Query Log Analysis for Enhancing Web Search

Salvatore Orlando, University of Venice, Italy
Fabrizio Silvestri, ISTI - CNR, Pisa, Italy

From tutorials given at IEEE / WIC / ACM WI/IAT'09 and ECIR’09
Query Log Analysis for Enhancing Web Search

Salvatore Orlando, University of Venice, Italy
Fabrizio Silvestri, ISTI - CNR, Pisa, Italy

From tutorials given at IEEE / WIC / ACM WI/IAT'09 and ECIR’09
History in Search Engines

Alphonse de Lamartine

History in Search Engines

Alphonse de Lamartine


History Teaches Everything... Even the Future!
History in Search Engines

• Past Queries
• Query Sessions
• Click-through Data
Tutorial Outline

• Query Logs
  • The Nature of Queries
  • User Actions
• Enhancing Effectiveness of Search Systems
• Enhancing Efficiency of Search Systems
What’s in Query Logs?

The 250 most frequent queried terms in the “famous” AOL query log!

Thanks to http://www.wordle.net for the tagcloud generator
## Query Logs Analyzed in the Literature

<table>
<thead>
<tr>
<th>Query log name</th>
<th>Public</th>
<th>Period</th>
<th># Queries</th>
<th># Sessions</th>
<th># Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>Excite ‘97</td>
<td>Y</td>
<td>Sep ’97</td>
<td>1,025,908</td>
<td>211,063</td>
<td>~ 410,360</td>
</tr>
<tr>
<td>Excite ‘97 (small)</td>
<td>Y</td>
<td>Sep ’97</td>
<td>51,473</td>
<td>N.D.</td>
<td>~ 18,113</td>
</tr>
<tr>
<td>Altavista</td>
<td>N</td>
<td>Aug 2\textsuperscript{nd} - Sep 13\textsuperscript{th} ‘98</td>
<td>993,208,159</td>
<td>285,474,117</td>
<td>N.D.</td>
</tr>
<tr>
<td>Excite ‘99</td>
<td>Y</td>
<td>Dec ‘99</td>
<td>1,025,910</td>
<td>325,711</td>
<td>~ 540,000</td>
</tr>
<tr>
<td>Excite ‘01</td>
<td>Y</td>
<td>May ‘01</td>
<td>1,025,910</td>
<td>262,025</td>
<td>~ 446,000</td>
</tr>
<tr>
<td>Altavista (public)</td>
<td>Y</td>
<td>Sep ‘01</td>
<td>7,175,648</td>
<td>N.D.</td>
<td>N.D.</td>
</tr>
<tr>
<td>Tiscali</td>
<td>N</td>
<td>Apr ‘02</td>
<td>3,278,211</td>
<td>N.D.</td>
<td>N.D.</td>
</tr>
<tr>
<td>TodoBR</td>
<td>Y</td>
<td>Jan - Oct ‘03</td>
<td>22,589,568</td>
<td>N.D.</td>
<td>N.D.</td>
</tr>
<tr>
<td>TodoCL</td>
<td>N</td>
<td>May – Nov ‘03</td>
<td>N.D.</td>
<td>N.D.</td>
<td>N.D.</td>
</tr>
<tr>
<td>AOL (big)</td>
<td>N</td>
<td>Dec 26\textsuperscript{th} ‘03 – Jan 1\textsuperscript{st} ‘04</td>
<td>~ 100,000,000</td>
<td>N.D.</td>
<td>~ 50,000,000</td>
</tr>
<tr>
<td>Yahoo!</td>
<td>N</td>
<td>Nov ‘05 – Nov ‘06</td>
<td>N.D.</td>
<td>N.D.</td>
<td>N.D.</td>
</tr>
<tr>
<td>AOL (small)</td>
<td>Y</td>
<td>Mar 1\textsuperscript{st} - May 31\textsuperscript{st} ‘06</td>
<td>36,389,567</td>
<td>N.D.</td>
<td>N.D.</td>
</tr>
</tbody>
</table>
AOL query log

The data is sorted by anonymous user ID and sequentially arranged.

The goal of this collection is to provide real query log data that is based on real users. It could be used for personalization, query reformulation or other types of search research.

The data set includes \{AnonID, Query, QueryTime, ItemRank, ClickURL\}.

- AnonID - an anonymous user ID number.
- Query - the query issued by the user, case shifted with most punctuation removed.
- QueryTime - the time at which the query was submitted for search.
- ItemRank - if the user clicked on a search result, the rank of the item on which they clicked is listed.
- ClickURL - if the user clicked on a search result, the domain portion of the URL in the clicked result is listed.

Each line in the data represents one of two types of events:

1. A query that was NOT followed by the user clicking on a result item.
2. A click through on an item in the result list returned from a query.

In the first case (query only) there is data in only the first three columns/fields -- namely AnonID, Query, and QueryTime (see above).

In the second case (click through), there is data in all five columns. For click through events, the query that preceded the click through is included. Note that if a user clicked on more than one result in the list returned from a single query, there will be TWO lines in the data to represent the two events. Also note that if the user requested the next "page" or results for some query, this appears as a subsequent identical query with a later time stamp.
Some Popular Terms: Excite and Altavista

<table>
<thead>
<tr>
<th>query</th>
<th>freq.</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Empty Query</em></td>
<td>2,586</td>
</tr>
<tr>
<td>sex</td>
<td>229</td>
</tr>
<tr>
<td>chat</td>
<td>58</td>
</tr>
<tr>
<td>lucky number generator</td>
<td>56</td>
</tr>
<tr>
<td>p****</td>
<td>55</td>
</tr>
<tr>
<td>porno</td>
<td>55</td>
</tr>
<tr>
<td>b****y</td>
<td>55</td>
</tr>
<tr>
<td>nude beaches</td>
<td>52</td>
</tr>
<tr>
<td>playboy</td>
<td>46</td>
</tr>
<tr>
<td>bondage</td>
<td>46</td>
</tr>
<tr>
<td>porn</td>
<td>45</td>
</tr>
<tr>
<td>rain forest restaurant</td>
<td>40</td>
</tr>
<tr>
<td>f****ing</td>
<td>40</td>
</tr>
<tr>
<td>crossdressing</td>
<td>39</td>
</tr>
<tr>
<td>crystal methamphetamine</td>
<td>36</td>
</tr>
<tr>
<td>consumer reports</td>
<td>35</td>
</tr>
<tr>
<td>xxx</td>
<td>34</td>
</tr>
<tr>
<td>nude tanya harding</td>
<td>33</td>
</tr>
<tr>
<td>music</td>
<td>33</td>
</tr>
<tr>
<td>sneaker stories</td>
<td>32</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>query</th>
<th>freq.</th>
</tr>
</thead>
<tbody>
<tr>
<td>christmas photos</td>
<td>31,554</td>
</tr>
<tr>
<td>lyrics</td>
<td>15,818</td>
</tr>
<tr>
<td>cracks</td>
<td>12,670</td>
</tr>
<tr>
<td>google</td>
<td>12,210</td>
</tr>
<tr>
<td>gay</td>
<td>10,945</td>
</tr>
<tr>
<td>harry potter</td>
<td>7,933</td>
</tr>
<tr>
<td>wallpapers</td>
<td>7,848</td>
</tr>
<tr>
<td>pornografia</td>
<td>6,893</td>
</tr>
<tr>
<td>“yahoo com”</td>
<td>6,753</td>
</tr>
<tr>
<td>juegos</td>
<td>6,559</td>
</tr>
<tr>
<td>lingerie</td>
<td>6,078</td>
</tr>
<tr>
<td>symbios logic 53c400a</td>
<td>5,701</td>
</tr>
<tr>
<td>letras de canciones</td>
<td>5,518</td>
</tr>
<tr>
<td>humor</td>
<td>5,400</td>
</tr>
<tr>
<td>pictures</td>
<td>5,293</td>
</tr>
<tr>
<td>preteen</td>
<td>5,137</td>
</tr>
<tr>
<td>hypnosis</td>
<td>4,556</td>
</tr>
<tr>
<td>cpc view registration key</td>
<td>4,553</td>
</tr>
<tr>
<td>sex stories</td>
<td>4,521</td>
</tr>
<tr>
<td>cd cover</td>
<td>4,267</td>
</tr>
</tbody>
</table>

(a) Excite.                        (b) Altavista.

Topic Distribution: Excite and AOL

<table>
<thead>
<tr>
<th>Topic</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entertainment or recreation</td>
<td>19.9%</td>
</tr>
<tr>
<td>Sex and pornography</td>
<td>16.8%</td>
</tr>
<tr>
<td>Commerce, travel, employment, or economy</td>
<td>13.3%</td>
</tr>
<tr>
<td>Computers or Internet</td>
<td>12.5%</td>
</tr>
<tr>
<td>Health or sciences</td>
<td>9.5%</td>
</tr>
<tr>
<td>People, places, or things</td>
<td>6.7%</td>
</tr>
<tr>
<td>Society, culture, ethnicity, or religion</td>
<td>5.7%</td>
</tr>
<tr>
<td>Education or humanities</td>
<td>5.6%</td>
</tr>
<tr>
<td>Performing or fine arts</td>
<td>5.4%</td>
</tr>
<tr>
<td>Non-English or unknown</td>
<td>4.1%</td>
</tr>
<tr>
<td>Government</td>
<td>3.4%</td>
</tr>
</tbody>
</table>

Excite


Long Tail Distribution
Long Tail Distribution

Terms ordered by popularity
Long Tail Distribution

Number of clicks

URLs ordered by number of clicks
Power-Law In Query Popularity: Altavista

Power-Law In Query Popularity: Excite

Power-Law In Query Popularity: Yahoo!

Query Resubmission

Frequency of Query Submission

## Query Statistics: Excite

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>1997</th>
<th>1999</th>
<th>2001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean terms per query</td>
<td>2.4</td>
<td>2.4</td>
<td>2.6</td>
</tr>
<tr>
<td>Terms per query</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 term</td>
<td>26.3%</td>
<td>29.8%</td>
<td>26.9%</td>
</tr>
<tr>
<td>2 terms</td>
<td>31.5%</td>
<td>33.8%</td>
<td>30.5%</td>
</tr>
<tr>
<td>3+ terms</td>
<td>43.1%</td>
<td>36.4%</td>
<td>42.6%</td>
</tr>
<tr>
<td>Mean queries per user</td>
<td>2.5</td>
<td>1.9</td>
<td>2.3</td>
</tr>
</tbody>
</table>

## Query Statistics: Excite

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>1997</th>
<th>1999</th>
<th>2001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean terms per query</td>
<td>2.4</td>
<td>2.4</td>
<td>2.6</td>
</tr>
<tr>
<td>Terms per query</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 term</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 terms</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3+ terms</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mean queries per user</td>
<td>2.5</td>
<td>1.9</td>
<td>2.3</td>
</tr>
</tbody>
</table>

In 2008: 2.5 terms per query.


Hourly Topic Distribution

Tutorial Outline

• Query Logs
• Enhancing Effectiveness of Search Systems
  • Query Expansion/Suggestion/Personalization
• Enhancing Efficiency of Search Systems
Query Expansion/Suggestion/Personalization

- Click-through data associated with past queries represent a sort of implicit relevance feedback information.

- The challenge is to exploit such information to mine knowledge and use it to improve the effectiveness of the search engines.

- The final goal is to improve the precision by expanding/suggesting/personalizing queries.
Can click-through data be useful relevance feedbacks?

- Joachims and Radlinski noted that the top position reported by WSE strongly influence user behavior, beyond snippets

- They registered the number of clicks a given position obtained in two different conditions: normal and swapping the first two top positions

Can click-through data be useful relevance feedbacks?

- Joachims and Radlinski noted that the top position reported by WSE strongly influence user behavior, beyond snippets.
- They registered the number of clicks a given position obtained in two different conditions: normal and swapping the first two top positions.

Research issues (1)

- The lack of query logs and well-defined effectiveness metrics may negatively influence the scientific value of research results
  - many times, such logs are not publicly available, and thus experiments may not be reproducible

- The effectiveness of the proposed solutions are often tested on a small group of homogeneous people, e.g., metrics are tested on small human-annotated testbeds
Research issues (2)

• Privacy is nowadays a big concern of user communities
  • many of the techniques presented need to build user profiles

• Profile-based (i.e. context-based, personalized) search
  • is computationally expensive
  • may prevent the adoption of global techniques aiming at enhancing performance (like caching)
Query Expansion

- Queries are short, poorly built, and sometimes mistyped
- *Cui et al.* observed that queries and documents are rather poorly correlated
  - by measuring the gap between the *document vector space* (the most important terms contained in each document according to $if \times idf$) and the *query vector space* (all the terms contained in the group of queries for which a document was clicked)
  - in most cases, the similarity values are between 0.1 and 0.4, and only a small percentage of documents have similarity above 0.8
- Solution: expanding a query by adding additional terms

Query Expansion

- In traditional IR systems, query expansion is a well-known technique.
- However, one of the first works making explicit use of past queries to improve the effectiveness of query expansion is the one by Fitzpatrick and Dent.
  - They build off-line an affinity pool made up of documents retrieved by similar past queries (the TREC queries and databases were used).
  - A submitted query is first checked against the affinity pool, and from the resulting top-scoring documents, a set of “important” terms is automatically extracted to enrich the query.
  - They achieved an improvement of 38.3% in average precision.

Query Expansion

- Cui et al. exploited correlations among terms in clicked documents and web search engine queries

- query session extracted from the query log:
  <query, (list of clicked docIDs)>

Query Expansion

- Cui et al. exploited correlations among terms in clicked documents and web search engine queries.

- *query session* extracted from the query log:
  \(<\text{query, (list of clicked docIDs)}>\)

Query Expansion

A link is inserted on the basis of query sessions.

Term $t_q$ occurs is a query of a session.
Term $t_d$ occurs in a clicked document within the same session.

Query Expansion

A link is inserted on the basis of query sessions. Term $t_q$ occurs is a query of a session. Term $t_d$ occurs in a clicked document within the same session.

Query Expansion

A link is inserted on the basis of query sessions.

Term $t_q$ occurs is a query of a session.

Term $t_d$ occurs in a clicked document within the same session.

Term $t_q$ occurs is a query of a session.

Term $t_d$ occurs in a clicked document within the same session.

$W = \text{degree of term correlation}$

Query Expansion

• Correlation is given by the conditional probability $P(t_d | t_q)$

• occurrence of term $t_d$ given the occurrence of $t_q$ in the query

Query Expansion

• The term correlation measure is then used to devise a query expansion method

• It exploits a so-called cohesion measure between a query $Q$ and a candidate term $t_d$ for query expansion

$$CoWeight (Q, t_d) = \log \left( \prod_{t_q \in Q} P(t_d|t_q) + 1 \right)$$

Query Expansion

- The term correlation measure is then used to devise a *query expansion method*.
- It exploits a so-called *cohesion measure* between a query $Q$ and a candidate term $t_d$ for query expansion.

$$\text{CoWeight} (Q, t_d) = \log \left( \prod_{t_q \in Q} P (t_d | t_q) + 1 \right)$$

Naïve hypothesis on independence of terms in a query.

Query Expansion

• The term correlation measure is then used to devise a query expansion method

• It exploits a so-called cohesion measure between a query $Q$ and a candidate term $t_d$ for query expansion

$$\text{CoWeight} (Q, t_d) = \log \left( \prod_{t_q \in Q} P (t_d | t_q) + 1 \right)$$

• The measure is used to build a list of weighted candidate terms

• The top-$k$ ranked terms (those with the highest weights) are selected as expansion terms for query $Q$

Query Expansion

- The log-based method was compared against two baseline methods
  - (a) not using query expansion at all, or
  - (b) using an expansion technique (local context method) that does not make use of logs to expand queries
- Indeed, the local context method (by Xu and Croft) exploits the top ranked documents retrieved for a query to expand the query itself
- A few queries were used for the tests (Encarta and TREC queries, and hand-crafted queries), and the following table summarizes the average results

<table>
<thead>
<tr>
<th>Method</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>baseline</td>
<td>17%</td>
</tr>
<tr>
<td>local context</td>
<td>22%</td>
</tr>
<tr>
<td>log-based</td>
<td>30%</td>
</tr>
</tbody>
</table>


Query Expansion

- Billerbeck et al. use the concept of Query Association already proposed by Scholer et al.
- Past user queries are associated with a document if they share a high statistically similarity
- Past queries associated with a document enrich the document itself
- All the queries associated with a document can be considered as Surrogate Documents, and can be used as a source of terms for query expansion

Query Expansion

Past Queries

Full Document Collection

Query Expansion

Past Queries

Full Document Collection

Each past queries $q$ is naturally associated with the $K$ most relevant documents returned by a search engine.

Each past queries $q$ is naturally associated with the $K$ most relevant documents returned by a search engine.

Query Expansion

Each past queries $q$ is naturally associated with the $K$ most relevant documents returned by a search engine.

Query Expansion

Each past query $q$ is naturally associated with the $K$ most relevant documents returned by a search engine.

Query Expansion

Each document $d$ can result to be associated with many queries. Only the $M$ closest queries are kept w.r.t. the Okapi BM25 similarity measure.

Each document $d$ can result to be associated with many queries. Only the $M$ closest queries are kept w.r.t. the Okapi BM25 similarity measure.

Query Expansion

• Why may surrogate documents be a viable source of terms for expanding queries?
  • The fact that the queries are associated with the document means that, in some sense, the query terms have topical relationships with each other.
  • It may be better than expanding directly from documents, because the terms contained in the associated surrogate documents have already been chosen by users as descriptors of topics.
  • It may be better than expanding directly from queries, because the surrogate document has many more terms than an individual query.
Query Expansion

- By using the surrogate documents
  - the expanded query is large and appears to contain only useful terms:
    - earthquakes  earthquake  recent  nevada  seismograph  tectonic  faults  perpetual  1812  kobe  magnitude  california  volcanic  activity  plates  past  motion  seismological

- By using the full documents
  - the expanded query is more narrow
    - earthquakes  tectonics  earthquake  geology  geological

Query suggestion

• Exploit information on past users' queries

• Propose to a user a list of queries related to the one (or the ones, considering past queries in the same session) submitted

• Query suggestion vs. expansion

• users can select the best similar query to refine their search, instead of having the query uncontrollably stuffed with a lot of terms
Query suggestion

• A naïve approach, as stated by Zaïane and Strilets, does not work
  • Query similarity simply based on sharing terms
  • The query “Salvatore Orlando” would be considered, to some extent, similar to “Florida Orlando”, since they share term “Orlando”

• In literature there are several proposals
  • queries suggested from those appearing frequently in query sessions
  • use clustering to devise similar queries on the basis of cluster membership
  • use click-through data information to devise query similarity

Query suggestion

• Exploiting query sessions
  • if a lot of previous users, when issuing the query $q_1$ also issue query $q_2$ afterwards, query $q_2$ is suggested for query $q_1$

• Fonseca et al. exploited association rule mining to generate query suggestions according to the above idea

Query suggestion

• The method used by Fonseca et al. is a straightforward application of association rules
• the input data set $D$ is composed of transactions, each corresponding to an unordered user session, where items are queries $q_i$
• In general, a rule extracted has the form $A \Rightarrow B$, where $A$ and $B$ are disjoint sets of queries
• To reduce the computational cost, only rules where both $A$ and $B$ are singletons are indeed extracted: $q_i \Rightarrow q_j$, where $q_i \neq q_j$

Query suggestion

• For each incoming query $q_i$
• all the rules extracted and sorted by confidence level
  
  $q_i \Rightarrow q_1, q_i \Rightarrow q_2, q_i \Rightarrow q_3, \ldots, q_i \Rightarrow q_m$

• the queries suggested are the top 5 ranked ones
• For experiments, they used a query log of 2,312,586 queries, coming from a real Brazilian search engine
• Low Minimum absolute support = 3 to mine the sets of frequent queries
• This means that, given an extracted rule $q_i \Rightarrow q_j$, the unordered pair $(q_i, q_j)$ appeared in at least 3 user sessions
• For validating the method, a survey among a small group of people,

Query suggestion

- Baeza-Yates et al. use clustering and exploits a two-tier system.
- An offline component builds clusters of past queries, using query text along with the text of clicked URLs.
- An online component recommends queries on the basis of the input one.
Query suggestion

• **Offline component:**

  • the clustering algorithm operates over **queries enriched by a selection of terms** extracted from the documents pointed by the **user clicked URLs**.

  • Clusters computed by using an implementation of the **k-means** algorithm

  • Similarity between queries computed according to a **vector-space** approach

  • Vectors $\vec{q}$ of $n$ dimensions, one for each term


*http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview*
Query suggestion

- Offline component:
  - $q_i$ is the i-th component of the vector $\vec{q}$ associated with term $t_i$ of the vocabulary (all different words are considered)

$$q_i = \sum_{u \in URLs} \frac{\text{Clicks}(q, u) \times \text{Tf}(t_i, u)}{\max_t \text{Tf}(t, u)}$$

Query suggestion

• Offline component:

• $q_i$ is the $i$-th component of the vector $\vec{q}$ associated with term $t_i$ of the vocabulary (all different words are considered)

$$q_i = \sum_{u \in URLs} \frac{\text{Clicks} (q, u) \times \text{Tf} (t_i, u)}{\max_t \text{Tf} (t, u)}$$

Percentage of clicks that URL $u$ receives when answered in response to query $q$

Query suggestion

- **Offline component:**

  - $q_i$ is the i-th component of the vector $\vec{q}$ associated with term $t_i$ of the vocabulary (all different words are considered)

  \[
  q_i = \sum_{u \in URLs} \frac{\text{Clicks} (q, u) \times \text{Tf} (t_i, u)}{\max_{t} \text{Tf} (t, u)}
  \]

  - Percentage of clicks that URL $u$ receives when answered in response to query $q$
  - Number of occurrences of the term in the document pointed to URL $u$

---

Query suggestion

- **Offline component:**

- \( q_i \) is the i-th component of the vector \( \vec{q} \) associated with term \( t_i \) of the vocabulary (all different words are considered)

\[
q_i = \sum_{u \in URLs} \frac{\text{Clicks} (q, u) \times \text{Tf} (t_i, u)}{\max_t \text{Tf} (t, u)}
\]

---

Query suggestion

• **Online component:**

  (I) for an input query the most similar cluster is selected
  
  • each cluster has a natural representative, i.e. its centroid

  (II) ranking of the queries of the cluster, according to:

  • **attractiveness** of query answer, i.e. the fraction of the documents returned by the query that captured the attention of users (clicked documents)

  • **similarity** w.r.t. the input query (the same distance used for clustering)

  • **popularity** of query, i.e. the frequency of the occurrences of queries

---

Query suggestion

- **Experiments:**
  - The query log (and the relative collection) comes from the *TodoCL* search engine
  - 6,042 unique queries along with associated click-throughs
  - 22,190 registered clicks spread over 18,527 different URLs
  - The algorithm was evaluated on ten different queries by a user study.
  - Presenting query suggestions ranked by attractiveness of queries yielded to more precise and high quality suggestions

Query personalization

• Personalization consists in presenting different ranked results for the same issued query, depending on
  • different searcher tastes
  • different contexts (places or times)

• For examples, a mathematician and an economist who issue the same query “game theory”
  • a mathematician would return many results on theory of games and theoretical studies
  • an economist would be rather interested in applications of game theory real-world economy problems

Query personalization

- One possible method to achieve **Personalization** is
  - “re-ranking” search results according to a specific user's profile, built automatically by exploiting knowledge mined from query logs
- We start from a negative results
  - Teevan et al. demonstrate that for queries which showed less variations among individuals, re-ranking results according to a personalization function may be insufficient (or even dangerous)

Query personalization

- Liu et al. categorize users and queries with a set of relevant categories
  - Return the top 3 categories for each user query
  - The categorization function is automatically computed on the basis of the retrieval history of each user
  - The set of different categories are the same as the ones used by the search engine to classify web pages
  - thus such user-based categorization is used to personalize results, since it allows to focus on the most relevant results for each user

- The two main concepts used are
  - User Search History
  - User Profile (automatically generated)

Query personalization

- Boydell and Smith use snippets of clicked results
- They argued that results (in a result list) are selected because the user recognizes in their snippets certain combinations of terms that are related to their information needs
- They propose to build a community-based snippet index that reflects the evolving interests of a group of searchers
- The index is used for (community-based) personalization through re-ranking of the search results
  - The index is built at the proxy-side
  - No usage information is stored at the server-side
  - Harmless with respect to issues of users' privacy

O. Boydell and B. Smyth, "Capturing community search expertise for personalized web search using snippet-indexes", in CIKM '06, pp. 277-286, ACM, 2006
Query personalization

Collaborative Web Search (CWS)

- A user $u$ in some community $C$
- The results of an initial meta-search, $R_M$, are revised with reference to the community's snippet index $I_C$
- A new result-list, $R_C$, is returned. This list is adapted to community preferences.
- $R_M$ and $R_C$ are combined and returned to the user as $R_T$

Query personalization

- A common method exploited by other CWS systems:
  - find a set of related queries $q_1, \ldots, q_k$ such that these queries share some minimal overlapping terms within $q_T$
  - the main issue of this method is that sometimes two related queries do not contain any common terms
  - e.g. “Captain Kirk” and “Starship Enterprise”;
Query personalization

- In the CWS by Boydell and Smyth, each past query is indexed along with the surrogate clicked documents (snippets)
- Main advantage:
  - A result $r$ that was previously selected for query $Q_1 =$ “Captain Kirk”, can potentially be returned in response to query $Q_2 =$ “Starship Enterprise”
  - If the terms in $Q_1$ occurred in the snippet of a result previously selected in response to $Q_2$
Tutorial Outline

- Query Logs
- Enhancing Effectiveness of Search Systems
- Enhancing Efficiency of Search Systems
  - Caching
  - Index Partitioning and Querying in Distributed IR Systems
Sketching a Distributed Search Engine
Caching in General

Larger, but slower memory

Smaller, but faster memory

CPU
W/O Caching

[Diagram showing a network of devices connected to a Broker, which then sends data to two IR Core nodes.]
With Caching
With Caching
With Caching

This is true on an ideal world.
Caching Performance Evaluation

- **Hit-Ratio**: i.e. how many times the cache is useful

- **Query Throughput**: i.e. the number of queries the cache can serve in a second

- But... what really impacts on caching performance?
“Things” to Cache in Search Engines

- Results
  - in answer to a user query

- Posting Lists
  - e.g. for the query “new york” cache the posting lists for term new and for term york

- Partial queries
  - cache subqueries, e.g. for “new york times” cache only “new york”
Traditional Replacement Policies

• LRU
• LFU
• SLRU
• ...

That is...
~80% of submitted queries represents the 20% of the unique queries submitted
That is...

Store these queries forever!
Static Caching
But...

Static Dynamic Caching

- SDC (Static Dynamic Caching) adds to the classical static caching schema a dynamically managed section.

- The idea:

  ![Diagram showing Static Set and Dynamic Set with f_static function]

Static Dynamic Caching

- SDC (Static Dynamic Caching) adds to the classical static caching schema a dynamically managed section.

- The idea:

  ![Diagram showing Static Set and Dynamic Set]

  \[ f_{\text{static}} \]

Static Dynamic Caching

- SDC (Static Dynamic Caching) adds to the classical static caching schema a dynamically managed section.

- The idea:

  
  
  ![Diagram](image)

  
  
  - LRU
  - SLRU
  - PDC
  - ...

Static Dynamic Caching

- SDC (Static Dynamic Caching) adds to the classical static caching schema a dynamically managed section.

- The idea:

---

SDC and Prefetching

- SDC adopts an “adaptive” prefetching technique:
  - For the first SERP do not prefetch
  - For the follow-up SERPs prefetch f pages

SDC and Prefetching

Fig. 3. Probability of the occurrence of a request for the $i^{th}$ page given a previous request for page $(i-1)$. When a query is submitted, it is necessary to browse many pages in order to find the relevant information.

3.3 Theoretical Upper Bounds on the Cache Hit Ratio

From the data reported in Table I it is easy to devise the theoretical upper bounds on the hit ratios achievable on the three query logs when prefetching is not used. To this end, let us suppose the availability of a cache of infinite size, so that only compulsory misses have to be taken into account, that is, cache misses corresponding to the first reference to each distinct query. The rate of compulsory misses is thus

$$m = \frac{D}{Q},$$

(1)

where $D$ is the number of distinct queries, and $Q$ is the total number of queries in the log. The value $m$ is the minimum miss ratio, while the maximum hit ratio, $H$, is obviously given by

$$H = 1 - m = 1 - \frac{D}{Q}.$$

(2)

The analysis becomes a little more complicated when we introduce prefetching [Lempel and Moran 2003]. It is worth recalling that a general user query has the form $(keywords, page\ no)$, and that, for this analysis, we are interested in considering queries characterized by identical keywords and distinct page no.

In particular, if we retrieve each time $k$ successive pages of results starting from the one requested by the user (hereinafter we will call $k$ the prefetching factor), we have to distinguish among queries requesting pages in the first block of $k$ pages ($1 \leq page\ no \leq k$), in the second block of $k$ pages ($k+1 \leq page\ no \leq 2k$), ...,

SDC Hit-Ratios

SDC’s Main Lessons Learned

SDC’s Main Lessons Learned

• Hit ratio benefits a lot from the use of historical data

SDC’s Main Lessons Learned

• Hit ratio benefits a lot from the use of historical data

• Prefetching helps a lot!

SDC’s Main Lessons Learned

• Hit ratio benefits a lot from the use of historical data
• Prefetching helps a lot!
• Static caching alone is not useful, yet...

SDC’s Main Lessons Learned

• Hit ratio benefits a lot from the use of historical data

• Prefetching helps a lot!

• Static caching alone is not useful, yet...

• A good combination of a static and a dynamic approach helps a lot!!!

That’s not All Folks!

That’s not All Folks!

Not Only Caching

- Improve efficiency using query logs can also be done by:
  - data/index partitioning
Sketching a Distributed Search Engine

query
\[ t_1, t_2, \ldots, t_q \]

results
\[ r_1, r_2, \ldots, r_r \]

Broker

IR Core 1
\[ \text{idx} \]

IR Core 2
\[ \text{idx} \]

IR Core \( k \)
\[ \text{idx} \]
Index Partitioning

Term Partition

Document Partition
Term Partitioning Systems

• Query routing is “trivial”...

• Whenever a query $Q=(t_1,t_2,\ldots,t_n)$ is received route it to the servers managing those terms.

• But..

  • not scalable (indexing is $n \log n$, term partitioning needs reindexing the entire collection from scratch when an update occurs)

  • load imbalance among IR cores
Why Studying Term Partitioned Systems?

- In principle:
  - less IR Cores queried
  - less operations performed
- Briefly...
  - More available capacity!
Term Partitioning
(Random Partitioning)

\[ T_2 \ T_4 \ T_3 \ T_1 \ T_1 \ T_8 \]

IR Core 1
\[ T_1 \ldots T_3 \]

IR Core 2
\[ T_4 \ldots T_6 \]

IR Core 3
\[ T_7 \ldots T_9 \]

IR Core 4
\[ T_{10} \ldots T_{12} \]
Term Partitioning
(Random Partitioning)

IR Core 1
\( T_1 \ldots T_3 \)

IR Core 2
\( T_4 \ldots T_6 \)

IR Core 3
\( T_7 \ldots T_9 \)

IR Core 4
\( T_{10} \ldots T_{12} \)
Term Partitioning
(Random Partitioning)

IR Core 1
T_1 T_2 T_3
T_1\ldots T_3

IR Core 2
T_4
T_4\ldots T_6

IR Core 3
T_8
T_7\ldots T_9

IR Core 4
T_1
T_{10}\ldots T_{12}
Term Partitioning
(Random Partitioning)

T₂ T₃ T₁ T₄ T₈
Term Partitioning
(Random Partitioning)

IR Core 1
T_1 \ldots T_3

IR Core 2
IR Core 3
IR Core 4
T_4 \ldots T_6
T_7 \ldots T_9
T_{10} \ldots T_{12}

T_4 \quad T_8
Term Partitioning
(Random Partitioning)

IR Core 2
$T_4$ $T_8$
$T_4...T_6$

IR Core 3
$T_8$
$T_7...T_9$

IR Core 4
$T_1$
$T_{10}...T_{12}$
Pipelined Term Part.
(Random Partitioning)

IR Core 1
T₁...T₃

IR Core 2
T₄...T₆

IR Core 3
T₇...T₉

IR Core 4
T₁₀...T₁₂
Pipelined Term Part.
(Random Partitioning)
Pipelined Term Part.
(Random Partitioning)
Pipelined Term Part.
(Random Partitioning)
Pipelined Term Part.
(Random Partitioning)

T2 T4 T3

IR Core 1
T1...T3

IR Core 2
T4...T6

IR Core 3
T7...T9

IR Core 4
T10...T12
How can we...

• Balance the load?
• Better exploit resources?
• In light of...
How can we...

- Balance the load?
- Better exploit resources?
- In light of... The Power Law!!!
Two Approaches in Literature

- Both exploiting past query log knowledge
