# Diversifying Query Suggestion Results

**Hao Ma** and **Michael R. Lyu** and **Irwin King**
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong
{hma, lyu, king}@cse.cuhk.edu.hk

**Abstract**

In order to improve the user search experience, *Query Suggestion*, a technique for generating alternative queries to Web users, has become an indispensable feature for commercial search engines. However, previous work mainly focuses on suggesting relevant queries to the original query while ignoring the diversity in the suggestions, which will potentially dissatisfy Web users' information needs. In this paper, we present a novel unified method to suggest both semantically relevant and diverse queries to Web users. The proposed approach is based on Markov random walk and hitting time analysis on the query-URL bipartite graph. It can effectively prevent semantically redundant queries from receiving a high rank, hence encouraging diversities in the results. We evaluate our method on a large commercial clickthrough dataset in terms of relevance measurement and diversity measurement. The experimental results show that our method is very effective in generating both relevant and diverse query suggestions.

## 1 Introduction

As the rapid expanding of information on the Web, it has become increasingly difficult for search engines to satisfy Web users' information needs. How to organize and utilize the Web information effectively and efficiently has become more and more critical. In order to enhance the user experience, *Query Suggestion*, a technique to generate alternative queries, has long been proved useful to help a user explore and express his/her information need. It plays an indispensable role in improving the usability of search engines, and it has been well-studied in academia and widely adopted in industry.

However, no matter what kind of methods are employed, previous efforts mainly focus on presenting users with the most relevant results based on some well-defined criteria. Actually, in query suggestion, only providing the "good" suggestions is far away from satisfying users' information needs. The top presented suggestions should also be semantically different from each other in order to have a broad coverage of the latent topics due to the following considerations. Queries may contain ambiguous terms which will confuse these approaches, producing recommendation results which do not match users' intents. For example, when a user wants to find some related information about "Apache Helicopter", and types a query "Apache" to the search engines, most probably, this user will be provided with suggestions like "Apache Web Server", "Apache Tomcat", "Apache Install", etc., which drift a lot from this user's information need. Moreover, users tend to submit short queries consisting of only one or two terms under most circumstances, and short queries are more likely to be ambiguous. Furthermore, in most cases, users have little or even no knowledge about the topics they are searching for, hence cannot clearly present their information needs by several query words. In order to find satisfactory answers, the users have to rephrase their queries constantly.

All the above evidences indicate that the suggestion results require not only relevant but also diverse choices. Furthermore, if we consider the circumstance in search engine advertising business, diversifying the query suggestion results can also benefit the customers who bid for query terms. The highly relevant queries cost more, while the diversified query suggestions which are also relevant cost less. This indicates that the well-defined diversified query suggestion results can boost other related queries, which provides a huge opportunity to maximize the benefits for both a search engine company and the customers of its advertising system.

In this paper, we propose a unified framework for generating relevant and diverse query suggestions by employing a Markov random walk and an iterative hitting time analysis on the query-URL clickthrough data. The proposed approach presents a novel solution to diversify the query suggestion results, which has the following advantages: (1) The suggested queries generated from our approach are semantically relevant to the original query. (2) Every highly ranked query is a representative of a local group or a latent topic related to the original query, hence the top suggested queries are diversified. (3) The proposed method is quite general, which not only can be applied to different graphs instead of clickthrough bipartite graph, but also can provide potential solutions to many other Web applications, such as document retrieval, image retrieval, tag recommendation, etc. The basic idea of our method is: (1) We first define a Markov random walk on the query-URL bipartite graph with the top ranked query added into a candidate set. (2) Other queries are then ranked by an algorithm which conducts a few rounds of the expected hitting time analysis. This algo-

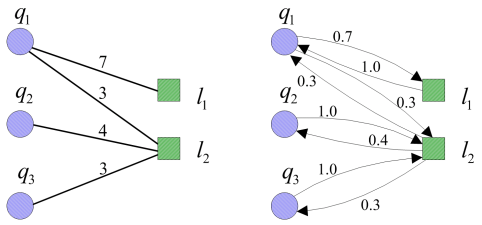



**Figure 1: Example for Graph Construction**

rithm actually will bring down the ranks of similar unranked queries, thus encouraging diversities in the suggestion results.

We evaluate our model for query suggestion using clickthrough data of a commercial search engine. We measure the performance from different aspects. In terms of the relevance measurement, we design two measurement methods to both automatically and manually evaluate the relevance between the original query and the suggested queries, where the results are assessed by the ground truth extracted from the ODP[1] (Open Directory Project) database and a panel of three experts. On the other hand, in order to judge the quality of diversity, we propose a novel metrics to examine the diversities among the suggested queries. The evaluation results show that our method is effective for generating both semantically related and diversified queries to the users.

## 2 Diversifying Suggestions

In this section, we introduce how to employ random walks and hitting time analysis to diversify the query suggestions using the clickthrough data.

### 2.1 Graph Construction

From a statistical point of view, in clickthrough data, the query word set corresponding to a Web page contains human knowledge on how the pages are related to their issued queries (Sun et al. 2005). Thus, in this paper, we utilize the relationships between queries ($q$) and Web pages ($l$) for the construction of bipartite graphs containing two types of vertices $\langle q, l \rangle$.

We cannot simply employ the bipartite graph extracted from the clickthrough data into our model, since this bipartite graph is an undirected graph, and cannot accurately interpret the relationships between queries and URLs. Fig. 1(a) illustrates a toy example. The values on the edges in Fig. 1(a) specify how many times a query is clicked on a URL. Hence, we convert this bipartite graph into Fig. 1(b). In this converted graph, every undirected edge in the original bipartite graph is converted into two directed edges. The weight on a directed query-URL edge is normalized by the number of times that the query is issued, while the weight on a directed URL-query edge is normalized by the number of times that URL is clicked.

### 2.2 Determining the First Suggested Query

In this section, we introduce how to determine the first suggested query by employing a Markov random walk.

[1]http://www.dmoz.org

Let $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ denote a directed graph, where $\mathcal{V} = \mathcal{Q} \cup \mathcal{L}$ is the vertex set, and $\mathcal{Q}$ represents the set of query nodes while $\mathcal{L}$ denotes the set of URL nodes. $\mathcal{E}$ is the edge set which consists of edges between two types of vertices.

For all the edges in the edge set $\mathcal{E}$, we define the initial transition probability $\widetilde{P}_0(i,j)$ from node $i$ to node $j$ as

$$\widetilde{P}_0(i,j) = \frac{C_{ij}}{\sum_p C_{ip}}, \tag{1}$$

where $C_{ij}$ is the number of click frequency between node $i$ and node $j$. If $i \in \mathcal{Q}, j \in \mathcal{L}$, the normalization term $\sum_p C_{ip}$ is the total number of times that the query node $i$ has been issued in the dataset. If $i \in \mathcal{L}, j \in \mathcal{Q}$, then the normalization term $\sum_p C_{ip}$ is the total number of times that the URL node $i$ has been clicked in the dataset. The notation $\widetilde{P}_0(i,j)$ denotes the initial transition probability from node $i$ to node $j$. While the counts $C_{ij}$ are symmetric, the transition probabilities $\widetilde{P}_0(i,j)$ generally are not, because the normalization varies across different nodes.

The random walk can then diffuse using the transition probability defined above. In fact, in addition to the transition probability, there are random relations among different queries, even if these queries are unrelated in their literal meaning: People of different cultures, genders, ages, and environments may implicitly link these queries together, but we do not know these latent relations. To capture these relations, we add a uniform random relation among different queries. More specifically, let $\alpha$ denote the probability that such phenomena happen, and $(1-\alpha)$ is the probability of taking a "random jump". Without any prior knowledge, we set $\mathbf{d} = \frac{1}{n}\mathbf{1}$, where $\mathbf{d}$ is a uniform stochastic distribution vector, $\mathbf{1}$ is the vector of all ones, and $n$ is the number of queries. Based on the above consideration, we modify the transition probability to

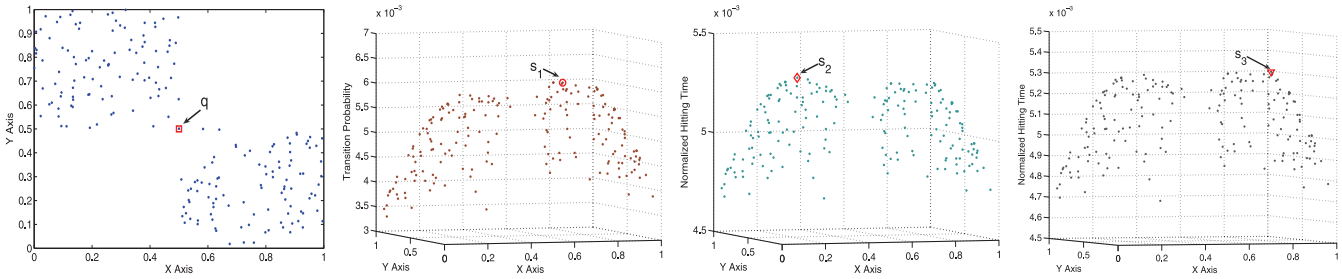
$$\mathbf{P} = \alpha\widetilde{\mathbf{P}} + (1-\alpha)\mathbf{d}\mathbf{1}^T, \tag{2}$$

where matrix $\mathbf{P}$ is the transition probability matrix with the entry of the $i$th row and $j$th column defined in Eq. (1). Following the setting of $\alpha$ in PageRank (Eiron, McCurley, and Tomlin 2004), we set $\alpha = 0.85$ in all of our experiments conducted in Section 3.

With the transition probabilistic matrix $\mathbf{P}$ defined using Eq. (2), we can now perform the random walk on the query-URL graph. We calculate the probability of transition from node $i$ to node $j$ after a $t$ step random walk as:

$$P_t(i,j) = [\mathbf{P}^t]_{ij}. \tag{3}$$

The random walk sums the probabilities of all paths of length $t$ between the two nodes. It gives a measure of the *volume of paths* between these two nodes; if there are many paths the transition probability will be high (Craswell and Szummer 2007). The larger the transition probability $P_t(i,j)$ is, the more the node $j$ is similar to the node $i$. Hence, in the query suggestion task, for a given query node $q \in \mathcal{Q}$, after performing a $t$-step random walk, we treat the node $s_1 \in \mathcal{Q}$ with the largest transition probability from node $q$ as the first suggested query, which is highly semantically relevant to the starting query node.




(a) (b) (c) (d)

**Figure 2: A Toy Example Showing How to Diversify the Suggestion Results.** (a) A toy dataset containing 200 data points. (b) The transition probability distribution after a 5-step random walk starting from node $q$. The data point with the largest probability is selected as the first suggestion $s_1$. Both $q$ and $s_1$ are added into a subset $\mathcal{S}$. (c) The normalized hitting time distribution to the subset $\mathcal{S}$. The data point with the largest hitting time is selected as the second suggestion $s_2$. Note the diversity in $s_1$ and $s_2$ since they come from different clusters. (d) After another round hitting time analysis, the data point with the largest hitting time is selected as the third suggestion $s_3$.

The parameter $t$ determines the resolution of the Markov random walk. If we choose a large $t$, the random walk will turn into a stationary distribution, where the final results depend more on the graph structure with little information about the starting query node. On the other hand, a short walk preserves information about the starting node at a fine scale. Since we wish to preserve the information about the starting query node, we need to choose a relatively small $t$, so that we are far from the stationary distribution.

### 2.3 Ranking the Rest Queries

After the selection of the first suggested query, as mentioned early, we then employ the hitting time to rank and diversify the rest of the queries.

Let $\mathcal{S}$ be a subset of vertex set $\mathcal{V}$, the expected hitting time $h(i|\mathcal{S})$ of the random walk is the expected number of steps before node $i$ is visiting the starting set $\mathcal{S}$. It can be easily verified that the hitting time satisfies the following system of linear equations:

$$\begin{cases} h(i|\mathcal{S}) = 0, & i \in \mathcal{S}, \\ h(i|\mathcal{S}) = 1 + \sum_{j \in \mathcal{N}(i)} P_0(i,j) h(j|\mathcal{S}), & i \notin \mathcal{S}, \end{cases} \tag{4}$$

where $\mathcal{N}(i)$ denotes the neighbors of node $i$. The recurrence relations in Eq. (4) can be used to iteratively compute the expected hitting time. We ignore the inference of the hitting time equation in this paper due to the space limitation. The meaning of these formulae is: In order to jump from node $i$ to the subset $\mathcal{S}$, one has to go to any adjacent state $j$ of node $i$ and proceed from there. The hitting time from node $i$ to node $j$ has the property of decreasing when the number of paths from $i$ to $j$ increases and the lengths of the paths decrease. This is a desirable property for diversifying the results since it brings down the values of very similar nodes of the targeting node. In the case of the query-URL bipartite graph, if the hitting time from a query node $i$ to the query node $j$ is small, it means the query $j$ is tightly relevant to the query $i$, which indicates that these two nodes probably belong to the same latent topic.

In Section 2.2, given a query $q \in \mathcal{Q}$, we extract the first suggestion node $s_1 \in \mathcal{Q}$ by conducting a $t$-step random walk. Intuitively, as mentioned above, those nodes strongly connected to $s_1$ will have many fewer visits by the random walk, because the walk tends to hit $s_1$ soon. In contrast, groups of nodes far away from $s_1$ still allow the random walk to move among them, and thus receive more visits. Hence, we select the second suggestion node $s_2 \in \mathcal{Q}$ with the largest expected hitting time to the subset $\mathcal{S}$ containing two nodes $q$ and $s_1$. This naturally inhibits queries closely connected to $q$ and $s_1$, hence encourages diversity. If we need the third suggestion, we can simply add $s_2$ into the subset $\mathcal{S}$, and compute the expected hitting time once more.

The complete algorithm for diversifying query suggestion is summarized in Algorithm 1.

**Algorithm 1**: Diversifying Query Suggestion Algorithm

**Input**: Transition probability matrix $\mathbf{P}$ of a click bipartite graph $\mathcal{G}$ and a query node $q \in \mathcal{Q}$

**Output**: A ranked list of $K$ suggested queries

1: Given the query node $q$, form a $1 \times n$ vector $v$ ($n$ is the total number of nodes), with the entry of query node $q$ equal to 1 while other entries equal to 0.

2: Perform a $t$-step random walk and get a new vector $v^* = v\mathbf{P}^t$. Get the top query node $s_1$ with the highest value in vector $v^*$ (notice that, this node should be in the query node set $\mathcal{Q}$).

3: Add query nodes $q$ and $s_1$ into a subset $\mathcal{S}$, and repeat the following statements until $K$ queries are suggested:

(a) For all queries except the ones in the subset $\mathcal{S}$, iterate

$$h_l(i|\mathcal{S}) = 1 + \sum_{j \in \mathcal{N}(i)} p_0(i,j) h_{l-1}(j|\mathcal{S})$$

for a predefined number of $l$ iterations starting with $h_0(i|\mathcal{S}) = 0$.

(b) Pick next suggested query as $\text{argmax}_i\ h_l(i|\mathcal{S})$, and add it into the subset $\mathcal{S}$.

In order to show the intuition of our method more directly, we create a toy dataset with 200 data points in Fig. 2(a). Roughly, there are totally two clusters in this dataset. We then create a fully connected graph on the data, with larger edge weights if points are closer to each other, using the following weighting scheme $w_{ij} = \exp(-||\mathbf{v}_i - \mathbf{v}_j||/0.16)$, where $\mathbf{v}_i$ an $\mathbf{v}_j$ represent two-dimensional vectors.

After a 5-step random walk starting at node $q$, the tran-

**Table 1: Query Suggestion Comparisons between DQS and other Methods**

| Query = nike | | | Query = mp3 | | |
|---|---|---|---|---|---|
| **DQS** | FRW | BRW | **DQS** | FRW | BRW |
| nike shoes | nike shoes | nike clothing | winamp | winamp | best mp3 player |
| nike yoga | nike shox | shoes nike | aol music | mp3 players | mp3 files |
| jordans | niketown | nike sandals | walmart music | free music downloads | lame mp3 |
| basketball shoes | nikes | nikes | mp3 download | limewire | lame |
| nike id | nike outlet | nike logo | creative | free music | lame mp3 encoder |
| ronaldinho | nike town | nike clothes | cdnow | sound click | lame encoder |
| **Query = sony** | | | **Query = nikon** | | |
| **DQS** | FRW | BRW | **DQS** | FRW | BRW |
| sony psp | sony psp | sony singapore | nikon cameras | nikon cameras | nikon products |
| pictures | psp | sony customer service | canon | nikon usa | nikonusa |
| ps3 | sony tv | sony corporation | ritz camera | nikon camera | nikon lenses |
| sony ericsson | sony vaio | sony vaio drivers | rifle scopes | nikonusa | nikon usa |
| psp games | sony style | sony e | nikon d300 | nikon lenses | nikon digital slr |
| sony canada | sonystyle | sony service | wolf camera | ritz camera | nikon digital |

sition probability distribution from node $q$ is shown in Fig. 2(b). We can observe that the transition probability distribution does not address diversity at all. If we need 5 suggestions, then the suggestion results will be dominated by the right part data points of Fig. 2(b). Therefore, we only use the transition probability to find the first suggestion $s_1$, which is annotated by a red circle in Fig. 2(b).

We then add the nodes $q$ and $s_1$ into a subset $\mathcal{S}$, and calculate the expected hitting time to the subset $\mathcal{S}$. The normalized hitting time distribution is shown in Fig. 2(c). We can observe that adding node $s_1$ into the subset $\mathcal{S}$ drops down the popularity of the right part of queries. The left part of nodes related to another topic which have higher hitting time values pop up. We can now select the data point marked as the red diamond as the second suggestion $s_2$, which obtains the largest hitting time in Fig. 2(c). If need the third suggestion, we can simply add $s_2$ to into subset $\mathcal{S}$, and start another round of hitting time analysis as shown in Fig. 2(d). We can see that our method naturally suggests both relevant and diverse queries without explicit data clustering.

Since the clickthrough data size is normally huge, performing Algorithm 1 may be time-consuming. Based on the intuition that nodes which are far away from the original node are less likely similar with this node, we can first create a subgraph around the original node, then perform Algorithm 1 on this subgraph.

# 3 Experimental Analysis

In this section, we will conduct several experiments to measure the effectiveness of our proposed query suggestion framework.

## 3.1 Data Description

We construct our dataset based on the clickthrough data of a commercial search engine. In summary, there are a total of 20 millions lines of clickthrough information, 7,288,746 unique queries, and 7,016,897 unique URLs.

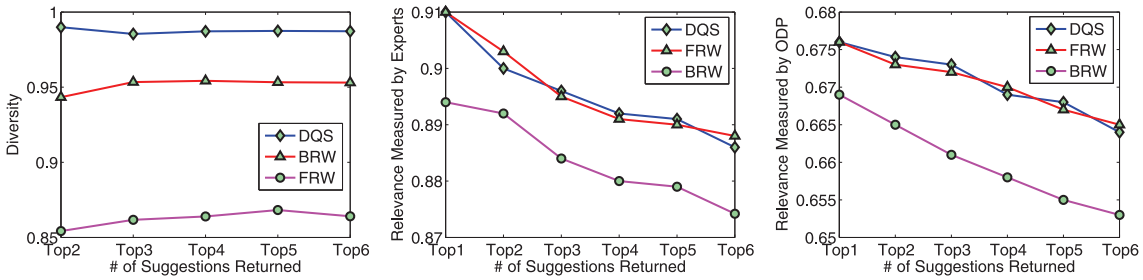
This dataset is the raw data recorded by the search engine, and contains a lot of noises which will potentially affect the effectiveness of our query suggestion algorithm. Hence, we conduct a similar method employed in (Wang and Zhai 2007) to clean the raw data. We process the data by only keeping those frequent, well-formatted, English queries (queries which only contain characters 'a', 'b', ..., 'z', numbers '0', '1', ..., '9' and space, and appear more than 3 times). After removing duplicates and cleaning, we get a total of 774,506 unique queries and 2,247,666 unique URLs in our data collection. We also observe a number of power-law distributions in our datasets, including clicks per query and clicks per URL.

## 3.2 Query Suggestion Results

Before presenting the query suggestion results, let us first discuss the settings of parameters $t$ and $l$, where $t$ is the random walk steps while $l$ denotes the iterations in calculating the hitting time. As discussed in Section 2.2, in order to keep the information of the original query, we should set $t$ to a relatively small value. In all of the experiments conducted in this paper, we empirically set $t = 40$. As to $l$, we set $l = 20$ in all the experiments since it is enough for solving the hitting time linear equations in our dataset. The solving process is very efficient due to the linear properties of Eq. (4).

We show some suggestion result examples generated by our Diversifying Query Suggestion (DQS) algorithm in Table 1. We also compare our results with Forward Random Walk (FRW) method and Backward Random Walk (BRW) method which are proposed in (Craswell and Szummer 2007).

From Table 1, given a query, we can observe that our suggestion results cover much more diverse topics than those suggestions generated by FRW and BRW. If the query is "nikon", for example, our algorithm can suggest the "nikon cameras" which is the most relevant query to "nikon", and can also suggest "canon", which is a strong competitor of Nikon company. It can also suggest different types of products of "nikon", such as "nikon d300" and "rifle scopes", where the former one is a type of nikon digital single-lens reflex camera while the latter is one type of lens for hunting. However, in the results suggested by FRW and BRW, in the top results, some of them represent the same topic, which are redundant. Another example is the query "nike".




(a) Diversity (b) Relevance Measured by Experts (c) Relevance Measured Using ODP

**Figure 3: Experimental Analysis**

In addition to the very relevant and diverse suggestions, like "nike shoes", "nike yoga", "jordans", "nike id", ect., our DQS algorithm can also suggest "ronaldinho", who is a famous soccer star as well as the celebrity of Nike company's advertising strategies. All of these examples show that our algorithm can suggest both semantically relevant and diverse queries to users while FRW and BRW are generally lack of diversity.

### 3.3 Diversity Measurement

In order to compare our method with other approaches, we randomly sample a set of 500 queries from our dataset as the testing queries.

We first evaluate DQS, FRW and BRW in terms of diversity among all the suggested queries, i.e., given a query $q$ and the suggested result set $\mathcal{S}_q$, we will evaluate the diversity in the result set $\mathcal{S}_q$.

Evaluating the diversity is a difficult task since there is no ground truth regarding which query belongs to which category. Moreover, even for human experts, it is also hard to determine the quality of diversity between two queries. However, the abundant information embedded in the click-through data actually provides some intuitions for us to examine the diversities between queries. Let $q_i$ and $q_j$ denote two queries, and assume $\mathcal{P}(q_i)$ denote the clicked Web page set of query $q_i$. If $q_i$ and $q_j$ have almost the same meanings, then $\mathcal{P}(q_i)$ and $\mathcal{P}(q_j)$ should share some of the same or similar click Web pages. On the other hand, in the extreme case if $q_i$ and $q_j$ represent two totally different concepts or topics, then $\mathcal{P}(q_i)$ and $\mathcal{P}(q_j)$ probably have no intersection, and the Web pages in the set $\mathcal{P}(q_i)$ are dissimilar with those in the set $\mathcal{P}(q_j)$. Hence, based on this intuition, we propose a novel diversity measurement as the following:

$$D(q_i, q_j) = 1 - \frac{\sum_{m=1}^{M}\sum_{n=1}^{N} sim(p_{im}, p_{jn})}{M \times N}, \quad (5)$$

where $p_{im} \in \mathcal{P}(q_i)$ and $p_{jn} \in \mathcal{P}(q_j)$. $M$ and $N$ are the sizes of $\mathcal{P}(q_i)$ and $\mathcal{P}(q_j)$. $sim(p_{im}, p_{jn})$ measures the similarity between Web pages $p_{im}$ and $p_{jn}$. We use the union of the queries which lead to clicking Web page $p_{im}$ to represent this Web page, then the similarity $sim(p_{im}, p_{jn})$ is calculated by Cosine similarity using the representations of Web pages $p_{im}$ and $p_{jn}$. The definition in Eq. (5) indicates that: a large value of $D(q_i, q_j)$ means queries $q_i$ and $q_j$ are diverse. For example, in our dataset, the diversity value between queries "msn travel" and "msntravel" is 0.350 while the diversity value between queries "msn travel" and "aol travel" are 0.969.

We then define the diversity over a query set $\mathcal{S}_q$ as:

$$SD(\mathcal{S}_q) = \frac{\sum_{i=1}^{K}\sum_{j=1, i\neq j}^{K} D(q_i, q_j)}{K \times (K-1)}, \quad (6)$$

where $K$ is the size of the query set $\mathcal{S}_q$. We evaluate the diversity of the suggestion result sets of all of the 500 testing queries based on Eq. (6), and we report only the average values. The comparison results are shown in Fig. 3(a). From the comparisons, we can observe that no matter how many suggestions are returned, our DQS algorithm consistently generates much diverse results than FRW and BRW methods. This shows the effectiveness of our DQS algorithm in terms of the quality of diversity.

### 3.4 Relevance Measurement

Obviously, only measuring the diversity of the suggestion results is not reasonable since the suggestions may be diverse but not relevant to the original query. Hence, in this section, we evaluate the relevance of our proposed algorithm with FRW and BRW.

Evaluating the quality of semantic relations is also a challenging task. In this paper, we conduct both a manual evaluation by a panel of three human experts, and automatic evaluation based on the ODP database.

In the evaluation by human experts, we ask all the experts to rate the query suggestion results (we use the same 50 testing queries adopted in Section 3.3). We define a 6-point scale (0, 0.2, 0.4, 0.6, 0.8, and 1) to measure the relevance between the testing queries and the suggested queries, in which 0 means "totally irrelevant" while 1 indicates "entirely relevant". The average values of evaluation results are shown in Fig. 3(b). We observe that, when measuring the results by human experts, our DQS algorithm generally performs at least as well as FRW while better than BRW.

For the automatic evaluation, we utilize the ODP database. ODP, also known as dmoz, is the largest, most comprehensive human-edited directory of the Web. In this paper, we adopt the same method used in (Baeza-Yates and Tiberi 2007) to evaluate the quality of suggested queries. When a user types a query in ODP, besides site matches, we can also find *category* matches in the form of paths between directories. Moreover, the matched categories are ordered by relevance. For instance, the query "nikon" would provide the hierarchical category "Arts: Photography: Equipment and Services: Cameras: 35mm: Nikon", where ":" is

used to separate different categories. One of the results for "canon" would be "Arts: Photography: Equipment and Services: Cameras: 35mm: Canon". Hence, to measure how related two queries are, we can use a notion of similarity between the corresponding categories (as provided by ODP). In particular, we measure the similarity between two categories $\mathcal{A}$ and $\mathcal{A}'$ as the length of their longest common prefix $\mathcal{P}(\mathcal{A}, \mathcal{A}')$ divided by the length of the longest path between $\mathcal{A}$ and $\mathcal{A}'$. More precisely, denoting the length of a path with $|\mathcal{A}|$, this similarity is defined as: $Sim(\mathcal{A}, \mathcal{A}') = |\mathcal{P}(\mathcal{A}, \mathcal{A}')|/max\{|\mathcal{A}|, |\mathcal{A}'|\}$. For instance, the similarity between the two queries above is $5/6$ since they share the path "Arts: Photography: Equipment and Services: Cameras: 35mm" with 5 directories and the longest one is made of 6 directories. We have evaluated the similarity between two queries by measuring the similarity between the most similar categories of the two queries, among the top 5 answers provided by ODP.

As shown in Fig 3(c), the trend of comparison results is similar to Fig 3(b). The main difference is that the accuracy scores using ODP database are generally smaller than the scores rated by the human experts. The reason is straightforward: Human experts have a better understanding of the latent semantic similarities between two queries than the ODP measure method does.

In general, the experimental results shown in Section 3.3 and Section 3.4 indicate that our proposed query suggestion algorithm can suggest both semantically relevant and diverse queries to the users. This also shows the promising future of our diverse ranking algorithm.

## 4 Related Work

In this section, we review research topics which are relevant to our work.

The intention of query suggestion is similar to that of query expansion (Chirita, Firan, and Nejdl 2007; Cui et al. 2003; Theobald, Schenkel, and Weikum 2005; Xu and Croft 1996), query substitution (Jones et al. 2006) and query refinement (Kraft and Zien 2004), which all focus on improving the queries submitted by users. In (Xu and Croft 1996), local and global documents are employed in query expansion by applying the measure of global analysis to the selection of query terms in local feedback. Although experiment shows that this method is generally more effective than global analysis, it performs worse than the query expansion method proposed in (Cui et al. 2003) based on user interactions recorded in user logs. In another approach reported in (Kraft and Zien 2004), anchor texts are employed for the purpose of query refinement. This work is based on the observation that Web queries and anchor texts are highly similar.

In (Baeza-Yates, Hurtado, and Mendoza 2004), a query recommendation method based on clickthrough data is proposed. The main disadvantage of this algorithm is that they ignore the rich information embedded in the query-URL bipartite graph, and consider only queries that appear in the query logs, potentially losing the opportunity to recommend highly semantically related queries to the users. In (Ma et al. 2008; Mei, Zhou, and Church 2008), two query suggestion approaches are proposed to suggest relevant queries to the issued query. However, all the above methods do not address the problem of results diversification.

## 5 Conclusions

In this paper, we present a novel ranking method which can diversify the ranking results by employing the Markov random walk process and hitting time analysis. We apply our method to the problem of diversifying the query suggestion results, and the experimental analysis on a large clickthrough data of a commercial search engine shows that our algorithm can generate both semantically diverse and relevant queries to the original query. The proposed approach includes a principled mathematical foundation and is quite general, which can be applied to various graphs and applications.

## 6 Acknowledgement

The work described in this paper was fully supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. CUHK 4128/08E and CUHK 4154/09E).

## References

Baeza-Yates, R., and Tiberi, A. 2007. Extracting semantic relations from query logs. In *KDD*, 76–85.

Baeza-Yates, R. A.; Hurtado, C. A.; and Mendoza, M. 2004. Query recommendation using query logs in search engines. In *EDBT*, 588–596.

Chirita, P. A.; Firan, C. S.; and Nejdl, W. 2007. Personalized query expansion for the web. In *SIGIR '07*, 7–14.

Craswell, N., and Szummer, M. 2007. Random walks on the click graph. In *SIGIR*, 239–246.

Cui, H.; Wen, J.-R.; Nie, J.-Y.; and Ma, W.-Y. 2003. Query expansion by mining user logs. *IEEE Trans. Knowl. Data Eng.* 15(4):829–839.

Eiron, N.; McCurley, K. S.; and Tomlin, J. A. 2004. Ranking the web frontier. In *WWW*, 309–318.

Jones, R.; Rey, B.; Madani, O.; and Greiner, W. 2006. Generating query substitutions. In *WWW*, 387–396.

Kraft, R., and Zien, J. 2004. Mining anchor text for query refinement. In *WWW*, 666–674.

Ma, H.; Yang, H.; King, I.; and Lyu, M. R. 2008. Learning latent semantic relations from clickthrough data for query suggestion. In *CIKM*, 709–718.

Mei, Q.; Zhou, D.; and Church, K. 2008. Query suggestion using hitting time. In *CIKM*, 469–478.

Sun, J.-T.; Shen, D.; Zeng, H.-J.; Yang, Q.; Lu, Y.; and Chen, Z. 2005. Web-page summarization using clickthrough data. In *SIGIR*, 194–201.

Theobald, M.; Schenkel, R.; and Weikum, G. 2005. Efficient and self-tuning incremental query expansion for top-k query processing. In *SIGIR*, 242–249.

Wang, X., and Zhai, C. 2007. Learn from web search logs to organize search results. In *SIGIR*, 87–94.

Xu, J., and Croft, W. B. 1996. Query expansion using local and global document analysis. In *SIGIR*, 4–11.