also require a considerable computation costs. Or, we might use the sum of weight of links connected to the corresponding node as an index considering that the popular items might have potential to bridge groups. However, popularity does not always correspond to the connectivity to other groups.

We proposed to apply an anomaly detection algorithm on a bipartite graph [30], which is based on the relevance scores described above, to any kinds of graph. The level of connectivity of a node on a graph is related to the level of anomaly on a graph because the nodes with considerable links to various, otherwise disjoint groups are considered to be anomalies. Moreover, the merit of using the same algorithm is that we can share the result of computing relevance scores. From equation (4), R expresses the relevance scores from each node to each node and we can re-use R for computing the bridging score. This is important, because TANGENT only needs a

¹ c is the only parameter in the proposed TANGENT. However, we find that the recommendation result of TANGENT is insensitive to c . Therefore, we fix c to be 0.5 throughout this paper.

few relevance scores, and thus it can exploit any of the fast, known algorithms, to compute such scores, like B_LIN/BB_LIN [32] and variants.

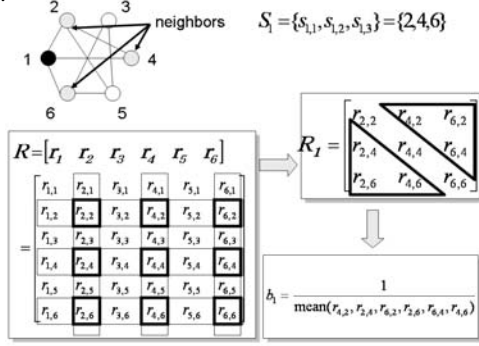


Figure 2: Illustration of the procedure of computing bridging score.

Figure 2 presents the procedure of computing bridging score. Let l_i be the number of edges connected to node i . Given a node i , we first find the set of nodes to which i links: $S_i = \{s_{i,j} | 0 \leq j \leq l_i\}$. Then we compute the relevance scores between any pair of elements in S_i and construct a $l_i \times l_i$ similarity matrix \mathbf{R}_i . After that, we take the mean of all non-diagonal elements in \mathbf{R}_i and get b_i by inverting it. If node i is connected by only one edge or it has no connection, we define its bridging score as $b_i = 0$, which means that it is not taken into account as a candidate of recommendation.

Figure 3 shows the intuition of computing bridging score. If i is connected to one group, then the relevance scores between any pair of elements in S_i should be high and as a consequence, b_i become small. On the other hand, if i has links to several groups, the relevance scores between different groups have small numbers and we get large b_i .

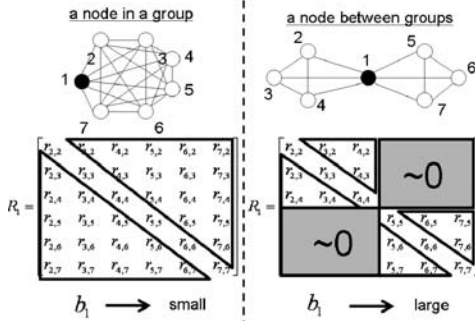


Figure 3: The intuition of computing bridging score.

Algorithm 1 summarizes the algorithm of computing bridging score. Note that \vec{b} is independent from query nodes Q . Therefore, it can be pre-computed.

3.6 TANGENT Score

There are several ways to combine the above two criteria. However, again, reasonable-looking choices may suffer from subtle issues. We discuss them first, and then give our proposed combination method.

Reasonable-Looking, but Unsuccessful Choices. One natural method to try is linear combination: i.e. $t_{Q,j} = r_{Q,j} + \alpha \times b_j$ (let α be the coefficient). However, (a) the two values typically differ by orders of magnitude, and the larger, b_j , typically overshadows $r_{Q,j}$, (b) the method is not parameter-free anymore: somebody, somehow, needs to determine a good value of α , which may need to be changed as the graph grows.

Our second approach, also reasonable-looking, but also unsuccessful, is to use *skyline queries*. Since we do not know how much

Algorithm 1 Bridging Score (BRS) Computation Algorithm

- 1: Given: Adjacency matrix A
- 2: **for** $i = 1$ to n **do**
- 3: Pick up node i and let $S_i = \{s_{i,j} | 0 \leq j \leq l_i\}$ be a set of nodes which have links to i ;
- 4: **if** ($l_i = 0$) OR ($l_i = 1$) **then**
- 5: $b_i = 0$;
- 6: **else**
- 7: Compute the relevance score vector $\vec{r}_{s_{i,1}}, \vec{r}_{s_{i,2}}, \dots, \vec{r}_{s_{i,l_i}}$ from each $S_i = s_1, \dots, s_{l_i}$;
- 8: Construct $l_i \times l_i$ matrix $\mathbf{R}_i = [\vec{r}_{s_{i,1}} \vec{r}_{s_{i,2}} \dots \vec{r}_{s_{i,l_i}}]$;
- 9: Take the average of all non-diagonal elements in \mathbf{R}_i and obtain \hat{r}_{S_i} ;
- 10: Divide one by \hat{r}_{S_i} , and obtain $b_i = 1/\hat{r}_{S_i}$;
- 11: **end if**
- 12: **end for**
- 13: return a vector of bridging scores \vec{b} .

weight to give to each of the two criteria (relevance vs. bridge-ness), we could try multi-objective optimization, which leads to skyline queries [27]: Given a set of multidimensional points D , a skyline query finds the skyline points from D , that is, those points that are not totally dominated by any other point in D . For example, if a point p is on the skyline, there is no data point p' in D such that p' is larger than p for the values in all dimensions. The example of a skyline query is shown in Figure 4. Movie A, B and C are on the skyline and there are no other nodes which have larger relevance score as well as larger bridging score than points on the skyline.

However, using skyline queries for our setting suffers from the following subtle drawbacks: (1) it cannot keep the balance between the two criteria; for instance, a skyline query extracts a node which has the highest relevance score even if it has zero bridging score (see Movie C in Figure 4) and vice versa, (2) the skyline also neglects second-best candidates. (In Figure 4, movie Z dominates everything else, and thus we would only return that movie, even if Movie Y is also a good candidate.)

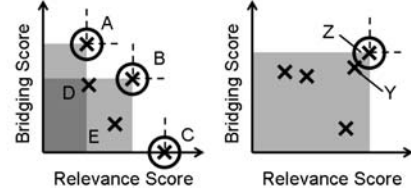


Figure 4: The examples of a skyline query.

Proposed Combination Method. A reasonable and well performing method is multiplication: i.e. $t_{Q,j} = r_{Q,j} \times b_j = r_{Q,j}/\hat{r}_{S_i}$. It can be seen from the equation above that $t_{Q,j}$ represents the ratio between relevance score to query nodes and relevance score among neighbors. Notice that all our proposed choices are parameter-free, making TANGENT parameter-free, as we required.

3.7 Scalability

We want TANGENT to be parameter-free², effective and fast. The first goal is achieved as we showed. The second goal is discussed in details later. Here we illustrate the achievement speed. Specifically, we discuss the scalability of the TANGENT algorithm by discussing each of the first two parts. The third part, merging, is $\mathcal{O}(n)$ since it needs just a multiplication for each of the $n - q = \mathcal{O}(n)$ candidate nodes.

Computing Relevance Score. Although computing \mathbf{R} in a straightforward way requires cubic computation cost to the number of node,

²As mentioned before, the only parameter c in TANGENT has little influence on the result and is fixed to be 0.5 throughout this paper.

there is a fast algorithm to get good approximate solution of random walk with restart proposed by [32]. Once we pre-compute \mathbf{R} , online computation cost for computing relevance score for each node is only matrix-vector multiplication shown in equation (3).

Computing Bridging Score. It can be seen in Algorithm 1 that, \mathbf{R} , which is pre-computed for computing relevance score, can be re-used in computing bridging score. Therefore, the online computation of this algorithm takes time $\mathcal{O}(n)$. Moreover, in case of a user-movie bipartite graph, we do not need to compute bridging scores of user nodes for recommendation; therefore, the computation cost can be reduced to the number of movies. If faster online computation is required, we can pre-compute it since the bridging score is independent of the query.

4. EXPERIMENTAL RESULTS

In this section, we evaluate the effectiveness of TANGENT. Since the objective of TANGENT is to find the neighbors of user preference with large connectivity to other groups, not to improve the accuracy of recommendation, conventional evaluation metrics such as degree of agreement[9], a percentile score[9], a recall score[9], mean absolute error[29], MSE[33], and NMSE[33] do not work on the evaluation of effectiveness for TANGENT algorithm, neither does questionnaires asking whether the recommendation made by the algorithm is satisfactory or not. Therefore, we evaluate its potential on a number of synthetic graphs as well as on real data sets. For both synthetic and real data set, we present (1) the case studies which show that TANGENT gives reasonable results; and (2) systematic comparisons with conventional recommendation algorithms (i.e., by relevance score) which show that TANGENT tends to return more surprising results. Let n_{tan} and n_{rel} be the surprising nodes in the returned recommendation list by TANGENT and by conventional recommendation algorithms, respectively. We define the ‘‘Surprising Gain’’ as $\frac{n_{tan} - n_{rel}}{n_{rel}}$. A big ‘‘Surprising Gain’’ means that TANGENT returns more surprising results compared with conventional recommendation algorithms.

4.1 Evaluations on Synthetic Data Sets

Case Studies on Unipartite Graphs. We start off by presenting unipartite graphs, which have several groups in them and are connected by nodes. In the experiment, we compare the ranked lists based on relevance score, bridging score, and TANGENT score.

The graph on the left side of Figure 5 has two groups; nodes 1 - 4 in Group 1 and nodes 6 - 9 in Group 2, which are connected through node 5, and we run a query for node 1. Node 2 gets the highest relevance score among nodes except the query node, followed by nodes 3 and 4. This result would be same as what conventional recommendation algorithm would give. On the other hand, comparing bridging scores, we find that node 5 gets the highest score, because the neighbors of node 5 (nodes 3, 4, 6 and 7) are separated in half into two groups and, as a consequence, the average of relevance scores among them becomes small. Notice that nodes 7, 3 and 4 also get relatively high bridging scores. As a result, nodes 3 and 4 are ranked as No. 1 in terms of TANGENT score, as reflects our intuition that these nodes are close enough to the query node as well as getting close to Group 2.

The graph on the right side of Figure 5 has four groups and two query nodes in different groups. In this case, both node 3 and node 12 are considered as bridging nodes; however node 3 should be considered as a better bridging node than node 12 because node 3 leads to Group 2, which is larger than Group 3. As we can see, TANGENT algorithm gives a result which follows our intuition.

Case Studies on Bipartite Graphs. We next evaluate the effectiveness of TANGENT algorithm on synthetic bipartite graphs. Figure 6 illustrates the structure of graphs and the results. As we

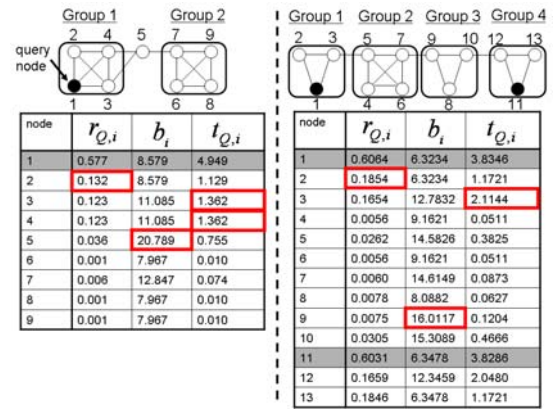


Figure 5: Relevance score, bridging score, and TANGENT score of each node on synthetic unipartite graphs. A query node is represented by a black node.

described in Figure 1, square nodes represent users, circular nodes denote movies, and edges correspond to ratings by users. Graph 1 and Graph 2 in Figure 6 have almost same structure, except that user 1 and movie 24 are linked with each other in Graph 2.

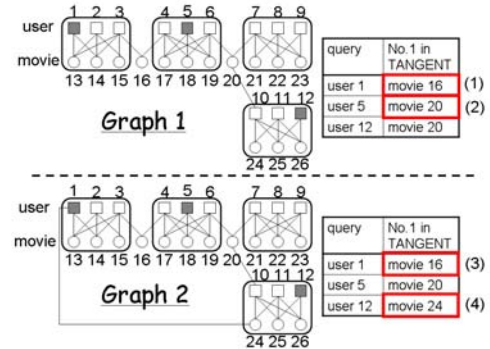


Figure 6: No.1-ranked movie for several query nodes on synthetic bipartite graph. Dark nodes are query nodes.

We can derive some fascinating observations from this figure as follows: (1) In Graph 1, movie 16 gets the highest TANGENT score for user 1. This result completely agrees with our motivation described in Figure 1. (2) In Graph 1, TANGENT algorithm concludes movie 20 to be the best recommendation for user 5, not movie 16. It is a reasonable result because movie 20 is watched by three groups; while movie 16 is watched by two groups. (3) In Graph 2, TANGENT algorithm still suggest movie 16 for user 1, not movie 20. Recommender system does not have to be sensitive to outliers. In this case, user 1’s preference still lies on the group where s/he involves and TANGENT takes it into account. This result indicates that TANGENT is robust to exceptionally favorite movie. (4) In Graph 2, movie 24 gets the highest TANGENT score for user 12, despite the fact that movie 20 is recommended in Graph 1. Comparing movie 20 and movie 24, we find that movie 24 has higher relevance score than movie 20, and also movie 24 has connectivity to another group in Graph 2. This structural change causes the difference of the recommendation results.

Comparison with Relevance Score. Finally, we systematically compare the proposed TANGENT with conventional recommendation algorithms. The goal of TANGENT is to provide some ‘‘surprising’’ recommendations, i.e., to find the neighbors of user preference with large connectivity to other groups. Formally, for a given query node q , we say the recommended node i is ‘‘surpris-

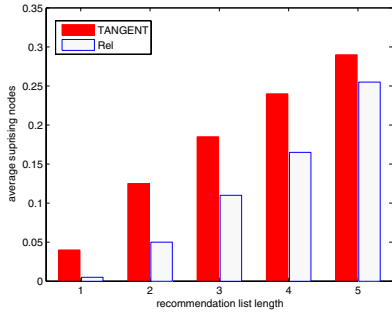


Figure 7: Comparison of TANGENT and conventional recommendation algorithms on synthetic graphs. Higher is better.

ing” if (1) nodes i and q belong to the *same* group, but the node i has some links to other groups; or (2) nodes i and q belong to the *different* groups, but the node i has some links to the same group as the query node q . For example, in the left graph of Figure 5, nodes 3-5 are “surprising” wrt the query node 1; while the nodes 2, 6-9 are not “surprising” results. Figure 7 present the comparison between TANGENT and conventional recommendation algorithms (Rel), where x-axis is the length of the recommendation list and y-axis is the average surprising nodes in the returned recommendation list. It can be seen that TANGENT consistently gives more surprising results. On average, the “Surprising Gain” of TANGENT on synthetic data sets is 1.95.

4.2 Evaluations on MovieLens Data Set

Here, we present the experimental result on a real movie-rating data set MovieLens (<http://www.movielens.org/>). MovieLens data set was collected by GroupLens Research Project at the University of Minnesota and contains 100,000 rating information from 943 users on 1682 movies. Each user has rated at least 20 movies from 1 (strongly unsatisfactory) to 5 (strongly satisfactory).

Using this data set, we construct a user-movie bipartite graph. We choose positive ratings (4 and 5) and connect users and movies by edges weighted by $4^{rating-4}$ in order to emphasize strongly satisfactory ratings. The reason why we do not take into account negative ratings (1 - 3) is that treating negative ratings requires another mechanism and we would like to focus on the evaluation of effectiveness of TANGENT algorithm. As a result, a user who gives only negative ratings and movies which receive only negative ratings are neglected; and 942 users, 1447 movies and 55375 ratings remain. We apply BB_Lin [32] to our system for fast computation of relevance scores and pre-compute only \mathbf{R} .

Case Studies. Figure 8 illustrates the transition of rankings between the relevance score and our TANGENT score. For each movie, we mention its genre(s) - the abbreviations are in the upper right of the figure. The top of Figure 8 are for users who prefer to slapstick movies. Notice that the top 10 movies by relevance score consist mainly of comedy movies. On the other hand, niche comedy movies such as “Ace Ventura: When Nature Calls” and “Renaissance Man” are out of top 10 ranking by TANGENT score and popular comedy movies such as “The Princess Bride”, “Toy Story”, “Monty Python and the Holy Grail” and “Men in Black” emerge instead. That is, the TANGENT algorithm recommends a wider variety of genres and gives No.1 rank to “Star Wars”, which scores high with both user preference as well as with the connectivity to other groups. Note that, especially “The Flintstones” decreases its ranking dramatically, because it has only one positive rating.

The lists on the bottom of Figure 8 are for users whose preference is horror movies. Although the transition between two lists is more moderate than that in case of slapstick movies, TANGENT al-

gorithm still recommends diverse genres of movies; not only horror but also thriller, crime, and drama.

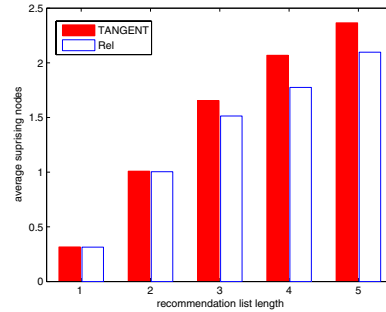


Figure 9: Comparison of TANGENT and conventional recommendation algorithms on MovieLens data set. Higher is better.

Comparison with Relevance Score. Here, for a query movie q (or the query user who likes movie q). We say a recommended movie i is a “surprising movie” if (1) movies i and q share some *common* genres; and (2) movie i has some *new* genres that the query movie q does not have. For example, for the query movie “Robin Hood: Men in Tights (Comedy)”, “Jack (Comedy and Drama)” is treated as a surprising recommendation; while neither “Spy Hard (Comedy)” or “Raiders of the Lost Ark (Action and Adventure)” is treated as surprising recommendations. Figure 9 presents the comparison between TANGENT and conventional recommendation algorithms (Rel), where x-axis is the length of the recommendation list and y-axis is the average surprising nodes in the returned recommendation list. Again, TANGENT consistently gives more surprising results and the “Surprising Gain” of TANGENT on this data set is 0.08.

4.3 Evaluations on CIKM Data Set

This data set is from CIKM proceedings (<http://www.informatik.uni-trier.de/~ley/db/conf/cikm/>). We construct an author-paper bipartite graph. The authors are annotated with one or more keywords as his/her attribute based on the session names where the corresponding paper was presented. For instance, “Tali Brodianskiy” has published a paper (titled with “Self-Correcting Queries for XML”) in CIKM 2007 and the paper was presented in “XML Query Processing” session. Therefore, he is associated with “XML” and “Query” as his attribute. Totally, we have 952 paper nodes, 1,895 author nodes, and 158 keywords.

Case Studies. Table 2 lists the top-5 recommended authors for “Frank Hing-Wah Luk” by TANGENT score and by relevance score, respectively. “Frank Hing-Wah Luk” has published one paper (titled with “Triple-Node Hierarchies for Object-Oriented Databases Indexing”) and he is associated with “Index” as his attribute. So, clearly, he is a databases person in CIKM community. From Table 2, it can be seen that while the two methods agree on the first two recommended authors (“Ada Wai-Chee Fu” and “Ke Wang”), TANGENT provides more diverse recommendations compared with relevance score, - the recommended authors by TANGENT (“Yabo Xu”, “Jiawei Han”, and “Jian Pei”) are cross-disciplinary in both databases and data mining. On the other hand, relevance score will recommend pure databases persons (“Weixiong Rao”, “Lei Chen”, and “Yingyi Bu”) instead.

Comparison with Relevance Score. For a given query author q , we say the recommended author i is “surprising” if (1) authors q and i share some *common* keywords, and (2) the author i has some *new* keywords that the query author q does not have. Figure 10 presents the comparison between TANGENT and conventional rec-

'Slapstick Movie Fan' case

User Preference (rating 5)

- Robin Hood: Men in Tights (1993) (Comedy)
- Young Frankenstein (1974) (Comedy, Horror)
- Naked Gun 33 1/3: The Final Insult (1994) (Comedy)
- Fatal Instinct (1993) (Comedy)

Abbreviation of genres

- "Com" : Comedy
- "Ch" : Children's
- "Ani" : Animation
- "Dr" : Drama
- "War" : War
- "Mus" : Musical
- "Rom" : Romance
- "Hor" : Horror
- "Adv" : Adventure
- "SF" : Sci-Fi
- "Thr" : Thriller
- "Cr" : Crime
- "Ac" : Action

Ranked list by relevance score

Rank	Title	Genre
1	The Flintstones (1994)	Ch,Com
2	Spy Hard (1996)	Com
3	Oliver & Company (1988)	Ani,Chi
4	Jack (1996)	Com,Dr
5	Son in Law (1993)	Com
6	Ace Ventura: When Nature Calls (1995)	Com
7	Renaissance Man (1994)	Com,Dr,War
8	Pocahontas (1995)	Ani,Chi,Mus,Rom
9	Corrina, Corrina (1994)	Com,Dr,Rom
10	Beverly Hillbillies, The (1993)	Com
11	Princess Bride, The (1987)	Ac,Adv,Com,Rom
15	Monty Python and the Holy Grail (1974)	Com
21	Empire Strikes Back, The (1980)	Ac,Adv,Dr
26	Raiders of the Lost Ark (1981)	Ac,Adv
29	Return of the Jedi (1983)	Ac,Adv,Rom,SF,War
32	Star Wars (1977)	Ac,Adv,Rom,SF,War
42	Toy Story (1995)	Ani,Chi,Com
53	Men in Black (1997)	Com,Dr

Ranked list by TANGENT score

Rank	Title	Genre
1	Star Wars (1977)	Ac,Adv,Rom,SF,War
2	Return of the Jedi (1983)	Ac,Adv,Rom,SF,War
3	The Princess Bride (1987)	Ac,Adv,Com,Rom
4	Toy Story (1995)	Ani,Chi,Com
5	Monty Python and the Holy Grail (1974)	Com
6	Spy Hard (1996)	Com
7	Raiders of the Lost Ark (1981)	Ac,Adv
8	Empire Strikes Back, The (1980)	Ac,Adv,Dr
9	Jack (1996)	Com,Dr
10	Men in Black (1997)	Ac,Adv,Com,SF
25	Ace Ventura: When Nature Calls (1995)	Com
27	Corrina, Corrina (1994)	Com,Dr,Rom
35	Son in Law (1993)	Com
42	Oliver & Company (1988)	Ani,Chi
43	Renaissance Man (1994)	Com,Dr,War
52	Pocahontas (1995)	Ani,Chi,Mus,Rom
166	The Beverly Hillbillies (1993)	Com
1439	The Flintstones (1994)	Ch,Com

'Horror Movie Fan' case

User Preference (rating 5)

- A Nightmare on Elm Street (1984) (Horror)
- The Shining (1980) (Horror)
- Jaws (1975) (Action, Horror)

Ranked list by relevance score

Rank	Title	Genre
1	The Silence of the Lamb (1991)	Dr, Thr
2	Psycho (1960)	Hor, Rom, Thr
3	Pulp Fiction (1994)	Cr, Dr
4	An American Werewolf in London (1981)	Hor
5	Natural Bom Killers (1994)	Ac, Thr
6	Carrie (1976)	Hor
7	Alien (1979)	Ac, Hor, SF, Thr
8	Twelve Monkeys (1995)	Dr, SF
9	Evil Dead II (1987)	Ac, Ad, Com, Hor
10	Scream (1996)	Hor, Thr
15	Star Wars (1977)	Ac,Adv,Rom,SF,War
17	Fargo (1996)	Cr, Dr, Thr
22	The Godfather (1972)	Ac, Cr, Dr
45	Contact (1997)	Dr, SF

Ranked list by TANGENT score

Rank	Title	Genre
1	The Silence of the Lambs (1991)	Dr, Thr
2	Scream (1996)	Hor, Thr
3	Pulp Fiction (1994)	Cr, Dr
4	Star Wars (1977)	Ac,Adv,Rom,SF,War
5	Fargo (1996)	Cr, Dr, Thr
6	Twelve Monkeys (1995)	Dr, SF
7	Psycho (1960)	Hor, Rom, Thr
8	The Godfather (1972)	Ac, Cr, Dr
9	Contact (1997)	Dr, SF
10	Alien (1979)	Ac, Hor, SF, Thr
13	An American Werewolf in London (1981)	Hor
12	Natural Bom Killers (1994)	Ac, Thr
16	Carrie (1976)	Hor
23	Evil Dead II (1987)	Ac, Ad, Com, Hor

Figure 8: Ranked lists by relevance score and one by TANGENT score for user who likes slapstick movies (top) and those who likes horror movies (bottom). Arrows highlight the transition of ranking. Nomenclature for genre is shown in the upper right ("Com":Comedy, and "Hor":Horror are in bold) TANGENT gives much higher diversity of genres, as desired.

Rank	by TANGENT	Attribute of Authors	by Relevance	Attribute of Authors
1	Ada Wai-Chee Fu	Index, Performance, Stream	Ada Wai-Chee Fu	Index, Performance Stream
2	Ke Wang	DB, Mine, Cluster, Pattern, Stream	Ke Wang	DB, Mine, Cluster, Pattern, Stream
3	Yabo Xu	DB, Mine, Stream	Weixiong Rao	Performance
4	Jiawei Han	DB, Mine	Lei Chen	Performance
5	Jian Pei	Mine, Pattern, Forecast, Stream	Yingyi Bu	Performance

Table 2: Ranked lists by relevance score and by TANGENT score for query author “Frank Hing-Wah Luk” on CIKM data set. “Frank Hing-Wah Luk” has published 1 paper in CIKM titled with “Triple-Node Hierarchies for Object-Oriented Databases Indexing” and he is associated with “Index” as his attribute.

ommendation algorithms (Rel), where x-axis is the length of the recommendation list and y-axis is the average surprising nodes in the returned recommendation list. Again, TANGENT consistently gives more surprising results and the “Surprising Gain” of TANGENT on this data set is 0.22.

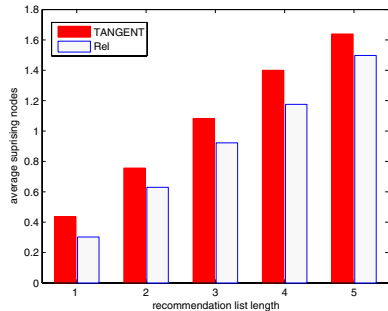


Figure 10: Comparison of TANGENT and conventional recommendation algorithms on CIKM data set. Higher is better.

4.4 Evaluations on DBLP Data Set

Case Studies. We also test the proposed TANGENT on DBLP data set (<http://www.informatik.uni-trier.de/ley/db/>). Here, we construct a series of different author-paper bipartite graphs from two conferences. One conference is always “KDD”, and the other conference is either “SIGMOD”, “ICML”, “WWW”, “SIGIR”, “CIKM”, “SIGCOMM”, “SIGGRAPH” or “ISMB”. Table 3 gives the 1st recommended authors on such bipartite graphs by TANGENT score and by relevance score, respectively. We can see that TANGENT provides more surprising recommendations compared with relevance score. For example, on the bipartite graph from “KDD” and “ICML”, conventional recommendation algorithms will recommend “Moiss Goldszmidt” for “Lise Getoor”, which makes sense since both of them are interested in probabilistic reasoning and graphical models. On the other hand, TANGENT will recommend “Charu C. Aggarwal” instead. “Charu C. Aggarwal” is mainly interested in performance, data mining and databases. So, compared with “Moiss Goldszmidt”, “Charu C. Aggarwal” is a more surprising recommendation for “Lise Getoor”. Yet, the recommendation by TANGENT is still close enough to the query author. For instance, “Charu C. Aggarwal” is also interested in uncertainty in databases which is closely related to probabilistic reasoning, - one of the research interest of “Lise Getoor”.

Comparison with Relevance Score. Here, we use the authors who publish in only one of the two conferences as the query authors. And if the recommended author publishes in both conferences, we say that it is a surprising recommendation. Figure 11 presents the comparison between TANGENT and traditional recommendation algorithms (Rel), where we always fix the length of the ranked list to be 5 and the number below the corresponding bars is the “Surprising Gain”. Again, TANGENT consistently gives more surprising results. For example, on the bipartite graph constructed from “KDD” and “SIGMOD”, the “Surprising Gain” of TANGENT is

0.78; while on the bipartite graph constructed from “KDD” and “SIGGRAPH”, TANGENT achieves 0.70 “Surprising Gain” etc.

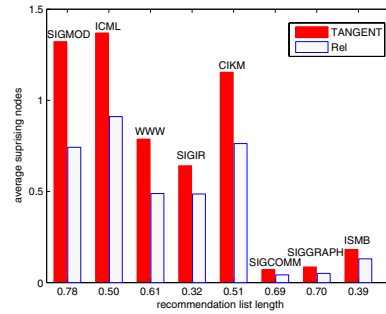


Figure 11: Comparison of TANGENT and conventional recommendation algorithms on DBLP data set. Author-Paper bipartite graph is constructed from “KDD” plus one more conference. The number below the corresponding bars is the “Surprising Gain”. Higher is better.

5. CONCLUSIONS

We define a novel recommendation problem, namely, how to make a recommendation that broadens the horizons of the user, in the sense that it is close enough to his/her current interests to be pleasant, but also towards a new area that the user has not discovered yet. The motivation is how to respond to a user that says “surprise me”: suppose that a user that consistently chooses, say, slapstick “comedy” movies, may occasionally get bored with that genre, and s/he would like to try something slightly different - what should we recommend? As humans, we would probably recommend, say, “horror comedy”, “cartoons”, or a “comedy-musical”. Our goal is to automate the response to such “surprise me” query. Our approach is to find items that are close to the user preferences, while they also have high connectivity to other groups.

One major contribution of this work is exactly the problem definition. Additional contributions are the following:

1. Careful design decisions, so that the resulting method is (a) parameter-free, (b) effective and (c) fast.
2. Extensive comparison of the numerous alternatives; while all seem reasonable on paper, several of them suffer from subtle issues.
3. Experiments on synthetic and real data sets, illustrating the effectiveness of the proposed method. On the synthetic data set, our proposed method indeed spots “bridge” nodes. On the real data set, our method does not follow blindly the user preferences (as conventional algorithms do), but instead it makes recommendations that are reasonable and yet off the beaten path.

Future work could focus on the implementation of TANGENT on the emerging Hadoop/MapReduce architecture [7], for Tera- and Peta-byte scale recommender systems.

Querying Author	By TANGENT	By Relevance	Conferences
Soumen Chakrabarti	Divesh Srivastava	William W. Cohen	KDD, SIGMOD
Andrew Y. Ng	Chengxiang Zhai	David M. Blei	KDD, SIGIR
Jiawei Han	Wei-Yin Ma	Mohammed J. Zaki	KDD, SIGIR
Xiaojin Zhu	Naoki Abe	Guy Lebanon	KDD, ICML
Pedro Domingos	Eamonn J. Keogh	Christopher Meek	KDD, ICML
Lise Getoor	Charu C. Aggarwal	Moiss Goldszmidt	KDD, ICML
Flip Korn	Philip S. Yu	Michail Vlachos	KDD, ICML
Christos Faloutsos	Michael I. Jordan	Ravi Kumar	KDD, ICML
Michael I. Jordan	Christos Faloutsos	John D. Lafferty	KDD, ICML
Robert E. Schapire	Philip S. Yu	Leslie P. Kaelbling	KDD, ICML

Table 3: 1st recommended author by relevance score and by TANGENT score on DBLP data set. The co-authors of the query author are blanked out. The last column shows the conferences that we use to construct the bipartite graphs.

6. ACKNOWLEDGEMENT

This material is based upon work supported by the National Science Foundation under Grants No. DBI-0640543 IIS-0705359 CNS-0721736 IIS0808661. Also, by the iCAST project sponsored by the National Science Council, Taiwan, under the Grants No. NSC97-2745-P-001-001 and by the Ministry of Economic Affairs, Taiwan, under the Grants No. 97-EC-17-A-02-R7-0823. Also, under the auspices of the U.S. Department of Energy by University of California Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344 (LLNL-CONF-404625), subcontracts B579447, B580840. This work is also partially supported by an IBM Faculty Award, a Yahoo Research Alliance Gift, a SPRINT gift, and a gift from Sony, with additional funding from Intel, NTT and Hewlett-Packard. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, or other funding parties.

7. REFERENCES

- [1] R. Albert, H. Jeong, and A.-L. Barabasi. Diameter of the world wide web. *Nature*, (401):130–131, 1999.
- [2] L. Backstrom, D. P. Huttenlocher, J. M. Kleinberg, and X. Lan. Group formation in large social networks: membership, growth, and evolution. In *KDD*, pages 44–54, 2006.
- [3] R. Bell, Y. Koren, and C. Volinsky. Modeling relationships at multiple scales to improve accuracy of large recommender systems. In *KDD*, pages 95–104, 2007.
- [4] M. Bilgic and L. Getoor. Effective label acquisition for collective classification. In *KDD*, pages 43–51, 2008.
- [5] U. Brandes. A Faster Algorithm for Betweenness Centrality. *Mathematical Sociology*, 25(2):163–177, 2001.
- [6] A. Broder, R. Kumar, F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, and J. Wiener. Graph structure in the web: experiments and models. In *WWW Conf.*, 2000.
- [7] J. Dean and S. Ghemawat. MapReduce: simplified data processing on large clusters. *Communications of the ACM*, 51(1):107–113, 2008.
- [8] P. G. Doyle and J. L. Snell. *Random Walks and Electric Networks*. Mathematical Association of America, 1984.
- [9] F. Fouss, J.-M. Renders, A. Pirotte, and M. Saerens. Random-Walk Computation of Similarities between Nodes of a Graph with Application to Collaborative Recommendation. *IEEE Transaction on Knowledge and Data Engineering*, 19(3):355–369, 2007.
- [10] L. C. Freeman. A Set of Measures of Centrality Based on Betweenness. *Sociometry*, 40(1):35–41, March 1977.
- [11] D. Gibson, J. Kleinberg, and P. Raghavan. Inferring web communities from link topology. In *Ninth ACM Conference on Hypertext and Hypermedia*, pages 225–234, New York, 1998.
- [12] M. Girvan and M. E. J. Newman. Community structure is social and biological networks.
- [13] T. H. Haveliwala. Topic-sensitive PageRank. In *WWW*, pages 517–526, New York, NY, USA, 2002.
- [14] R. Jin, C. Wang, D. Polshakov, S. Parthasarathy, and G. Agrawal. Discovering frequent topological structures from graph datasets. In *KDD*, pages 606–611, 2005.
- [15] J. Kamahara, T. Asakawa, S. Shimojo, and H. Miyahara. A Community-Based Recommendation System to Reveal Unexpected Interests. In *Proceedings of the 11th International Multimedia Modelling Conference*, pages 433–438, Washington, DC, USA, 2005.
- [16] G. Karypis and V. Kumar. Multilevel -way hypergraph partitioning. In *DAC*, pages 343–348, 1999.
- [17] Y. Koren, S. C. North, and C. Volinsky. Measuring and Extracting Proximity in Networks. In *KDD*, pages 245–255, 2006.
- [18] J. Leskovec, J. M. Kleinberg, and C. Faloutsos. Graphs over time: densification laws, shrinking diameters and possible explanations. In *KDD*, pages 177–187, 2005.
- [19] J. Leskovec, K. J. Lang, A. Dasgupta, and M. W. Mahoney. Statistical properties of community structure in large social and information networks. In *WWW*, pages 695–704, 2008.
- [20] S. M. McNee, J. Riedl, and J. A. Konstan. Being Accurate is Not Enough: How Accuracy Metrics have hurt Recommender Systems. In *CHI '06 extended abstracts on Human factors in computing systems*, pages 1097–1101, 2006.
- [21] J. Neville, Ö. Simsek, D. Jensen, J. Komoroske, K. Palmer, and H. G. Goldberg. Using relational knowledge discovery to prevent securities fraud. In *KDD*, pages 449–458, 2005.
- [22] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45:167–256, 2003.
- [23] M. E. J. Newman. A measure of betweenness centrality based on random walks. *Social Networks*, 27(1):39–54, January 2005.
- [24] C. C. Noble and D. J. Cook. Graph-based anomaly detection. In *KDD*, pages 631–636, 2003.
- [25] L. Page, S. Brin, R. Motwani, and T. Winograd. The PageRank Citation Ranking: Bringing Order to the Web. *Stanford Digital Library Technologies Project, SIDL-WP-1999-0120*, 1999.
- [26] C. R. Palmer and C. Faloutsos. Electricity Based External Similarity of Categorical Attributes. In *PAKDD*, pages 486–500, 2003.
- [27] J. Pei, A. W.-C. Fu, X. Lin, and H. Wang. Computing Compressed Multidimensional Skyline Cubes Efficiently. In *ICDE*, pages 96–105, 2007.
- [28] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: an open architecture for collaborative filtering of netnews. In *Proceedings of the 1994 ACM conference on Computer supported cooperative work*, pages 175–186, New York, NY, USA, 1994.
- [29] B. M. Sarwar, G. Karypis, J. Konstan, and J. Riedl. Recommender Systems for Large-Scale E-Commerce: Scalable Neighborhood Formation Using Clustering. In *Proceedings of the Fifth International Conference on Computer and Information Technology*, 2002.
- [30] J. Sun, H. Qu, D. Chakrabarti, and C. Faloutsos. Relevance Search and Anomaly Detection in Bipartite Graphs. *SIGKDD Explorations Special Issue on Link Mining*, 7(7):48–55, January 2005.
- [31] H. Tong and C. Faloutsos. Center-piece subgraphs: problem definition and fast solutions. In *KDD*, pages 404–413, 2006.
- [32] H. Tong, C. Faloutsos, and J.-Y. Pan. Random walk with restart: fast solutions and applications. *Knowledge and Information Systems*, 14(3):327–346, 2008.
- [33] F. Wang, S. Ma, L. Yang, and T. Li. Recommendation on Item Graphs. In *ICDM*, pages 1119–1123, 2006.
- [34] D. Xin, J. Han, X. Yan, and H. Cheng. Mining compressed frequent-pattern sets. In *VLDB*, pages 709–720, 2005.