A Generic Graph-based Multidimensional Recommendation Framework and Its Implementations

Sangkeun Lee
supervised by Sang-goo Lee
Intelligent Data Systems Lab
School of Computer Science and Engineering, Seoul National University, Seoul, Korea
{liza183, sglee}@europa.snu.ac.kr

ABSTRACT
As the volume of information on the Web is explosively growing, recommender systems have become essential tools for helping users to find what they need or prefer. Most existing systems are *two-dimensional* in that they only exploit *User* and *Item* dimensions and perform a typical form of recommendation 'Recommending *Item* to *User'*. Yet, in many applications, the capabilities of dealing with multidimensional information and of adapting to various forms of recommendation requests are very important. In this paper, we take a graph-based approach to accomplishing such requirements in recommender systems and present a generic graph-based multidimensional recommendation framework. Based on the framework, we propose two homogeneous graph-based and one heterogeneous graph-based multidimensional recommendation methods. We expect our approach will be useful for increasing recommendation performance and enabling *flexibility* of recommender systems so that they can incorporate various user intentions into their recommendation process. We present our research result that we have reached and discuss remaining challenges and future work.

Categories and Subject Descriptors
H.3.3 [**Information Search and Retrieval**]: Information Search and Retrieval – *Information filtering, Retrieval models*

General Terms
Algorithms, Design, Experimentation.

Keywords
Multidimensional, Recommender systems, Context-aware recommender systems, Random walks, Implicit feedback, Usage log, Context-awareness.

1. INTRODUCTION
As the information on the Web becomes larger and more diverse, it has become more difficult for users to find what they really want. As a result, recommender systems have become necessary for dealing with such information overload. So far, recommender systems have been widely researched both by industry and academia. Typical recommender systems consider *User* and *Item* dimensions in recommendation process and perform a particular form of recommendation that is 'Recommending *Item* to *User'*. More formally, they consider the recommendation problem as a problem of finding an effective utility function $u: User \times Item \rightarrow Utility$, which estimates utilities of items for a given user [1]. They have focused on improving the performance of recommendation with respect to various criteria such as accuracy, diversity, serendipity, and etc. [2]; but, there have been few researches focusing on realizing *Multidimensional recommendation (*MDR*)*.

Recently, the importance of MDR has been emphasized, as it is considered to be very effective for achieving better users' satisfaction. There are two main goals in MDR [3]. Firstly, MDR systems aim to more accurately identify user preferences by exploiting additional dimensions (e.g., *Location*, *Time*, *Season*, etc.) to the *User* and *Item* dimensions. For example, when a user requested a recommendation for *Vacation*, a user's preference on a ski resort or a beach may largely depend on *Season*. Thus, in this case, MDR systems aim to consider *Season* dimension into recommendation process for more accurate recommendation result. Secondly, MDR systems aim to perform not only 'Recommending *Item* to *User*' but also other various queries by incorporating multidimensional information. There can be many queries with different user intentions. For example, a user may want to request a recommendation for songs that are often played by the user's friends or for songs whose genres are similar with the user's favorite genres. Similarly, a user may want to incorporate the user's contextual information into recommendation or not depending on situations. MDR systems need an expressive and unambiguous way for users and systems to represent and process various queries.

However, achieving MDR is challenging. In many cases, it is not straightforward to extend well-known and commonly used existing two-dimensional recommendation models (e.g. k-NN CF, matrix factorization, etc.) to support MDR. Also, it is not trivial to define a language for representing various user queries with different intentions. Furthermore, because the search space becomes significantly larger when dealing with multidimensional data, the data sparsity issue becomes very critical.

We tackle such challenges by taking a graph-based approach and propose a generic graph-based MDR framework. Then, based on our framework, we introduce three methods that construct and utilize different types of graphs that are *Co-occurrence Graph*, *Contextual Bipartite Graph*, and *Heterogeneous Data Graph*. First two methods are based on homogeneous graph model, and the last one is based on heterogeneous graph model. For each method, we demand for a node ranking measure that can reasonably quantify the proximity between a given *query* and *nodes* that represent target entities. For the first two homogeneous-graph based methods, we adapt *Personalized PageRank* [4], which is widely used and known to be an effective node ranking measure. For the heterogeneous graph-based method, we define a new node ranking measure, named *PathRank* by extending *Personalized PageRank* to take advantage of various types of edges in heterogeneous graph, and we adapt *PathRank* for our third method.

WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

The remainder of this paper is organized as follow. Section 2 presents research background and related work. In section 3, we present a generic graph-based MDR framework and its expected advantages. In section 4, we introduce our MDR methods based on the proposed framework. In section 5, we explain the result that we have reached so far. Section 6 concludes our work and discusses future work.

2. RELATED WORK

Traditional recommendation methods are generally classified into two main categories, Content-based Filtering (CBF) and Collaborative Filtering (CF). For a given user, CBF recommends items that are similar with the ones previously liked by the user. On the other hand, CF utilizes the other users' item preferences and aggregates them to estimate the given user's unknown item preferences. Many variations of CF have been introduced such as user-based k-NN CF, item-based k-NN CF, matrix factorization, and etc [5]. CF approaches have been acknowledged to be practical and popularly used in many real-world applications [6]; but they do not consider additional information other than *User* and *Item* for their recommendation process.

Since Adomavicius et al.[7] introduced the concept, several studies on MDR have been presented. Reduction-based approach [3] recommends items according to the given user's context by only using the users' prior item preferences that match the current context. It can be straightforwardly applied to many existing algorithms, but the data sparsity problem often becomes more serious due to too much filtering. To resolve the problem, some methods have been introduced such as Disjunction-based approach [8] and Learning-to-Rank approach [9]. Yet, all of these methods fix the recommendation form and do not allow users to express their intentions. There have been few approaches that aim to achieve flexibility of recommendation [10][11]. Such approaches require an ordinary user to learn structured language or usages of operators in order to request a recommendation.

Earlier graph-based recommender methods have a common point in that they are based on homogeneous graphs. Generally, they take advantage of existing node ranking measures defined on homogeneous graph such as *PageRank* [12], *Personalized PageRank* [4], and *HITS* [13]. Fouss et al. [14] compared several random walk based quantities for measuring relatedness between users and movies on bipartite graph. Gori et al. [15] presented a random walk based item scoring algorithm, *ItemRank*, reflecting propagation and attenuation properties. Both approaches only consider *User* and *Item*. Lately, some studies have attempted to utilize additional information. Xiang et al. [16] presented a method that aim to improve accuracy by blending users' short-term and long-term preferences.

Although heterogeneous graph can express real-world objects and their relationships much more expressively, heterogeneous graph-based multidimensional recommenders systems and node ranking measures defined on heterogeneous graph are relatively less studied [17]. *ObjectRank* [18] first introduced that considering many types of edges differently could influence node ranking results. It computes revised *Personalized PageRank* based on differently assigned authority flows to each type of edge. There have been a few studies that consider paths importantly for ranking on heterogeneous graph. *PathSim* [17] presents a novel node similarity measure on heterogeneous graph and gives more freedoms to users to express their requirement by defining meta paths. However, because it can be defined only between the same types of objects, it cannot be directly used for our method in that we need to measure relatedness between different types of objects such as *User* and *Item*.

3. PROPOSED APPROACH

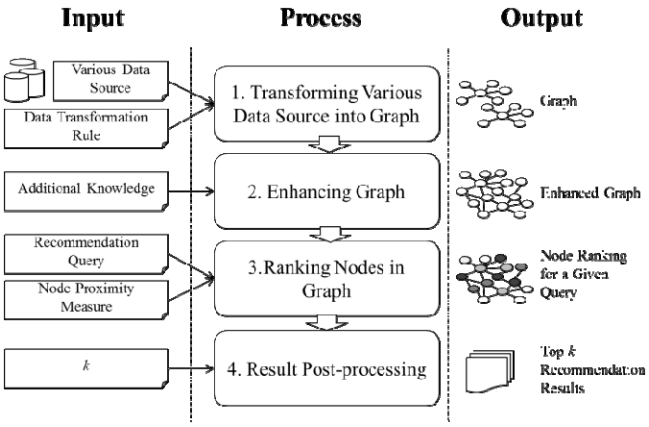

Figure 1: A Generic Graph-based Multidimensional Recommendation Framework

Figure 1 depicts a generic framework of our general graph-based MDR describing inputs and outputs in the overall process of graph-based MDR. In this framework, MDR problem is considered as a problem of finding most relevant nodes for a given query that is a set of nodes on graph, where the nodes represent entities. We first transform various data sources (e.g. user ratings, item consumption logs, etc.) into a graph based on data transformation rules. is a set of nodes that represent entities, and is a set of edges that represent relationships between entities. Then, we enrich the created graph by adding nodes and edges based on additional knowledge. For example, when we created *User* nodes, we could add *SimilarTaste* edges between them by comparing their preferences on items. A user can express his or her recommendation request as a recommendation query , and it can be represented using various features of graph model such as nodes, paths, and their weights.

By adapting a node ranking measure defined on graph, we can define a utility function that estimates the relatedness between given query and each entity that is represented as a node in graph. After ranking nodes in the graph, we perform a post-processing to generate a top-*k* entity list as the recommendation results. In this framework, the step 2 process can be omitted if not needed. Several aspects like data transformation rules, query representations and expressiveness, and node ranking measures can vary depending on implementations of our framework

There are three main benefits of adopting graph-based approach for MDR. Firstly, because graph data model is suitable to deal with heterogeneous information, we can reasonably represent various real-world objects and their relationships as nodes and edges in graph. Secondly, without understanding complicated language, users can describe their recommendation queries using graph features (e.g., nodes, paths, etc.). Lastly, we can benefit from not only direct relationships but also the indirect relationships between nodes. This is very helpful for resolving data sparsity problem that is known to be a serious problem when dealing with multidimensional data [8].

4. METHODOLOGY

In this section, we introduce three different implementations of our framework. These implementations can be categorized into

two groups that are homogeneous graph-based and heterogeneous graph-based methods.

4.1 Homogeneous Graph-based Method

We introduce two methods that are based on homogeneous graphs that have only one edge type. We use a log table L as an input for the both methods. Each attribute in table L can be considered as a dimension, and its values can be considered as its entities. We denote an entity o whose dimension is D by $o \in D$. We assume that a recommendation query Q is given as $\{O, D_T\}$, where O is a set of entities $\{o_1, o_2, \dots o_{n_q}\}$ and D_T is recommendation target dimension (e.g. $SONG$). For instance, when $O = \{'Matt' \in USER, 'Seoul' \in LOCATION\}$ and $D_T = SONG$ for a query $Q = \{O, D_T\}$, the query can be interpreted as a recommendation request for songs that are suitable for the user 'Matt' and location '*Seoul*'. Both methods transform the log table L into graphs $G(V, E)$, where each node $v_i \in V$ corresponds to a set of entities, and weight of each edge $e_{ij} \in E$ represents the strength of relationship between two entities o_i and o_j ; however they differentiate the strategies to create nodes and assign edges in graphs.

***Co-occurrence Graph-based* Method (CG) [19]:** It transforms a given log table L into a graph based on co-occurrence of entities in the same tuples of L. For every entity o appeared in the log table L, we create a node v_o that corresponds to the entity. Then, we compute $|V| \times |V|$ transition matrix P_{CG} whose element p_{ij} is the weight of an edge between node v_i and v_j for constructed graph as follow:

$$p_{ij} = \begin{cases} p(o_i, o_j) \times \dfrac{1}{l-1}, & p(o_i, o_j) > 0 \\ 0, & p(o_i, o_j) = 0 \end{cases}$$

where $p(o_i, o_j)$ is the probability of co-occurrence of two entities o_i and o_j in the tuples of L, and l is the number of dimensions in L ($l \geq 2$). Edge e_{ij} between v_i and v_j exists iff $p_{ij} > 0$.

***Contextual Bipartite Graph-based* Method (CBG) [20]:** This method transforms a given log table L into a bipartite graph based on recommendation factor set F. Each recommendation factor $f_i \in F$ is defined as $\{D_{f_i}, w_i\}$. D_{f_i} is a set of one or more than one non-target dimensions whose combination will affect recommendation results, and w_i is the weight of f_i, in other words, how much this factor will affect the recommendation results. We assume that users can reflect their intentions by changing the recommendation factors. To create a graph for this method, we first create a node set $V = V_1 \cup \dots \cup V_i \cup \dots \cup V_{|F|} \cup V_T$. The nodes in V_i correspond to the entity combination whose dimensions match the dimensions in D_{f_i}, and the nodes in V_T correspond to the entities of target dimension D_T. Next, we compute a $|V| \times |V|$ weighted adjacent matrix M whose element m_{ij} is the weight of an edge between node v_i and v_j as follow:

$$m_{ij} = \begin{cases} C(i,j) & , if\ v_i \notin V_T\ and\ v_j \in V_T \\ C(i,j) \times w_k & , if\ v_i \in V_T, v_j \notin V_T, and\ v_j \in V_k \\ 0 & , otherwise \end{cases}$$

where $C(i,j)$ is the co-occurrence number of entities o_i and o_j in the tuples of L, and w_k is the weight of f_k that V_k corresponds to.

Now, we normalize the matrix M to achieve the transition matrix P_{CBG} as follow:

$$p_{ij} = \begin{cases} \dfrac{m_{ij}}{\sum_{v_k \in outlink[v_i]} m_{ik}}, if\ \ outlink[v_i] \neq 0 \\ 0, otherwise \end{cases}$$

The final constructed graph G that is represented by the matrix P is bipartite in that there are no edges between nodes $v_i \in V_T\ and\ v_j \in V_T$.

For both CG and CBG, we adapt Personalized PageRank to compute utility of each entity for a given query. From the query $Q = \{O, D_T\}$, we compute a vector $\tilde{\vec{q}}$ as:

$$\tilde{q}_i = \begin{cases} 1, if\ \mathcal{E}_{v_i} \subseteq O \\ 0, otherwise \end{cases}$$

where $\mathcal{E}_{v_i}$ is a set of entities that each node $v_i \in V$ corresponds to. Next, we achieve a normalized vector $\vec{q}$ from $\tilde{\vec{q}}$ such that elements in $\vec{q}$ sum up to 1. With a transition matrix P (P_{CG} or P_{CBG}), we calculate utility vector as $\vec{r} = cP^T\vec{r} + (1-c)\vec{q}$ (typically, c is set to 0.85). Then, we define a utility function $u(Q, o_i)$ as follow:

$$u(Q, o_i) = \begin{cases} r_i, if\ o_i \in D_T \\ 0, o_i \notin D_T \end{cases}$$

Finally, we sort the entities by $u(Q, o_i)$ in descending order to generate top-k entities for the query. By adapting *Personalized PageRank*, these two methods can reasonably quantify the utility of each entity by fusing authority of each nodes and relatedness to the given query. CG is based on the simple assumption that the more two objects appears in the same tuples in the log table, the more they are related. CBG can be considered as a more sophisticated modification of CG in that it can reflect various recommendation factors depending on user intention.

4.2 Heterogeneous Graph-based Method

In this section, we introduce a **Heterogeneous Data Graph and Path-based Method (HGP)**. Unlike previous two methods, we do not constraint our input as a log table. Instead, we transform various types of data sources (e.g. relational data, XML, RDF, OWL) into a heterogeneous data graph that consists of different types of nodes and edges. In this paper, we assume that we already have constructed the graph and focus more on entity ranking. For a graph $G(V, E)$, V is a set of nodes $\{v_1, , \dots, v_i, \dots, v_{|V|}\}$, and each v_i represents entity o_i. Each node has one node type $t_k \in T_V = \{t_1, \dots, t_k, \dots, t_{|T_V|}\}$, where a node type t_k represents a dimension of entities. E is a set of edges, and each edge e_{ij} from node v_i to v_j is represented as $(v_i, v_j, t_l, w_{e_{ij}})$. t_l and $w_{e_{ij}}$ are the edge's type and weight, where $t_l \in T_E = \{t_1, \dots, t_l, \dots, t_{|T_E|}\}$.

Although one matrix can represent a homogeneous graph, when we deal with more than one type of edge in heterogeneous graph, we need more than one matrix to represent to the graph. So, we represent the heterogeneous graph G as a set of $|V| \times |V|$ weighted adjacent matrices $\mathbb{A} = \{A_{t_1}, \dots, A_{t_l}, \dots, A_{|T_E|}\}$, where each A_{t_l} represents the graph's edges whose type is t_l. We can derive a transition matrix set $\mathbb{P} = \{P_{t_1}, \dots, P_{t_l}, \dots, P_{|T_E|}\}$ by normalizing the adjacent matrices.

The intuition of HGP is that paths defined on heterogeneous graph can represent various user intentions expressively. For example, we can define several paths as follow:

Path 1: $(User) \xrightarrow{SimilarTaste} (User) \xrightarrow{Prefer} (Song)$

Path 2: $(User) \xrightarrow{Prefer} (Song) \xrightarrow{SimilarContents} (Song)$

Path 3: $(User) \xrightarrow{LivesIn} (Location) \xrightarrow{hasUsers} (User) \xrightarrow{Prefer} (Song)$

where parentheses include a node type and arrows represent edges in the graph.

As we previously mentioned that ranking objects by following different paths could be useful for recommender systems [21], each path can be interpreted as a different recommendation procedure. If we follow the defined *path 1* on the graph from a *User* type node, then we will reach the songs that are preferred by the users who have similar taste with the user that the start node represents. In other words, we can consider the *path 1* represents collaborative filtering process. Similarly, the *path 2* represents a content-based filtering process that aims to find items whose contents are similar to the items previously liked by the given user. The *path 3* can be understood as a location-based recommendation procedure, so we can find the songs that are preferred by the users who live in the same location by following it. As we can see, paths can be used to describe different recommendation procedures. So, we aim to allow users to define paths and use them as part of their query.

For the HGP, we assume that a query is given as $Q = \{O, P, D_T\}$, where P is a set of pairs $< p_i, w_i >$, p_i is a path, w_i is its weight, and $\sum_{k=1}^{n} w_k = 1$. $\vec{q}$ can be computed based on O in the same way as we explained in the previous section.

As we need to consider paths in node proximity measuring, we define a novel node ranking measure ***PathRank*** as follow:

$$\vec{r} = w_{trans} P^T \vec{r} + w_{path}\left(w_1 P_{p_1}^T + w_2 P_{p_2}^T + \cdots + w_n P_{p_n}^T\right)\vec{r} + w_{restart}\vec{q}$$

$$\text{where } w_{trans} + w_{path} + w_{restart} = 1$$

P is the transition matrix when we ignore the edge types and consider it as homogeneous, so it can be computed by normalizing the sum of adjacent matrix $A = \sum_{t_i \in T} A_{t_i}$. Each P_{p_i} represents a transition matrix for path *i*, and it can be computed by sequentially multiplying transition matrix of each edges on the path. For example, a transition matrix for the path 1 can be computed as $P_{path\ 1} = P_{similarTaste} \times P_{prefer}$. Like *Personalized PageRank*, *PathRank* r_i of a node v_i is the stationary probability of random walker's being at the node after enough number of iterations of random walk with restart. w_{trans} is the weight of following graph ignoring edge types, and $w_{restart}$ is weight of restarting the walk. Additionally, during the random walk with restart process, the random walker also jumps to the destination nodes following each path *i* with a given probability $w_{path} \times w_i$. All these weights are given as probability values. Intuitively, *PathRank* can be understood as an extension of *Personalized PageRank*, because it inherits the ability of fusing node authority and query relatedness. But, moreover, it reflects path semantics, thus, the nodes that can be reached by following paths achieve higher scores. By adapting *PathRank*, we can define the utility function $u(Q, o_i)$ for HGP as in the same way we explained earlier so that we can achieve a top-k recommendation results which reflected more flexible forms of MDR requests.

5. RESULTS

In our previous studies, we aimed to formalize the MDR problem and propose solutions by taking graph-based approaches. So far, we have implemented CG [19] and CBG [20] and performed experiments to validate our methods using several real world datasets gathered from different application domains such as online music streaming services (e.g. last.fm [22], Bugs Music [23]), and a mobile application market (LG Oz Store [24]). One important insight we achieved from the experiments is that our methods can increase recommendation *accuracy* (*HR@k)* [8] by exploiting the contextual information in log tables. Particularly, CBG showed better accuracy than the state-of-art two-dimensional recommendation methods by using time information in logs [20]. To validate the flexibility of our methods, we have identified many interesting use cases of our methods, such as friend recommendation, context-aware recommendation, group recommendation, etc. These promising results convinced us to extend and improve our previous work. By extending the previous work [19][20][21], we proposed our new HGP model in the earlier section. We are currently working on implementing a prototype system for a proof-of-concept and experiments.

6. CONCLUSIONS AND FUTURE WORK

In summary, we first presented a unified framework that can cover general graph-based MDR methods. Then, we introduced two homogeneous graph-based methods CG and CBG. We also outlined a novel MDR method HGP and a node ranking measure *PathRank* that are based on paths in heterogeneous graph. Graph-based MDR methods have significant advantages in that they give much more freedom to express their recommendation queries with various intentions. Especially, we insist that exploiting paths in graphs is significantly important for considering various semantics of user intentions to recommendation process. Preliminary experimental results and identified use cases of our methods show the great potential of graph-based MDR methods. We expect the further research will lead to better user satisfaction. Still, there are remaining interesting subjects that we are planning to tackle in the future work as follow:

Experiments with Respect to Various Criteria: We have compared the recommendation accuracy of our methods with other existing methods. However, recommendation performance can be assessed with respect to, not just accuracy, but also with various other criteria such as serendipity, novelty, and etc [25]. So, we plan to conduct thorough experiments to evaluate our proposed methods with a wide range of recommendation performance metrics. By taking advantage of flexibility, we can adjust many settings for recommendation in our methods. So, it will be meaningful to look into the how the changes in query node types, paths, and weights affect such recommendation metrics. We expect the experimental results will be a meaningful reference for finding good settings in various applications and enhancing our methods.

More Expressive Graph & Path Models: We plan to adapt more expressive graph and path models for our future work. There are many ways to extend the graph and path models, for instance, we can extend each node in heterogeneous graph to have attributes and its values (e.g., a user node can have attributes such as name, age, address, and so on.). Then, this will allow users to define more detailed paths by adding filtering conditions on each step on the paths. For example, a path for a modified collaborative filtering can be defined as:

$$(USER) \xrightarrow{SimilarTaste} (USER)_{gender='Male'} \xrightarrow{Prefer} (SONG)_{yearOfRelease=2011 \wedge genre='Jazz'}$$

Other extensions are possible. However, because path definition should be comprehensive and intuitive for users, we will try to balance the expressiveness and complexity and look for most suitable graph and path models.

Incremental Graph Update: All of our methods need to perform transforming data sources into graph, and it requires full scan on

input data. However, when input data is extremely large and often updated, so, re-building whole graph is tedious and impractical. Thus, we need to develop a method so that it can synchronize input data and constructed graph.

Intuitive and User-friendly Interface: Our goal is to build a unified graph-based recommender system that allows users to flexibly perform various recommendation requests by easily changing various queries and parameters. To realize such system, we need to design an intuitive and user-friendly interface for graph-based MDR system. We expect this will give great opportunities to content providers and consumers in that they can test various recommendation requests to empirically find best recommendation results.

7. ACKNOWLEDGEMENTS

This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MEST) (No. 20110017480).

8. REFERENCES

[1] G. Adomavicius and A. Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. IEEE Transactions on Knowledge and Data Engineering, Vol. 17(6), pp. 734–749, 2005.

[2] H. Stack. Training and testing of recommender systems on data missing not at random. In KDD '10: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2010.

[3] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin. Incorporating Contextual Information in Recommender Systems Using a Multidimensional Approach. ACM Trans. Information Systems, vol. 23, no. 1, Jan. 2005

[4] T. Haveliwala, S. Kamvar and G. Jeh. An analytical comparison of approaches to personalizing pagerank. Technical Report. Stanford. 2003.

[5] Xiaoyuan Su , Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. Advances in Artificial Intelligence. Jan. 2009. http://doi.ac.org/10.1155/2009/421425

[6] M. Deshpande and G. Karypis. Item-Based Top-N Recommendation Algorithms. ACM Trans. Information Systems, Vol. 22, no. 1, pp. 143-177, 2004.

[7] G. Adomavicius and A. Tuzhilin. Multidimensional Recommender Systems: A Data Warehousing Approach. Proc. Second Int'l Workshop Electronic Commerce (WELCOM '01), 2001b.

[8] D. Lee, S. E. Park, M. Kahng, S. Lee, and S.-g. Lee. Exploiting contextual information from event logs for personalized recommendation. In Computer and Information Science, Studies in Computational Intelligence, pp. 121-139. Springer. 2010.

[9] M. Kahng, S. Lee, and S.-g. Lee. Ranking in context-aware recommender systems. In WWW '11: Proceedings of the 20th International Conference Companion on World Wide Web. 2011. http://doi.acm.org/10.1145/1963192.1963226

[10] G. Adomavicius, A. Tuzhilin, and R. Zheng. REQUEST: A query language for customizing recommendations. Information System Research. Information Systems Research, Vol. 22(1), pp. 99-117, 2011.

[11] G. Koutrika, B. Bercovitz, and H. Garcia-Molina. FlexRecs: expressing and combining flexible recommendations. In SIGMOD '09: Proceedings of the 35th SIGMOD International Conference on Management of Data . 2009. http://doi.acm.org/10.1145/1559845.1559923

[12] S. Kamvar , H. Haveliwala, C. D. Manning, and Gene H. Golub. Extrapolation methods for accelerating PageRank computations. In WWW '03: Proceedings of the 12th International Conference on World Wide Web. 2003.

[13] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. In Proceedings of the 9th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA), pp. 668-677, 1998.

[14] F. Fouss, A. Pirotte, J.-M. Renders, and M. Saerens. Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation. IEEE Transactions on Knowledge and Data Engineering, Vol. 19(3), pp. 355-369, 2007.

[15] M. Gori and A. Pucci. ItemRank, A random-walk based scoring algorithm for recommender engines, In 20th International Joint Conference on Artificial Intelligences. 2007.

[16] L. Xiang, Q. Yuan, S. Zhao, L. Chen, X. Zhang, Q. Yang, and J. Sun. Temporal recommendation on graphs via long- and short-term preference fusion. In KDD '10: Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 2010. http://doi.acm.org/ 10.1145/1835804.1835896.

[17] Y. Sun, J. Han, X. Yan, P. S. Yu, and T. Wu. Pathsim: Meta path-based top-k similarity search in heterogeneous information networks. In VLDB' 11, 2011.

[18] A. Balmin, V. Hristidis, and Y. Papakonstantinou. ObjectRank: authority-based keyword search in databases. In VLDB, 2004.

[19] Lee, S., Kahng, M., and Lee, S. Flexible recommendation using random walks on implicit feedback graph. In Proceedings of ICUIMC. 2011, 3-3.

[20] S. Lee, S. Song, M. Kahng, D. Lee, S.-g Lee. Random Walk based Entity Ranking on Graph for Multidimensional Recommendation. Proc. of the 5th ACM Conference on Recommender Systems (RecSys 2011), Page 93-100. 2011.

[21] M. Kahng, S. Lee, and S.-g Lee, Ranking Objects By Following Paths In Entity-Relationship Graphs, Proc. of the 4th PIKM 2011 in conjunction with CIKM 2011, vol , Page 9-16.

[22] Last.fm, http://www.last.fm/

[23] Bugs Music, http://www.bugs.com/

[24] LG U+ Oz Store, http://ozstore.uplus.co.kr/

[25] N. Kawamae. Serendipitous recommendations via innovators. In proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval (SIGIR'10), pages 218-225, 2010.