Università di Pisa

Facoltá di Scienze MM.FF.NN.

Corso di laurea in Tecnologie Informatiche

Tesi di laurea

An efficient algorithm to generate Search Shortcuts

Lorenzo Marcon

Relatori	Controrelatore
Dott. Raffaele Perego	Prof. Roberto Grossi
Dott. Fabrizio Silvestri	

Anno Accademico 2009/2010

Contents

Acknowledgements **1**

1 Introduction **3**

1.1 Earlier studies 4

1.2 Outline 8

2 Related work **9**

2.1 A broad classification 9

2.2 Implicit methods 10

2.3 Association rules 11

2.4 Cover Graph 13

2.5 Query Flow Graph 17

2.6 Successful sessions 24

3 Search Shortcuts: theoretical model **27**

3.1 The Search Shortcuts Problem 27

3.2 Previous solution proposals 30

4 Search Shortcuts: our shortcuts generation method **33**

4.1 Inverted indexes 37

4.2 BM25 Ranking model 38

4.3 Final results ranking 39

5 Implementation details **41**

5.1 Datasets 41

5.2 Preprocessing 44

5.3 Terrier IR engine 45

5.4 Virtual Documents 46

5.5 SS Interactive Interface 49

5.6 Results processing 50

6 Evaluation methodology **53**

6.1 Evaluation models 53

6.2 TREC topics coverage 54

6.3 Experimental Settings 56

7 Results **57**

7.1 Recommendations for TREC queries 62

8 Conclusions **81**

8.1 Extracting sessions with Query Flow Graph 82

8.2 Future work 84

A Glossary **87**

Acknowledgements

This work wouldn't exist without the helpful support of some as talented as kind people. It is my actual intention to reward these people as they really deserve, but this place wouldn't be enough to express all the gratitude I owe to everyone. Additionally, in every task of our life, there are always 'hidden' people who help you, people that doesn't even imagine that a simple word, a gesture, or just a smile can give you new energy to keep on going through your life with renewed strength. I'm thinking about this right now. This work about *query suggestions* has a fascinating parallel with real life: people you don't even know can address you to the right way.

This thesis is dedicated to all the people that, consciously or not, helped me to get through these last years.

I'd like to explicitly thank my supervisors Raffaele Perego and Fabrizio Silvestri, who put great enthusiasm and daily effort to support and follow the evolution of this thesis, and all the nice people from the HPC Lab at CNR in Pisa.
Finally, an acknowledgement goes to the authors of [6, 7] and the Yahoo! Research Lab of Barcelona for providing the possibility of using their Query Flow Graph implementation for evaluation purposes.

Chapter 1

Introduction

There is a paradox at the ground of the actual state of knowledge of search engine users: their information needs often arise because they ”don’t know something”; information retrieval systems, and search engines in particular, are designed to satisfy these needs, but the users need to know what they are looking for. However, if the users know what they’re looking for, there may not be a need to search in the first place. Thus, in these cases, computing similarity between queries and documents is fundamentally wrong, or at least not useful enough. For example, a user would want a search for ”aircraft” to match ”plane”. Users often attempt to address this problem themselves by manually refining a query, but this process of refinement could be automated. Every search engine nowadays has got their own query recommendation features, ranging from auto-completion to related topics suggestion, from spelling correction to similar queries proposal.

Therefore, giving suggestions to users of Web Search Engines (WSEs) is a common practice aimed at “*driving*” users toward the information bits they may need. Suggestions are normally provided as queries that are, to some extent, related to those recently submitted by the user. The generation process of such queries, basically, exploits the expertise of “smart” users to help inexperienced ones. The knowledge mined for making this possible is contained in WSE logs which store all the past interactions of users with the search system.

1.1 Earlier studies

An original solution for query suggestion based on the model called *Search Shortcuts* has been proposed in 2009 by Baraglia *et al.* [5]. Considering a *query path* as the set of time-ordered queries performed by the same user in a time interval, the authors wanted to analyze the query path followed by different users who started with the same query, assuming they could have the same information need.

The basic idea is that some users follow a ”right” path, and end their session visiting some document proposed by the search engine; some other users may end the search session without visiting any result. These two cases are what the authors call, respectively, a *satisfactory* and a *unsatisfactory* session, as explained in detail in chapter 3.

Figure 1.1 shows the percentage of satisfactory query paths, sorted by the logarithm of the rank of the initial queries: this plot was made with queries extracted from AOL query log, and it includes $140,165$ initial queries (not unique) which are the starting point for at least two sessions. Considering this data, 64% of the sessions were satisfactory, while 36% ended with a query that did not produce any user click.

In the left hand side of the figure, which represents the query paths associated with the most frequent initial queries, we can see that the majority of sessions ended successfully, but at the same time there are several sessions ended up without clicking on any document, although they started with the

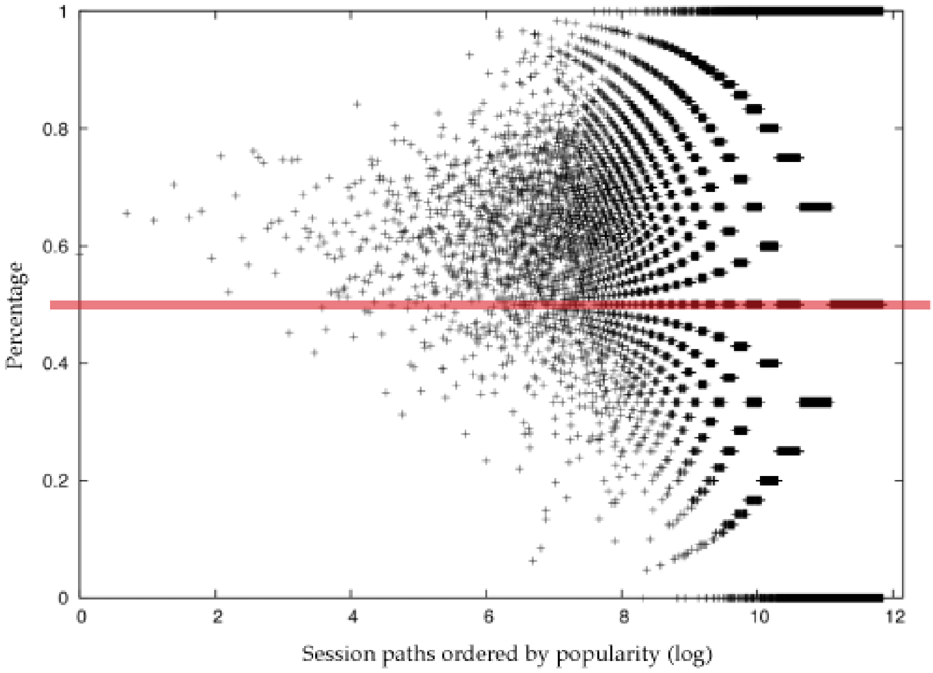


Figure 1.1: Percentage of satisfactory query paths in the AOL query log, by popularity of the first query.

same query.

Hence, the search shortcut model provides a way of exploiting the information provided by the satisfactory sessions, which could lead the failed sessions to a successful ending point.

The particular shape visible in the right hand side of the plot is produced by less frequent queries: for all initial queries repeated two times, we will have some paths 100% satisfactory, some others at 50%, some others at 0%; thus, we have dots at three percentages. Same consideration for queries repeated three times: we have dots at 100%, 66%, 33% and 0%, and so on for the other less popular queries.

In the same work, the problem of *Search Shortcuts* was formally defined: it was also proposed an original evaluation metric for assessing the effectiveness of suggested queries, and investigated the use of *Collaborative Filtering* methods to address this problem. However, some limitations were pointed out in the solution based on collaborative filtering mainly due to the poor scoring information available in query logs, and to the sparsity of data. As a result, using CF based method was able to "*cover*", i.e. generate suggestions for, only a limited number of queries. We worked to overcome these issues and in this work we propose a very efficient and effective algorithm specifically designed for generating search shortcuts. We firstly introduce in the shortcut model a weak function for assessing query similarity. We then relax the query similarity constraint. Finally, we re-conduct the shortcut generation phase to the processing of a full-text query over an inverted file that indexes satisfactory user sessions recorded in query log. Differently from most state-of-the art proposals, our shortcuts generation algorithm results to be very efficient, making it suitable for large-scale implementations in real-world search engines. Moreover, our solution can provide effective recommendations also for queries that were never processed in the past, thus solving the data-sparsity problem that often affects recommending techniques [1].

Another important contribution of this work consists in a novel methodology for assessing the effectiveness of query suggestion techniques. The methodology exploits the query topics and the human judgements provided by the National Institute of Standards and Technology (NIST) for running the TREC Web diversity track. For the purposes of the diversity track, the NIST assessors provide 50 queries, and, for each of them, they identify a representative set of subtopics, based on information extracted from the logs of a commercial search engine. We claim that *given a query topic A with all its subtopics* $\{a_1, a_2, \ldots, a_n\}$*, and a query suggestion technique* $\mathcal{T}$*, the more the queries suggested by* $\mathcal{T}$ *for A cover the human-assessed subtopics* $\{a_1, a_2, \ldots, a_n\}$*, the more* $\mathcal{T}$ *is effec-*

tive. To assess the effectiveness of a given query suggestion technique, we thus propose to simply count how many subtopics are actually covered by the suggestions generated by $\mathcal{T}$ for all the TREC diversity track queries. This methodology is entirely based on a publicly-available data. It can be thus considered fair and constitute a good shared base for testing query recommendation systems.

In all the experiments conducted the solution proposed outperformed remarkably the two state-of-the-art algorithms (presented in [3] and [6],[7]) chosen for performance comparison purposes. Differently from these competitor algorithms, our solution generated relevant suggestions for **all** the 50 TREC queries, and the suggested queries covered a high percentage of possible subtopics.

1.2 Outline

The rest of this thesis is organized as follows:

Chapter 2, Related work, a detailed overview on the current methods to generate query recommendations, including an in-depth description of two main state-of-the-art algorithms used for comparison;

Chapter 3, Search Shortcuts: theoretical model, a formal definition of Search Shortcuts problem;

Chapter 4, Search Shortcuts: out shortcuts generation method, a preliminary yet intensive explanation of the approach we use to resolve Search Shortcuts problem discussed in the previous chapter;

Chapter 5, Implementation details, a more exhaustive and thorough description of the how we implemented our method and about the datasets we used;

Chapter 6, Evaluation methodology, a presentation of a novel evaluation methodology;

Chapter 7, Results, an extensive report of results obtained, including a comparison with other algorithms;

Chapter 8, Conclusions, a final review of the whole work, focused on the results and possible improvements of this approach.

Chapter 2

Related work

Different approaches have been advanced to perform and improve query suggestion, which is a challenging problem. These techniques can be used to improve more than one aspect of information retrieval, but basically, they all aim to boost the performances of the search engine, to better fit the user needs.

2.1 A broad classification

A first distinction could be done between explicit and implicit approaches:

- **explicit methods** rely on actively soliciting data by recording queries and then asking users to provide relevance judgements on retrieved documents. The main idea is to present to the users a list of documents related to an initial query: after examining them, the user selects those which are relevant;

- **implicit methods** are based on extracting implicit information from different source, mainly query logs: the system attempts to infer user intentions based on observable behaviour (e.g.: click-through data, time spent on a page, input reformulation). These approaches usually need a preprocessing phase, consisting in performing static analysis on the information sources available, extracting useful data that will be used later to recommend queries to users.

Few users are willing to give explicit feedback, making significant amounts of such data difficult to obtain; implicit techniques allow virtually unlimited data to be collected at very low cost, although interpretation is more complex. We will focus on these latter, both because the more interest they have in research and the real benefits they potentially provide.

2.2 Implicit methods

Considering implicit methods, we discern between *query expansion* approaches and methods that *get knowledge by query log exploitation*. The former ones basically adapt query expansion techniques to give suggestion of new queries possibly related to the input query. This strategy is different from finding related queries because the methods based on expansion construct artificial queries, while by leveraging query log knowledge, it is possible to give *actual* related queries formulated by other users that had the same information need in the past. In particular, these methods are based on the idea that it is possible to make automatic predictions about the interests of a user by collecting and analyzing pattern information from many users; we will focus on the second category, as it includes our Search Shortcuts approach.

Following, a narrower classification among implicit methods that use query logs:

- **Association rules** based methods, proposed by some authors as a technique to generate lists of related queries;

- **Collaborative filtering** methods try to make automatic predictions of queries: a collaborative filtering approach has been proposed to resolve the Search Shortcuts problem we define in section 3; we will briefly discuss the method provided by [5] in section 3.2;

- **Clustering** methods use a formal definition of ′similarity′ to build sets of ′similar′ queries.

2.3 Association rules

Fonseca et al. [10] used an algorithm for mining association rules from the log of past submitted queries to a search engine. Their approach can be used for spelling correction and query expansion as well.

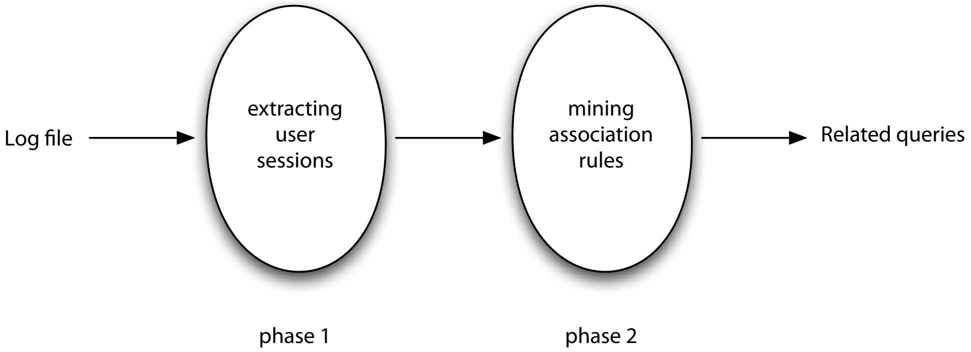


Figure 2.1: Identifying related queries process

Their method is divided in two phases; in the first one, search engine logs are analyzed and user sessions are extracted. A user session is the set of all queries made by a user in a pre-defined time interval. In this work, the definition of user session is strictly related to our query session definition, except that they consider a time interval of 10 minutes instead of 5. To avoid queries from different users with the same IP address, they only use sessions with a low number of queries (10 or less). Once the set of user sessions s is characterized, the second phase can be performed. The intuition behind this method is as follows: during a session, the user defines (roughly) his information need submitting a set of queries. If distinct queries occur simultaneously in many user sessions, these queries may be related.

The simple definition they propose allow to compute the relation between queries in an extremely fast way, which means new association rules can be updated periodically to identify new groups of related queries. The evaluation of the quality of this method is made performing some experiments using a log with 2,312,586 queries from a popular search engine in Brazil (Farejador IG). They show related queries extracted for the top 5 most popular queries in the period analyzed.

The general problem of mining association rules, based on the problem of mining sales data, can be refined for the problem of finding related queries. Given a set of queries I from log files and a set of user sessions T, let X and Y be subsets of I: the implication $X \Rightarrow Y$, where $X \cap Y = \emptyset$ is an association rule with a confidence factor of c if $c\%$ of the sessions in T that contains X also contain Y; this association rule also has a support factor of s if $s\%$ of sessions in T contain $X \cup Y$. The problem of mining association rules is to generate all the association rules that have a support greater than a specified minimum support, or *minsup*.

The authors evaluates their method performing experiments using a value of *minsup* = 3. The judgement about the relationship between queries was performed by five people from their laboratory, who analysed each query and the suggestion provided by the program, assigning as related the suggestions they believed cold be interesting for users who formulated the original query; a second evaluation to check the degree of relation between two queries is made evaluating the precision-recall curve of the original query compared against the curve for the related queries. The results are quite good, however two problems arise: first, it is difficult to determine sessions of successive queries that belong to the same search process; on the other hand, the most interesting related queries, those submitted by different users, cannot be discovered. This is because the support of a rule increases only if its queries appear in the same query session, and thus they must be submitted by the same user.

2.4 Cover Graph

Baeza-Yates et al. [2, 3] propose a clustering method that uses the content of historical preferences of users registered in the query logs to group semantically similar queries: they define a graph based on the notion of query distance using common clicked URL's. We will focus on this approach to compare the results with the Search Shortcuts method we propose. They start with a few definitions:

- Query instance: query (set of words or sentence) plus zero or more clicks related to that query. Formally: $QI = (q, u*)$ where $q = \{words\ or\ phrase\}$ being q the query, and u a clicked URL. Moreover, given a query instance QI they denote with QI_q the query associated to QI and with $QI_{c(u)}$ the set of its clicked URLs.
- URL Cover: set of all URLs clicked by a query. That is:

$$UC_p = \bigcup_{QI_q = p} QI_{c(u)}$$

Our definition of session is quite different than theirs: in fact, a session in our work is strictly related to time, and not to clicked results, furthermore it usually contains different queries. We recall that we consider a query session as a set of queries performed by the same user in a 5 minutes period of time.

Taking a step back to [3] work, the authors start considering only queries that appear in the query log: a single query may be submitted to the search engine several times, and each submission induces a different query session.

Then they introduce a vectorial representation for the queries: these latter are represented as points in a high dimensional space, where each dimension corresponds to a unique URL u. That is, a query is based on all the different URLs in its URL cover. Given a query q, they denote its representation with $\overline{q}$. To each component of the vector $\overline{q}$ is assigned a weight equal to the frequency with which the corresponding URL u has been clicked for that query q. Based on this vectorial representation, it is possible to define a graph: each query is a node of the graph; two nodes (queries) are connected by an edge if they share at least one URL u. Hence, the graph obtained is undirected. Edges are weighted according to the cosine similarity of the queries they connect: thus, if $e = \{q, q'\}$ and the URL space has D dimensions (total number of different URLs), the weight of e is given by:

$$W(e) = \frac{\overline{q} \cdot \overline{q}'}{|\overline{q}||\overline{q}'|} = \frac{\sum_{i \leq D} q(i) \cdot q'(i)}{\sqrt{\sum_{i \leq D} q(i)^2} \cdot \sqrt{\sum_{i \leq D} q'(i)^2}}$$

The quality of the so obtained graph could be improved using a different types of edges connecting the nodes: they classify the types of edges as follows:

- **Identical cover**: $UC_{q1} = UC_{q2}$, a undirected edge implying that both queries $q1$ and $q2$ are in practice equivalent;
- **Strict complete cover**: $UC_{q1} \subset UC_{q2}$, a directed edge from $q1$ to $q2$, semantically implying that $q1$ is more specific than $q2$;

- **Partial cover**: $UC_{q1} \cap UC_{q2} \neq 0$, but does not fulfill any of the previous two conditions. This is the most typical edge and can exist for many reasons, such as due to multi-topic URLs to truly related queries.

One of the problems of this approach is the sparsity of the model; in fact, all the queries that have been clicked at least once, become part of the model. To lower its dimension, the authors use a filter both on nodes and edges of the graph, pruning queries with a few clicks and the edges with a low weight. This technique has also the effect of lowering the noise of the data.

Another possible improvement is based on multi-topical URL recognition: this kind of URLs brings usually weak relations between queries, because the URL used is shared among weakly semantically related queries or unrelated at all; the authors propose a heuristic to lower the impact of this phenomenon on the results by deleting URL that are implied in weak relations.

Their query recommending algorithm operates in the following steps:

1. Queries along with the text of their clicked URLs extracted from the query log are clustered. This preprocessing phase can be conducted periodically.
2. Given an input query, it first finds the cluster to which the input query belongs; then, it computes a rank score for each query in the cluster.
3. The related queries are returned ordered according to their rank score.

It results that is critically important to define a good ranking function: they measure the rank score of a related query combining two notations:

a. **Similarity of the query**. The similarity of a query to the input query is measured using the following method: they first build a term-weight vector for each query, using as vocabulary the set of all distinct words in the clicked URLs, not considering stopwords. Each term is weighted according to the number of occurrences and the number of clicks of the documents in which the term appears. Given a query q, and a URL u, let $Pop(q,u)$ be the popularity of u (fraction of clicks) in the answers of q. Let $Tf(t,u)$ be the number of occurrences of term t in URL u: they now define a vector representation for q, where $q[i]$ is the $i-th$ component of the vector associated to the $i-th$ term of the vocabulary as follows:

$$q[i] = \sum_{URLu} \frac{Pop(q,u)Tf(t_i,u)}{max_t Tf(t,u)}$$

b. **Support of the query**. This is a measure of how relevant is the query in the cluster. The support of the query is measured as the fraction of the documents returned by the query that captured the attention of users (clicked documents). It is estimated from the query log as well.

Given these definitions, they compute the clusters using a 15 days query log of TodoCL, a search engine of Chile; it contains $6,042$ queries with relative clicks; $22,190$ clicks in total, over $18,527$ different URLs, for an average of 3.67 URLs per query. The algorithm used to compute the clusters is the well known k-means, chosen both for simplicity and low computational cost, compared to other clustering algorithms. Since the value k is fixed in k-means, they performed successive runs of the algorithm with different number of clusters, represented by k. They measured the quality of the clusters using a common adopted criterion function in k-means implementations, which is a function that measures the total sum of the similarities between the vectors and the centroids of the clusters that are assigned to. The following figure shows the quality of the cluster related to the number of clusters (k value); $Diff$ curve shows the incremental gain of the overall quality of the clusters:

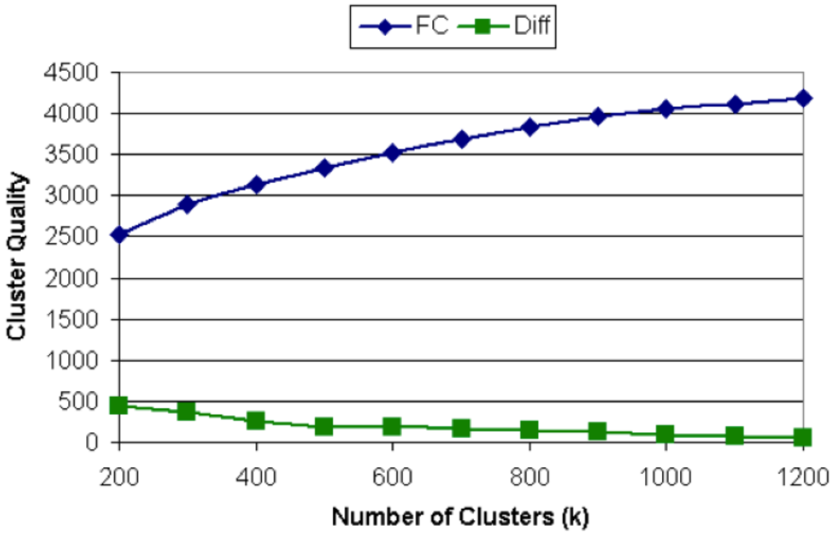


Figure 2.2: Plot of cluster quality (vertical axis) for number of clusters (horizontal axis)

The authors followed a similar approach to [10] in order to assess the quality of results: the relevance of each query to the input query were judged by members of their department; the results are given in a graph that shows precision vs. number of recommended queries. The average support measure is 80% for the first 3 recommended queries; for both popularity and similarity, the precision decreases as the rank of results decreases.

2.5 Query Flow Graph

Boldi et al. [7] introduce the *Query-Flow Graph*, a graph representation of the interesting knowledge about latent querying behaviour. Intuitively, in the query-flow graph, a directed edge from query q_i to query q_j means that the two queries are likely to be part of the same "search mission". Any path over the query-flow graph may be seen as a searching behaviour, whose likelihood is given by the strength of the edges along the path.

The Query-Flow Graph is an outcome of query-log mining and, at the same time, a useful tool for it. The methodology proposed builds a real-world query-flow graph from a large-scale query log that can be applied to two concrete applications: *query recommendation* and *finding logical sessions*.

The Query-Flow Graph is an actionable, aggregated representation of the interesting information contained in a large query-log. In particular, the phenomenon of interest is the *sequentiality of similar queries*: the fundamental two dimensions that drive the construction of the query-flow graph are the temporal order of queries and their similarity.

Given a query log, the nodes of the query-flow graph are all the queries contained in the log, and a directed edge between two queries q_i, q_j has a weight $w(q_i, q_j)$. The authors propose two weighting schemes, one that represents the probability that the two queries are part of the same search mission given that they appear in the same session, and another that represents the probability that query q_j follows query q_i. In both cases, when $w(q_i, q_j)$ is high, one may think of q_j as a typical reformulation of q_i: this a step ahead towards the successful completion of a possible search mission.

The first problem, query recommendations, is strictly related to our work. We will see that the second problem, finding logical sessions, could be useful if combined to the Search Shortcuts problem solution we propose: in fact, we provide a simple and naive methodology for user sessions extraction, while a more effective approach could improve the quality of recommendations.

With respect to query recommendation, they propose an algorithm that builds on the concept of query-flow graph and allows leveraging not only similarity between queries, but the overall complex structure in a neighbourhood of the graph. Their recommendation algorithm is based on performing a random walk with restart to the original query of the user or to a small set of queries representing the recent querying history.

They list some definitions to introduce their approach:

Sessions: a user query session, or session, is defined as the sequence of queries of one particular user within a specific time limit. More formally, if t_θ is a timeout threshold, a user query session S is a maximal ordered sequence

$$S = \langle\langle q_{i_1}, u_{i_1}, t_{i_1}\rangle, ..., \langle q_{i_k}, u_{i_k}, t_{i_k}\rangle\rangle,$$

where $u_{i_1} = ... = u_{i_k} \in U, t_{i_1} \leq ... \leq t_{i_k}, andt_{i_{j+1}} - t_{i_j} \leq t_\theta$, for all $j = 1, 2, ..., k-1$.

Given a query log L, the corresponding set of sessions can be constructed by sorting all records of the query log first by userid u_i, and then by timestamp t_i, and by performing one additional pass to split sessions of the same user if the time difference of two queries exceeds the timeout threshold. Whenever they used a timeout threshold for splitting sessions, they set $t_\theta = 30$ minutes.

Supersessions: the sequence of all the queries of a user in the query log, ordered by timestamp, is called a *supersession*. Thus, a supersession is a sequence of sessions in which consecutive sessions have time difference larger than t_θ.

Chains: it is a topically coherent sequence of queries of one user. For instance, a query chain may contain the following sequence of queries: `"brake pads"; "auto repair"; "auto body shop"; "batteries"; "car batteries"; "buy car battery online"`. Unlike the concept of session, chains involve relating queries based on the user information need, which is an extremely hard problem. Thus, a session may contain queries from many chains, and inversely, a chain may contain queries from many sessions.

The query-flow graph: it is a directed graph $G_qf = (V, E, w)$ where:

- the set of nodes is $V = Q \cup \{s, t\}$, i.e., the distinct set of queries Q submitted to the search engine and two special nodes s and t, representing a starting state and a terminal state which can be seen as the begin and the end of a chain;
- $E \subseteq V \times V$ is the set of directed edges;
- $w : E \to (0..1]$ is a weighting function that assigns to every pair of queries $(q, q') \in E$ a weight $w(q, q')$.

It is important to notice that in their settings, even if a query has been submitted multiple times to the search engine, possibly by many different users, it is anyway represented by a single node in the query-flow graph. The two special nodes s and t are used to capture the begin and the end of query chains. In other words, the existence of an edge (s, q_i) represents that q_i may be potentially a starting query in a chain, and an edge (q_i, t) indicates that q_i may be a terminal query in a chain.

They built the query-flow graph extracting a set of sessions from a query log L from Yahoo! UK search engine in early 2008. Given two queries q, q', they *tentatively* connect them with an edge if there is at least one session in $S(L)$ in which q and q' are consecutive. In other words, they form the set of tentative edges T as:

$$T = \left\{(q, q') \mid \exists S_j \in S(L) s.t. q = q_i \in S_j \land q' = q_{i+1} \in S_j\right\}$$

The key aspect of the construction of the query-flow graph is to define the weighting function $w : E \to (0..1]$.

The two weighting schemes proposed are based, respectively, on the *chaining probability*, i. e. the probability that q and q' belong to the same chain (given that they belong to the same session) and the *relative frequencies* of the pair (q, q') and the query q.

Weights based on chaining probabilities. The approach used is a machine learning method. The first step is to extract for each edge $(q, q') \in T$ a set of features associated with the edge. Those features are computed over all sessions in $S(L)$ that contain the queries q and q' appearing in this order and consecutively.

For learning the weighting function from the features, they use training data: this data is created by picking at random a set of edges (q, q') (excluding the edges where $q = s$ or $q' = t$), and manually assigning them a label `same_chain`. This label, or target variable, is assigned by human editors and is 0 if q and q' are not part of the same chain. The probability of having an edge included in the training set is proportional to the number of times the queries forming that edge occur in that order and consecutively in the query log. Then they use this training data to learn the function $w(-, -)$, given the set of features and the label for each edge in T.

They use 18 features to compute the function $w(-, -)$ for each edge in T, which can be summarized as follows:

- Textual features: they compute the textual similarity of queries q and q' using various measures, including cosine similarity, Jaccard coefficient, and size of intersection. Those measures are computed on sets of stemmed words and on character-level 3-grams;

- Session features: they compute the number of sessions in which the pair (q, q') appears. They also compute other statistics of those sessions, such as, average session length, average number of clicks in the sessions, average position of the queries in the sessions, etc;

- Time-related features: they compute average time difference between q

and q' in the sessions in which (q, q') appears, and the sum of reciprocals of time difference over all appearences of the pair (q, q').

The last step for constructing the query-flow graph is to train a machine learning model to predict the label `same_chain`. The training dataset consists of approximately $5,000$ labeled examples; the labels were assigned by the authors.

Weight based on relative frequencies. The second weighting scheme considered turns the query flow graph into a Markov chain. Let $f(q)$ be the number of times the query q appears in the query log, and $f(q, q')$ the number of times the query q' follows immediately q in a session. Let $f(s, q)$ and $f(q, t)$ indicate the number of times the query q is the first and last query of a session, respectively.

The weight used is:

$$w'(q, q') = \begin{cases} \frac{f(q,q')}{f(q)} & if(w(q, q') > \theta) \vee (q = s) \vee (q = t) \\ 0 & otherwise \end{cases}$$

which uses the chaining probabilities $w(q, q')$ basically to discard pairs that have a probability of less than θ to be part of the same chain.

By construction, the sum of the weights of the edges going out from each node is equal to 1. Following, an example of the query flow graph produced with this weighting scheme: notice that this snapshot contains the query "`barcelona`" and some of its followers up to a depth of 2, and not all the outgoing edges are reported.

In respect to the application of the query flow graph we're examining, query recommendation, a simple scheme is to pick, for an input query q, the node having the largest $w(q, q')$. An issue with this method is that it tends to

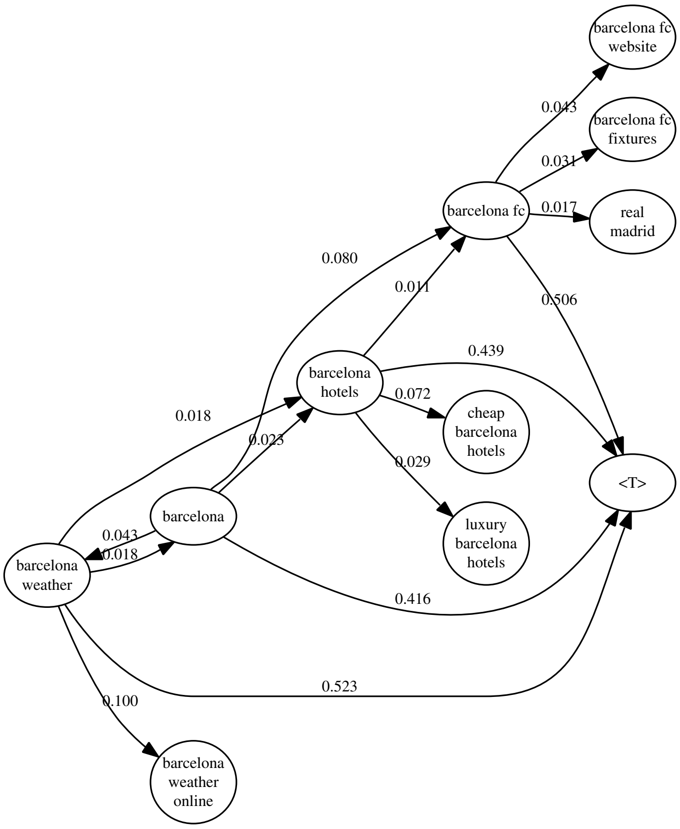


Figure 2.3: A portion of the query flow graph using the weighting scheme based on relative frequencies

"drift" towards those queries that are popular in the query log, but unrelated with the input query. Another recommendation algorithm can be instead built upon a measure of relative importance: when a user submits a query q to the engine, the recommendation that the engine provides should be the most important query q' relatively to q. This can be described as a random walk with restart to a single node: a random surfer starts at the initial query q; then, at each step, with probability $\alpha < 1$, the surfer follows one of the outlinks from the current node chosen proportionally to the weights present on the arcs, and with probability $1 - \alpha$ s(he) instead jumps back to q. This point of view reminds a form of personalized PageRank: recommendations can be deduced from the random-walk score by tanking either the single top-scored query, or the best queries up to a certain lower score threshold. Notice that, in particular, if the most relevant query for q is t, this means that the engine will not give any suggestion, because the query flow graph is showing that the chain at that point is more likely to end than to continue. A third and last recommendation scheme is taken into account by authors: the idea is to provide recommendations not only relying on the last input query, but on some of the last queries in the user's history. This approach may help to alleviate the data sparsity problem and help to solve ambiguous queries, adjusting the score of the query q' in relation to q obtained in the random walk model.

The authors of the query flow graph do not assess the results obtained with their approach, they just show the possible applications of the query flow graph; we are mainly interested to compare the results provided by our Search Shortcuts approach to the application of the query flow graph with respect to the *query recommendation task*. Furthermore, in section 8.1, we show a preliminary analysis on the application of query flow graph to the problem of *splitting in logical sessions* a query log: this is a good starting point for future work aiming to improve Search Shortcuts quality.

2.6 Successful sessions

A few words to mention the works of Smyth *et al.* [26, 25], about collaborative web searches. The authors refer extensively to *Successful sessions*, an idea which is pretty the same of our concept of *Satisfactory sessions*. In these works, a session is considered successful if at least one result has been selected; oppositely, if no results were selected, a session is considered *failed*. As we also do, they do not distinguish between sessions with different numbers of selected results, mainly because it is not possible to conclude much from the frequency of result selections. In fact, for example, one might be tempted to conclude that users selecting more results is a sign of *increasing* result relevance, but a similar argument can be made in support of *decreasing* result relevance, on the basis that the initial selections must not have satisfied the users.

Chapter 3

Search Shortcuts: theoretical model

In the following sections we recall the *Search Shortcuts Problem* (SSP) proposed in [5], discussing the first proposal to resolve the problem, based on collaborative filtering, included in the same work. After formally defining SSP problem, we examine the weak points and limitations of the collaborative filtering approach.

3.1 The Search Shortcuts Problem

The SSP is formally defined as a problem related to the recommendation of queries in search engines and the potential reductions obtained in the users session length. This problem formulation allows a precise goal for query suggestion to be devised: *recommend queries that allowed "similar" users, i.e., users which in the past followed a similar search process, to successfully find the information they were looking for.* The problem has a nice parallel in computer systems: *prefetching*. Similarly to prefetching, search shortcuts anticipate requests to the search engine with suggestion of queries that a user would have likely issued at the end of her session.

We now introduce the notations and we recap the formal definition of the SSP.

Let $\mathcal{U}$ be the set of users of a WSE whose activities are recorded in a query log QL, and $\mathcal{Q}$ be the set of queries in QL. We suppose QL is preprocessed by using some session splitting method (e.g. one of those designed by Jones *et al.* [14] or Boldi *et al.* [6]) in order to extract query $sessions$, i.e., sequences of queries which are related to the same user search task. Formally, we denote by $\mathcal{S}$ the set of all sessions in QL, and σ^u a session issued by user u. Moreover, let us denote with σ_i^u the i-th query of σ^u. For a session σ^u of length n, its **final query** is the query σ_n^u, i.e. the last query issued by u in the session. To simplify the notation, in the following we will drop the superscript u whenever user u is clear from the context.

We say that a session σ is ***satisfactory*** if and only if the user has clicked on at least a link shown in the result page returned by the WSE for the final query σ_n, ***unsatisfactory*** otherwise.

Finally, given a session σ of length n we denote $\sigma_{t|}$ the **head** of σ, i.e., the sequence of the first t, $t \leq n$, queries, and $\sigma_{|t}$ the **tail** of σ given by the sequence of the remaining $n - t$ queries.

Definition 1 *We define **k-way shortcut** a function h taking as argument the head of a session $\sigma_{t|}$, and returning as result a set $h\left(\sigma_{t|}\right)$ of k queries belonging to Q.*

Such definition allows a simple ex-post evaluation methodology to be introduced by means of the following similarity function [5]:

Definition 2 *Given a satisfactory session of length n $\sigma \in \mathcal{S}$, and a k-way shortcut function h, the similarity between $h\left(\sigma_{t|}\right)$ and a tail $\sigma_{|t}$ is defined as:*

$$s\left(h\left(\sigma_{t|}\right),\sigma_{|t}\right)=\frac{\sum\limits_{q\in h\left(\sigma_{t|}\right)}\sum\limits_{m=1}^{n-t}[\![q=\left(\sigma_{|t}\right)_{m}]\!]f\left(m\right)}{|h(\sigma_{t|})|}\qquad(3.1)$$

Where $f\left(m\right)$ is a monotonic increasing function, and function $[\![q=\sigma_m]\!]=1$ if and only if q is equal to σ_m.

For example, to evaluate the effectiveness of a given shortcut function h, the sum (or average) of the value of s computed on all satisfactory sessions in $\mathcal{S}$ can be computed.

Definition 3 *Given the set of all possible shortcut functions $\mathcal{H}$, we define* ***Search Shortcut Problem*** *(SSP) the problem of finding a function $h\in\mathcal{H}$ which maximizes the sum of the values computed by Eq. (3.1) on all satisfactory sessions in $\mathcal{S}$.*

A difference between search shortcuts and query suggestion is actually represented by the function $[\![q=\left(\sigma_{|t}\right)_m]\!]$ in Eq. (3.1). By relaxing the strict $equality$ requirement, and by replacing it with a similarity relation – i.e., $[\![q\sim\left(\sigma_{|t}\right)_m]\!]=1$ if and only if the similarity between q and σ_m is greater than some threshold – the problem reduces, basically, to query suggestion. By defining appropriate similarity functions, the equation in (3.1) can be thus used to evaluate query suggestion effectiveness as well.

Finally, we should consider the influence the function $f\left(m\right)$ has in the definition of scoring functions. Actually, depending on how f is chosen, different features of a shortcuts generating algorithm may be tested. For instance, by setting $f\left(m\right)$ to be the constant function $f\left(m\right)=c$, we simply measure the number of queries in common between the query shortcut set and the queries submitted by the user. A non-constant function can be used to give an higher score to queries that a user would have submitted later in the session. For example, in the tests discussed in [5], the exponential function $f\left(m\right)=e^m$

was chosen to assign an higher score to shortcuts suggested early. Smoother f functions can be used to modulate position effects.

3.2 Previous solution proposals

A previous solution to the Search Shortcut problem was provided by Baraglia *et al.* [5]; the solution offered is based on the application of Collaborative Filtering techniques, which seemed a natural way to approach the Search Shortcuts problem, given the fact emphasized at the beginning of section 3.1: *to recommend queries that allowed "similar" users, i.e., users which in the past followed a similar search process, to successfully find the information they were looking for*. Baraglia *et al.* [5] applied CF as a proposal to solve the Search Shorcuts problem and the results obtained are evaluated on large query logs from AOL and Microsoft.

Collaborative filtering algorithms, based on the preferences of other users, can be classified in two main types: *memory-based* and *model-based*. Memory-based approaches use the whole past data to identify similar users [22], items [21], or both [30]. Generally, memory-based algorithms are quite simple and produce good recommendations, but they usually face serious scalability problems. On the other hand, model-based algorithms construct in advance a model to represent the behaviour of users, allowing to predict more efficiently their preferences. However, the model building phase can be highly time-consuming, and models are generally hard to tune, sensitive to data changes, and highly dependent on the application domain. In the literature different approaches can be found: based on algebra methods [9, 16] and clustering [29].

Collaborative filtering deals with a set of users U, and a set of items I. User preferences are taken into account as item ratings, a numeric value representing the utility of an item to a given user. The subset of valid ratings is denoted as R. Ratings can be explicitly introduces by users, or implicitly extracted from user interaction (e.g. from query log data). Preferences for all users are stored in a user-item matrix, known as the rating matrix V. Each entry v_{ui} of V represents the rating of user u for item i, with $u_{ui} \in R \cup \{\emptyset\}$, where $\{\emptyset\}$ indicates that the user has not rated the item yet.

Thus, to apply collaborative filtering to the SSP, we need to fill such matrix with the information in the query log data.

First, the concept of the SSP (users, queries, terms and sessions) have to be mapped to the pure collaborative filtering problem (users and items). As the goal in the SSP is to recommend queries for a given session, it seems reasonable to treat each session as a *user*, and each query as in *item*. Sessions are extracted from query log collecting all the queries performed by the same user in a time span of 30 minutes.

Second, the query ratings must be inferred from the information in the query log. As a preliminary approach, in this work the Baraglia et al. rate the queries focusing in the last query of each session. If such last query was successful (the user has clicked at least one result), then a positive rating (10.0) is given to the query. Otherwise, it is given a negative rating (0.0). All remaining queries are considered neutral (5.0).

The main problem of this approach is that in query session logs there are many queries that only appear in a single session. This lack of information is the well-known *sparsity* problem [12], and it brings to low coverage of results. In addition, web search query logs usually contain much more data than those collected in traditional collaborative filtering domains like e-commerce, and their size grows continuously at a very high rate. Furthermore, several limitations are related to the three-level rating chosen (positive, negative, neutral), which does not perform as expected for collaborative filtering algorithms, especially because most queries are neutral.

Some techniques to limit the sparsity problem include stemming and stop-words removal; using a threshold to cut off the sessions with a low number of queries is another approach to partially narrow down the sparsity problem: experimental results show that when only session with at least 3 queries are considered, sparsity is highly reduced.

Chapter 4

Search Shortcuts: our shortcuts generation method

We approach the SSP previously described using a novel algorithm that aims to generate suggestions containing *only* those queries appearing as final in satisfactory sessions. The goal is to suggest queries having a high potentiality of being useful for people to reach their initial goal. As hinted by the problem definition, suggesting queries appearing as finals in satisfactory sessions, in our view is a good strategy to accomplish this task. In order to validate this hypothesis, we analyzed the Microsoft RFP 2006 dataset, a query log from the MSN Search engine containing about 15 million queries sampled over one month of 2006 (hereinafter QL).

First, we measured that the number of distinct queries that appear as final query in satisfactory sessions of QL is relatively small if compared to the overall number of submitted queries: only about 10% of the total number of distinct queries in QL occur in the last position of satisfactory user sessions. As expected, the distribution of the occurrences of such final queries in satisfactory user sessions is very skewed (as shown in Figure 4.1), thus confirming once more that the set of final queries *actually* used by people is limited.

Queries which are $final$ in some satisfactory sessions may obviously appear also in positions different from the last in other satisfactory sessions. We verified that when this happens, these queries appear much more frequently in positions very close to the final one: about 60% of the distinct queries appearing in the penultimate position of satisfactory sessions are also among the final queries, about 40% in positions second to the last, 20% as third to the last, and so on. We can thus argue that $final$ queries are usually $close$ to the achievement of the user information goal. We can thus consider these queries as highly valued and high quality short pieces of text expressing actual user needs.

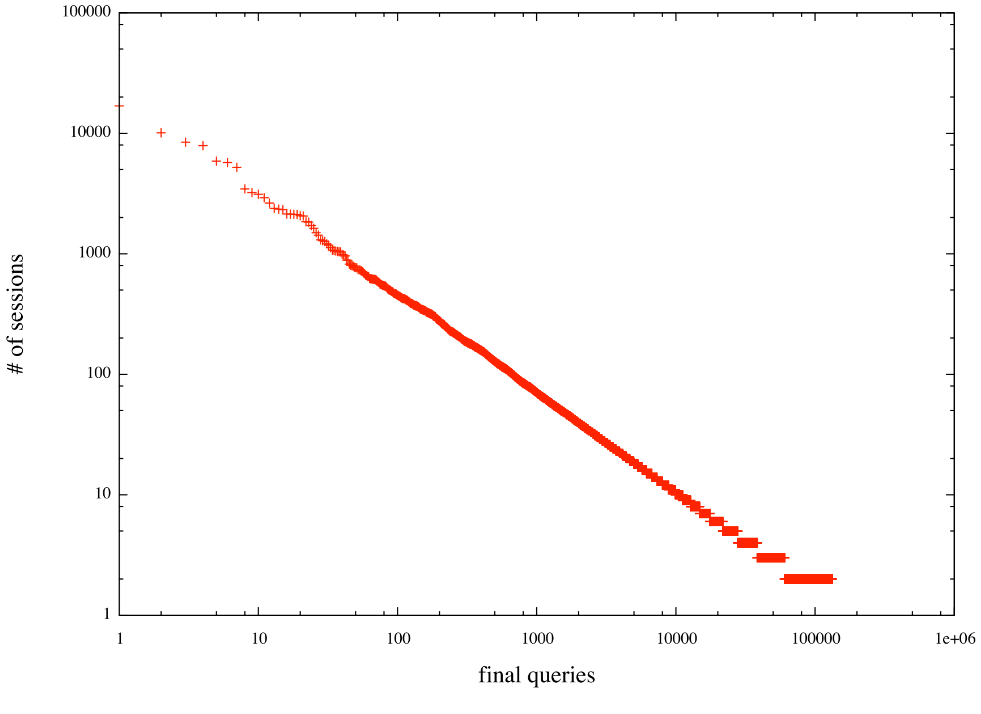


Figure 4.1: Popularity of final queries in satisfactory sessions.

The SSP algorithm proposed works by efficiently computing similarities between partial user sessions (the one currently performed) and historical satisfactory sessions recorded in a query log. Final queries of most similar satisfactory sessions are suggested to users as search shortcuts. Let us better formalize this concept.

Let σ' be the current session performed by the user, and let us consider the sequence τ of the concatenation of all terms with possible repetitions appearing in $\sigma'_{t|}$, i.e. the head of length t of session σ'. We now compute the value of a scoring function $\delta\left(\tau, \sigma^s\right)$, which for each satisfactory session measures the similarity between its queries and the set of terms τ. Intuitively, this similarity measures how much a previously seen session overlaps with the user need expressed so far (the concatenation of terms τ serves as a bag-of-words model of user need). Sessions are ranked according to δ scores and from the subset of the top ranked sessions we suggest their final queries. It is obvious that depending on how the function δ is chosen we may have different recommendation methods. In our particular case, we opt for δ to be the similarity computed as in the BM25 metrics [17]. We opt for an IR-like metric because we want to take into much consideration words that are discriminant in the context of the session to which we are comparing. BM25, and other IR-related metrics, have been designed specifically to account for that property in the context of query/documents similarity. We borrow from them the same attitude to adapt to this conditions. The shortcuts generation problem has been, thus, reduced to an information retrieval task of finding highly similar sessions in response to a given sequence of queries.

The idea described above is thus translated into the following process. For each unique "*final query*" q_f contained in satisfactory sessions we define what we have called a *virtual document* identified by its *virtual title* and its *virtual content*. The virtual title, i.e. the identifier of the document, is exactly q_f. The virtual content of the document, instead, is made up of all the terms that have appeared in queries of all the sessions ending with the query q_f representing the virtual title. At the end of this procedure we have a set of virtual documents, one for each final query in satisfactory sessions. Just to make things clearer, let us introduce a toy example. Consider the two following satisfactory sessions: (*gambling*, *gambling places*, *las vegas*, *bellagio*), and (*las vegas*, *strip*, *las vegas hotels*, *bellagio*). We then create the virtual document identified by the virtual title **bellagio** and whose content is the text (*gambling gambling places las vegas las vegas strip las vegas hotels*). As you can see the text actually contains repetitions that are also considered in the context of BM25 metrics. All virtual documents are indexed with the preferred Information Retrieval system, and generating shortcuts for a given user session σ' becomes simply processing the query $\sigma'_{t|}$ over the inverted file indexing the virtual documents. We know that processing queries over inverted indexes is very fast and scalable, and these characteristics are inherited by our query suggestion technique; further information about inverted index is provided in section 4.1.

It is worth noticing another, very important, characteristics of our method for extracting query suggestion. Query shortcuts generation through IR-like methods is very robust with respect to singleton queries. Singleton queries account for almost 50% of the submitted queries [24], and their presence is responsible for what it is known as the issue of the sparsity of models [1]. This phenomenon has been accounted as an issue by many papers in the field (also in the already cited work from Baraglia *et al.* [5]). Since we match τ with the text obtained by concatenating all the queries in each session we are not bound to look for previously submitted queries as in the case of other models (e.g. [3], [6], [10]). We will report in chapter 7 about the coverage of different models, including ours, discussing the results obtained.

4.1 Inverted indexes

Figure 4.2 shows briefly how inverted indexes work, expanding the toy example discussed above: firstly, the virtual documents are processed and split in tokens. Then, each entry of the index, *i.e.* a token, has a *posting list* associated: this is a sequence of documents containing that entry, and the number of times that token appears in the document.

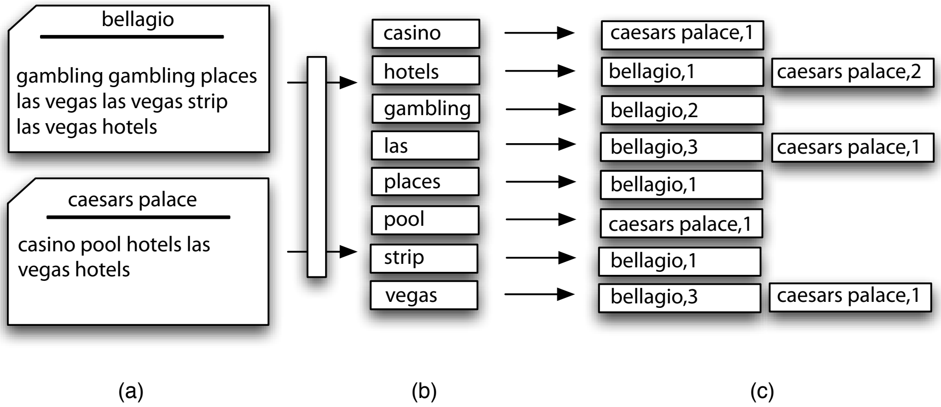


Figure 4.2: Virtual documents (a) are split in tokens (b); each token has a posting list (c) associated, which reports the name of the related document and the frequency of the token.

The retrieval process starts from a query q performed by a user: q is submitted to the retrieval engine, which extracts from the inverted index the documents containing the term(s) of q. Before showing the output to the user, results are ordered using a ranking function, or model. A very basic measure of relevance could be based on the frequency of the terms: a document containing more occurrences of the requested term t with regards to another document, is considered *more relevant* for t. This model is definitely not reliable enough in the *real world*, because it is simply too easy to deceive, opening the way to spammers. However, frequency of terms is still somehow taken into account even in more complex weighting models.

Examples of queries performed on this inverted index could be:

- ″**las vegas**″, returning both *bellagio*, *caesars palace* (in this exact order, if frequency-based rank is used);
- ″**gambling places**″, returning only *bellagio*;
- ″**casino pool**″, returning only *caesars palace*.

This is just an example to show a simplified version of the inverted indexing and retrieval process: obviously, real implementations take usually into account some other information about tokens besides frequency, *e.g.* their position inside the document. In the next section we discuss the ranking model we used in this work.

4.2 BM25 Ranking model

In information retrieval theory, BM25 ([20]) is a widely used ranking function based on the probabilistic retrieval framework developed in the 1970s and 1980s by Stephen E. Robertson, Karen Spärck Jones, and others. Often also called *Okapi weighting*, getting this name from the first retrieval system in which it was implemented, it was developed as a way of building a probabilistic model sensitive to both term frequency and document length, while not introducing too many additional parameters in the model. This weighting function is based on a previous work [18] from the same authors, who presented a first version called BM1; afterwards, they improved it with two other functions called BM11 and BM15 [19], and, by combining these latter into a single function, they finally obtained BM25, which, at the moment, represents the state-of-the-art TF/IDF-like retrieval functions used in document retrieval.

Going deeper, BM25 is a bag-of-words retrieval function that ranks a set of documents based on the query terms appearing in each document, regardless of the inter-relationship between the query terms within a document, as their relative proximity. Actually, it is not a single function, but a whole family of scoring functions with slightly different components and parameters. One of the most used functions is formally defined as follows:

Given a query Q, containing keywords $q_1, ..., q_n$, the BM25 score of a document $\mathcal{D}$ is:

$$BM25(\mathcal{D}, Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{f(q_i, \mathcal{D})(k_1 + 1}{f(q_i, \mathcal{D}) + k_1(1 - b + b \cdot \frac{|\mathcal{D}|}{avgdl})} \quad ,$$

where $f(q_i, \mathcal{D})$ is q_i's term frequency in the document $\mathcal{D}$, $|\mathcal{D}|$ is the number of words contained in $\mathcal{D}$, and $avgdl$ is the average document length in the text collection from which documents are drawn. k_1 and b are free parameters, usually chosen as $k_1 = 2.0$ and $b = 0.75$. Note that setting parameter $b = 1$ turns BM25 to BM11, and $b = 0$ turns it to BM15. $IDF(q_i)$ is the inverse document frequency weight of the query term q_i, usually computed as:

$$IDF(q_i) = log \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} \quad ,$$

where N is the total number of documents in the collection, and $n(q_i)$ is the number of documents containing q_i.

4.3 Final results ranking

We observed that using only BM25 to rank suggestions for a query, we were not taking into account the frequency of suggestions, *i.e.* the number of satisfactory sessions having a query q_f as final. Since we are providing recommendations for user queries, we believe that popular queries should have more weight, and thus have their rank raised with respect to less popular final queries.

The resulting weighting formula is a linear combination of the BM25 score and the frequency of the suggestion; the rank value for the generic q_{f_i} recommendation in relation to τ, defined above in section 4 as the concatenation of the terms appearing in the head of the current search session, is computed as following:

$$w(\tau, q_{f_i}) = \alpha \cdot BM25(\tau, q_{f_i}) + \beta \cdot freq(q_{f_i})$$

Notice that both BM25 rank and frequency are normalized values, so $w(\tau, q_{f_i})$ domain is defined by the range $(0..2]$.

In our experimental settings we used $\alpha = \beta = 1$, giving the same emphasis to both the parameters; obviously, further tests aimed to find the best values of α and β coefficients in the above formula could be performed in future, giving thus more or less importance, respectively, to BM25 ranking or frequency-based ranking.

Chapter 5

Implementation details

5.1 Datasets

In order to implement our Search Shortcuts generation method we had to choose a dataset to work on. We initially had two possible choices: the first one was the well known America Online query log, a publicly available dataset released on early 2006, and a Microsoft query log, the MSN Search Asset Data Spring 2006. The first one contains $36,389,567$ queries, sampled from March 1st to May 31st, while the second includes $14,921,286$ records; a first cleansing was performed by Microsoft researchers before making the query log available, pruning $78,769$ adult queries that are provided in a separate file; despite that, lots of queries that should have been filtered are still present in the query log. In AOL query log, instead, it doesn't exists any kind of pre-filtering process.

For both query logs is available click-through information: in the MSN QL this data is provided in a separate file, while in the AOL QL it is included in the main file: in the latter, if a query generated a click, the clicked url and the result rank (ordinal number) is simply appended to the query line.

The following table shows the format of the MSN query log:

timestamp	query	queryID	sessionID	number of results

- **timestamp**: the date and minute the query (or click) occurred, in the format ”YYYY-MM-DD HH:MM:SS”;

- **query**: the query string, trimmed and with spaces reduced to 1 character (normalization performed by Microsoft); no change to character case;

- **queryID**: a unique identifier of the query, a 16 numbers hexadecimal hash;

- **sessionID**: in the documentation Microsoft provided there is no explanation about how this identifier is generated: so we made some assumptions about it after a simple analysis. We sorted the query log by sessionID and then measured the number of queries and the time span of a ’session’, as they call it: a lot of them are short and include only a few queries, but many of them last hours or days, and include hundreds of queries; for both number of queries and time interval, the variance is extremely high. A possible explanation of this could be that sessionID is obtained from browser cookies; in this case, we can’t know the exact nature of this parameter, but we can still rely on it as a measure of sessions. Another chance could be that Microsoft researchers exploited some ”session splitting” technique, and also in this case we can’t know how they performed this task. In the end, we assumed that this is surely a basic user identifier, thus is useful for our purposes. We will execute a further sessionization step, using a time interval. As queryID, sessionID is represented by a 16 hex digits hash;

- **number of results** on results page: the meaning of this parameter is unclear: it varies from 0 to 67, and no explanation for it is given from the attached Microsoft QL docs. However, it has not been used for Search Shortcuts generation.

The click query log has instead the following format:

queryID	query	timestamp	clickedURL	position

The first three parameters have the same meaning already explained for the main query log, while the last two are, respectively, the url clicked for the relative query, and the url position in the results page. As expected, position has extremely low values, confirming that people usually select the first results provided by the search engine. As in the case of ”number of results”, this parameter is not relevant for our algorithm of shortcuts generation.

We don't report results obtained on AOL QL because they are comparable with those computed on MSN QL; anyway, just for completeness, we report some statistics we initially obtained from the former one, such that it is possible to compare them with the latter:

	MSN	AOL
Total number of queries	14,921,286	36,389,567
Total sessions	9,461,423	16,218,017
Satisfactory sessions	1,949,320	2,814,449
Average number of queries per session	2.71	2.39

Notice that session maximum time interval between the first and the last query is set to 5 minutes; the number of satisfactory sessions is actually higher than the one shown in the table, but we merged sessions that share the ”final query”, considering those sessions as part of the same search need; lastly, we discarded sessions including only one query, as they are not interesting in shortcuts generation process, although formally being a (one-query) session.

Comparing the number of satisfactory sessions to total sessions ratio extracted from the two query logs, we have 4.85 for MSN and 5.76 for AOL, which is quite the same; we got similar results comparing the average number of queries per session, respectively 2.71 and 2.39; thus we assume that users behaviour, with respect to session length, is similar in both search engines, and this result is coherent with previous studies [13], [23] on search engine users' sessions.

5.2 Preprocessing

The Microsoft RFP 2006 query log has been preprocessed by applying standard data cleaning techniques: lowercase conversion, removal of stopwords and of punctuation/control characters. We tested different combinations of stemmer and stopwords modules to spot differences in results suggestions. We obtained good results with all combinations, anyway, stemming and stopwords removal provides the smaller index, as expected.

Then, we sorted the queries by user and timestamp, and segmented them into sessions on the basis of the already described splitting algorithm which simply groups in the same session all the queries issued by the same users in a time span of 5 minutes. Any other more advanced session splitting method [14] could be used with expected improvements also in the quality of shortcuts generated by our solution. The investigation of session splitting methods is however out of the scope of this work. For our purposes, we considered only the $9, 461, 423$ sessions made up of less than 30 queries in order to clean the log from highly-populated sessions surely performed by software robots. Then, we devised satisfactory sessions present in the log and grouped them on the basis of the final query. Thus, for each distinct final query its corresponding *virtual document* was built with the terms (with possible repetitions) belonging to all the queries of all the associated satisfactory sessions, as we will show in more detail in section 5.4.

5.3 Terrier IR engine

To implement our query suggestion technique we exploited the open source Terrier search engine (`http://terrier.org/`). This approach has several benefits with respect to query suggestions generation: in fact, both the algorithms compared to our method use a *"query-based approach"*, which means that if we are willing to get suggestions for a query that is not present in their model, these algorithms are unable to provide any recommendation. In other words, if the string for which we want to get suggestions is not included in the query log from which they extract their knowledge, implying that the query has not been performed before in that query log, those two approaches cannot generate any suggestion.

Using a IR engine such as Terrier, we are able to provide recommendations for queries that has never been performed before, starting from a knowledge base made up by simply building *virtual documents* from the extracted sessions.

Following, is shown the main configuration file, read by Terrier in indexing and in retrieval phase either:

Listing 5.1: configuration file: terrier.properties

```
terrier.home=/Users/stc/Documents/tesi/terrier
querying.postprocesses.order=QueryExpansion
querying.postprocesses.controls=qe:QueryExpansion
querying.default.controls=start:0,end:999
querying.allowed.controls=c,scope,qe,qemodel,start,end
TrecDocTags.doctag=DOC
TrecDocTags.idtag=DOCNO
TrecDocTags.skip=DOCHDR
TrecDocTags.casesensitive=
TrecQueryTags.doctag=TOP
TrecQueryTags.idtag=NUM
TrecQueryTags.process=TOP,NUM,TITLE
TrecQueryTags.skip=DESC,NARR
bundle.size=2500
termpipelines=PorterStemmer,Stopwords
block.indexing=
matching.retrieved_set_size=50
interactive.model=BM25
```

5.4 Virtual Documents

We wrote a collection of Python scripts to extract sessions from a given query log, already preprocessed and ordered by user id and timestamp; following the Search Shorcuts model defined in chapter 4, if the last query of the session extracted produced a click, the script checks if there is already a *virtual* document associated to that last query, previously defined as *final query* or *virtual title*. If already exists a bag-of-words for that query, the script simply merges all the other queries of the current session with the related and existing *virtual content*; if there is no virtual document defined for that query, the script creates it.

A virtual document is a plain text file with the following format:

```
<DOC>
<DOCNO>doc_identifier</DOCNO>
bag-of-words
</DOC>
```

This format has been chosen because it is easily parsable by Terrier; in fact, it builds up its index using the `doc_identifiers` as elements of the posting list associated to each term of the bag of words, in our case the queries of the sessions. Terrier has been configured to build the index without block indexing feature, because, for the current Search Shortcut implementation, we don't need positional information about words inside the queries; Terrier has also been configured to stem the tokens extracted form the bag-of-words, and to exclude stopwords from indexing, using a stopwords list provided by Terrier developers.

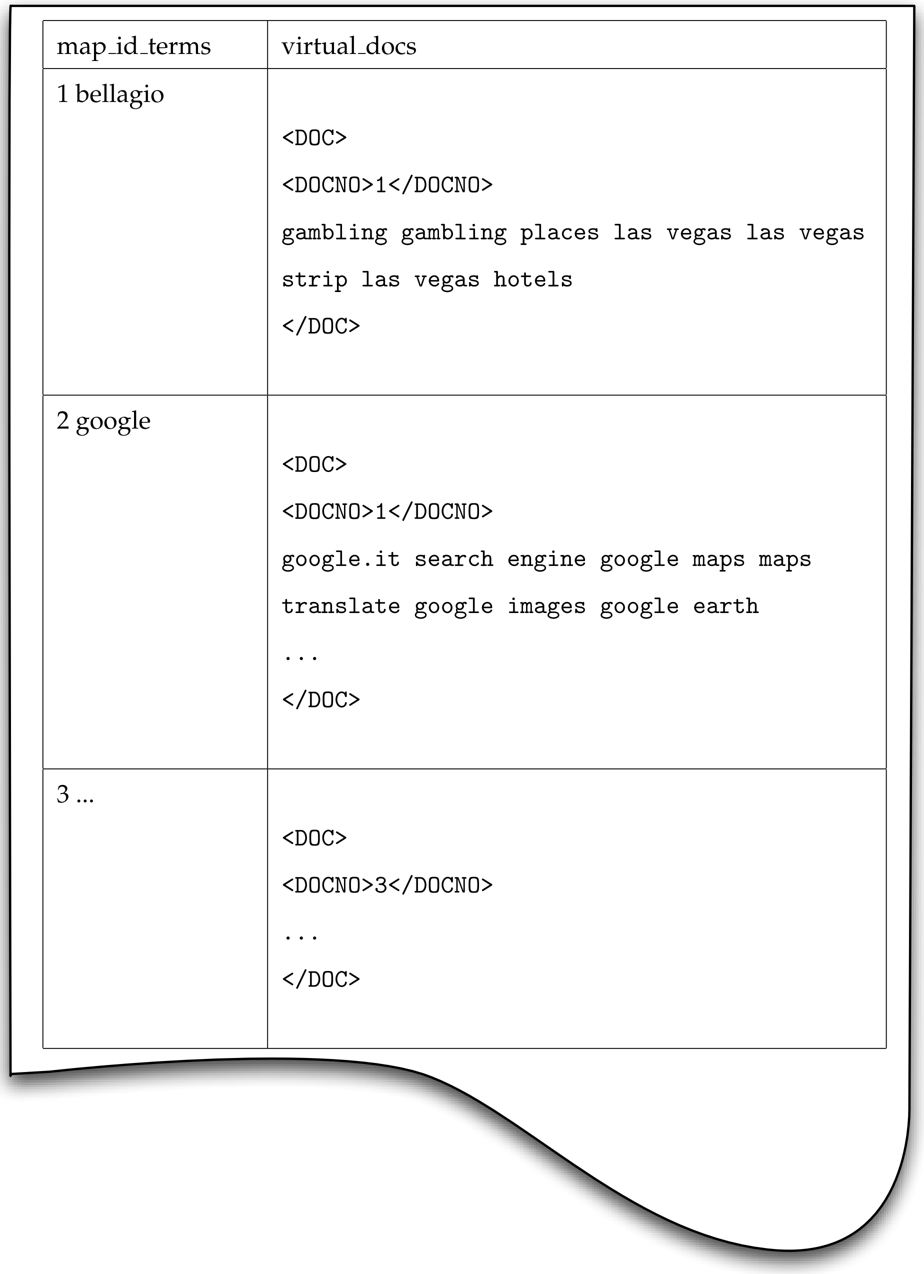
Notice that `doc_identifier` is supposed to be an *integer* value: however, in our IR-based model, the identifier of every session is its *final query*.

To make such model work, when our scripts create the virtual documents, a unique integer identifier is assigned to every final query. Hence, before indexing with Terrier, we have two files containing all the information to build the index and to make possible to know the associated query after retrieval.

For each virtual document we also store the number of sessions that includes; in other words, we save the number of times a query appears as final among all *satisfactory sessions*. As expected, the distribution of these frequencies follow a power-law. We will use this value for tuning the rank of our recommendations, as already explained in more detail in section 4.3.

Both *id-terms map* and *final queries frequencies* are stored in a SQLite database, (`http://www.sqlite.org`), a lightweight, single-file based database engine. We opted for this solution to perform our tests, because SQLite fit well our needs; by adopting other solutions it would be possible to get improvements in scalability and speed.

Following, we show the structure of id-terms map and of virtual documents files:



map_id_terms	virtual_docs

1 bellagio	`<DOC>`
	`<DOCNO>1</DOCNO>`
	`gambling gambling places las vegas las vegas strip las vegas hotels`
	`</DOC>`

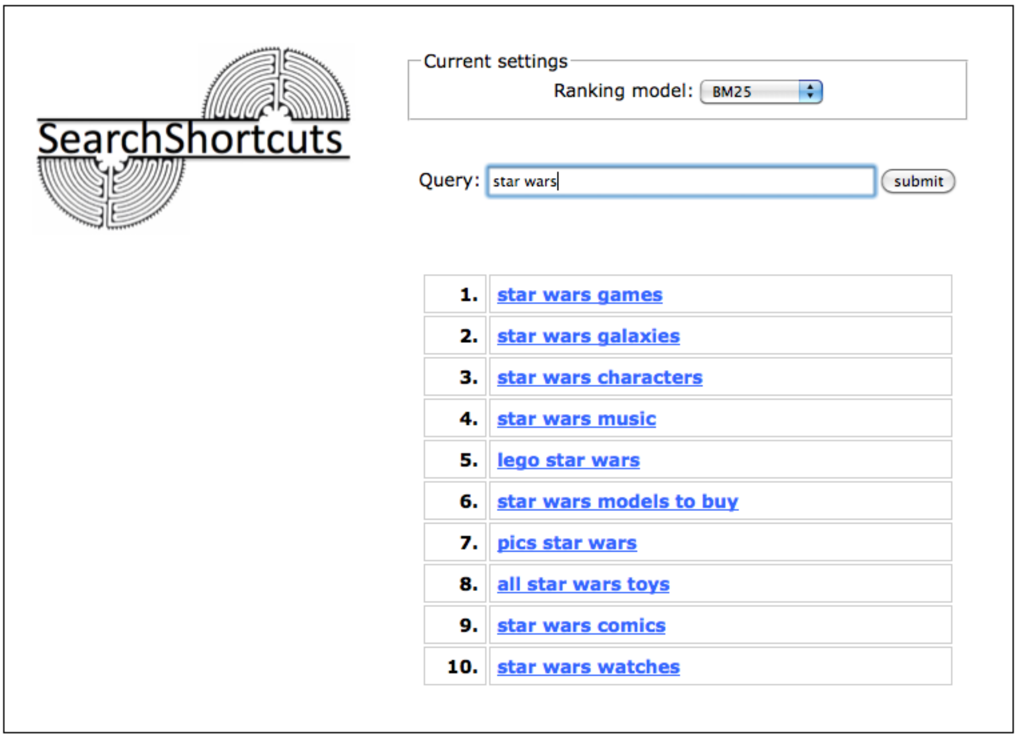
2 google	`<DOC>`
	`<DOCNO>1</DOCNO>`
	`google.it search engine google maps maps translate google images google earth`
	`...`
	`</DOC>`

3 ...	`<DOC>`
	`<DOCNO>3</DOCNO>`
	`...`
	`</DOC>`

5.5 SS Interactive Interface

Through the Terrier search engine we indexed the resulting $1,191,143$ virtual documents, and the index was made available for our testing purposes.

The possibility of processing queries on such index is provided to interested readers through a simple web interface available at the address `http://searchshortcuts.isti.cnr.it`. The web-based wrapper accepts user queries, interact with Terrier to get the list of final queries (id of virtual documents) provided as top-k[1] results, and retrieves and visualizes the associated query strings.



Current settings
Ranking model: BM25

Query: star wars

1.	star wars games
2.	star wars galaxies
3.	star wars characters
4.	star wars music
5.	lego star wars
6.	star wars models to buy
7.	pics star wars
8.	all star wars toys
9.	star wars comics
10.	star wars watches

Figure 5.1: A sample query and its relative recommendations provided by the SS web interface

[1] k is set to 10 in our experimental settings.

The web interface is developed in *PHP5*, and it acts as a wrapper to the interactive terrier command-line interface; the input query is sent to the interactive version of Terrier, which reads a previously defined configuration shown in the listing reported in section 5.3. The web interface allows to change the ranking model, although, as we will explain in section 4.3, this is not the global ranking value for the suggestions.

5.6 Results processing

The output produced by Terrier, including document identifiers and their IR-Ranks, is then processed by a PHP script following these steps:

- **extraction of the recommended query** string by matching the document identifier contained in the id-terms map table, stored in the SQLite database;
- **results filtering**, using some techniques described below;
- **results reordering**, sorting them by the rank value computed as explained in detail in section 4.3.

As the careful reader may have noticed, in the listing proposed in section 5.5, the parameter `matching.retrieved_set_size` is set to 50, while we previously stated that in our experimental settings we set $k = 10$; the main reason is that we always retrieve *at most* 50 results from Terrier, but then we filter out some results with some criteria explained below, showing in the end only the *top-k*.

The main purpose of filtering is to provide a basic **topic diversification**: in just 10 recommendations provided, we don't want to allow very similar queries such as, for example, *"Yahoo! mail"* and *"Yahoo mail"*; it won't be useful to the user, thus we want to discard one of those, creating room for another, different suggestion. This solution also avoids that the almost-useless suggestion "Yahoo!" is provided when the input query is "Yahoo".

Every recommendation is compared with all the recommendations given until then, and if their *Levenshtein distance*, or edit distance, is shorter than a threshold c_θ, the shortest query is discarded. Experimental tests with human-assessed results show that $c_\theta = 2$ behaves good.

Chapter 6

Evaluation methodology

6.1 Evaluation models

The evaluation of recommender systems effectiveness is an hard task that is usually addressed by means of *user-studies* or through the adoption of some performance metrics. In many works, for example [2, 11, 4], manual assessment of results is the most reliable evaluation of effectiveness, however limited to a small test set. Usually, in these cases, the results obtained are submitted to human judges, a role often played by the researchers themselves, who patiently spend some of their time assigning values and labels. Some other evaluations are based on performance metrics [15], for example the well known precision and recall measurement of results. However, depending on how the object algorithm was designed, it is not always possible to apply these metrics; furthermore, unfortunately, both these methodologies may lack of generality and incur in the risk of being over-fitted on the system object of the evaluation. The evaluation methodology proposed and used in this work aims to solve the above issues, still maintaining a simple human results assessment task which guarantees transparency as being possible to evaluate by everyone.

6.2 TREC topics coverage

The idea is based on exploiting the query topics and the human judgements provided by the National Institute of Standards and Technology (NIST, an agency of the U.S. Commerce Department), for running the TREC diversity track 2009 (`http://trec.nist.gov/data/web09.html`). The TREC Web Track explores and evaluates Web retrieval technologies: the 2009 Web Track includes two different tasks, a traditional *adhoc* retrieval task and a *diversity* task. We are interested to the latter: in particular, the goal of this diversity task is to return a ranked list of pages that together provide complete coverage for a query, while avoiding excessive redundancy in the result list. Even if in this work we are not interested to participate to the Web Track, we noticed that for the purposes of the TREC diversity track, the NIST assessors provided 50 queries, and, for each of them, they identified a representative set of subtopics covering the whole spectrum of different user needs/intentions. Subtopics are based on information extracted from the logs of a commercial search engine, and are roughly balanced in terms of popularity.

Obviously, the queries chosen are very different and from various categories: *ambiguous* or *faceted* in order to allow the overall performance of diversification methods to be evaluated and compared. Ambiguous queries are those that have multiple distinct interpretations, while faceted ones refer to a single meaning, but from different points of view. For example, the query ″*KCS*″ is considered an ambiguous query, because it could be related to the ″Kansas City Southern railroad″, or ″Kanawha County Schools in West Virginia″, or ″Knox County School system in Tennessee″, or even something else; on the other side, a query like ″*Volvo*″ is considered faceted because all its subtopics are somehow related to the Swedish car company and (almost surely) to nothing else. When selecting the subtopics, strange and unusual interpretations and aspects were avoided as much as possible; the set of subtopics is intended to be representative, not exhaustive, with the number of subtopics per topic ranging from three to eight, with a mean of 4.9.

Since diversity and topic coverage are the key issues also for the query recommendation task, we propose to use the same dataset for evaluating query suggestion effectiveness also. Given a query topic A with subtopics $\{a_1, a_2, \ldots, a_n\}$, and a query suggestion technique $\mathcal{T}$, we claim that the more the top-k queries suggested by $\mathcal{T}$ for A cover the human-assessed subtopics $\{a_1, a_2, \ldots, a_n\}$, the more $\mathcal{T}$ can be considered effective. To assess effectiveness of $\mathcal{T}$, we thus simply count how many subtopics are actually covered by the top-k suggestions generated by $\mathcal{T}$ for all the 50 TREC diversity track queries.

A last comment about this evaluation method: one might be tempted to say that the pertinence of the 50 TREC queries could be time-sensitive; some of these queries, in fact, refer to events or people of a certain period, (*e.g.* "obama family tree"), affecting the effectiveness of evaluations performed on datasets extracted in a different period, as we do. Our dataset, for instance, is extracted from MSN Search Engine in 2006, when the keyword "obama" was surely less searched than in 2009. However, the basic idea of this methodology is still valid, as long as all the results are obtained from the same dataset: if used to compare results from different approaches, and not as an absolute effectiveness value, even if some queries report poor or no results, this methodology still provide an accurate relative effectiveness measure.

In conclusion, this evaluation methodology has some clear advantages. It is based on a publicly-available test collection which is provided by a well reputed third-party organization. Moreover, it grants to all the researchers the possibility of measuring the performance of their solution under exactly the same conditions, with the same dataset and the same reproducible evaluation criterion.

6.3 Experimental Settings

In order to compare the performance of our *Search Shortcuts* (SS) solution with other state-of-the-art proposals, we selected two algorithms: *Cover Graph* (CG) proposed by Baeza Yates et Al. [3], and *Query Flow Graph* (QFG), proposed by Boldi et Al. [7]. These algorithms are recent and highly reputed representatives of the best practice in the field of query recommendation. The implementation of the CG algorithm was done by ourselves, while for testing the QFG query suggestion technique we used the original implementation kindly provided by the authors. Obviously, either CG and QFG models were trained with the same Microsoft RFP 2006 query log in order to conduct a fair comparison.

The relevance of each suggestion w.r.t. the TREC query subtopics was assessed manually. Given the limited number of queries and the precise definition of subtopics provided by NIST assessors, this manual evaluation task was not cumbersome at all.

The results have been human-assessed using a binary label: each recommendation has been labelled as *"related"* or *"unrelated"* to the query that produced it; additionally, if the recommendation is somehow associated to one or more topics, we consider such topic as *"covered"*. In the end, we have a list of related recommendations and covered topics; from such data, we obtain graphs and performance statistics, which are discussed in chapter 7.

Chapter 7

Results

Table 7.1 reports for each of the 50 TREC queries, the coverage (in percentage) of the associated subtopics measured for the top-10 suggestions returned by SS, CG, and QFG; the same data is plotted in the area chart shown in Figure 7.1. By looking at such results, we can see that SS outperforms remarkably its competitors. On 27 queries out of 50 SS was able to cover more than a half of the subtopics, while CG in no case reached the 50% of coverage, and QFG only on 5 queries out of 50. Moreover, SS covered the same number or more subtopics than its competitors in all the cases but 4, and in 34 cases the number of subtopics covered by SS was strictly greater. Only in 4 cases (query topics 15, 19, 25, and 45), QFG outperformed SS in subtopic coverage.

Table 7.2 and figure 7.2 reports instead the number of relevant suggestions returned among the top-10 ones generated by CG, SS, and QFG. A recommendation is considered relevant for a query if pertinent to the initial query. Also considering this performance metric our Search Shortcuts solution results the clear winner. All the $top - 10$ queries suggested by SS are relevant in 40 cases out of 50, against the 5 of both CG and QFG. The average number of relevant suggestions returned (a sort of P@10 metric) was 9.52, 4.72, and 2.46 for SS, QFG, and CG, respectively. This difference is really impressive, but we must consider that both CG and QFG are not able to generate suggestions for queries which were not encountered in the training log, and are thus not present in the model. SS on the other hand, adopts an IR-based approach based on a similarity score to select from the inverted index the final queries which are the closest to the current user query. For this reason, *the method results to be very robust to data sparsity which strongly penalizes the other two algorithms, and it is able to produce significant suggestions also for singleton queries which were not previously submitted to the WSE*.

We recall that singleton queries account for almost half of the whole volume of unique queries submitted to a WSE, and are often the hardest to answer since they ask for "rare" or badly expressed information needs. The possibility of suggesting relevant alternatives to these queries is more valuable than the one of suggesting relevant alternatives to frequent queries, which express common and often easier to satisfy needs.

Table 7.1: Subtopics coverage for the 50 TREC queries, shown in percentage (truncated). Comparison between three algorithms.

TREC query	CG	SS	QFG	TREC query	CG	SS	QFG
1	0	.33	0	26	0	.75	0
2	0	.50	0	27	.16	.50	.33
3	0	.66	.66	28	0	.60	.40
4	0	.16	0	29	0	.40	0
5	0	.25	0	30	0	.66	.16
6	.20	.40	0	31	0	.75	.25
7	0	0	0	32	0	.60	0
8	0	.75	.50	33	0	.50	.25
9	.16	.50	.33	34	0	.50	0
10	.12	.25	.12	35	0	.33	0
11	0	.50	0	36	0	.25	.25
12	0	.25	.25	37	.40	.60	0
13	.14	.14	.14	38	0	.33	.33
14	.20	.80	.40	39	.20	.20	.20
15	.16	.16	.33	40	0	1	0
16	.25	.25	0	41	0	.25	0
17	0	.50	.33	42	0	.50	.50
18	0	.80	0	43	0	.25	.25
19	.25	0	.25	44	0	.80	.60
20	.16	.33	.16	45	0	.16	.33
21	.20	1	.40	46	.33	.66	.33
22	.20	.20	0	47	0	.66	0
23	.14	.57	0	48	.40	.40	0
24	0	.75	.25	49	0	.33	0
25	.20	.50	.75	50	.33	1	.33

Table 7.2: Number of related recommendations among the top-k for the 50 TREC queries. Comparison between three algorithms.

TREC query	CG	SS	QFG	TREC query	CG	SS	QFG
1	0	10	0	26	0	10	0
2	4	10	0	27	9	10	5
3	0	5	4	28	0	10	10
4	0	10	10	29	0	10	0
5	1	10	0	30	0	10	5
6	5	6	4	31	0	10	9
7	0	10	0	32	0	9	0
8	1	10	9	33	1	10	5
9	5	10	8	34	2	10	9
10	1	9	7	35	0	10	8
11	0	10	0	36	0	10	6
12	0	9	7	37	7	8	6
13	10	10	4	38	0	10	7
14	3	10	9	39	5	10	5
15	10	10	6	40	0	3	0
16	10	10	8	41	0	9	1
17	0	10	10	42	1	10	5
18	0	10	0	43	0	9	2
19	4	10	4	44	0	9	9
20	10	10	5	45	0	10	7
21	6	10	5	46	6	10	4
22	5	10	6	47	0	10	0
23	10	10	0	48	3	10	0
24	0	10	10	49	0	10	0
25	1	10	10	50	3	10	7

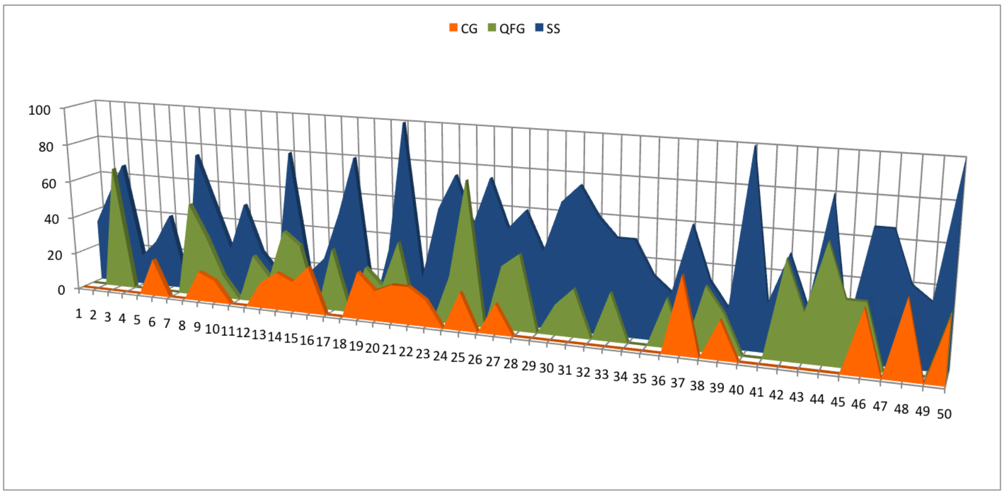


Figure 7.1: Coverage of the subtopics associated with the 50 TREC diversity-track queries measured on the top-10 suggestions provided by the Cover Graph (CG), Search Shortcuts (SS), and Query Flow Graph (QFG) algorithms.

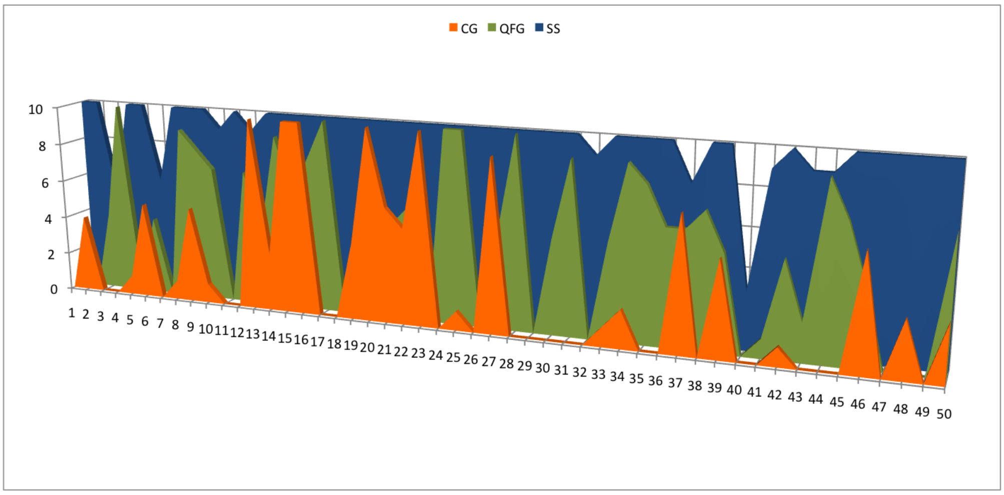


Figure 7.2: Number of suggestions relevant for some of the TREC query subtopics among the top-10 suggestions returned by the Cover Graph (CG), Search Shortcuts (SS), and Query Flow Graph (QFG) algorithms.

7.1 Recommendations for TREC queries

In the tables that follow, we present the results obtained for the 50 TREC queries, used in the evaluation model we proposed; as discussed above and in section 6.2, we count how many sub-topics are covered by the suggestions produced by our Search Shortcuts algorithm and the two other approaches, for comparison.

We report the top-10 suggestions provided by the three algorithm run against the MSN query log. For each query topic, the first column of the table lists the associated subtopics.

The manual evaluation of results produced the figures discussed above: SS computed mostly relevant suggestions covering a significant subset of the subtopics.

Table 7.3: Query suggestions provided by Search Shortcuts, Cover Graph, and Query Flow Graph for all the 50 TREC diversity-track query topics.

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
1 - *obama family tree* (faceted)			
1. Find the TIME magazine photo essay ”Barack Obama’s Family Tree”. 2. Where did Barack Obama’s parents and grandparents come from? 3. Find biographical information on Barack Obama’s mother.	family trees family tree maker free family tree barach obama family family tree chart blank family tree free family tree chart family tree baltimore maryland family tree designs	*no suggestion provided*	*no suggestion provided*
2 - *french lick resort and casino* (faceted)			
1. Find the homepage for French Lick Resort and Casino. 2. What casinos are located within a day’s drive of French Lick Resort and Casino? 3. What jobs are available at French Lick Casino and Resort? 4. Are there discounted packages for staying at French Lick Resort and Casino?	resorts casino island resort and casino riverside casino and golf resort french lick resorts french lick resort package french lick indiana visitors indianapolis getaway french lick french lick springs villas indiana french lick campground	hospitality hound vujc georgia perimeter college online degrees junior colleges in cobb county kim yang low cost photography classes sharona nolton social security office lithonia ga southern poly university in atlanta ga	french lick french lick indiana french lick, indiana french lick in
3 - *getting organized* (faceted)			
1. Find tips on getting organized, both reducing clutter and managing time. 2. Take me to the Container Store homepage. 3. Find catalogs of office supplies for organization and decluttering.	organic organic chemistry get organized how to get organized what is organic how to organize an office pipe organ organizing home organization products organ donation	eliminating clutter north fork realty office organization office layout how to organize an office daytimer www.target.com padfolio renees garden seeds aveda hair care products	*no suggestion provided*

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
4 - *toilet* (faceted)			
1. What different kinds of toilets exist, and how do they differ? 2. I'm looking for companies that manufacture residential toilets. 3. Where can I buy parts for American Standard toilets? 4. How do I fix a toilet that isn't working properly? 5. What companies manufacture bidets? 6. I'm looking for a Kohler wall-hung toilet. Where can I buy one?	toto toilets toilet partitions old fashion toilets antique toilets installing a toilet mansfield toilets toilet-to-go cadet toilets composting toilets toilet history	composting toilets ladies bathroom sigh toilet accessories toilet girls toilet history installing a toilet cartoon toilet tib toilets ladies bathroon sign	*no suggestion provided*
5 - *mitchell college* (faceted)			
1. Find the homepage for Mitchell College. 2. Find the homepage for the athletics department at Mitchell College. 3. Find web pages that compare Mitchell College to other colleges in Connecticut. 4. Find information on admissions to Mitchell College. How do I become a student there?	paul mitchell mitchell daily republic joni mitchell mitchell community college mitchell gold beverley mitchell jacks campers mitchell mitchell fuerst robert mitchell mitchell sd	mitchell	*no suggestion provided*
6 - *kcs* (ambiguous)			
1. Find the homepage for the Kansas City Southern railroad. 2. I'm looking for a job with the Kansas City Southern railroad. 3. Find the homepage for Kanawha County Schools in West Virginia. 4. Find the homepage for the Knox County School system in Tennessee. 5. Find information on KCS Energy, Inc., and their merger with Petrohawk Energy Corporation.	kannapolis city schools kcs autocad applications kcs railroad shreveport louisiana kcs railroad http://kcs.kana.k12.wv.us rally 800 williams communications gps south charleston middle school	kaul tronics us cellular kcs autocad applications kannapolis city schools gdp kannapolis intermediate school gdp mexico nfb per capita income gdp cia bfsb	kansas city railroad kansas city southern railroad kcs railroad www.kcsi.com kansas city southern

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
7 - *air travel information* (faceted)			
1. What restrictions are there for checked baggage during air travel? 2. What are the rules for liquids in carry-on luggage? 3. Find sites that collect statistics and reports about airports, such as flight delays, weather conditions, etc. 4. Find the AAA's website with air travel tips. 5. Find the website at the Transportation Security Administration (TSA) that offers air travel tips.	air travel air pacific sherman travel student air fare air travel forums bel air travel cheap air travel european air travel prague to innsbruck www.travelzoo.com air travel spartan travel	*no suggestion provided*	*no suggestion provided*
8 - *appraisals* (ambiguous)			
1. What companies can give an appraisal of my home's value? 2. I'm looking for companies that appraise jewelry. 3. Find examples of employee performance appraisals. 4. I'm looking for web sites that do antique appraisals.	performance appraisal hernando county property appraiser antique appraisal appraisers in colorado appraisals etc appraisers.com find appraiser wachovia bank appraisals appraisersdotcom drive by appraisals	online appraisals	appraisersdotom employee appraisals real estate appraisals appraisers employee appraisals forms appraisers.com gmac appraisers beverly wv picket fence appraisal fossillo creek san antonio
9 - *used car parts* (faceted)			
1. I'm looking for online sites that sell car parts. 2. I'm looking for the car-part.com web site. 3. I want to find a salvage or junk yard. 4. I'm looking for parts for commercial vehicles such as heavy trucks and semis. 5. I'm looking for the autozone.com web site. 6. I want to find online sources for NAPA parts.	car parts car-parts.com r/c car parts www.car-part.com spalding car parts car parts.com 1999 honda car parts car parts orileys volkswagen car parts in ireland car parts stores in kentucky	ford bumpers used parts request helm publications wheel covers richie sambora salvage yards 1998 ford ford expedition front door mirror left side gmgood car parts telephone numbers	used auto parts buyer used auto parts "used car parts" car parts used used parts

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
10 - *cheap internet* (faceted)			
1. What are some low-cost broadband internet providers? 2. Do any internet providers still sell dial-up? 3. Who can provide inexpensive digital cable television bundled with internet service? 4. I'm looking for the Vonage homepage. 5. Find me some providers of free wireless internet access. 6. I want to find cheap DSL providers. 7. Is there a way to get internet access without phone service? 8. Take me to Comcast's homepage.	internet explorer cheap internet services (aol cheap cable internet cheap internet services atrial fib cheap wireless internet for laptops cheapest long distance, local, internet packages internet cheap cigarettes cheap books	atrial fib dsl att dsl accelerators centurytel dsl charterdsl corvallis public library d s l shipment delivery service diagnostic systems l	cheap internet services
11 - *gmap prep classes* (faceted)			
1. Who are some companies that offer GMAT prep classes? 2. I'm looking for some free sample GMAT exams to practice on. 3. I'd like to find some tips to help me do well on the GMAT. 4. I'm looking for the BeatTheGMAT blog and forums. 5. Take me to the VeritasPrep home page. 6. What's the difference between the GRE and the GMAT?	gmat gmat prep nj gmat atlanta free gmat test prep psat prep classes kaplan prep courses sat prep class ' maynard, ma coding ccs prep class gmat+help sat test prep classes in virginia	*no suggestion provided*	*no suggestion provided*
12 - *djs* (faceted)			
1. I'm looking for DJs that specialize in hip-hop music. 2. I want to hire a DJ for a wedding. 3. How do I become a radio disc jockey? 4. What jobs are available for disc jockeys?	thunder and lightning djs djs, raleigh, nc atlanta black djs djs unlimited in houston djs in waco, tx djs wanted los angeles djs teens pornography music djs elite entertainment wedding djs chicago	dj angel dj dave mccollough djskennesaw ga free dj link novaspace albums free dj links milk inc. wedding dj albums cline	*no suggestion provided*

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
13 - *map* (ambiguous)			
1. Take me to the Google Maps homepage. 2. Take me to the MSN Maps homepage. 3. I want to go to the Yahoo Maps homepage. 4. I'm looking for MapQuest's homepage. 5. I want to find free printable maps. 6. I want to find sources for satellite maps and live satellite photos. 7. I'm looking for an online world atlas.	maps msn map map of the usa aerial maps u.s. map topographic map europe map c&c generals maps c-map maps us	msn map peach creek texas real estate 4056 sweetbriar dr. martinez ga 30907 address cliffs valley course address map apache lake campsites archer daniels midland company area 51 air port nv. area map service auction rossville tn	map it mapquest.com maps map quest map quest.com map and directions maps & directions maps and directions directions map directions
14 - *dinosaurs* (faceted)			
1. Go to the Discovery Channel's dinosaur site, which has pictures of dinosaurs and games. 2. I'm looking for free pictures of dinosaurs. 3. I want to find pictures of dinosaurs that I can color in, as in a coloring book. 4. I'm looking for a list of all (or many of) the different kinds of dinosaurs, with pictures. 5. Take me to the homepage for the BBC series, "Walking with Dinosaurs".	dinosaur pictures dinosaur worksheets dinosaur games all about dinosaurs walking with dinosaurs poetry dinosaurs dinosaur clip art trooden dinosaurs dinosaurs list tyrannosaurus dinosaur	dinosaur clip art dinosaur pictures ....elephant big t rex birthdays toys books on dinosaurs dinosaur color dinosaur coloring pages for kids dinosaur merchandise dinosaur photos	dinosaurs com all of the dinosaurs zoom dinosaur com
15 - *espn sports* (ambiguous)			
1. Take me to the ESPN Sports home page. 2. I'm looking for college football and basketball scores. 3. I want to find NBA basketball standings. 4. I'm looking for baseball scores and information on upcoming live broadcast games. 5. I'm looking for information on NASCAR races. 6. I'm looking for information on fantasy football leagues.	espn soccer espn mlb espn sports center oln sports good sports espn soccernet espn classic x-games comforter cdm sports sports legends	americas cup yachting cnn business espn outdoor sports figure skating news job service wisconsin kentuck derby listings kentucty nba las vegas odds tonight nhra cory mc picture north shore stone	espn espn.go.com "espn.com" espn.go buccigross apology entertainment sports network espn 1 espn college espn home espn home page

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
16 - *arizona game and fish* (faceted)			
1. Take me to the Arizona Game and Fish Department homepage. 2. What are the regulations for hunting and fishing in Arizona? 3. I'm looking for the Arizona Fishing Report site. 4. I'd like to find guides and outfitters for hunting trips in Arizona.	big fish games arizona fish and game wyoming game and fish nm game and fish arizona mvd arizona game and fish department california fish and game arizona saguaro lake arizona fishing crappie fishing san carlos lake, arizona	arizona saguaro lake a frame arizona lakes map cibeque creek arizona pro.sports news com. san carlos lake arizona www.navajofishandwildlife.org nm game and fish movie post arizona mvd	arizona department of wildlife arizona game & fish dept arizona game fish az dept. fish and game az fish &game az fishing report az game & fish az game & fish dept az game an fish az game and fish
17 - *poker tournaments* (faceted)			
1. I want to find information on the World Series of Poker. 2. I'm looking for a schedule of poker tournaments in Las Vegas. 3. Take me to the Full Tilt Poker website. 4. I'm looking for a schedule of poker tournaments in Atlantic City. 5. I want to find Texas Hold-Em tournaments. 6. Find books on tournament poker playing.	freeroll poker tournaments tropicana casino poker tournaments tilt poker tournament world poker tournament poker tournament timer poker tournament-binions bellagio poker tournament winners dd tournament poker .iso free poker tournaments cash poker tournaments in atlantic city	poker blogs learning poker paradise poker tournaments poker tournaments at harrahs superstar poker tournaments casino games poker forum poker tournaments at harrahs casino superstar poker tournaments results texas holdem	*no suggestion provided*
18 - *wedding budget calculator* (faceted)			
1. I want to find online guides, tips, and checklists for planning a wedding. 2. I am looking for spreadsheets or templates to help me tabulate a budget for a wedding. 3. I want to find some example wedding budgets. 4. I'm looking for information on planning a wedding shower, like theme ideas and budget guidelines. 5. How can I plan an inexpensive wedding?	budget wedding budget sheet sample wedding budgets budget calculator budget outside wedding wedding planning budget wedding bouquets how to have a celebrity wedding on a budget planning a wedding on a budget wedding costs	*no suggestion provided*	*no suggestion provided*

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
19 - *the current* (ambiguous)			
1. Take me to the homepage of The Current, a program on Minnesota Public Radio. 2. I'm looking for the homepage of The Current newspaper in New Jersey. 3. I want to find the homepage of The Current newspaper in Hartford. 4. I want to find the homepage of The Current magazine in San Antonio.	current current events current time current river current.com current labels current supreme justices ocean currents current science news currents catalog	the current magazine oklahoma thecurrent select account sa current thecurrent89.3 tahlequah ok miilife the hague water company apocalyptic horror san antonio college	3 the current 89.3 the current the current 89.3 thecurrent
20 - *defender* (ambiguous)			
1. I'm looking for the homepage of Windows Defender, an anti-spyware program. 2. Find information on the Land Rover Defender sport-utility vehicle. 3. I want to go to the homepage for Defender Marine Supplies. 4. I'm looking for information on Defender, an arcade game by Williams. Is it possible to play it online? 5. I'd like to find user reports about Windows Defender, particularly problems with the software. 6. Take me to the homepage for the Chicago Defender newspaper.	windows defender microsoft defender defender pro windows defender gdi multnomah defenders inc free defender pro the defender movie dynasty defenders msn defender windows defender troubleshooting	abyc.com defender land rover heinz 57 tshirt high definition plug in inflatable boats of the keys bit defender windows defender microsoft defender superbrightleds.com windows xp bitdefender	microsoft windows defender 'microsoft anti spyware' anit spy anti beta anti spy beta anti spy ware anti spyware beta 2 anti spyware microsoft anti spyware microsoft beta anti-spy ware
21 - *volvo* (faceted)			
1. I'm looking for Volvo's homepage. 2. Find reviews of the Volvo XC90 SUV. 3. Where can I find Volvo semi trucks for sale (new or used)? 4. Find a Volvo dealer. 5. Find an online source for Volvo parts.	volvo.com volvo usa volvo parts volvo suv volvo xc70 volvo of las vegas volvo marine k big volvo truck parts scottsdale volvo	volvo xc90 volvo trucks 03 volvo s60 a heart full of lies baby on board barbie lee photography centeal ohio minority trade fair diacro diesel powered suvs don beyer	volvo cars volvocars.com volvo suv ”volvo” volvocars www.volvo cars.com

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
22 - *rick warren* (faceted)			
1. Take me to Rick Warren's homepage. 2. I'm looking for the homepage for Rick Warren's book, "The Purpose Driven Life". 3. I'm looking for background and biographical information on Rick Warren. 4. I want to see articles and web pages about the controversy over Rick Warren's invocation at the Obama inauguration. 5. I want to read about the debate between John McCain and Barack Obama hosted by Rick Warren.	warren buffett rockbridge seminary what sin is rick warren rick astley rick bayless warren kimble rick warrens book bible study methods colorado springs christianity"rick warren rick boucher rick jeannaret	benny hin rockbridge seminary television evangelist platos closet help from the bible tv prechers platos closet in lexington ky platos closet plano diesel clothing fox tv church programs	purposedrivenlife.com www.purposedrivenlife.com www.rickwarren.com the purpose driven life
23 - *yahoo* (ambiguous)			
1. Take me to the Yahoo! homepage. 2. Take me to Yahoo! Mail. 3. I'm looking for the Yahoo! Messenger homepage. 4. Take me to Yahoo! Finance. 5. I'm looking for the Yahoo! Music homepage. 6. I want to log in to my Yahoo! account. 7. Find information about Yahoo!, the company.	my yahoo yahoo chat yahoo! finance yahoo mexico yahoo pool yahoo jobs yahoo.games yahoo e mail yahoo canada yahoo messanger	cablelynx wesh tv cupidbay ahoo gayphoenix.com ideal bite blog javairc gonzaga law school holt international craig list new hampshire	yahoo.com yahoo! http://yahoo.com www. yahoo.com yahoo -warehouses-employment and 1 http/____________ ireland: pitcures of clothing local roller blade stores near trenton, nj
24 - *diversity* (faceted)			
1. How is workplace diversity achieved and managed? 2. Find free activities and materials for running a diversity training program in my office. 3. What is cultural diversity? What is prejudice? 4. Find quotes, poems, and/or artwork illustrating and promoting diversity.	diversity in education diversity inclusion cultural diversity diversity test accepting diversity diversity poem diversity skills diverse learners presentation picture of diverse childern advantages of diversity	accepting diversity dispariging remarks diverse world diversity director diversity poem diversity test minority & women civil liberties inclusion gender and racial bias	*no suggestion provided*

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
25 - *euclid* (ambiguous)			
1. Find information on the Greek mathematician Euclid. 2. I'm looking for a source for Euclid truck parts. 3. Take me to the homepage for Euclid Industries. 4. Take me to the homepage for the Euclid Chemical company.	terex euclid euclid high school euclid algorithum euclid industries euclid computers euclid ohio dentists euclid municipal court euclid speed switch non euclid java euclid fish & seafood- ohio	euclid algorithum euclid automotive euclid brake euclid truck parts thearvinmeritor who is the father of geometry ythagoras arvinmeritor euclid industries archimedes	euclid chemicals
26 - *lower heart rate* (faceted)			
1. What causes the heart to beat faster or slower? 2. What is a normal heart rate when a person is resting? 3. How can I lower my heart rate? 4. Is a higher heart rate related to high blood pressure or cholesterol?	normal heart rate target heart rate heart problems heart rate fetal heart rate heart rate chart accelerated heart rate and pregnant how to figure heart rate heart rate calculate heart rate medication	*no suggestion provided*	*no suggestion provided*
27 - *starbucks* (faceted)			
1. Take me to the Starbucks homepage. 2. What is the balance on my Starbucks gift card? 3. Find the menu from Starbucks, with prices. 4. Find calorie counts and other nutritional information about Starbucks products. 5. Find recipes from Starbucks, either for making or using Starbucks products. 6. I'm looking for locations of Starbucks stores worldwide.	starbucks franchise starbucks locations starbucks benefits starbucks nutritional info starbucks jobs starbucks partners jobs at starbucks how to franchise starbucks starbucks airpot starbucks recipes	starbucks franchise starbucks partners latte starbucks menu caribou coffee baha bobs belmar blockbuster job applicants bowing bucksteep manor	starbucks.com how to maket art oregon starbuck starbuck's starbuck's coffee starbucks coffee www.starbucks www.starbucks coffee.com star bucks www.starbucks.com

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
28 - *inuyasha* (ambiguous)			
1. I'm looking for pictures and wallpaper images from InuYasha. 2. Find InuYasha anime episodes for download. 3. Find games based on the InuYasha series, either online or for game systems. 4. I'm looking for InuYasha fan forums and websites. 5. Find music from the InuYasha television series.	inuyasha pictures inuyasha world pics inuyasha radio inuyasha games inuyasha avatars inuyasha myspace inuyasha screensavers inuyasha in love inuyasha media pictures of inuyasha	inuyasha and kagome inuyasha pics inuyasha wallpaper amuro namie anime people gundam seed cagalli inuyasha and his girl friend inuyasha and kagome church of lemons inuyasha epesods inuyasha episode guide	*no suggestion provided*
29 - *ps 2 games* (faceted)			
1. Find reviews of PlayStation 2 games. 2. Where can I find cheat codes for PlayStation 2 games? 3. I'm looking for sites that announce new PlayStation 2 games. 4. Where can I buy used PlayStation 2 games? 5. What are the specifications of the PlayStation 2 console?	mouse drivers playstation 2 games ps 2 gaming cheats sp 2 ps 2 cheat codes diner dash 2 game game cheats for medal of honor european assault for ps 2 usb to ps/2 adapter ps/2 compatible mouse driver unplug mouse ps/2	*no suggestion provided*	*no suggestion provided*
30 - *diabetes education* (faceted)			
1. Find free diabetes education materials such as videos, pamphlets, and books. 2. Take me to the NIH National Diabetes Education Program homepage. 3. Take me to the American Association of Diabetes Educators homepage. 4. I'm looking for nutrition and diet information for diabetics. 5. Where can I get free diabetes education posters? 6. How can I become a diabetes educator?	diabetes diabetic education nutrition and diabetes education diabetes educator exam educating the insulin dependent diabetic international diabetes center educational materials diabetic diet diabetes education powerpoint american association of diabetes educators tele-ed program diabetes education	diabetes education survival skills nutrition and diabetes education diabetes education survival skills pdf diabetes survival skills pdf diabetes handhouts pdf	*no suggestion provided*

Query & subtopics	Search Shortcuts	Cover Graph [3]	Query Flow Graph [7]
31 - *atari* (faceted)			
1. I want to read about the history of the Atari 2600 and other Atari game consoles. 2. I am looking for a site where I can play old Atari games online. 3. Find information about classic Atari games. 4. Find information about Atari arcade games.	atari play the ataris pong atari machine atari emulator stella mystique atari 2600 atari.us/dragon ball z egypt pharoh atari atari's original asteroids game atari games	dranium atari battletank atari games beginings atari machine atari rller coaster tycoon 3 big easy spokane wa zool 2 atari games mortal kombat armageddon grand theft auto	*no suggestion provided*
32 - *website design hosting* (faceted)			
1. What are the cheapest web hosting companies? Who offers free web hosting? 2. Where can I register a domain name? 3. Find sites that offer free DNS hosting. 4. Find reviews of web hosting services, geared towards small business needs. 5. I'm looking for information and courses on designing web sites.	free website hosting website hosting msn ranking beta how do i host my own website financial website design website design monterey website design company military website designs design websites for teens web design hosting services.com.mx	*no suggestion provided*	*no suggestion provided*
33 - *elliptical trainer* (faceted)			
1. I'm looking for reviews of elliptical machines. 2. Where can I buy a used or discounted elliptical trainer? 3. What are the benefits of an elliptical trainer compared to other fitness machines? 4. What are the best elliptical trainers for home use?	elliptical machines elliptical machine reviews elliptical trainers with tv's elliptical elliptical trainer reviews consumer reports elliptical trainers increase running speed with elliptical trainer free elliptical training life fitness elliptical precor elliptical trainers	elliptical trainer calories burned inkjet cartridge kfrc radio elliptical machines stair climber calories burned kfrc radio live kfog radio kmart.com krty elliptical exercise machines	elliptical

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
34 - *cell phones* (faceted)			
1. What free phones are available from different vendors? 2. Go to AT&T's cell phones page. 3. Go to Verizon's page that lists phones for sale. 4. Find information on prepaid cell phones. What companies offer them? What kind of phones are available? 5. Go to Nokia's home page. 6. What cell phone companies offer Motorola phones? 7. Go to Sprint's page that lists phones for sale. 8. Where can I find information on buying unlocked phones?	samsung cell phones prepaid cell phones cingular cell phones cell phone reviews alltel cell phones sprint cell phones unlocked cell phones cell phone lookup nokia cell phones used cell phones	cell phones without plans camera phones cell phones without a plan a1 cellular advertisement ancient bell. ca cartoon cell phone cell phone batt cell phone companies in toronto cell phone history	cell phones and what the latest cell phones
35 - *hoboken* (faceted)			
1. Find restaurants in Hoboken. 2. Find the homepage for the city of Hoboken, NJ. 3. I'm looking for the history of Hoboken, NJ. 4. I'm looking for information on bars and nightclubs in Hoboken, NJ. 5. Find real estate listings for Hoboken, NJ. 6. Find a street-level map of Hoboken, NJ.	pet grooming - hoboken hoboken floors hoboken man dead hoboken apartments hoboken nj madisons in hoboken w hoboken residence prices hoboken chinese precious hoboken nj hotels lucie marciano hoboken nj	hoboken popullation hoboken nj hoboken nj hotels marciano law hoboken nj north bergen nj hotels within 50 miles of manhattan ny lucie marciano hoboken nj north bergen nj hotels super 8 hotels best western hotel reservations	*no suggestion provided*
36 - *gps* (faceted)			
1. Find reviews of GPS units and car navigation systems. 2. Take me to the Garmin homepage. 3. Take me to GPS Magazine. 4. Find reviews of digital cameras with built-in GPS.	magellan gps sony gps gps garmin palm gps www.gps.edu gps for teenagers gps microsoft satelitte pictures of addresses cheap gps ford gps	gps garmin gps tracker magellan gps 50 ra air grinder aplications of triginomitry bar code mount best motorcycle gps bluetooth gps buying gps compare prices for	*no suggestion provided*

Query & subtopics	Search Shortcuts	Cover Graph [3]	Query Flow Graph [7]
37 - *pampered chef* (faceted)			
1. Take me to the Pampered Chef homepage. 2. How do I host a Pampered Chef cooking show? 3. How do I become a Pampered Chef consultant? 4. Find some recipes from The Pampered Chef. 5. I would like to find reviews of The Pampered Chef programs, products, and recipes.	pampered chef recipes the pampered chef pamper chef parties glastonbury ct zip code pampered chef lemonade pampered chef consultants free pampered chef recipes joyce's fine cooking swing arm lamp pampered chef may guest specials	the pampered chef 1509611 ontario inc. chicken caesar pizza dale yeckley glastonbury ct zip code nancy lambert pampered chef catalog pampered chef cookware pampered chef distributor in nampa idaho pampered chef lisa fritz	"pampered chef" www.pamperedchef.com the pampered chef pamperedchef.com pamperedchef pampered chef.com pampered chef recipes
38 - *dogs for adoption* (faceted)			
1. Find organizations that offer dogs for adoption. 2. Take me to the homepage of the Humane Society. 3. What should I know about adopting a dog?	dog adoption gwinnett county dogs for adoption dogs for adoption georgia dog adoption atlanta dog adoption in mn oklahoma dog adoptions dogs for adoption sacramento california faces dog adoption of springfield pets on parade arizona oahu dog adoption	dogs for adoption in illinois pet adoption in joplin mo. st bernard rescue dog.com st bernard puppies cliff notes st bernard casa grande dmv ddog.com how to track down your family	*no suggestion provided*
39 - *disneyland hotel* (faceted)			
1. What hotels are near Disneyland? 2. Find information on package deals from hotels near Disneyland. 3. Find reviews of Disneyland hotels. 4. Find special offers such as reduced ticket rates at Disneyland. 5. Take me to the hotel listing at the Disneyland web site.	disneyland disneyland hotels map hotels near disneyland disneyland hotels bunkbeds disneyland hotels - grand californian disneyland hotel fragrance disneyland hotels, anaheim hotels near disneyland, california hotels inside disneyland in california hilton hotel in disneyland	vegas towers casino kinecta disneyland hotels disneyland six flags hilton gv la canasta ocxoffroad.com pacific resource credit union ahaheim fairfield inn hotel	vegas towers casino kinecta disneyland hotels disneyland six flags hilton gv la canasta ocxoffroad.com pacific resource credit union ahaheim fairfield inn hotel

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
40 - *michworks* (faceted)			
1. Take me to the michworks Michigan Talent Bank homepage. 2. What jobs are available in Michigan? 3. Find career resources and information on job seeking in Michigan. 4. Find information about services available to the unemployed in Michigan.	www.michworks.org michworks.org www.michworks.com anderson speedway	grand haven weaterh grand haven michigan weather statistics	*no suggestion provided*
41 - *orange county convention center* (faceted)			
1. Take me to the Orange County Convention Center homepage. 2. Find a schedule of events taking place at the Orange County Convention Center. 3. How do I reserve the Orange County Convention Center for an event? 4. What hotels are near the Orange County Convention Center?	dallas convention center orange county performing arts center tampa convention center la convention center philadelphia convention center orlando, fl+embassy suites hotel hawaii convention center tucson convention center great american homeowners challenge marriott courtyard orange county convention center	international drive dallas convention center hotrod.com trinity rail express hemming news.com oldcartrader magical midway resturants on international drive universal studios islands of adventure discovery cove	*no suggestion provided*
42 - *the music man* (faceted)			
1. Find lyrics for songs from The Music Man. 2. Find current performances of The Music Man. 3. Find recordings of songs from The Music Man. 4. I'm looking for the script for The Music Man.	till there was you musical music man lyrics the music man " soundtrack the music man summary elephant man music 70's music rubberband man encino man, songs music man lyrics free music on msn music man trouble in river city	the music man on broadway the music man summary state fair musical till there was you dizzy gilelespee oysters rockefeller recipe archnid female whale brewski fats dominos first name	the music man movie

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
43 - *the secret garden* (faceted)			
1. Find reviews of the book The Secret Garden. 2. Find reviews of the various TV and movie adaptations of The Secret Garden. 3. I'm looking for biographical notes about Frances Hodgson Burnett. 4. Find information about the Broadway musical The Secret Garden.	secret garden cd secret garden-quincy jones lyrics silent screams bdsm savannah secret gardens secret house vineyards winery my secret garden secret garden springfield mo secret garden musical songs secret garden party secret garden" lyrics musical	the secret garden play orchestra the secret garden stroy keys wout wynnants door keys wout wynants dod schools honda dealerships in maryland keys to hell keys tucson	*no suggestion provided*
44 - *map of the united states* (ambiguous)			
1. Find US road maps. 2. Find detailed geographic maps of the United States. 3. Find political maps of the United States showing the states and their capitals. 4. Find printable maps of the United States. 5. Find a black-and-white outline map of the United States such that a child could color.	map of united states blank map of the united states map of united states of america united states maps outline map of the united states united states of america map printable united states map united states region map political map of the united states updated wrestling news	blank map of the united states flag of the united states free map of the united states highway map of the united states kel_tan163 map of the missouri map of the u map of the united states state names map of time zones in united states map of united states by region	*no suggestion provided*
45 - *solar panels* (faceted)			
1. What kinds of solar panels and photovoltaic cells are there? 2. Go to the JA Solar homepage. 3. Go to the Solarfun homepage. 4. Find information about solar panels that I can install on my home. 5. Go to the homepage for Evergreen Solar. 6. Find information about nanotechnological solar power.	solar panels how to make a solar panel free solar panels rv solar panel solar panels boats how to build solar pool panel building solar panels solar panel manufact solar panels branson solar panels for home use	drawbacks of building with adobe flexable solar panels free solar panels high meadow ranch home solar panels shelby county al library solar panels boats solar panels branson solar panels electris solar power generators	*no suggestion provided*

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
46 - *alexian brothers hospital* (faceted)			
1. Go to the Alexian Brothers Health System homepage. 2. Find information about the Alexian Brothers lay Catholic order. 3. Find Alexian Brothers hospitals.	big brother brother alexian brothers san jose alexian brothers hospital network brother.com alexian brother eating disorders program band of brothers alexian brothers medical center illinois healey brothers alexian.org	24 hour health clinic in schaumburg illinois alexian brothers san jose rumuda ranch 24 hr emergency care near schaumburg illinois immediate care services near schaumburg illinois google barbizon star power site:www.google.com google icy madness google home	alexian brothers alexian brothers hospital elk grove il alexian brothers hospital, illinois alexian brothers illinois alexian.org alexian brothers medical center
47 - *indexed annuity* (faceted)			
1. What is an indexed annuity? What are their advantages and disadvantages? What kinds of indexed annuities are there? 2. Where can I buy an indexed annuity? What investment companies offer them? 3. Find ratings of indexed annuities.	annuities annuity lincoln benefit life" index annuity travelers annuity aig annuity what is an annuity aig annuities annuity vs bond equity deferred annuities fidelity guaranty annuity	what commpetitor took bryant gumbel to dinner the night after the cbs early show debuted what competitor took bryant gumbel to dinner who is byrnat gumbels competitor who is byrnat gumbel competitor competitor took bryant gumbel to dinner cbs early show debuted 1965 portrait of the assassin	*no suggestion provided*
48 - *wilson antenna* (faceted)			
1. Go to the Wilson Antenna homepage. 2. What kinds of CB antennas does Wilson Antenna sell? 3. Where can I buy used Wilson Antennas? 4. What is the best antenna from Wilson for a big truck? 5. Find reviews of Wilson antennas.	wilson antenna for cingular 8125 wilson cb antenna tips antenna decor scanner antennas antenna noise factor cadillac srx antenna antenna decorations cellular antennas dipole antenna everhardt antennae	*no suggestion provided*	wilson antennas wilson cb antenna wilsonelectronics.com

Query & subtopics	**Search Shortcuts**	**Cover Graph [3]**	**Query Flow Graph [7]**
49 - *flame designs* (ambiguous)			
1. Find free flame design clipart I can use on a website. 2. How do I make realistic flame images using Photoshop? 3. I'm looking for good flame tattoo designs. 4. Find pictures of flames and fire. 5. I want to find flame design decals I can put on my car or motorcycle. 6. I'm looking for flame design stencils.	flames flame fonts penny flame flame illustration flaming text flaming fonts tattoo flame designs flame pattern comforters in flames band flame art	*no suggestion provided*	*no suggestion provided*
50 - *dog heat* (ambiguous)			
1. What is the effect of excessive heat on dogs? 2. What are symptoms of heat stroke and other heat-related illnesses in dogs? 3. Find information on dogs' reproductive cycle. What does it mean when a dog is "in heat"?	heat heat cycle of a dog can a dog be spayed in heat dogs/heat dog in heat symptoms do female dogs howl in heat cycle dog heat exhaustion signs of a female dog in heat heat cycle of dogs when do dauschound go into heat	dogs in heat do female dogs howl in heat cycle how to tell if your dog is pregnant can a dog be spayed in heat memphis spaying pet vacs pet vax myspace yspace myspac	female dogs in heat how long? how often do dogs come in heat female dogs in heat

Chapter 8

Conclusions

We have proposed a very efficient solution for generating effective suggestions to WSE users based on the model of *Search Shortcuts*. Our original formulation of the problem allows the query suggestion generation phase to be re-conducted to the simple processing of a full-text query over an inverted index. Final queries of most similar satisfactory sessions are thus efficiently selected to be proposed to the user. An additional contribution of this work regards the evaluation methodology used, based on a publicly-available test collection provided by a highly reputed organization such as the NIST. The proposed methodology is objective and very general, and, if accepted in the query recommendation scientific community, it would grant researchers the possibility of measuring the performances of their solutions under exactly the same conditions, with the same dataset and the same evaluation criterion.

On the basis of the above evaluation method, the algorithm (SS) proposed in this work remarkably outperformed two well-known representatives of the best practice in the field of query recommendation in almost all the tests conducted. In particular, suggestions generated by SS covered the same number or more TREC subtopics than its two counterparts in 46 cases out of 50. In 34 cases the number of subtopics covered by SS suggestions was strictly greater. Only in 4 cases QFG outperformed SS. Also when considering the number of relevant suggestions among the top-10 returned, SS resulted the clear winner with an average number of relevant suggestions equal to 9.52, versus 4.72 and 2.46 for QFG, and CG, respectively. Moreover, differently from its competitors, SS resulted to be very robust w.r.t data sparsity, and can produce relevant suggestions also to queries which were not present in the query log used for training.

8.1 Extracting sessions with Query Flow Graph

Another application of the Query Flow Graph described in [7] is *finding logical sessions*. This is a very important problem, as it allows improving of query-log analysis, user profiling and more: in the current Search Shortcuts implementation we use a naive approach to extract user sessions, so its performances could be improved by introducing a more sophisticated way to segment the query log into user sessions (also called *chains*). We already described in section 2.5 the weighting models used in the query flow graph while operating in the task of query recommendation: for the second application - finding chains - the authors use the first weighting scheme, the one based on chaining probabilities.

They separate the problem of finding chains into two subproblems: *session reordering* and *session breaking*. The session reordering problem is to ensure that all the queries belonging to the same search mission are consecutive; in fact, the authors allow chains to be intertwined in a supersession[1]. Then, the session breaking problem is much easier, as it only needs to deal with non-intertwined chains.

[1] For supersession definition, cp. section 2.5.

The first subproblem is modelled as an instance of the Asymmetric Traveler Salesman Problem (ATSP): instead of trying to produce exact solutions, they adopt a greedy heuristic that every time chooses the arc with minimum weight going out of the current node.

After reordering, session breaking corresponds to the determination of a series of cut-off points in the re-ordered session. They apply a threshold η to break a reordered session whenever the weight of the edge connecting q and q' is less than η.

We performed a preliminary study on session breaking using the query flow graph, applying different thresholds η; as explained by the authors of query flow graph, we performed a first task of session splitting based on breaking the list of queries from the same user using a time threshold $t_\theta = 30$ minutes against the same MSN query log we used in our experiments (see section 5.1); from these sessions we extracted a sample set including the first $10,000$ sessions, and we got the following results:

	$\eta = 0.1$	$\eta = 0.2$	$\eta = 0.4$	$\eta = 0.75$
total sessions	$10,222$	$10,311$	$10,639$	$12,442$
session length ≥ 2	$6,152$	$6,149$	$6,142$	$6,081$

These results show that applying session reordering + session breaking on $10,000$ 30-minutes sessions, we obtained a larger number of sessions, which means that the algorithm based on the query flow graph split some of them. Raising the threshold, the number of sessions obtained raises as well: anyway, the number of sessions with 2 queries or more is almost the same. This leads us to the assumption that adjusting this threshold could be useful to ”clean up” the query log, removing from sessions some noise originated by *weakly connected* queries.

Finally, we run the session reordering and breaking processes applying the threshold $\eta = 0.75$ on the whole query log, obtaining $9,214,476$ total sessions. This result is actually smaller than the one we obtained[2] with our simple *5-minutes based* procedure ($9,461,423$). As a first hypothesis, we believe that this way to generate sessions could improve Shortcuts quality as well.

8.2 Future work

As future works we intend to investigate if the sharing of the same final queries induces a sort of "clustering" of the queries composing the satisfactory user sessions. By studying such relation, which is at the basis of our query shortcut implementation, we could probably find ways to improve our methodology. Moreover, we currently use a very simple session splitting technique based on a fixed time-window, and we plan to study the possible enhancements to the effectiveness of suggestions deriving from the exploitation of more precise session splitting heuristics such as the ones discussed in [14] or in section 8.1. At the moment, in fact, we did not perform any evaluation task on the quality of Shortcuts obtained from sessions extracted with query flow graph: an interesting improvement of our Shortcuts algorithm surely relies on a more effective sessions extraction. Previous studies about multitasking in web searches [8], [28] and [27] showed that real search engine users don't limit their searches to a single topic within a session. Hence, a topic identification process could be an important area to investigate to improve Shortcuts quality: multitasking sessions, in fact, tend to introduce noise into the bag-of-words associated to the final query, which represents the actual query recommendation.

We made another consideration about scalability of Shortcuts generation: with the current implementation, we have to pre-process the whole query log

[2] Cp. section 5.1

to extract the Shortcuts; a way to avoid to reprocess all the data should be implemented in future versions of this algorithm. An incremental indexing could be a simple and effective solution: at defined time intervals, we could merge the new users' satisfactory sessions with the already extracted virtual documents, then simply re-index them.

In the current version of Search Shortcuts algorithm we don't take into account the history of the input queries for which we want to generate suggestions: for example, if a user asks suggestions for ″apple″, and ″banana″ in the following query, we would want to avoid recommendations related to *apple computers*, and prefer suggestions about *the fruits*. Search-history driven topic disambiguation is a good basis to develop in future improvements of Shortcuts algorithm.

Finally, some considerations about the web interface: a useful expansion to introduce would be to add a wrapper for *actual* search engines: people willing to perform a search would use our interface, which would behave just as a man-in-the-middle between the *real* search engine and the users. This would give us the possibility to both suggest our own recommendations and enrich our knowledge base with *real* user sessions. In other words, it would let us collect useful data to improve the quality of Shortcuts.

Appendix A

Glossary

Association Rules mining is the process of extraction of relations between elements in large data sets. Widely used in marketing field to discovery information patterns about users tastes and purchases, A.R. mining can be transposed in IR world, for example, to find relations among the queries in a search engine query log, improving query recommendation. The basic idea is that if a certain percentage of users who searched information about topic T1 and searched for topic T2 in the same session, T1 and T2 are related topics. An A.R. is formally written as $X \Rightarrow Y$.

Click-Through Data is an important part of a search engine query log, that includes all the information about user activity related to clicks. This information is at the base of different approaches with the intent of improving the results provided (e.g. in implicit relevance feedback) or in the recommendation of related queries.

Collaborative Filtering is the process of filtering information or patterns using techniques involving collaboration among multiple sources. Typically used in large data sets, C.F. tries to make automatic predictions about the interests of a user by collecting taste information from many users. The more information there is in the data set, the more accurate will probably be the prediction.

Precision and Recall are classification in IR world.

- *Precision* is the number of relevant documents retrieved divided by the number of documents retrieved;
- *Recall* is the number of relevant documents retrieved divided by the total number of existing relevant documents.

They both can be used to give a measure of the performances of a IR system, and they are often considered in the Precision/Recall tradeoff: some features can increase one of them by decreasing the other one.

Query Expansion includes all the techniques used to improve the quality of the results in a search engine, or in a information retrieval system in general. Q.E. is the process of reformulating a user query by evaluating and expanding it in order to match additional documents. In this way, the results may not exactly match the original query, but they hopefully better fit the user needs. Examples of Q.E. are use of synonyms (and searching for synonyms besides the original query), stemming, spelling correction. Q.E. methods usually increase recall at the expense of reducing the precision.

Query Chain is a sequence of queries about the same topic. It is related to the idea of the refinement process manually performed by a user, that could lead to the discovery of more information if compared to considering the queries in the chain independently. Sometimes it is possible to use query chains and query sessions as synonyms, but only if assumed that a session, which is time-based and not topic-based, contains searches only about one topic.

Query Clustering: there are several ways to make cluster of queries; a cluster is a set of items that are similar each other in a formally defined way. Thus, there must be a definition of similarity, which makes the difference between clustering methods.

Relevance Feedback is a feature of some IR systems, based on the idea of taking the results initially returned from a given query and using information about whether or not those results are relevant to perform a new query. R.F. can be classified in explicit feedback, implicit feedback and pseudo (or blind) feedback.

- *Explicit feedback*: users explicitly mark relevant and irrelevant documents;
- *Implicit feedback*: the system attempts to infer user intentions based on observable behavior (e.g.: click-through data, time spent on a page, input reformulation);
- *Blind feedback*: the idea is to take the top n documents and assume they

are relevant, and then perform the query as usual. If the initial hits are good, blind feedback will improve the results.

When R.F. is used to benefit all users of the search engine, then it can be considered collaborative filtering. Relevance feedback from one user indicates that a document is considered relevant for their current need. If that user's information need can be matched to others' information needs, then relevance feedback can help improve the others' search results.

Stemming: is the process for reducing inflected (or sometimes derived) words to their stem, base or root form, e.g. *getting* → *get*, or *dogs* → *dog*. The S. process is useful in search engines for query expansion or indexing and other natural language processing problems.

Bibliography

[1] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE TKDE*, 17(6):734–749, 2005.

[2] Ricardo Baeza-Yates, Carlos Hurtado, and Marcelo Mendoza. *Query Recommendation Using Query Logs in Search Engines*, volume 3268/2004 of *LNCS*, pages 588–596. Springer Berlin / Heidelberg, November 2004.

[3] Ricardo Baeza-Yates and Alessandro Tiberi. Extracting semantic relations from query logs. In *Proc. KDD'07*, pages 76–85, New York, NY, USA, 2007. ACM.

[4] Evelyn Balfe and Barry Smyth. Improving web search through collaborative query recommendation. In Ramon López de Mántaras and Lorenza Saitta, editors, *Proc. ECAI'04*, pages 268–272. IOS Press, 2004.

[5] Ranieri Baraglia, Fidel Cacheda, Victor Carneiro, Diego Fernandez, Vreixo Formoso, Raffaele Perego, and Fabrizio Silvestri. Search shortcuts: a new approach to the recommendation of queries. In *Proc. RecSys'09*, pages 77–84, New York, NY, USA, 2009. ACM.

[6] Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, Aristides Gionis, and Sebastiano Vigna. The query-flow graph: model and applications. In *Proc. CIKM'08*, pages 609–618, New York, NY, USA, 2008. ACM.

[7] Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, and Sebastiano Vigna. Query suggestions using query-flow graphs. In *Proc. WSCD'09*, pages 56–63, New York, NY, USA, 2009. ACM.

[8] Nikolai Buzikashvili. Automatic task detection in the web logs and analysis of multitasking. In *ICADL*, pages 131–140, 2006.

[9] John Canny. Collaborative filtering with privacy via factor analysis. In *Proceedings of the 25th annual international ACM SIGIR 2002 Conference*, pages 238–245, New York, NY, USA, 2002. ACM.

[10] Bruno M. Fonseca, Paulo B. Golgher, Edleno S. de Moura, and Nivio Ziviani. Using association rules to discover search engines related queries. In *LA-WEB '03: Proceedings of the First Conference on Latin American Web Congress*, page 66, Washington, DC, USA, 2003. IEEE Computer Society.

[11] Shunkai Fu, Bingfeng Pi, Ying Zhou, Michel C. Desmarais, Weilei Wang, Song Han, and Xunrong Rao. Cross-channel query recommendation on commercial mobile search engine: Why, how and empirical evaluation. In *PAKDD*, pages 883–890, 2009.

[12] Zan Huang, Hsinchun Chen, and Daniel Dajun Zeng. Applying associative retrieval techniques to alleviate the sparsity problem in collaborative filtering. *ACM Trans. Inf. Syst.*, 22(1):116–142, 2004.

[13] Bernard J. Jansen, Amanda Spink, Judy Bateman, and Tefko Saracevic. Real life information retrieval: a study of user queries on the web. *SIGIR Forum*, 32(1):5–17, 1998.

[14] Rosie Jones and Kristina Lisa Klinkner. Beyond the session timeout: automatic hierarchical segmentation of search topics in query logs. In *Proc. CIKM'08*, pages 699–708, New York, NY, USA, 2008. ACM.

[15] John Makhoul, Francis Kubala, Richard Schwartz, and Ralph Weischedel. Performance measures for information extraction. In *In Proceedings of DARPA Broadcast News Workshop*, pages 249–252, 1999.

[16] Jasson D. M. Rennie and Nathan Srebro. Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the 22nd ICML 2005 Conference*, pages 713–719, New York, NY, USA, 2005. ACM.

[17] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Proc. SIGIR'94*, pages 232–241, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[18] Stephen E. Robertson, Steve Walker, Micheline Hancock-Beaulieu, Aarron Gull, and Marianna Lau. Okapi at trec. In *TREC*, pages 21–30, 1992.

[19] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-2. In *TREC*, pages 21–34, 1993.

[20] Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. Okapi at trec-3. In *TREC*, pages 0–, 1994.

[21] Badrul Sarwar, George Karypis, Joseph Konstan, and John Reidl. Item-based collaborative filtering recommendation algorithms. In *Proc. WWW'01*, pages 285–295, New York, NY, USA, 2001. ACM.

[22] Upendra Shardanand and Pattie Maes. Social information filtering: algorithms for automating ”word of mouth”. In *Proc. SIGCHI'95*, pages 210–217, New York, NY, USA, 1995. ACM Press.

[23] Craig Silverstein, Hannes Marais, Monika Henzinger, and Michael Moricz. Analysis of a very large web search engine query log. *SIGIR Forum*, 33(1):6–12, 1999.

[24] Fabrizio Silvestri. Mining query logs: Turning search usage data into knowledge. *Foundations and Trends in Information Retrieval*, 1(1-2):1–174, 2010.

[25] Barry Smyth. A community-based approach to personalizing web search. *Computer*, 40(8):42–50, 2007.

[26] Barry Smyth, Evelyn Balfe, Oisin Boydell, Keith Bradley, Peter Briggs, Maurice Coyle, and Jill Freyne. A live-user evaluation of collaborative web search. In *IJCAI*, pages 1419–1424, 2005.

[27] Amanda Spink, Sherry Koshman, Minsoo Park, Chris Field, and Bernard J. Jansen. Multitasking web search on vivisimo.com. In *ITCC (2)*, pages 486–490, 2005.

[28] Amanda Spink, Minsoo Park, Bernard J. Jansen, and Jan O. Pedersen. Multitasking during web search sessions. *Inf. Process. Manage.*, 42(1):264–275, 2006.

[29] L. Ungar and D. Foster. Clustering methods for collaborative filtering. In *Proceedings of the Workshop on Recommendation Systems*. AAAI Press, Menlo Park California, 1998.

[30] Jun Wang, Arjen P. de Vries, and Marcel J. T. Reinders. Unifying user-based and item-based collaborative filtering approaches by similarity fusion. In *Proc. SIGIR'06*, pages 501–508, New York, NY, USA, 2006. ACM.