Efficient Location-aware Web Search

J. Shane Culpepper

Department of Computer Science and Information Technology,
RMIT University, Australia.

November 9th, 2015

Research Agenda

Efficient, Scalable Algorithm Design

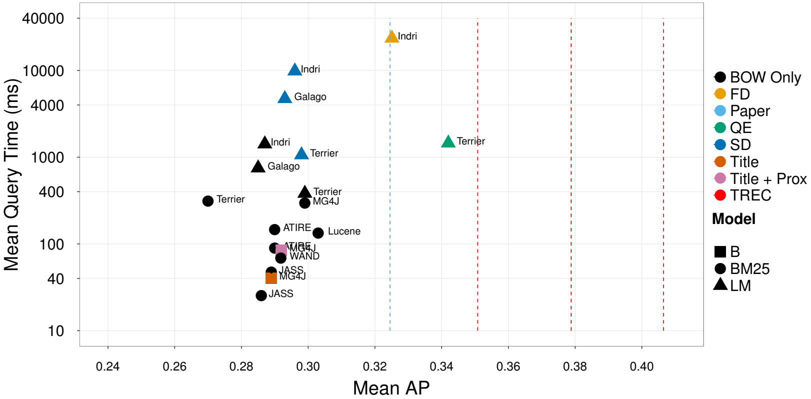
1. Create new algorithms and data representations to support efficient query processing (verbose queries).
2. Extend and refine in-memory indexing algorithms to support parallel, distributed, and dynamic indexing of massive data sets more efficiently.
3. Explore efficiency and effectiveness trade-offs in large scale search algorithms.
4. Investigate new approaches to combining structured and unstructured search.
5. Develop and understand IR evaluation best practices.

Interesting Facts

1. Energy costs can now exceed hardware costs in many “big data” applications.
2. Every 100 ms boost in search speed increases revenue by 0.6% at Bing.[†]
3. Commodity computer systems can now support over 1 TB RAM and 40+ CPU cores in a single machine, but trends in practical algorithmic design have not mirrored these advances.

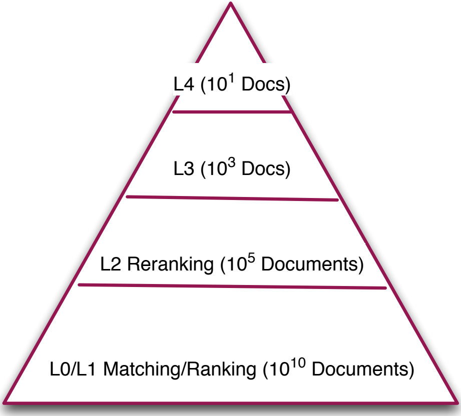
[†] R. Kohavi, A. Deng, B. Frasca, T. Walker, Y. Xu, and N. Pohlmann: “Online controlled experiments at large scale.” In Proc. KDD, pages 1168–1176, 2013.

State-of-the-Art in GOV2



Data from the RIGOR Workshop at SIGIR 2015. All systems are ran on a single EC2 instance using the TREC GOV2 collection with queries 701-850. More at `http://github.com/lintool/IR-Reproducibility`.

Multi-Stage Retrieval

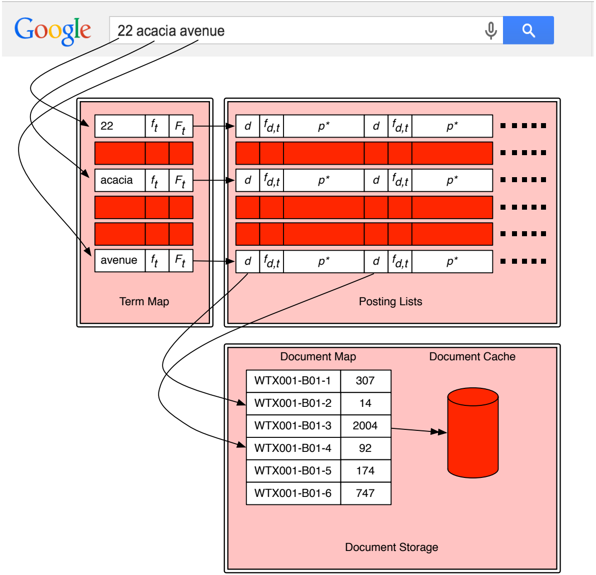

L4 (10^1 Docs)
L3 (10^3 Docs)
L2 Reranking (10^5 Documents)
L0/L1 Matching/Ranking (10^{10} Documents)

Bing – Sequence of select, rank, prune steps

- L0 Boolean Logic
- L1 BOW IR Scoring
- L2/L3/L4 Learning to Rank

1. Burges et al. “Learning to Rank Using Gradient Descent.” In ICML 2005.
2. Jan Pedersen: “Query Understanding at Bing.” SIGIR Industry Keynote, 2010.

The Inverted Index



WTX001-B01-1	307
WTX001-B01-2	14
WTX001-B01-3	2004
WTX001-B01-4	92
WTX001-B01-5	174
WTX001-B01-6	747

Processing Regimes for Inverted Lists

- **Score-safe processing** guarantees that an identical top-k ordering will occur for a given set ranking metric.
  - MAXSCORE
  - WAND
  - BLOCK-MAX
- **Heuristic processing** approximates the true top-k list.
  - STOP and CONTINUE
  - SCORE-AT-A-TIME / impact ordered indexing

The WAND operator

Weak AND, or Weighted AND – WAND

Given a list of Boolean variables $X_1, X_2, \ldots, X_t$ and a list of associated positive *weights*, $w_1, w_2, \ldots, w_t$ and a threshold θ, WAND$(X_1, w_1, \ldots, X_t, w_t, \theta)$ is true iff

$$\sum_{1 \le i \le t} x_i w_i \ge \theta,$$

where x_i is the indicator variable for X_i, that is

$$x_i = \begin{cases} 1 & \text{if } X_i \text{ is true} \\ 0 & \text{otherwise.} \end{cases}$$

A. Z. Broder, D. Carmel, H. Herscovici, A. Soffer, and J. Zien: “Efficient query evaluation using a two-level retrieval process.” In Proc. CIKM, pages 426–434, 2003.

The WAND operator

$$\sum_{1 \leq i \leq t} U_i > \theta$$

- Assume the U_t is the maximum score a term $t \in q$ can contribute in an additive scoring regime.
- If we set θ to the current score of the k·th smallest item in a heap of k items, only items that *might* be able to enter the heap will ever be scored.
- If we set $\theta = \infty$, then only the first k items evaluated can ever enter the heap.
- By varying the threshold, WAND can move from being close to an **OR** operation to being close to an **AND** operation.

How to use WAND (Sketch)

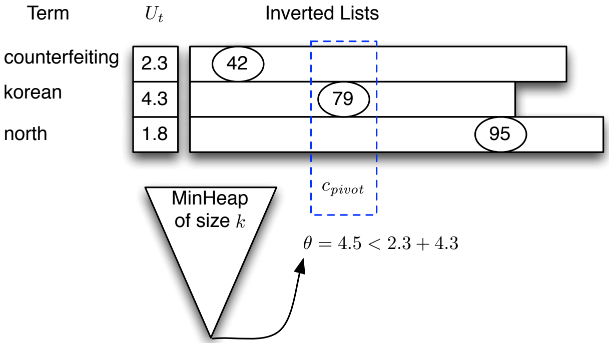
Index Time:

1. Pick an additive ranking metric and build a document ordered inverted index.
2. For each inverted list, pre-calculate a value U_t which represents the maximum contribution term t can have for that scoring metric.

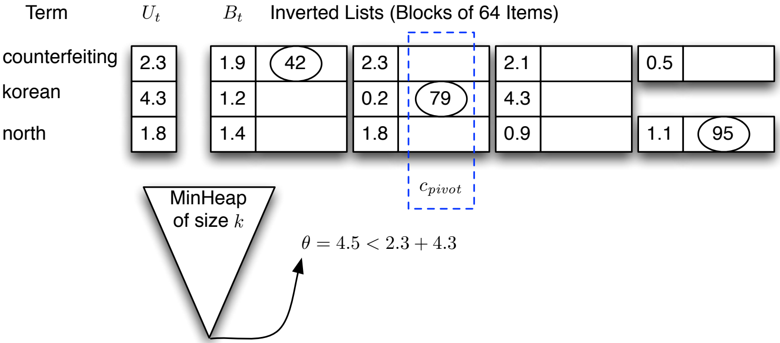
Query Time:

1. Perform a standard *document-at-a-time* traversal with one twist.
2. Set a *finger* pointing to the first unevaluated document in each list.
3. Set the term processing order of the lists using the document id – smallest to largest.
4. While the sum of the U_t values are less than θ step to the next list.
5. As soon as we exceed θ, the document ID in the current list is selected as the *pivot*.
6. A finger search is initiated for all lists evaluated before the pivot, and the current pivot document ID is scored if all of the previous list document IDs match the pivot.
7. If the real document score exceeds the minimum value in a k heap, it is added to the heap, and θ is set to the new minimum heap score.

WAND in action



BM-WAND in action



Term	U_t	B_t	Inverted Lists (Blocks of 64 Items)						
counterfeiting	2.3	1.9	42	2.3		2.1		0.5	
korean	4.3	1.2		0.2	79	4.3			
north	1.8	1.4		1.8		0.9		1.1	95

c_{pivot}

MinHeap of size k

$\theta = 4.5 < 2.3 + 4.3$

[2] S. Ding and T. Suel: “Faster top-k retrieval using block-max indexes.” In Proc. SIGIR, pages 993–1002, 2011.

Motivation for this Work

- Recent talks at SIGIR and CIKM have highlighted the fact that the number of mobile searches have surpassed the number of desktop searches in all of the major search engines.
- For mobile searches, more than 50% of all queries have local intent.
- Leveraging GPS data for search is now a key feature in mobile applications.
- We know that the interaction between 10s or even 100s of features improves effectiveness. There is value in studying important features in isolation to better understand when and where they work best.

Spatial-Textual Queries

Two common spatial queries:

- **Range query** – Given a location *p* and a maximum travel distance, a range query returns all items whose location fall within that distance from *p*. An example of a spatial range query is when a user wants to find all of the petrol stations within 5 km of their current location.
- **KNN query** – Given a specific location *p*, the query returns the *k* items closest to *p*.

Two common textual queries:

- **Conjunctive Boolean (AND) query** – Given a set of query terms $t_1 \ldots t_q$, return all documents containing all terms.
- **Disjunctive Boolean (OR) query** – Given a set of query terms $t_1 \ldots t_q$, return all documents containing any of the terms.

Research Questions

Research Question 1: Which indexing approaches provide the best efficiency trade-offs for bag-of-words, top-*k* *knn* search in large document collections?

Research Question 2: What are the efficiency and effectiveness trade-offs for the two most common spatially constrained query types when applied to bag-of-words, location-aware search?

Approach Overview

- **Filter-based Methods** – Apply a spatial filter (KNN or MBR) to identify a subset of candidate documents. Reorder the documents using a textual similarity function to get the final top-*k* document list.
    - Use an R-Tree to do efficient range filtering.
    - Use a KD-Tree or an R*-Tree to do efficient KNN filtering.
    - Score the filter set using BM25 or any TF·IDF ranking metric.

- **Top-*k* KNN Methods** – Score the top-*k* documents in a single pass using both components to prune to search space.
    - Use a hybrid index such as an IR-Tree.
    - Use an augmented inverted index.

IR Tree

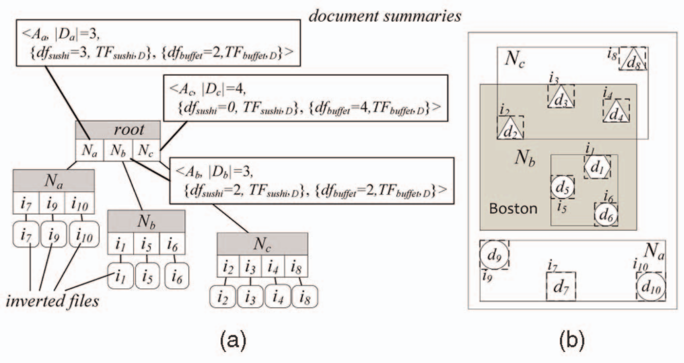

document summaries
<A_a, |D_a|=3, {df_sushi=3, TF_sushi, D}, {df_buffet=2,TF_buffet, D}>
<A_c, |D_c|=4, {df_sushi=0, TF_sushi, D}, {df_buffet=4,TF_buffet, D}>
<A_b, |D_b|=3, {df_sushi=2, TF_sushi, D}, {df_buffet=2,TF_buffet, D}>
root
N_a N_b N_c
N_a
i_7 i_9 i_10
N_b
i_1 i_5 i_6
N_c
i_2 i_3 i_4 i_8
inverted files
(a)
N_c
i_8 d_8
i_3 d_3
i_4 d_4
i_2 d_2
N_b
i_1 d_1
d_5 i_5
i_6 d_6
Boston
d_9 i_9
i_7 d_7
N_a
i_10 d_10
(b)

Fig. 4. An example IR-tree. (a) IR-tree content. (b) Node MBBs.

G. Cong, C. S. Jensen, and D. Wu: “Efficient retrieval of the top-*k* most relevant spatial web objects.” PVLDB, 2(1):337348, Aug. 2009.

Z. Li, K. C. K. Lee, B. Zheng, W.-C. Lee, D. Lee, and X. Wang: “IR-tree: An efficient index for geographic document search.” TKDE, 23(4):585599, Apr. 2011.

Combining Text and Spatial Similarity

Spatial-Textual Ranking

Given a spatial-textual query q and a document d, the spatial-textual similarity can be computed as:

$$S(q, c) = \alpha \cdot \beta(\ell_q, \ell_d) + (1 - \alpha) \cdot \sum_{t \in q} \gamma_t$$

where $\beta(\ell_q, \ell_d)$ is the spatial similarity, γ_t is the textual similarity, and $\alpha \in (0, 1)$ is a parameter that can be used to add weight the importance of the two components.

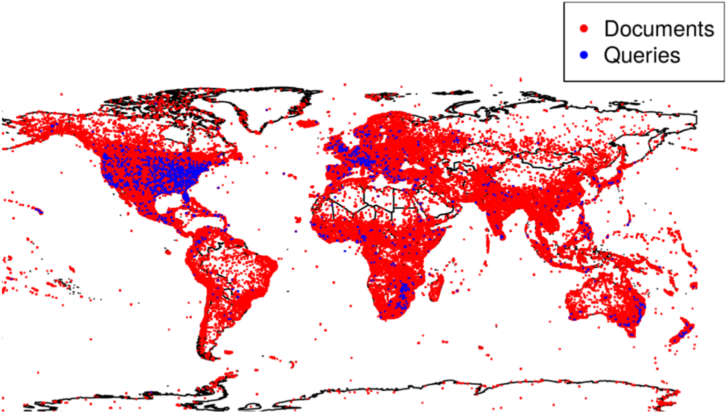
- $\beta(\ell_q, \ell_d) = (1 - \frac{Euclidean(\ell_q, \ell_d)}{d_{\max}}) \cdot \sum_{t \in q} U(\gamma_t)$
- In this work, γ_t is the BM25 ranking function. However, any TF·IDF scoring function could be used.
- This is referred to as a top-*k knn* query.

GEOWAND Algorithm

Algorithm 2: WAND processing with geospatial weights, replacing steps 10–21 in Algorithm 1 of Petri et al. [23].

```
2.1  $text\_limit \leftarrow 0$; $spatial\_limit \leftarrow 0$; $pivot \leftarrow 0$
2.2  while $pivot < |q| - 1$ do
2.3      $text\_limit \leftarrow text\_limit + (1 - \alpha) \cdot U[pivot]$
2.4      $spatial\_limit \leftarrow spatial\_limit + \alpha \cdot U[pivot]$
2.5      if $text\_limit + spatial\_limit > \theta$ then
2.6          $s \leftarrow text\_limit + spatial\_limit$
2.7          break, and continue from Step 2.9.
2.8      $pivot \leftarrow pivot + 1$
2.9  if $c_0 = c_{pivot}$ then
2.10     $t \leftarrow 0$
2.11     $s \leftarrow s - spatial\_limit + \alpha \cdot \beta(\ell_q, \ell_{pivot})$
2.12     while $t < |q| - 1$ and $c_0 = c_{pivot}$ do
2.13         $s \leftarrow s - (1 - \alpha) \cdot U[t] + (1 - \alpha) \cdot \gamma_t$
2.14         if $s < \theta$ then
2.15             break, and reset all pointers to $c_{pivot} + 1$.
2.16         ▷ Update heap and $\theta$ ... remaining computation similar to the
             approach described in Algorithm 1 of of Petri et al. [23].
```

Spatial datasets



- WikiGeo – 1 Million Geotagged Documents (9 GB Uncompressed)
- CW09BGeo – 24 Million Geotagged Documents (230 GB Uncompressed)

Collection Creation

- Used a combination of the GeoNames gazetteer and the Freebase annotations for the ClueWeb Corpora.
- We also used the TREC Freebase parse of the Million query track.
- If multiple locations were found in a document, we used the first one.
- If multiple locations were in a query, we replicated the query, once for each location.
- For documents, many were clustered on one exact location (New York). So, we also did location fuzzing where we randomly added a very small epsilon to the Latitude / Longitude coordinates.
- This produced 24 million unique, geotagged documents, and 5,315 geotagged queries.
- A subset of the main collection was created by taking the first million documents that were ordered by spam score, many of which turned out to be Wikipedia documents.

Systems Evaluated

System	Abbreviation
Standard WAND (text only)	W
GeoWAND	G-W
IR-Tree	IR
R*-Tree (*knn* queries)	W-R*
kd-Tree (*knn* queries)	W-kd
R*-Tree (1km MBR queries)	MBR-1
R*-Tree (10km MBR queries)	MBR-10
R*-Tree (100km MBR queries)	MBR-100

Table: System configurations in all experiments. The W-R*, W-kd, and MBR-*k* methods all use WAND over an inverted index to rank queries once the spatial filtering has finished. The W-R* and W-kd methods retrieve the 2.5 × *k* closest documents before reording.

Experiment - WikiGeo Performance

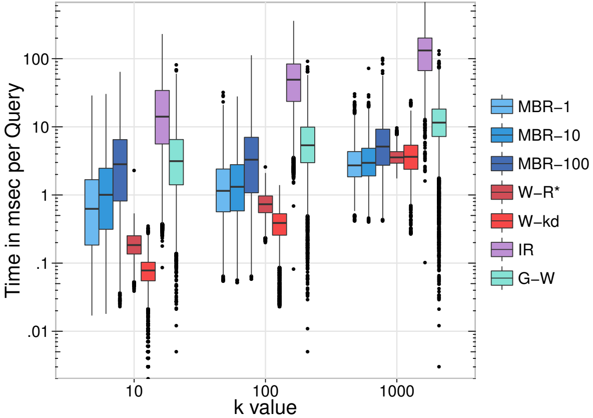


Figure 4: The mean running time per query as a function of k for all query type and index combinations on the WikiGeo dataset, with $\alpha = 0.5$.

Experiment - WIKIGEO Index Size

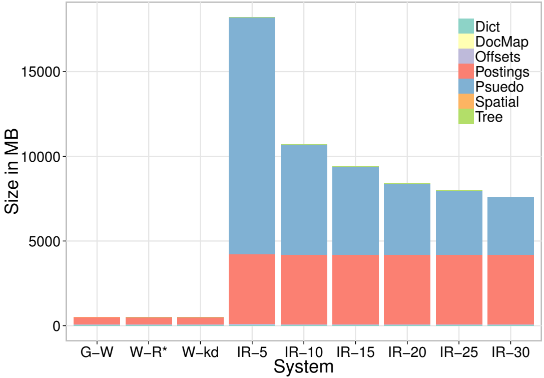


Figure 3: A comparison of the space usage for a GEOWAND inverted index, a loosely coupled WAND inverted index combined with an R-Tree and k-d-Tree, and IR-Tree of varying fanouts. The space difference between the different WAND and GEOWAND indexes is negligible.

Experiment - Tree Traversal

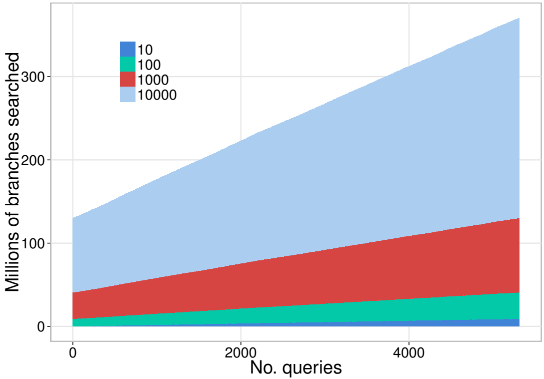
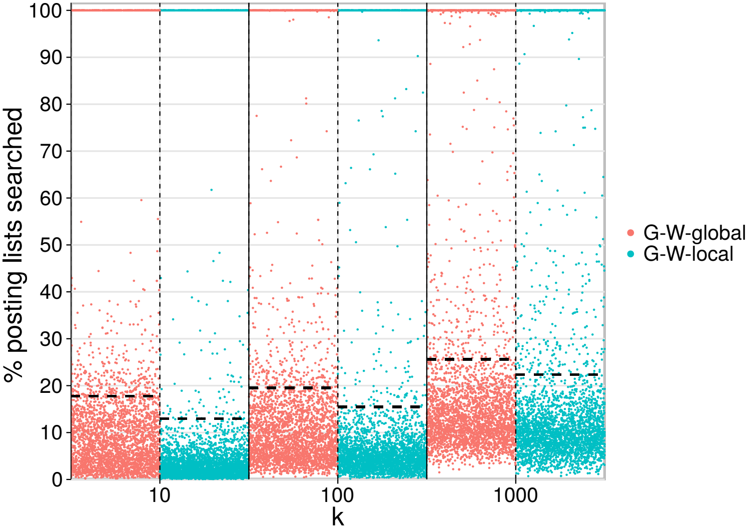


Figure 5: The cumulative number of branches evaluated for the IR-Tree traversal for $\alpha = 0.5$ on the WikiGeo collection. The greedy search must search many more nodes as k increases, which results in a significant increase in the number of cache misses incurred when walking the tree.

WIKIGEO Summary

- The IR-Tree can use up to 2 times the space of the uncompressed collection.
- A WAND-based approach is around 20% of the size of the uncompressed collection.
- So, there is an order of magnitude difference in space usage between the two approaches.
- To be fair, the IR-Tree was designed to be ran on-disk, and object sizes are often much smaller than web documents.
- The vocabularies are therefore much smaller, and the pseudo nodes can be made smaller by reading only terms for the current query into memory at processing time.
- We were unable to build the IR-Tree on the full CW09BGEO collection.

Optimizing GEOWAND



Effectiveness

α	MBRF	MBRF	MBRF	KNNF	KNNF	KNNF
	1 km	10 km	100 km	$k = 10$	$k = 100$	$k = 1000$
$\text{MED}_{\text{RBP0.95}}$						
0.2	0.7164	0.6978	0.6601	0.9947	0.9936	0.9703
0.5	0.5991	0.5746	0.5246	0.9921	0.9902	0.9551
0.8	0.4412	0.4085	0.3432	0.9880	0.9849	0.9319
$\text{MED}_{\text{ERR@20}}$						
0.2	0.6844	0.6646	0.6264	0.9655	0.8892	0.8746
0.5	0.5589	0.5332	0.4825	0.9473	0.8354	0.8149
0.8	0.4021	0.3682	0.3042	0.9220	0.7625	0.7406

Table 3: The Maximum Expected Difference (MED) as measured with $\text{MED}_{\text{RBP0.95}}$ and $\text{MED}_{\text{ERR@20}}$. A MED value less than 0.2 is considered negligible. The higher the value, the more likely there is a noticeable effectiveness difference.

Efficiency vs Effectiveness

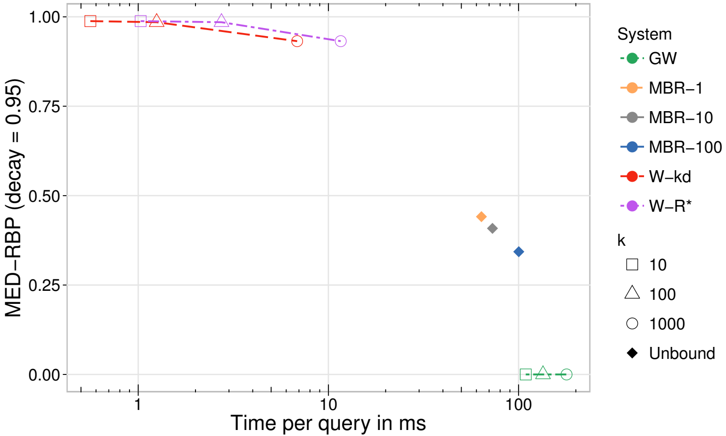


Figure 8: Effectiveness versus Efficiency for the CW09BGeo collection for $\alpha = 0.8$.

Summary, Limitations, and Future Work

- GEOWAND is simple to implement, rank-safe, and efficient. We believe users like things as simple as possible.
- The efficiency of all of the methods can be improved by reordering documents in the collection using a Space Filling Curve.
- It is not clear how to use document reordering when documents and/or queries have multiple locations.
- Our data set is synthetic. No good publicly available test collections exist for experiments on location-aware search. We are talking to industry partners about this problem now.
- No relevance judgments are available, so we do not have good tools to evaluate new ranking algorithms combining non-textual features.
- Location is one of many important non-textual features that can be used for ranking.

Thanks

Thanks to my amazing collaborators, without which most of this work would not be possible – Joel Mackenzie, Farhana Choudhury, Alistair Moffat, Matthias Petri, Xiaolu Lu, Gaya Jayasighe, Falk Scholer, William Webber, Timos Sellis, Mark Sanderson, Andrew Turpin, Gonzalo Navarro, Simon Puglisi, Charlie Clarke, Jamie Callan, Yubin Kim, and many others.

Questions?