Adaptive Query Routing in Peer Web Search

Le-Shin Wu †
lewu@cs.indiana.edu

Ruj Akavipat †
rakavipa@cs.indiana.edu

Filippo Menczer †‡
fil@indiana.edu

†Department of Computer Science and ‡School of Informatics
Indiana University, Bloomington

ABSTRACT

An unstructured peer network application was proposed to address the query forwarding problem of distributed search engines and scalability limitations of centralized search engines. Here we present novel techniques to improve local adaptive routing, showing they perform significantly better than a simple learning scheme driven by query response interactions among neighbors. We validate prototypes of our peer network application via simulations with 500 model users based on actual Web crawls. We finally compare the quality of the results with those obtained by centralized search engines, suggesting that our application can draw advantages from the context and coverage of the peer collective.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval; H.3.4 [**Information Storage and Retrieval**]: Systems and Software—*Distributed systems, information networks, performance evaluation (efficiency and effectiveness)*

Keywords

Peer collaborative search, topical crawlers, adaptive query routing

1. INTRODUCTION

Peer networks are increasingly seen as candidate frameworks for distributed Web search applications. One approach for peer-based search is to maintain a centralized registry for query routing [2]. This makes it difficult to adapt the search process to the heterogeneous and dynamic contexts of the peer users. The opposite, completely decentralized approach (as in early versions of Gnutella) has the well-known disadvantage that peers are flooded with traffic from queries and responses. An intermediate approach between the flood network and the centralized registry is to utilize index lists in distributed, shared hash tables [5]. Adaptive, content based routing has been proposed as an alternative; NeuroGrid [3] uses LSI to assign search keywords and route queries in a file sharing setting. We have presented a different model for peer-based Web search with a learning algorithm by which each peer uses the results of its interactions with its neighbors to refine a model of its neighbors [1]. This model is used to dynamically route queries according to the predicted match with other peers' knowledge. The network topology is thus modified on the fly based on learned contexts and current information needs.

WWW 2005, May 10–14, 2005, Chiba, Japan.
ACM 1-59593-051-5/05/0005.

2. PEER MODEL AND QUERY ROUTING

Each peer has a unique local search engine and it can send queries Q to other peers and respond to queries from other peers with messages containing search results, scores, and a peer location.

Each peer learns and stores two neighbor profile matrices, W^f and W^e for *focused* and *expanded* information, respectively. Each profile matrix has the same structure; rows correspond to terms and columns to peers. Thus an element $w_{t,p}$ of W is the contribution of term t to the profile of known peer p. In the focused profile, $t \in Q$ (query terms); in the expanded profile, t includes terms that co-occur with Q within the hits. For query forwarding, known peers are ranked by similarity σ between Q and the peer profiles computed as follows: $\sigma(p, Q) = \sum_{t \in Q} \left[\alpha \cdot w^f_{t,p} + (1-\alpha) \cdot w^e_{t,p}\right]$ where α is a reliability parameter regulating the contributions of focused and expanded profiles. The top N_n ranked among known peers are selected as neighbors and sent Q.

When a response is received, a peer uses the following learning rule to update the weights of the query terms in each neighbor profile matrix: $w_{t,p}(i+1) = (1-\gamma) \cdot w_{t,p}(i) + \gamma \cdot \frac{S_p+1}{S_l+1}$ where i is a time step, S_p and S_l are the average scores of p's hits and the local hits respectively in response to the query Q, and γ is a learning rate parameter. If the sender was not known, a list of most frequent keywords in the sender's search database is requested as its initial focused profile. Many other details of the proposed peer Web search framework are omitted for brevity.

3. EXPERIMENTAL SETUP

We developed a simulator with $N = 500$ peers belonging to 50 different groups of 10 peers each. We ran their queries over real indexes obtained from actual topical Web crawls. Our simulator takes a snapshot of the network topology for every time step.

Each group is associated with a general topic. For the peers in a given group, their own search engines are built by topical crawlers focusing on the same topic. The group topics are chosen from the ODP (`dmoz.org`). For each group, we extract a set of 100–200 URLs from the ODP subtree rooted at the category node corresponding to the group's topic. Random subsets of these URLs are assigned to the peer crawlers as seeds.

Each peer has 10 local queries, each of them generated by extracting the title words of a Web site in its group's ODP subtree. The peer uses its group topic and its own seed URLs to crawl 1,000 pages (for a total of 500,000 pages). The topical crawlers employ a *best-N-first* crawling algorithm [4]. The `nutch.org` indexer is used to build each peer's search engine from its crawled pages.

Each peer can forward queries to $N_n = 5$ neighbors. At the beginning of each experiment, the peer network is initialized as a random *Erdos-Renyi* graph, i.e., each peer is assigned 5 random neigh-

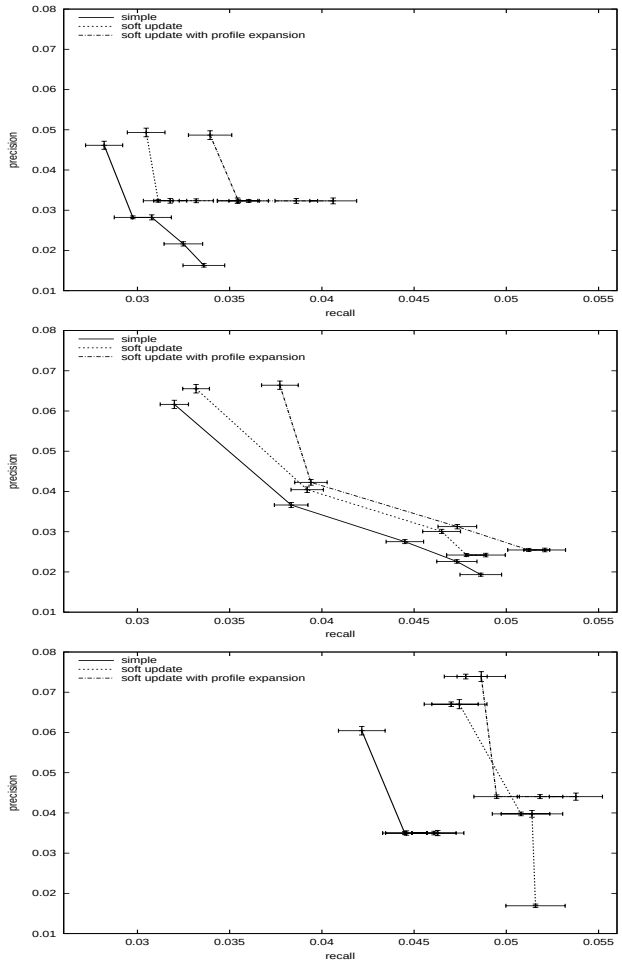

Figure 1: Precision-recall plots for three learning schemes, taken at the start of the simulation (top), after 504 time steps (middle), and after 1000 time steps (bottom).

bors drawn from a uniform distribution, irrespective of groups. A query can be forwarded at most TTL=3 times from one peer. We used three different query routing schemes: (1) *simple* is a baseline algorithm that updates W^f by replacing $w^f_{t,p}$ with the best hit score from p [1]; (2) *soft update* uses W^f with the update rule described in § 2; and (3) *soft update with profile expansion* uses both W^f and W^e. For each query routing scheme, we ran the simulator for about 1,200 time steps, corresponding to 120 queries issued per peer. Since there are only 10 distinct queries per peer, each query is submitted 12 times in the course of a simulation. Finally we empirically set the profile parameters to $\gamma = 0.3$ and $\alpha = 0.8$.

4. ANALYSIS OF RESULTS

We computed precision and recall of the results for each simulation using the three query routing schemes. Each query's relevant set in our simulation is simply the set of URLs classified by the ODP under the same topic as the page whose title is used as query. We show precision-recall snapshots in Figure 1.

Already at the start we observe a difference in performance between the learning algorithms. One might be surprised by such a difference after the first query since all peers in each simulation begin with empty profiles. However, during the 4 time steps the first query took to propagate, adaptive peers in the query path had already learned about their neighbors, hence they could better forward the query. Beside showing that all query routing schemes take advantage of the learning and improve their performance over time, Figure 1 also confirms that the more sophisticated learning algorithms outperform the simpler ones, with the best performance achieved by combining expanded profiles and the soft profile update rule.

Table 1: Average precision @ 10 of Google and peer search.

	$\langle P_{10} \rangle$	$\sigma_{\langle P_{10} \rangle}$	95% Confidence Interval
Google	0.079678	0.00095	(0.0778, 0.0816)
Peer search	0.078380	0.00062	(0.07714, 0.07962)

As a last analysis we compare the quality of the results obtained by our model with those returned by a real-world search engine. To this end we queried the Google Web API. As a summary performance measure we employed the commonly used *average precision @ 10*, $\langle P_{10} \rangle$.

As shown in Table 1 the difference in performance between the two systems is not statistically significant, suggesting that our model can be competitive with much larger search engines — the number of pages indexed by Google is about 10^4 times larger than those of the entire peer network in our simulation. The pages used as relevant sets in this experiment are well known to Google, and using their titles as queries allowed it to retrieve and rank very highly the pages with those titles. However, our model users can exploit their context and share their knowledge via collaboration during the search process, while Google has a single, universal ranking function and cannot exploit such context. Another factor to be considered is that Google may have returned other relevant pages which were not in our relevant sets; our automatic assessment methodology would not allow us to give credit for those. Despite this caveat, we find the comparative result very encouraging.

5. CONCLUSION

The experiment results show that the collective search performance of our peer network improves as more sophisticated learning algorithms are employed by the peers to route queries. Additionally the results suggest that our model scales well up to 500 peers, the maximum number of users we were able to simulate in a closely controlled testing environment, giving us confidence for the imminent public release of an open prototype.

6. ACKNOWLEDGMENTS

We are grateful to Nutch for its open source search engine code, to Gautam Pant for the topical crawler libraries, and to the ODP for the data used to model our simulated users. This work was supported in part by NSF Career Grant IIS-0348940.

7. REFERENCES

[1] R. Akavipat, L.-S. Wu, and F. Menczer. Small world peer networks in distributed Web search. In *Alt. Track Papers and Posters Proc. 13th International World Wide Web Conference*, pages 396–397, 2004.

[2] M. Bawa, R. Bayardo Jr, S. Rajagoplan, and E. Shekita. Make it fresh, make it quick — searching a network of personal webservers. In *Proc. 12th International World Wide Web Conference*, 2003.

[3] S. Joseph. Neurogrid: Semantically routing queries in Peer-to-Peer networks. In *Proc. Intl. Workshop on Peer-to-Peer Computing*, 2002.

[4] F. Menczer, G. Pant, and P. Srinivasan. Topical web crawlers: Evaluating adaptive algorithms. *ACM Transactions on Internet Technology*, 4(4):378–419, 2004.

[5] C. Tang, Z. Xu, and S. Dwarkadas. Peer-to-peer information retrieval using self-organizing semantic overlay networks. In *Proc. ACM SIGCOMM '03*, 2003.