LSG: A Unified Multi-dimensional Latent Semantic Graph for Personal Information Retrieval

Yang Huangfu[1,2], Kuien Liu[1], Wen Zhang[1,3], Peng Zhou[1], Yanjun Wu[1], Qing Wang[1], and Jia Zhu[4]

[1]Institute of Software, Chinese Academy of Sciences, Beijing, 100190, China
[2]University of Chinese Academy of Sciences, Beijing, 100190, China
[3]State Key Laboratory of Software Engineering of Wuhan University, Wuhan, 430072, China
{huangfuyang,kuien,zhangwen,zhoupeng}@nfs.iscas.ac.cn,
yanjun@iscas.ac.cn, wq@itechs.iscas.ac.cn
[4]School of Computer Science, South China Normal University, Guangzhou, 510631, China
Jia@intelligentforecast.com

Abstract. Traditional desktop search engines can merely support keyword-based search as they don't utilize any other information, such as contextual/semantic information, which has been commonly used in internet search. We observe that *a user usually operates some files to complete a task related to a certain topic and organizes these files in some directories*. Inspired by the observation, we propose an approach that considers three relations among personal files to improve desktop search, namely Topic, Task and Location. Each relation is derived from topics of files, user activities log and hierarchy of file system respectively. The heart of our approach is Latent Semantic Graph (LSG), which can measure the three relations with associated score. Based on LSG, we develop a personalized ranking schema to improve traditional keyword-based desktop search and design a novel recommendation algorithm to expand search results semantically. Experiments reveal that the performance of proposed approach is superior to that of traditional keyword-based desktop search.

Keywords: Latent Semantic Discovery, Graph Model, Information Retrieval.

1 Introduction

Personal Information Retrieval, also known as Desktop Search, aims to search personal data stored in the local disks. In recent years, with the explosion of personal data, desktop search has become a hot topic. In order to pinpoint the resources (files), rich meaningful information is needed to be introduced to retrieval model. In Web search, linkage structure has been extensively studied to improve search performance, such as PageRank and HITS. Nevertheless, there is no direct and explicit association structure in local disks. Intuitively, it seems that personal resources are independent with each other. In fact, implicit associations among personal resources exist extensively. These associations can be further used to improve traditional keyword-based search. We observe that users usually operate PC in a common pattern: Operating

F. Li et al. (Eds.): WAIM 2014, LNCS 8485, pp. 540–552, 2014.

some resources to finish a specific task related to a certain topic, and organizing these resources in some directories. For instance, in order to write a paper about desktop search, I read the references stored in D:/research/literature, write and store this ".docx" in D:/research/paper. This observation inspires us that *topic*, *user behaviors* and *directory structure* are quite useful information for locating resources. We also carried out a well-designed user investigation of 20 skilled researchers. They were asked to select 4 most useful information items for locating local resources in their mind. Fig.1 illustrates the investigation result that strongly supports our observation. Motivated by the upper facts, we propose an approach that exploiting the three kinds of information to improve traditional desktop search. As shown in Fig.2, we denote the three implicit information as {Task, Topic, Location} Relations respectively. The heart of our approach is Latent Semantic Graph (LSG), which is used to measure the three relations with associated score. Based on LSG, we develop a personalized rank schema to improve tradition keyword-based desktop search and design a creative semantic recommendation algorithm to expand the query results.

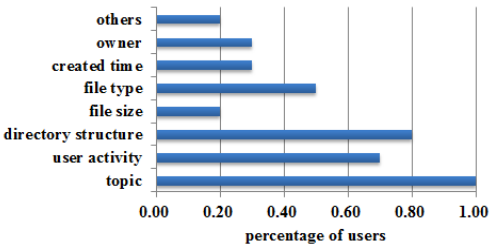


Fig. 1. Result of User Investigation. The user ratio selecting each information item.

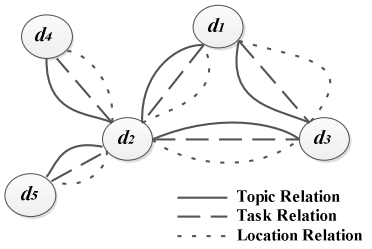


Fig. 2. Overview of LSG with {Topic, Task, Location} Relations between Resources $\{d_1, d_2, d_3, d_4, d_5\}$

Related Work. A number of researches have tried to introduce extra information into Personal Information Retrieval. Chau et al. [8] exploited the explicit metadata to build the contextual cues and designed navigation-style search. They didn't build any ranking algorithms that exploit these links. Other researchers [2, 9] proposed methods to discover the contextual information from file system hierarchy. Peney et al. [9] mapped the authority and hub nodes to resources and directories respectively and then ranked results with authority score. However, this method can't distinguish resources in the same directory. Recently, many researches [7, 10] put their views on user behaviors. Chen et al. [7] built different relations with respect to different user behaviors and assigned the relations of different type with fixed weights. In this paper, we design novel algorithms to measure the relations. Sawyer et al. [3] applied physical-social interactions to build resource associations, but their approach was limited to emails context. Deng et al. [4] built contextual hierarchies for explicit metadata, but they ignored the implicit information like topic, user behavior pattern etc. In summary, none of above work sufficiently utilized the 3 relations mentioned and presented proper methods to measure these relations. Kim et al. [5] combined many

similarity metrics to create association, which is close to our work. However, their approach was strongly dependent on resource tags contributed by users actively. Our approach can measure implicit relations without extra user involvement.

Contribution. The main contribution of this paper includes:

- We design novel methods to measure the {Topic, Task, Location} Relations and propose LSG to integrate them into a unified score. We measure the Topic Relation using a LDA-based method, Task Relation using a Surfing Graph and Location Relation from three aspects as Depth, Transfer Length and Depth Difference of resource organization.
- Based on LSG, we design a new personalized ranking schema, which can reflect user preference and improve keyword-based desktop search results.
- Based on LSG, we design a graph-based recommendation algorithm for query result expansion. We recommend 5 relevant documents for each query result to help user recover memory cues and facilitate search.

The remainder of the paper is organized as follows. In section 2, we propose Latent Semantic Graph. Section 3 discusses how to search using LSG. Section 4 describes the experiments and discusses the results. We conclude the paper and discuss future work in Section 5.

2 Unified Multi-dimensional Latent Semantic Graph

LSG is a multi-dimensional integrated Graph. In this section, we define the {Topic, Task, Location} Relations and present methods to score these relations to propose LSG.

- ***Topic Relation*** is used to measure the topic similarity between two resources (e.g., d_1, d_2). We denote this relation as $Score_{topic}(d_1, d_2)$.
- ***Task Relation*** is used to measure the collaborative strength between two resources in history. We denote this relation as $Score_{task}(d_1, d_2)$.
- ***Location Relation*** is used to measure the relation hidden in the two resources' directory organization. We denote this relation as $Score_{location}(d_1, d_2)$.

2.1 Scoring Topic Relation

In our context, the size of personal resources is varying and large, which results in that term vectors are sparse and high-dimensional. Therefore, cosine similarity between term vectors is not a good way to measure the similarity between resources. In this paper we employ a LDA-based method to measure the similarity of two extractable resources. LDA is a hierarchical Bayesian model which allows us to model a text document as a mixture of topics. After LDA modeling, each resource can be mapped to a latent topic distribution of N dimensions (100 by default). To measure the topic similarity between two resources, we calculate the similarity between the distributions of topics associated with each document, which can effectively resolve the problem

due to data sparsity. We use the KL divergence [1], a non-symmetric measure of the difference between two probability distributions X and Y, to measure the distance between two resources. Given distributions X and Y, the KL divergence between them is formalized as: $D_{kl}(X||Y) = \sum_{n=1}^{N} p(x=n) \log \frac{p(x=n)}{p(y=n)}$.

Considering the non-symmetry of KL divergence, namely $D_{kl}(X||Y) \neq D_{kl}(Y||X)$, we use Jensen-Shannon divergence, a symmetric variant of the KL divergence , instead in this paper: $D_{js}(X||Y) = [D_{kl}(X||M) + D_{kl}(Y||M)]/2$, Where $M = \frac{1}{2}(X+Y)$. Smaller JS value means the two resources are more relevant. We use formula 1 to model the Topic Relation between two resources d_1 and d_2 and obtain a normalized score. The obtained JS value can be viewed as the distance between two resources. The curve of formula 1 is steep near 0 and become flat with the growth of JS value. This characteristic ensures that we can easily distinguish the low similar pairs from the high similar ones.

$$Score_{topic}(d_1, d_2) = 1/e^{D_{js}(d_1||d_2)} \tag{1}$$

2.2 Scoring Task Relation

In order to discover the collaborative relation between different resources, we employ a system-level API to monitor and record resource access activities. The recorded information about an access activity is a 4-tuple {***Name***, ***Start***, ***End***, ***Duration***} (see Fig.3(a)). Here, we define a resource being accessed once the resource gets focus. Users tend to access different resources to complete a purpose. Therefore, we define a ***Task*** as a set of resources for the same purpose. The order of access log in a task reflects the surfing traces of users and collaborative relationships between adjacent resources. Before introducing the task identification algorithm, we first define the concepts used in the algorithm:

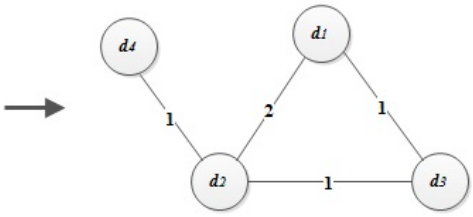
Valid Duration (s)**:** *The duration of a log must be larger than a given Valid Duration threshold (10s by default). This setting is based on a common assumption that users usually focus on a resource for a relatively long time while finishing a task. The invalid one will be remove from the sequence and split the sequence naturally.*
Task Interval (s)**:** *A new task starts while the interval between the adjacent logs is larger than a given Task Interval threshold (600s by default).*
Task Similarity**:** *We merge adjacent tasks T1 and T2 together and view them as the same task if the similarity between T1 and T2 is larger than a given threshold (0.5 by default). We define the task similarity as*

$$\mathrm{Sim}(\mathrm{T1}, \mathrm{T2}) = \frac{|Number\ of\ common\ resources\ in\ two\ task|}{|Total\ number\ of\ resources\ in\ the\ shorter\ task|} \tag{2}$$

Surfing Graph (SG)**:** *We propose a Surfing Graph (SG) to denote each task identified. A surfing graph* G *is an undirected graph. The node set* V_G *corresponds to all resources in the task. There is an edge* $e(u,v) \in E_G$ *for* $u, v \in V_G$ *iff there is an*

access transfer from u to adjacent v. The weight of edge is the transfer frequency happened on this edge regardless of direction.

Name	Start	End	Duration
d_1	08:02:03	08:02:30	27
d_2	08:02:30	08:03:30	60
d_1	08:04:00	08:04:20	20
d_3	08:04:25	08:06:00	95
d_2	08:06:00	08:07:30	90
d_4	08:07:35	08:07:50	15



(a) An Example of Task including 4 resources $\{d_1, d_2, d_3, d_4\}$ (b) Surfing Graph

Fig. 3. An Example of Surfing Graph

Taking Fig.3 for example, we have a task of 4 resources (e.g., d_1, d_2, d_3, d_4). It is obvious that the surfing trace in this task is $d_1 \rightarrow d_2 \rightarrow d_1 \rightarrow d_3 \rightarrow d_2 \rightarrow d_4$. Thus, the transfer pairs are $\{(d_1, d_2), (d_2, d_1), (d_1, d_3), (d_3, d_2), (d_2, d_4)\}$. Each pair corresponds to an edge in Surfing Graph. (d_1, d_2) and (d_2, d_1) correspond to the same edge. Therefore, the weight of each edge is measured by the number of transfer pairs corresponding to it.

Algorithm 1. Construction of Task Relation Graph

Input: A user activity log sequence *R*, Valid Duration *VD*, Task Interval *TI*, Task Similarity *TS*

Output: Task Relation Graph *TARG(V, E)*, where V is vertex set and E is edge set.

1: $V \leftarrow \emptyset, E \leftarrow \emptyset, TaskList_{new} \leftarrow \emptyset$
2: $TaskList_{raw} \leftarrow$ **rawTaskList**(R, VD, TI) //split R according to VD and TI
3: $t \leftarrow TaskList_{raw}[0]$
4: **for**$(i = 1;\ i < TaskList_{raw}.size;\ i++)$ **do**
5: $t' \leftarrow TaskList_{raw}[i]$
6: **if** t', t are splited by VD && $sim(t', t) > TS$ ***then*** //merge similar raw tasks
7: $t \leftarrow t + t'$
8: **else** add t to $TaskList_{new}$ $if\ t.size > 2, t \leftarrow t'$ //remove invalid task
9: **foreach** $t \in TaskList_{new}$ **do**
10: $SG(V_{sg}, E_{sg}) \leftarrow$ **generateSurfingGraph**(t)
11: $V \leftarrow V \cup V_{sg}, E \leftarrow E \cup E_{sg}$
12: **return** $TARG(V, E)$

Algorithm 1 describes the process of task identification (step 1 to 8) and SG merging (step 9 to 12). The identification consists of 3 major steps: (1) split the accessing sequence into clusters according to *Valid Duration* and *Task Interval*; (2) merge similar adjacent clusters (raw tasks) split by *Valid Duration*; (3) remove invalid tasks (size < 3). After identification, we convert all identified tasks to Surfing Graphs and then combine them together as a Task Relation Graph (TARG) where weight of the same edges appearing in the different SG will be added. The TARG reflects the Task Relation between different resources through the weight of edge connecting them. In this

paper, the weight of edge is defined as ***Transfer Frequency (TF)***. Given the TF of two resources d_1 and d_2, we use Eq.(3) to model Task Relation.

For a certain resource X, we are intent to distinguish those low TF resources interacting with it. A small increment on low TF will result in a big difference on task relation strength. When the TF is high enough, the difference on relation strength with the increment of TF will become small. So those high TF resources can be nearly viewed as the same category, namely highly relevant ones to X. In our model, we use TF^2 to enlarge the difference between small TFs and *log* to shrink them towards steep interval. Therefore, the model can reveal the difference between small TFs.

$$Score_{task}(d_1, d_2) = \frac{\log_4(\mathrm{TF}(d_1,d_2)^2+1)}{1+\log_4(\mathrm{TF}(d_1,d_2)^2+1)} \tag{3}$$

2.3 Scoring Location Relation

Empirically, users tend to arrange their information according to a certain implicit relationship among them, such as related content, same target, etc. In this section, we model the Location Relation between resources from 3 aspects mentioned in [12].

Aspect (a): *Depth*. Users usually organize resources into subdirectories. The resources organized in the subdirectories have closer relationship among each other than those in the parent directories. Recursively, the deeper the two resources stored in the hierarchy of file system, the tighter the relationship between them is. We use the average depth of two resources to denote this aspect: $\mathrm{E}(d_1, d_2) = (|d_1| + |d_2|)/2$, where |n| is depth of n.

Aspect (b): *Transfer Length*. Users usually group related resources together. In file system, every resource has a route to every other, which is equivalent to the route between the directories containing the two resources. We refer to the length of the shortest path between two directories as their degree-of-association or transfer-length. Shorter transfer-length means two resources are more related. Thus, if transfer-length is 0, that means the two resources are organized in the same directory. We denote this aspect as equation: $\mathrm{D}(d_1, d_2) = (|d_1| - |\lambda|) + (|d_2| - |\lambda|)$, where λ is their lowest common ancestor-or-self directory.

Aspect (c): *Depth Difference*. Users usually group related directories nearby, which means the depths of these directories are close. Under prerequisite of large *Depth* and short *Transfer Length*, if the depths of directories containing the given two resources are closer, the two resources are more relevant. We denote this aspect as equation: $\mathrm{M}(d_1, d_2) = \big||d_1| - |d_2|\big|$.

By integrating the above aspects, we now obtain the following formula which models Location Relation of 2 resources and returns a normalized score. According to the importance of each aspect, the function penalizes *Transfer Length* and *Depth Difference* harshly and *Depth* leniently. In fact, users won't organize their **valuable** resources at deep level, generally 4-5 level at most. So the influence of *Depth* is limited.

$$Score_{location}(d_1, d_2) = \frac{\sqrt{E(d_1,d_2)}}{1+D(d_1,d_2)^2+M(d_1,d_2)^2+\sqrt{E(d_1,d_2)}} \quad (4)$$

2.4 Score Aggregation with Linear Regression

After measuring the three relations, we can obtain 3 Relation Graphs. We name them as Topic Relation Graph, Task Relation Graph and Location Relation Graph. The final **Latent Semantic Graph (LSG)** we proposed is a mixture graph to integrate the above 3 Relation Graphs. In this paper, the final association score, namely, the weight of edge in LSG is simply formulized as a linear combination:

$$Score_{LSG}(d_1, d_2) = \rho Score_{topic}(d_1, d_2) + \varphi Score_{task}(d_1, d_2) + \omega Score_{location}(d_1, d_2) \quad (5)$$

Therefore, the parameters ρ, φ, ω can be estimated by a linear regression. We conduct a training set including K (K=200, 40 pairs per person) resource pairs $<d_1, d_2>$ selected from data set provided by participants of our experiments. We ask participants to rate those pairs selected from their own data set with a grade between 0 and 1. Suppose the training set is $T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_K, y_K)\}$, where x_i is triple including 3 relation scores and y_i is relation score rated by user. So we estimate the parameters by solve the following optimization problem.

$$\min_W \frac{1}{2}\sum_{i=1}^{K}(y_i - W^T x_i)^2, \quad W = (\rho, \varphi, \omega), \ \text{s.t. } \rho + \varphi + \omega = 1 \quad (6)$$

With the limitation of space, we omit some detail about learning process. After a 5-fold cross validation, we average the parameters of 5 models and obtain the approximate parameters as (0.3, 0.6, 0.1) in our context. With the estimated parameters, we combine the three relation graphs and generate the Latent Semantic Graph (LSG).

3 Searching with LSG

When user submits keywords, the content-based search module firstly finds results from index and then feeds them to LSG module. LSG module resorts these results with a personalized ranking schema and gets recommended resources for each result.

Fig. 4. Process of Searching with LSG

Personalized Ranking. In case of Web search, personalized PageRank has been widely used. The original PageRank can be expressed as the solution: $\vec{R}_k = d * M^T \times \vec{R}_{k-1} + (1 - \mathrm{d}) * \vec{E}$. Here, we design a LSG-based Personalized Ranking Schema to rank the results by rebuilding the Transition Matrix M and personaliza-

tion vector $\vec{E}$. Generally, user would like to jump to the node which is highly relevant to the current one. Let N_j be the set of resources which the resource j links to in LSG. Then when computing the transfer probability from j to any one in N_j, e.g. resource i, we use the following formula:

$$P(j \to i) = rM_{ji} = \frac{Score_{LSG}(j,i)}{\sum_{k \in N_j} Score_{LSG}(j,k)} \tag{7}$$

To rebuild $\vec{E}$, let L be the set of resources recorded in user activity log, o_i be the number of occurrences of resource i in the user activity log and N_{index} be the number of indexed resources. If resource i belongs to L, e.g., $o_i \neq 0$, we set $e_i = o_i / \sum_{j \in L} o_j$, otherwise, $e_i = 1/N_{index}$. The final query score is a combination of TF-IDF score and personalized ranking score. Suppose q is the given query and p is the resource in our disk, then final ranking score: $\text{Score}_q(p) = \text{Score}_{cosine}(q,p) * \text{Score}_{PR}(p)$, where $\text{Score}_{cosine}(q,p)$ is cosine similarity between q and p, $\text{Score}_{PR}(p)$ is the importance score of p calculated by personalized ranking schema.

Semantic Recommending. Based on LSG, graph-based recommendation methods can be introduced to compute the resource association from a global perspective instead of local pairwise computation of neighborhood. We transplant a graph-based recommendation algorithm IPF [6] into our scenario to recommend semantically-relevant resources for each query result. We name the transplanted IPF as ***TIPF*** in this paper. Given a query result u and a unknown resource i, there are many propagation paths from u to i in LSG. For u, the recommending score of i is sum of weights of all paths. The path weight can be viewed as the visited probability of i from the resource u. Suppose $\text{P}\{\text{d}_0, \text{d}_1, \ldots, \text{d}_n\}$ is the path from the query result u $(\text{d}_0 = u)$ to resource i $(\text{d}_n = i)$, the path weight is defined as: $\psi(\text{p}) = \prod_{k=0}^{n-1} P(d_k \to d_{k+1})$, where $P(d_k \to d_{k+1})$ is a propagation function defined in Eq.(7). Similarly with IPF, we only consider the short paths (distance < 3) to measure the visited probability because the long path contributes little and is prone to bring in noise. Consequently, suppose $\Gamma(u,i)$ is the set of short paths from u to i, then the recommending score of i to u is defined as:

$$Score_{recommend}(\text{u},\text{i}) = \sum_{p \in \Gamma(\text{u},\text{i})} \psi(\text{p}) \tag{8}$$

The ***TIPF*** is implemented by Bread-First-Search on LSG. Finally, we sort the candidate resources according to $Score_{recommend}(\text{u},\text{i})$ and then return the top 5 resources to u. Fig. 8 illustrates an example of semantic recommendation.

4 Experiments

In order to evaluate the effectiveness of LSG in enhancing result ranking and semantic recommendation for query result expansion, we develop a prototype, called IDSE[1], to implement the searching process with LSG and construct a small scale user study. Because the experiment on personal data is sensitive, the scale of experiment is hard to be large. Referring to the experimental scale mentioned in references (e.g, 3 in [5], 5 in [7] and 6 in [10] etc.), we invited five **volunteers** in our institution to participate in our experiments.

Table 1. An Overview of Data set

User	*U1*	*U2*	*U3*	*U4*	*U5*
Data Size (GB)	10.27	3.08	2.38	15.26	0.81
Accessed Resources	926	830	193	508	234
Extractable Resources	2059	4765	1817	3311	349

Data Set. To gather the activities log, we trace participants' behaviors on resources for two months. The average data set contains *40346* resources in *6783* directories. The average directory depth is *10* (max *23*). In addition, there are totally *12301* extractable resources and *2691* resources are recorded in the user log.

Metrics. We take the traditional metrics $\text{Precision} = \frac{|\{retrieved\ relevant\ documents\}|}{|\{retrieved\ documents\}|}$ and $\text{Recall} = \frac{|\{retrieved\ relevant\ documents\}|}{|\{relevant\ documents\}|}$ to evaluate the IR performance.

4.1 Experimental Results

Ranking Performance Evaluation

Comparison. We compare IDSE with two of the state-of-art [11] Desktop Search tools: Copernic Desktop Search [13] and Google Desktop Search (GDS) [14]. In addition, we provide two versions of prototype: IDSE-based (content-based ranking only) and IDSE_Imecho (a similar method in [7], which uses fixed relation weight).

Setup. Each participant is asked to design 10 search queries related to their activities and then send each of queries to the 5 tools respectively. For each query, each participants rate the top 10 results for each tools using grades {0, 1, 2, 3, 4, 5}[2] where 0 for an irrelevant result and 5 for a highly relevant one. At each rank, the average precision and recall can be calculated by using the grades rated by our participants. For each tool, we can calculate the average precision and recall of 50 queries at any rank *k*. We use the pooling technology [7] to generate the relevant resources set.

Fig. 5(a) and Fig. 5(b) depict the average precision and recall levels of the 5 tools at each rank from one to ten respectively. Intuitively, the prototype IDSE with LSG

[1] We share the project of IDSE at GitHub: `https://github.com/HarryHuang1990/idse.git`

[2] The relevant grade will be normalized to [0, 1] when calculating the measure.

outperforms others on both metrics at every rank level. This indicates that IDSE finds more relevant resources and ranks them higher than content-based search tools (GDS, Copernic and IDSE-based). The interpretation of this result is that the GDS, Copernic and IDSE-based are only concerned about hitting key-words and rank their results using TF-IDF models, regardless of any global importance measures for resources. Whereas with semantic links modeled in LSG, IDSE can exactly measure importance for every resource and push resources of interest towards the top of list.

IDSE also outperforms IDSE_Imecho on both metrics. This indicates that exact measurement of relation is better for desktop search than using fixed weight and our approach is effective. IDSE-based is implemented by using an optimized TF-IDF model. Therefore it can find more relevant resources than GDS and Copernic. But it may not rank the high relevant resources towards the top and cause a flat precision curve from top 6 to 10 (depicted in Fig. 5(a)). After adding the LSG, those high relevant but low ranked resources will be ranked high. This results in a visible improvement in terms of precision and recall.

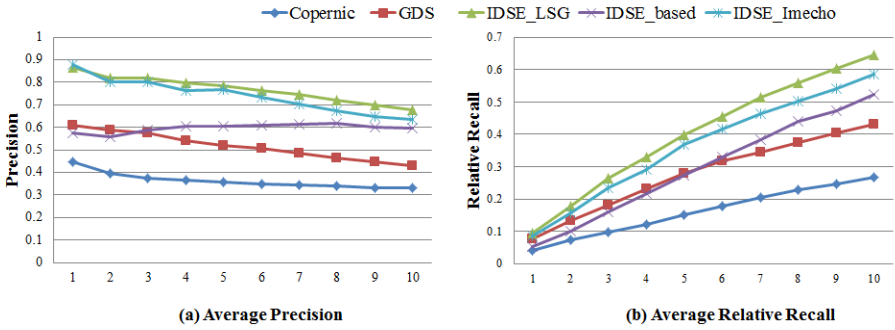


Fig. 5. Comparison of Average Precision and Average Relative Recall from Top 1 to 10

Effectiveness of Semantic Recommendation

Recommendation for each query result is a creative attempt. It can help user to recall the memory cues among resources. The performance of the recommendation mainly relies on users' judgment. We ask each volunteer to rate the each recommended resource for top 5 results returned by IDSE using the same grade method as above. The relevant grade reflects the user satisfaction to the recommended one. Finally, we average the all satisfaction grades in 10 queries to measure a user's satisfaction to our method.

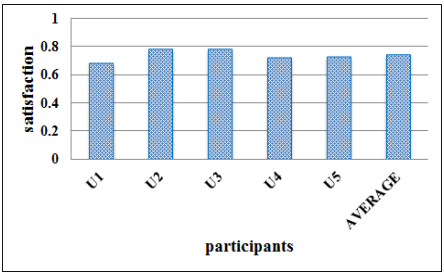


Fig. 6. Average User Satisfaction to Semantic Recommendation for Query Result Expansion.

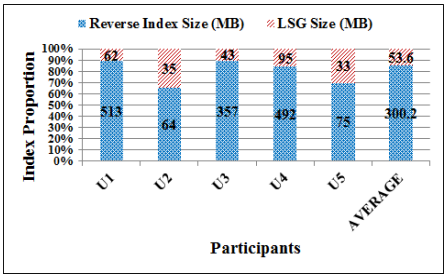
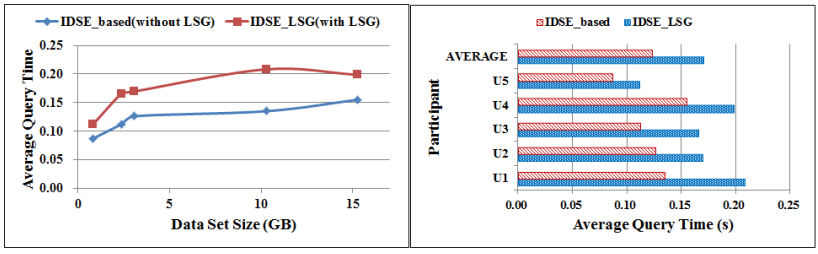


Fig. 7. Performance on Indexing Space

Fig. 6 illustrates the average assessment result with a five score system where [0, 0.2] for "Very Poor", (0.2, 0.4] for "Poor", (0.4, 0.6] for "Not too Poor", (0.6, 0.8] for "Acceptable", and (0.8, 1] for "Perfect". As we see, all participants feel that our semantic recommendation is "acceptable". Although without an acknowledged evaluation method, such user measure can, to some extent, demonstrate the effectiveness of this approach. Fig. 8 shows an example of Semantic Recommendation on query result "WAIM2014-IDSE paper.docx". 5 resources recommended are all related to the topic "desktop search". In other examples, the recommended resources may be task-related or location-related.

Fig. 8. An Example of Semantic Recommendation for Query Result Expansion



(a) Average Query Time with the Increasment of Data Set Size

(b) Query Time Comparison between IDSE_based (without LSG) and IDSE_LSG (with LSG)

Fig. 9. Performance on Query Time

Scalability

Querying Performance. In Fig. 9(b), the average query time of IDSE across all queries (50 queries) is 0.17s, which is much less than one second. Query time averages are calculated across 5 runs of each query. Fig. 9(a) shows that with the data set size increasing, the average query time appears to grow slowly, especially on big data sets. Although, the average time of search with LSG is a little longer than content-search alone (0.12 on average), it is acceptable. Even though compared with GDS, the participants agree that the performance is also acceptable.

Indexing Space. The IDSE's index consists of reverse index and LSG. Fig. 7 shows the space required by the LSG and reverse index for each data set and on average. This result indicates that the LSG is not a barrier to search. On average, the LSG size is less than 0.85% of the size of the user's data size.

5 Conclusion and Future Work

In this paper, we attempt to exploit {Topic, Task, Location} Relations to improve traditional desktop search performance. We design novel methods to measure these 3 relations properly using information derived from resource latent topics, user access activities and directory hierarchy respectively and then propose LSG, a multi-dimension integrated graph, to integrate the 3 relations using a linear combination. With LSG, a personalized ranking schema is designed to enhance the full-text keyword desktop search and the recommending algorithm TIPF is adapted to help user extend and associate the results returned by the keyword search. Experiments reveal the prototype embedding LSG is superior to the traditional keyword-search desktop search tools. In future work, we are intent to explore Web-behavior-based topic semantic identification for those unextractable resources.

Acknowledgements. We would like to thank Dr. Xianpei Han for providing valuable suggestions. This work is supported by the NSF of China (Nos. 71101138, 61202064 and 91324008), Beijing NSF (No. 4122087) and the Strategic Priority Research Program of the Chinese Academy of Sciences (No. XDA06010600).

References

1. Cover, T., Thomas, J.: Elements of information theory. John Wiley & Sons (2012)
2. Wang, W., Peery, C., Marian, A., Nguyen, T.: Efficient Multidimensional Fuzzy Search for Personal Information Management Systems. TKDE 24(9), 1584–1597 (2012)
3. Sawyer, B., Quek, F., Wong, W.C.: Using physical-social interactions to support information re-finding. In: CHI, pp. 885–910 (2012)
4. Deng, T., Zhao, L., Feng, L., Xue, W.: Information re-finding by context: a brain memory inspired approach. In: CIKM, pp. 1553–1558 (2011)
5. Kim, J., Croft, W.B., Smith, D., Bakalov, A.: Evaluating an associative browsing model for personal information. In: CIKM, pp. 647–652 (2011)
6. Xiang, L., Yuan, Q., Zhao, S., Chen, L.: Temporal recommendation on graphs via long- and short-term preference fusion. In: SIGKDD, pp. 723–732 (2010)

7. Chen, J., Guo, H., Wu, W., Wang, W.: iMecho: an associative memory based desktop search system. In: CIKM, pp. 731–740 (2009)
8. Chau, D., Myers, B., Faulring, A.: What to do when search fails: finding information by association. In: CHI, pp. 999–1008 (2008)
9. Penev, A., Gebski, M., Wong, R.K.: Topic distillation in desktop search. In: Bressan, S., Küng, J., Wagner, R. (eds.) DEXA 2006. LNCS, vol. 4080, pp. 478–488. Springer, Heidelberg (2006)
10. Soules, C., Ganger, G.: Connections: using context to enhance file search. SOSP 39(5), 119–132 (2005)
11. Noda, T., Helwig, S.: Benchmark study of desktop search tools.University of Wisconsin-Madison E-Business Consortium (2005)
12. Ravasio, P., Schär, S.G., Krueger, H.: In pursuit of desktop evolution: User problems and practices with modern desktop systems. TOCHI 11(2), 156–180 (2004)
13. Copernic desktop search, `http://www.copernic.com/en/products/desktop-search/`
14. Google desktop search, `http://desktop.google.com/`