Exploiting Global Impact Ordering for Higher Throughput in Selective Search

Michał Siedlaczek Juan Rodriguez Torsten Suel

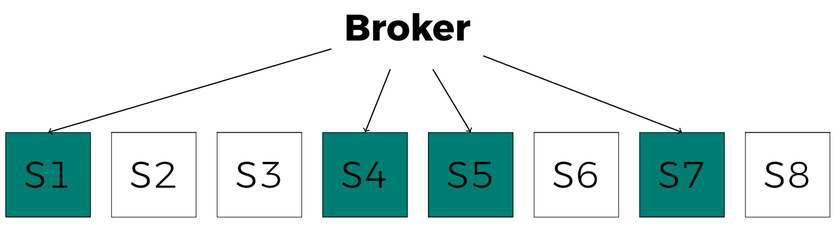
New York University · michal.siedlaczek@nyu.edu

Abstract

We exploit a global impact ordering in a selective search architecture for better cost-quality trade-off. Our solution generalizes the learning-to-rank-resources framework [4] and is suitable for increasing query throughput during periods of peak load or in low-resource systems.

Selective Search

- Collection clustered into **topical shards**
- At query time, a non-exhaustive subset of shards, relevant to the query, selected to process

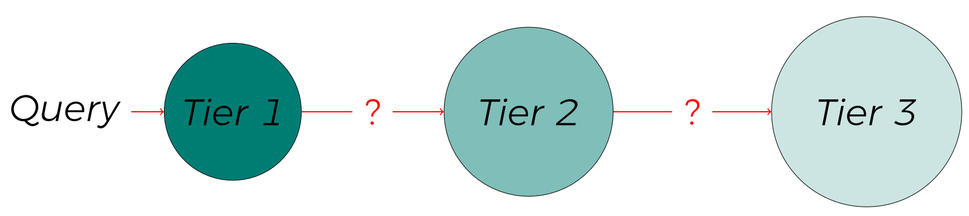


Global Impact Ordering

Global impact ordering (or **static document ranking**) is a query-independent ordering of the documents in a collection in terms of quality or importance. Examples: PageRank [6], spam scores [3], query logs-based methods [2, 5], and learning models [7].

Index Tiering

- Collection partitioned into **tiers** based on document quality
- First tier always processed
- Following tiers only processed when the relevance from the first is found to be or predicted to be insufficient

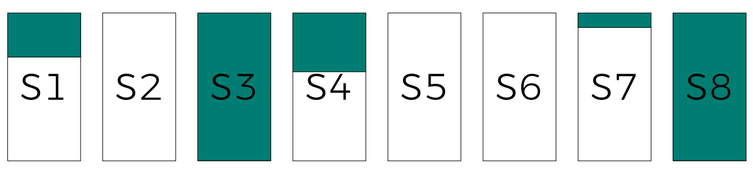


Global Rank Cutoff

- Documents reordered according to overall quality
- Processing a subset of best quality documents, e.g.:
  - all documents lower than certain d (like tier 1 but can be different each time),
  - process until reached score or time threshold.

Ordering-Aware Selective Search

- **Topical shards** like in selective search
- **Documents reordered** in each shard according to a static ranking
- At query time, **selecting a document cutoff** for each shard



Reordering

Given a **query log** Q and a document d, we define a hit count $h(d) = \sum_{q \in Q} I_q(d)$ where $I_q(d)$ is equal to 1 if d is among the top k results retrieved for query q, or 0 otherwise (we used $k = 1000$). In each shard, IDs are assigned to documents based on decreasing hit count.

Acknowledgement. This research was partially supported by NSF Grant IIS-1718680 and a grant from Amazon.

Ordered Partitioning

We simplify training and selection by setting b evenly spaced cutoff points, effectively creating b subranges we call **buckets**.

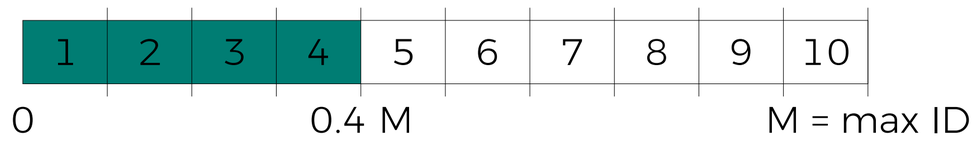


Figure 1: An illustration of an ordered partitioning with $b = 10$. Evenly spaced cutoff points determine b buckets. Processing 4 buckets is equivalent to processing all documents with IDs at most $0.4M$ where M is the number of documents in the collection.

Features

Following [4], we use the following features (CSI-based features were ignored as slow and having limited impact):

- Overall popularity of a shard
- Taily [1] (3 features)
- Champion list — number of documents each shard contributes to term's top-k (2 features: $k \in \{10, 100\}$)
- Query likelihood features (9 features)
- Query term statistics (12 features)
- Bigram log frequency
- Additional feature: **bucket number**

At query time, bucket selection is translated into numbers of leading buckets in each shard.

Experiments

- GOV2 and Clueweb09-B collections partitioned into 199 and 123 shards, respectively [4]
- Higher throughput at very low costs
- Improvements achieved by enabling partial shard access
- Resources for traversing unimportant documents in highly relevant shards put towards traversing highly important documents in less relevant shards.

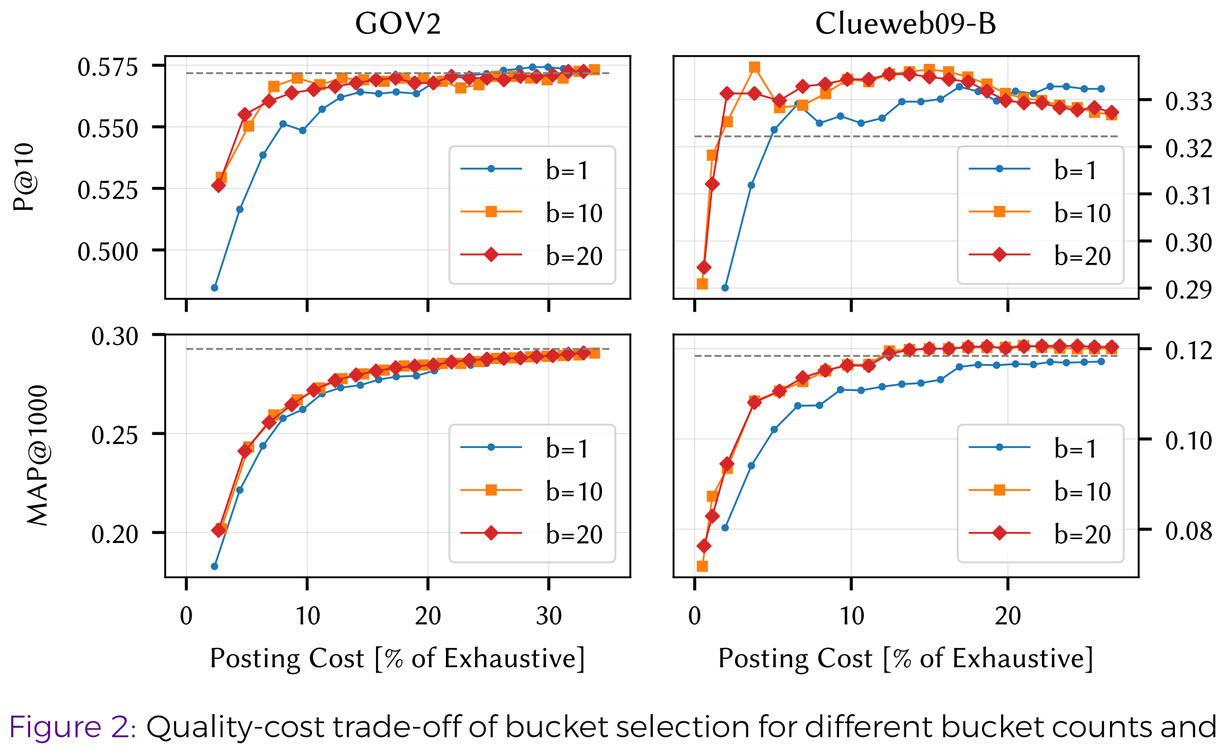


Figure 2: Quality-cost trade-off of bucket selection for different bucket counts and data sets. Shards were selected under the uniform cost model and with budgets from 1 to 20 shards. For Clueweb09-B, additional budgets of 0.25 and 0.5 were tested for $b > 1$. The quality of exhaustive search is indicated by a dashed line.

References

[1] R. Aly, D. Hiemstra, and T. Demeester. Taily: Shard selection using the tail of score distributions. In *Proc. of the 36th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2013.

[2] A. Anagnostopoulos, L. Becchetti, S. Leonardi, I. Mele, and P. Sankowski. Stochastic query covering. In *Proc. of the 4th ACM Intl. Conf. on Web Search and Data Mining*, 2011.

[3] G. V. Cormack, M. D. Smucker, and C. L. A. Clarke. Efficient and effective spam filtering and re-ranking for large web datasets. *Information Retrieval*, 14(5), 2011.

[4] Z. Dai, Y. Kim, and J. Callan. Learning to rank resources. In *Proc. of the 40th Ann. Intl. ACM SIGIR Conf. on Research and Development in Information Retrieval*, 2017.

[5] Steven Garcia, Hugh E Williams, and Adam Cannane. Access-ordered indexes. In *Proceedings of the 27th Australasian Conference on Computer Science*, 2004.

[6] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical Report 1999-66, November 1999.

[7] M. Richardson, A. Prakash, and E. Brill. Beyond pagerank: machine learning for static ranking. In *Proceedings of the 15th International Conference on World Wide Web*, 2006.