Query-log based techniques for optimizing WSE efficiency

Salvatore Orlando[+], Raffaele Perego[*], Fabrizio Silvestri[*]
[*]ISTI - CNR, Pisa, Italy
[+]Università Ca' Foscari Venezia, Italy

Course Plan

- Class 1: Query log analysis.
- Class 2: Query-log based techniques for optimizing WSE effectiveness.
- Class 3: Query-log based techniques for optimizing WSE efficiency.
- Class 4: Hands-on session.
- Class 5: Future Research Issues and the Web of Data.

Web Searching

Is one of the most complex data engineering challenges today:

- Distributed in nature
- Large volume of data
- Highly concurrent service
- Users expect very good & fast answers

A key goal is optimizing throughput but making sure that query response time is below a given threshold

Throughput up, latency down: How to?

Mine Query Logs to devise patterns and predictors allowing to optimize all aspects of the complex system, e.g.:

- Ad hoc caching strategies for
  - results' pages (SERPs), posting lists, intersections, document, surrogates, snippets
- Parallel query processing on multiple clusters
  - replication, index partitioning (document-based or term-based), query routing, index pruning, etc.

Outline

- Ad hoc caching strategies for
  - results' pages (SERPs), posting lists, intersections, document, surrogates, snippets
- Parallel query processing on multiple clusters
  - replication, index partitioning (document-based or term-based), query routing

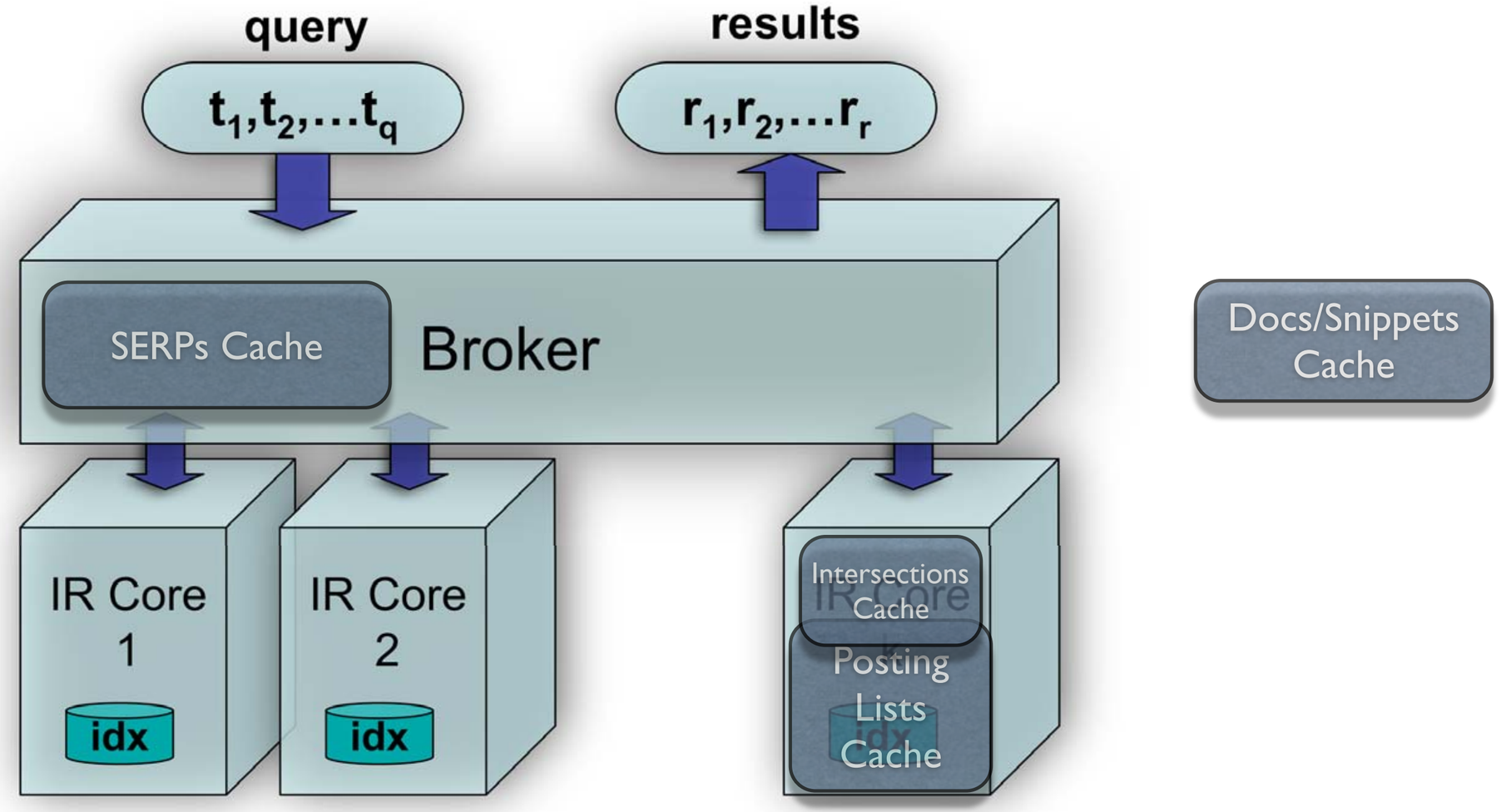
Caching

- Caching can save significant amounts of computational resources
  - Caching reduce latency and load on backend servers
  - Caching helps to make queries “local”
- Search engine with capacity of 1000 queries/second
  - Cache with 30% hit ratio increases capacity to 1400 queries/second
  - Almost for free!
- Caching is similar to replication on demand

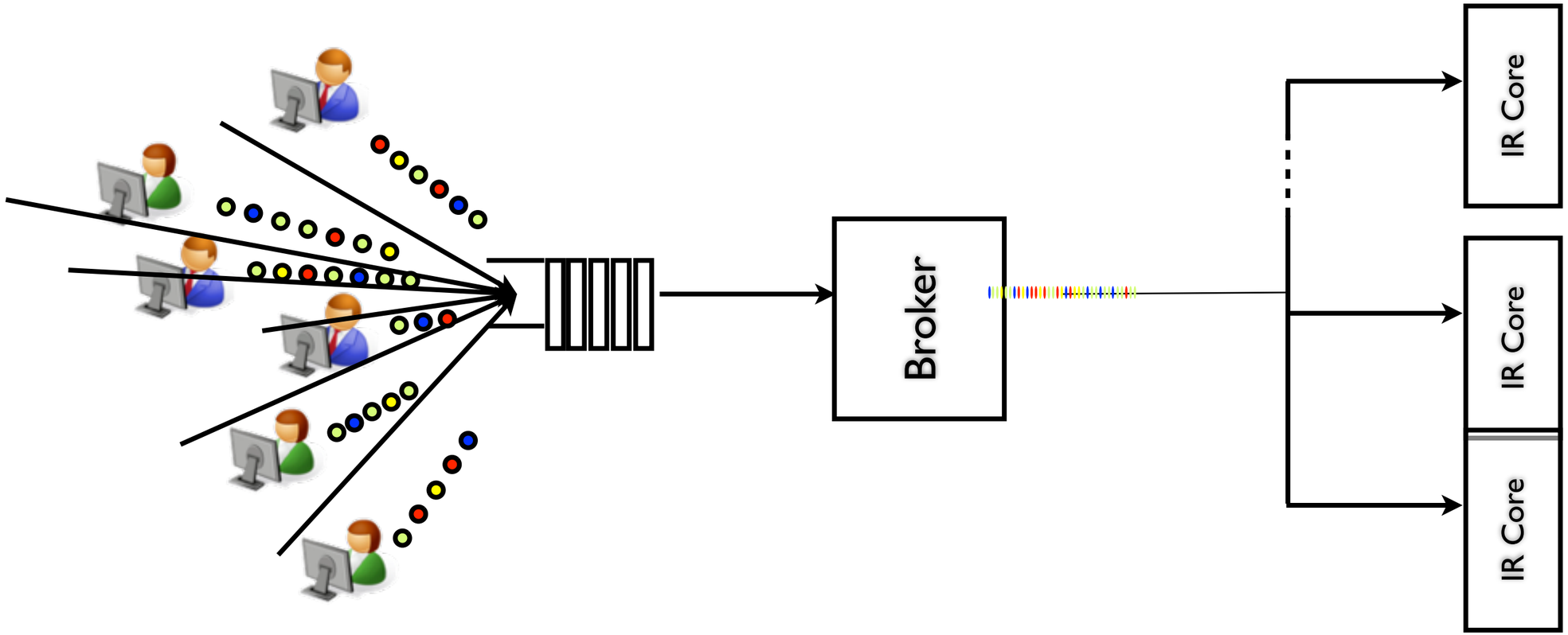
“Things” to Cache in Search Engines

- SERPs
  - 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 intersections for subqueries, e.g. for “new york times” cache only “new york”
- Documents or surrogates or snippets
- Ads, etc.

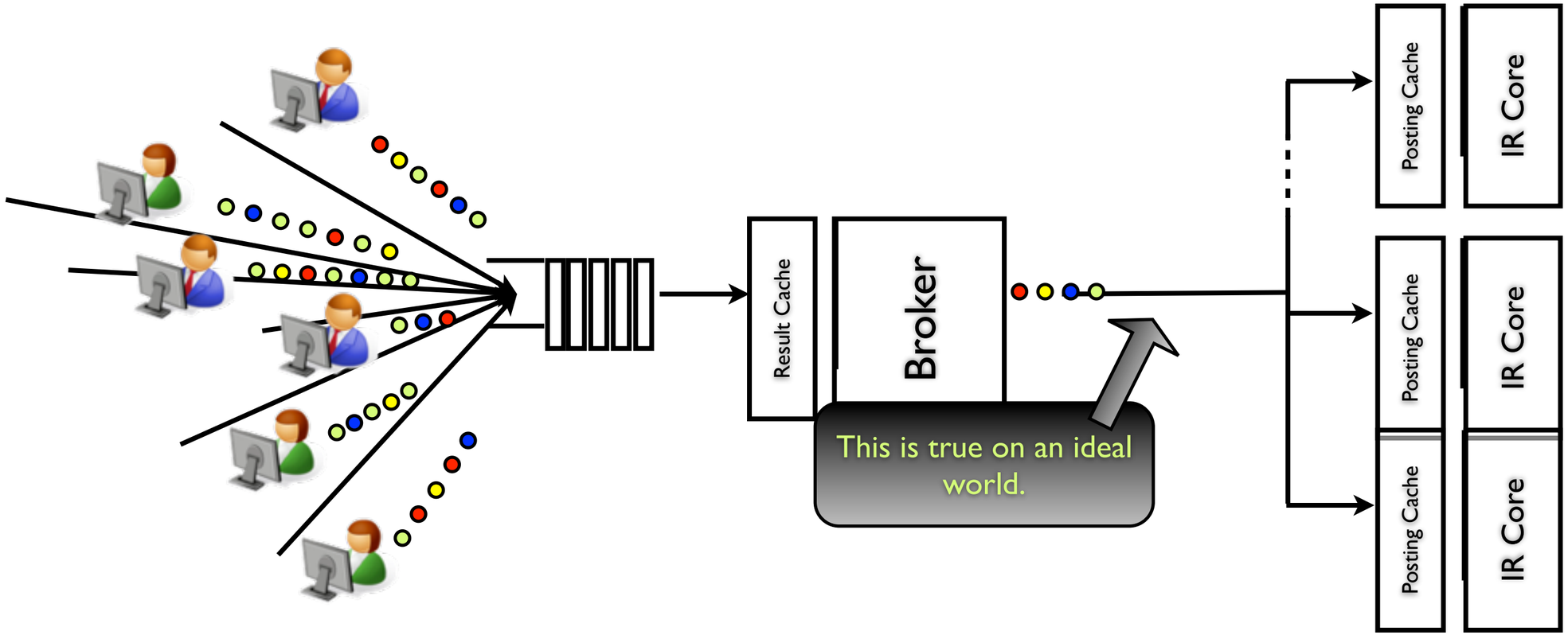
Caching in a Distributed Search Engine



SERP's Caching



Caching

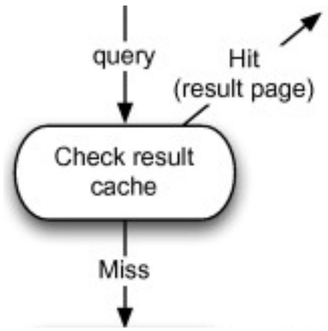

Result Cache
Broker
This is true on an ideal world.
Posting Cache
IR Core
Posting Cache
IR Core
Posting Cache
IR Core

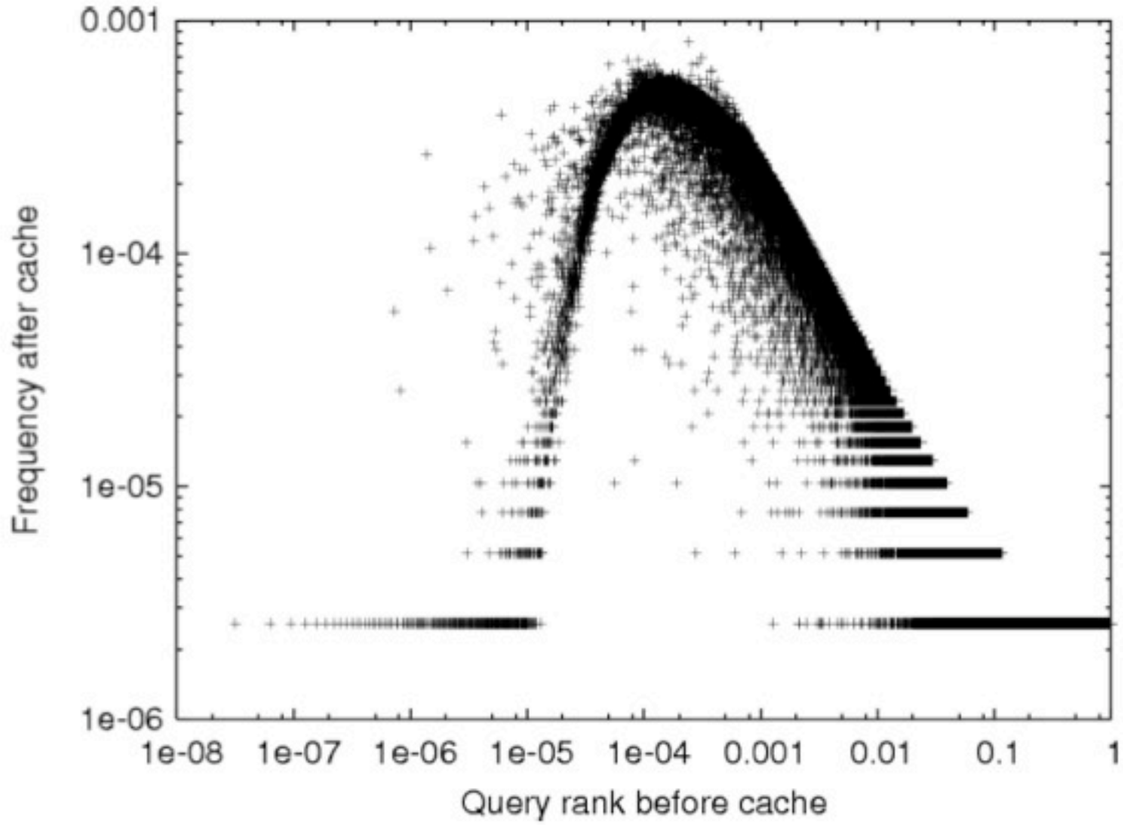
Caching Performance Evaluation

- **Hit-Ratio**: i.e. how many times the cache is useful
- **Query Throughput**: i.e. the number of queries the system can serve in a second

- But... what really impacts on caching performance?

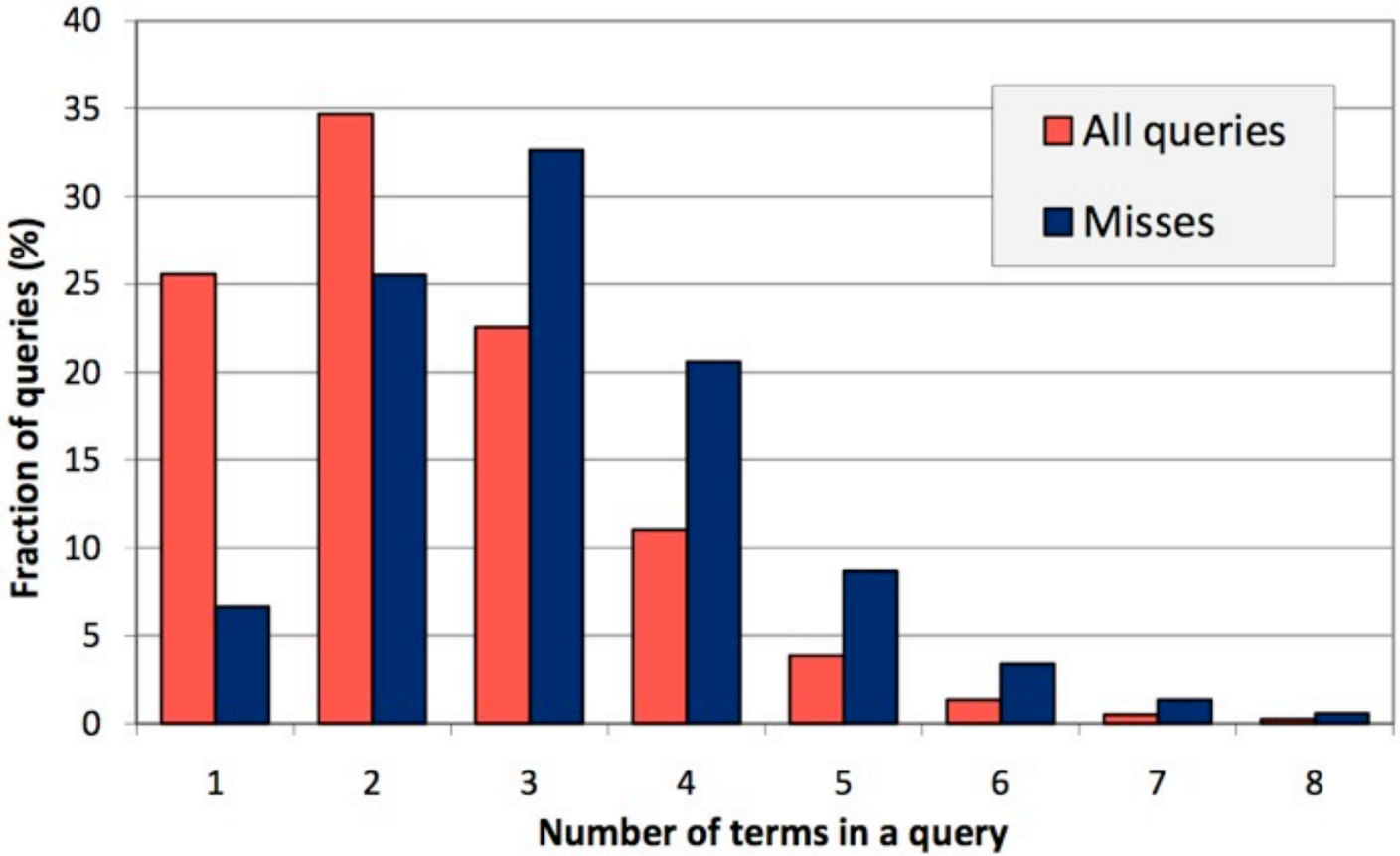
Filtering Effect of SERPs Caching





R. Baeza-Yates, A. Gionis, F. P. Junqueira, V. Murdock, V. Plachouras, and F. Silvestri, "**Design trade-offs for search engine caching**," ACM Trans. Web, vol. 2, no. 4, pp. 1–28, 2008.

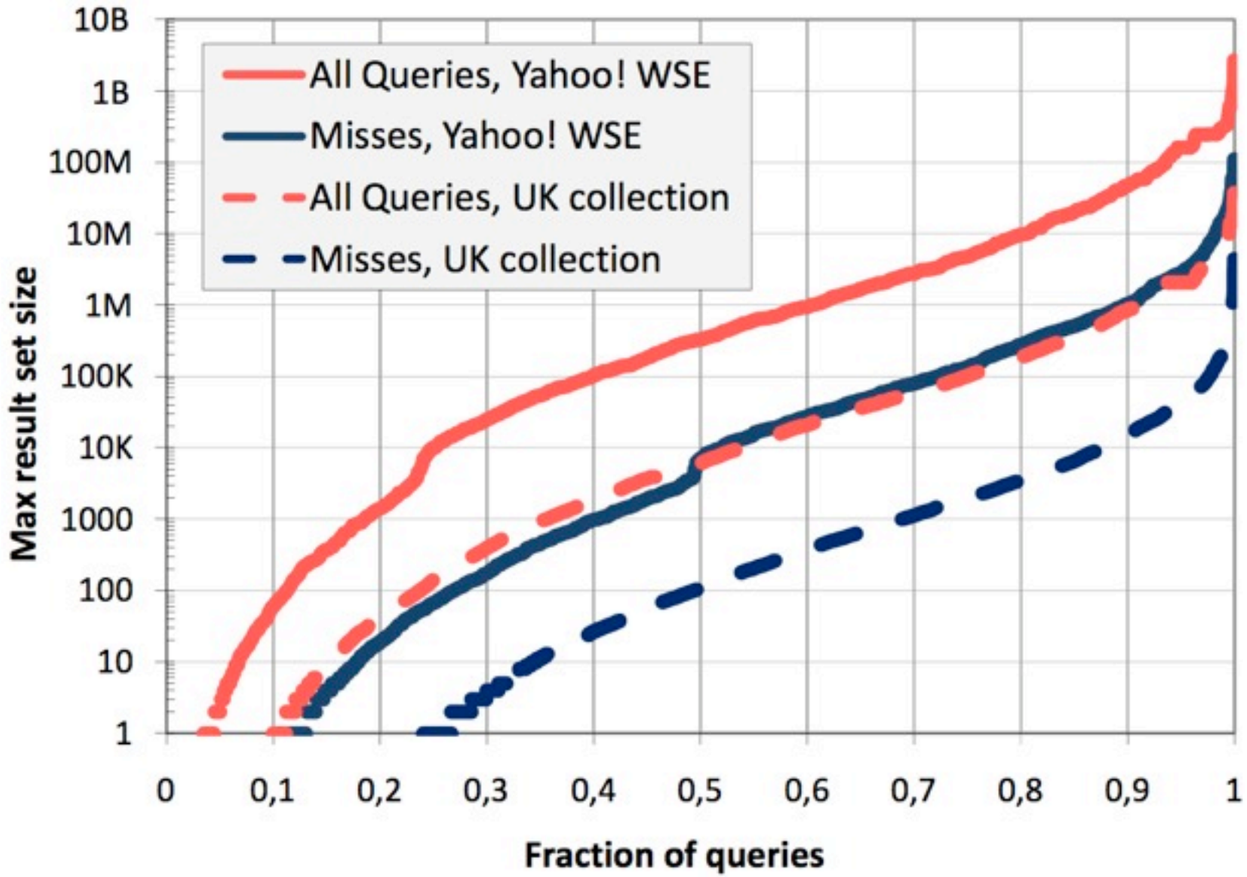
Filtering Effect of SERPs Caching



Skobeltsyn, G., Junqueira, F., Plachouras, V., and Baeza-Yates, R., "**ResIn: a combination of results caching and index pruning for high-performance web search engines**," SIGIR 2008, pp 131-138.

Filtering Effect of SERPs Caching

Avg. result size for misses is ~100 times smaller than for all queries

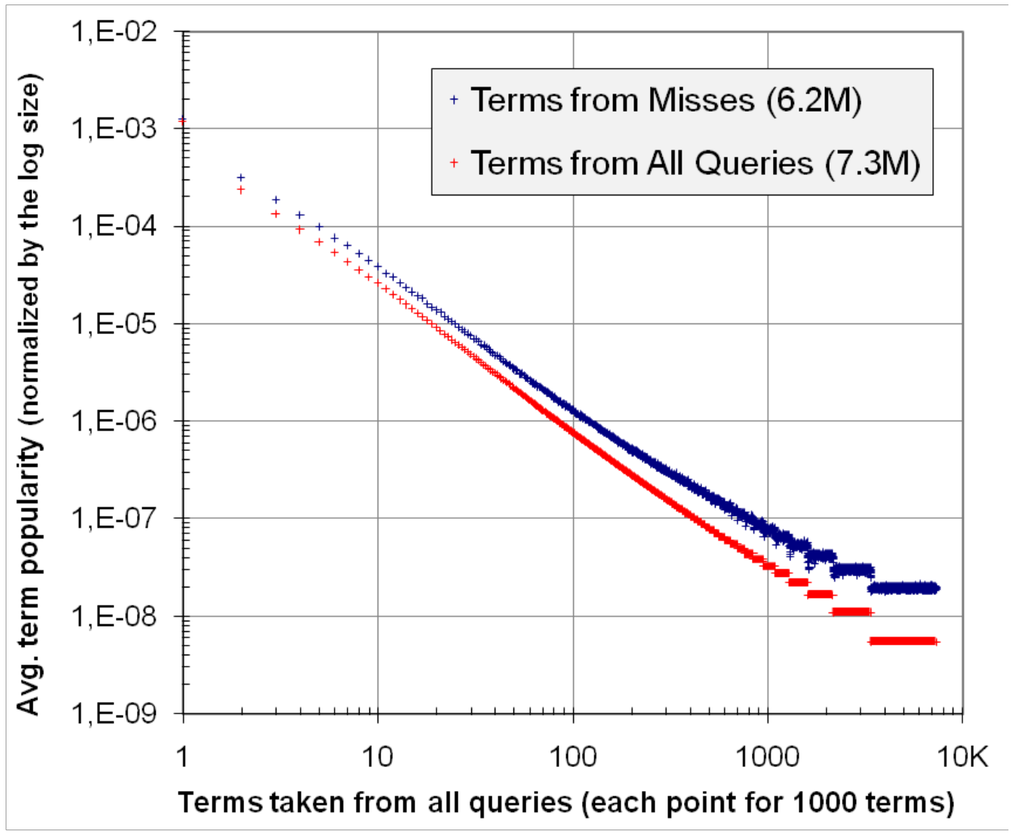


Skobeltsyn, G., Junqueira, F., Plachouras, V., and Baeza-Yates, R., "**ResIn: a combination of results caching and index pruning for high-performance web search engines**," SIGIR 2008, pp 131-138.

Terms popularity in hits/misses

- Term popularity does not change much
  - caching at lower levels!!

Log sizes: 185M – *all queries*, 41M - *misses*



Skobeltsyn, G., Junqueira, F., Plachouras, V., and Baeza-Yates, R., "**ResIn: a combination of results caching and index pruning for high-performance web search engines**," SIGIR 2008, pp 131-138.

Caching for Search Engines Workloads

- Caching Architectures:
  - Multi-Level Caching
- Caching Policies
  - PDC
  - SDC
  - AC

Two-Level Caching

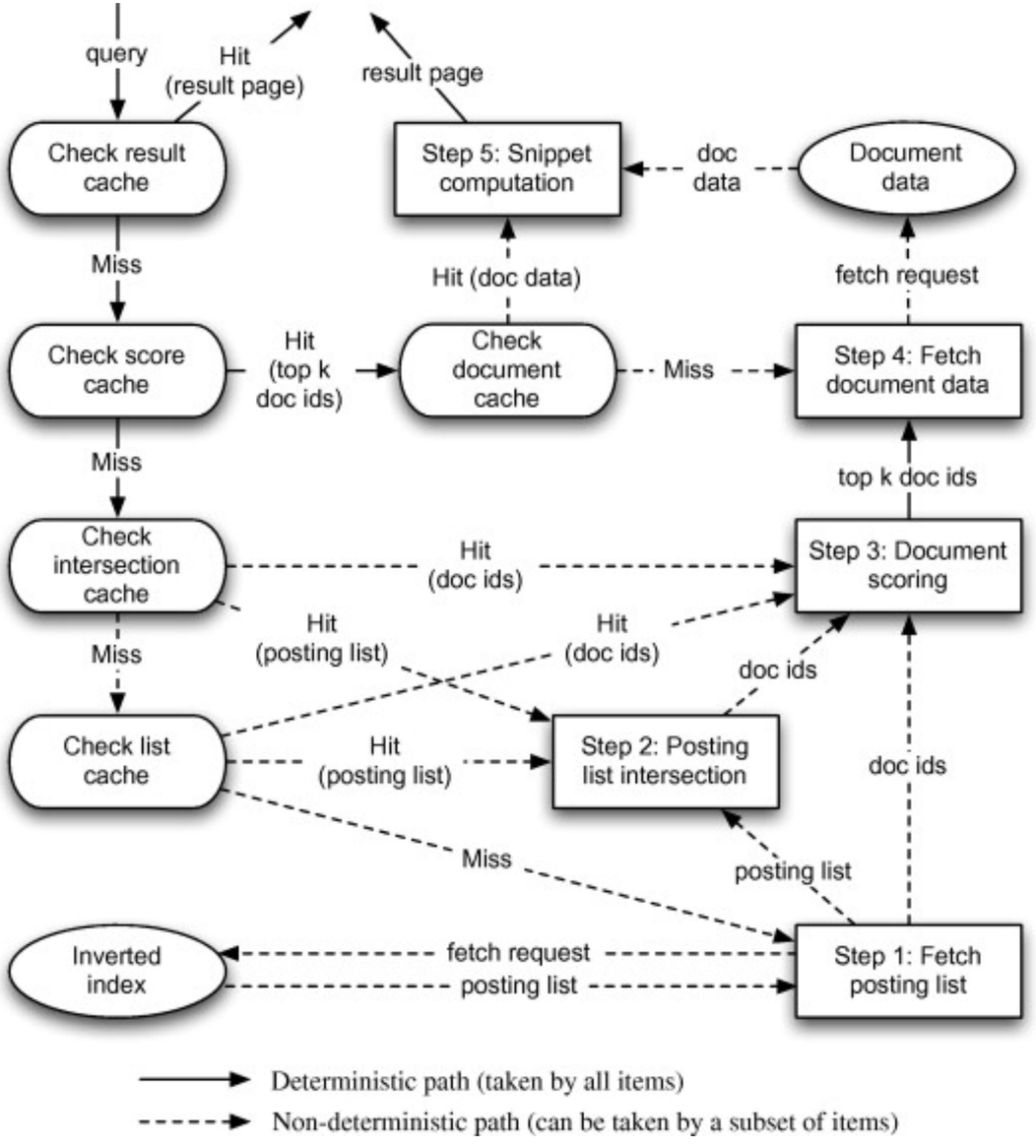
- Firstly studied in:
  - Saraiva, P. C., Silva de Moura, E., Ziviani, N., Meira, W., Fonseca, R., and Riberio-Neto, B. 2001. **Rank-preserving two-level caching for scalable search engines**. In Proceedings of ACM SIGIR '01. ACM, New York, NY, 51-58.
- Further analyzed in:
  - Baeza-Yates, R., Gionis, A., Junqueira, F. P., Murdock, V., Plachouras, V., and Silvestri, F. 2008. **Design trade-offs for search engine caching**. ACM Trans. Web 2, 4 (Oct. 2008), 1-28.

Three-level Caching

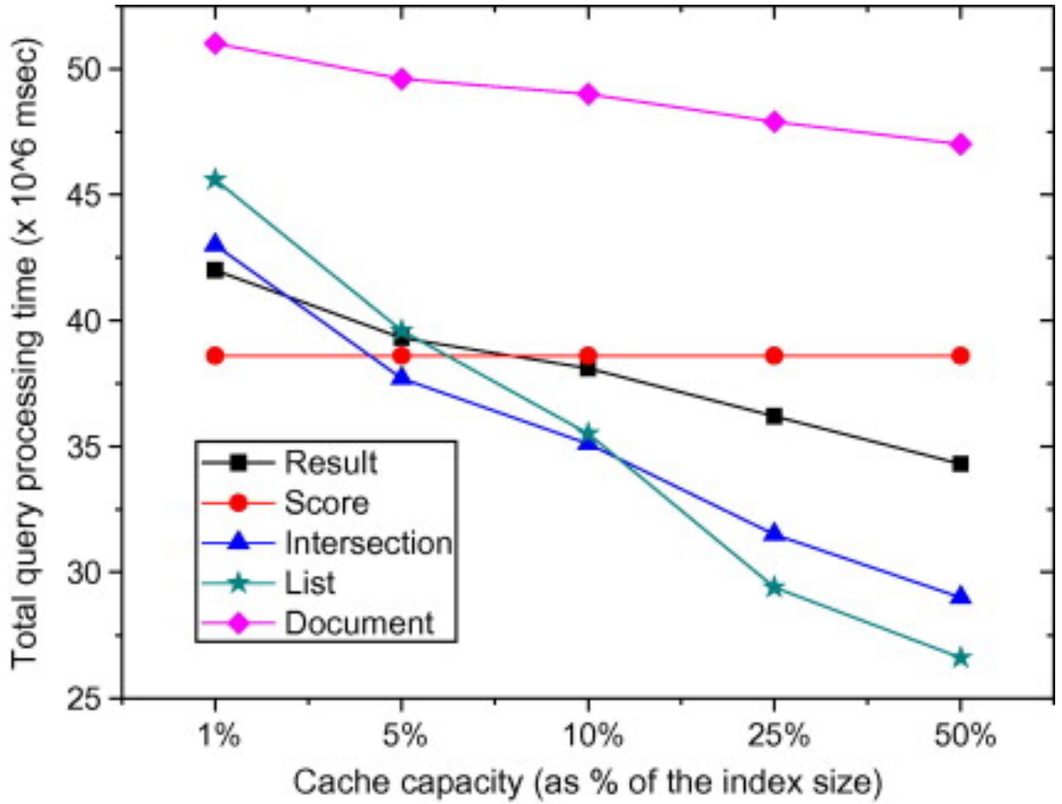
- Adds one level between results and posting lists cache.
- Usually stores frequently occurring pairs of terms.
  - Long, X. and Suel, T. 2005. **Three-level caching for efficient query processing in large Web search engines**. In Proceedings of the 14th international Conference WWW '05. 257-266.
  - Skobeltsyn, G., Junqueira, F., Plachouras, V., and Baeza-Yates, R. 2008. **ResIn: a combination of results caching and index pruning for high-performance web search engines**. In Proceedings of the 31st Annual international ACM SIGIR '08. 131-138.

Five-level Caching

- Adds on caching of documents and of top-k scores
- Nice experimental evaluation of relations between different cache levels
  - Rifat Ozcan, I. Sengor Altingovde, B. Barla Cambazoglu, Flavio P. Junqueira, Ozgur Ulusoy, **A five-level static cache architecture for web search engines**, Information Processing & Management, In Press, Available online 1 February 2011



Independent contributions of single caches



Contributions sum up in large search systems!

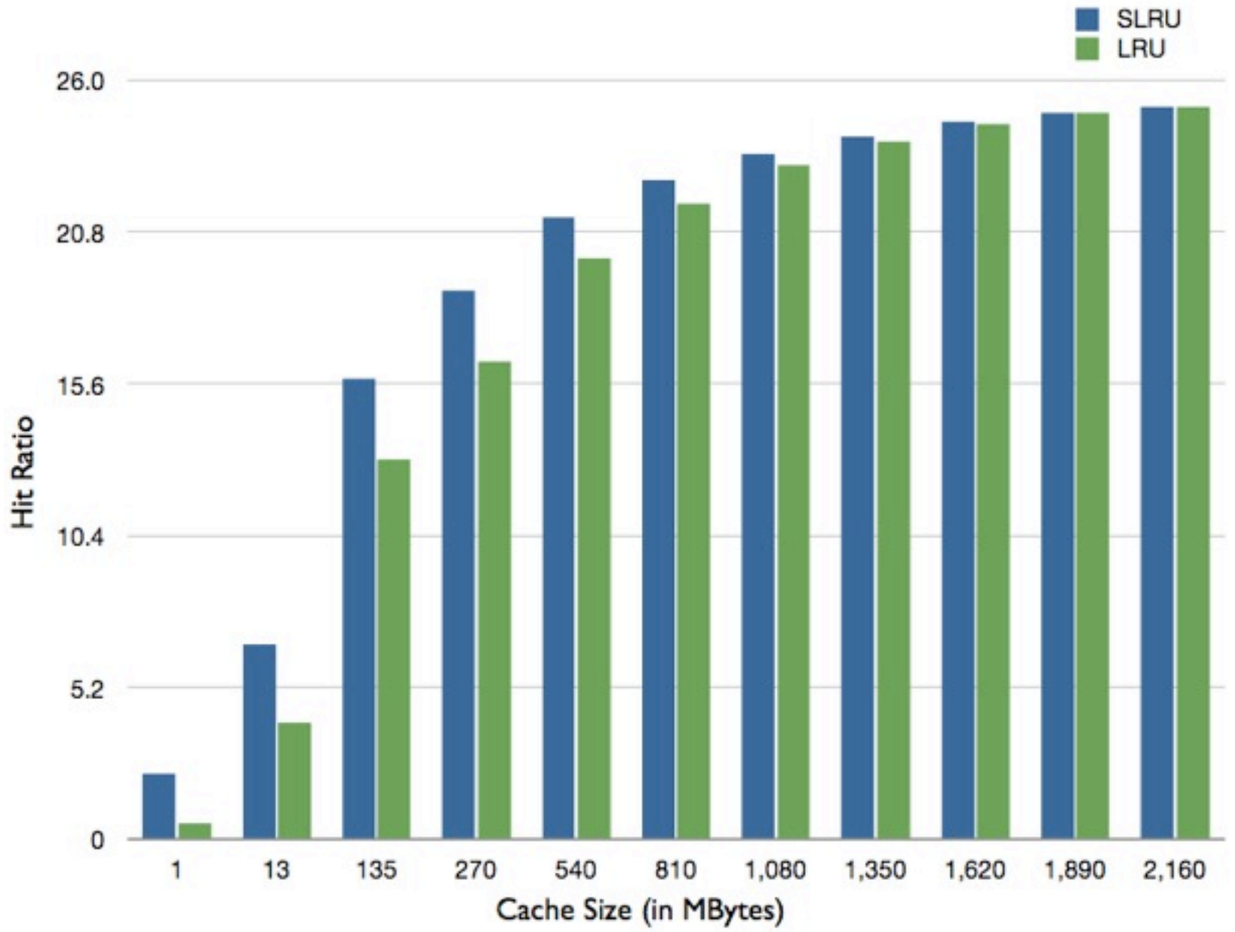
Rifat Ozcan, I. Sengor Altingovde, B. Barla Cambazoglu, Flavio P. Junqueira, Ozgur Ulusoy, **A five-level static cache architecture for web search engines**, Information Processing & Management, In Press, Available online 1 February 2011

Traditional Replacement Policies

- LRU
- LFU
- SLRU
- ...

Evangelos P. Markatos: **On caching search engine query results**. Computer Communications 24(2): 137-143 (2001)

SLRU vs. LRU on Excite

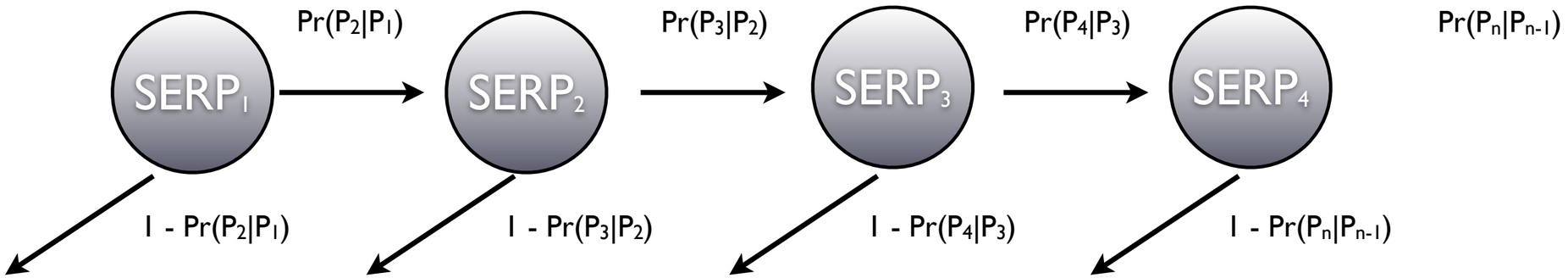


Evangelos P. Markatos: **On caching search engine query results**. Computer Communications 24(2): 137-143 (2001)

Search Engine Tailored Policies

- PDC
  - Probability Driven Caching
- SDC
  - Static Dynamic Caching
- AC
  - Admission Control

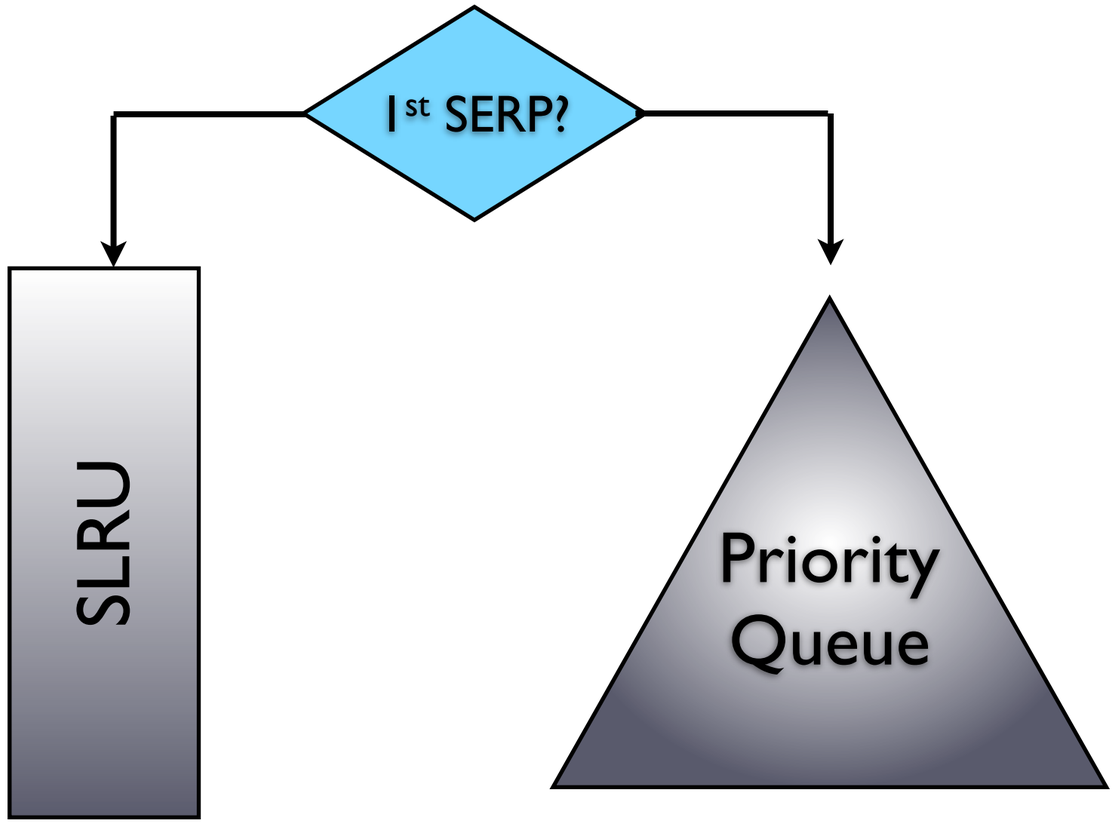
PDC



- IDEA: design a policy tailored over users' behavior on search pages
- With high probability users do not go beyond the first page of results
- For some query users browse many result pages.

Lempel, R. and Moran, S. 2003. **Predictive caching and prefetching of query results in search engines.** In Proceedings of WWW '03. 19-28.

PDC



PDC Priorities

- Priorities are assigned using an approximation of the Markovian SERP request model
  - Each SERP different from the first one has a priority computed on historical queries (query log)
  - PDC caches pages that has follow-up queries more likely to be submitted. Why?

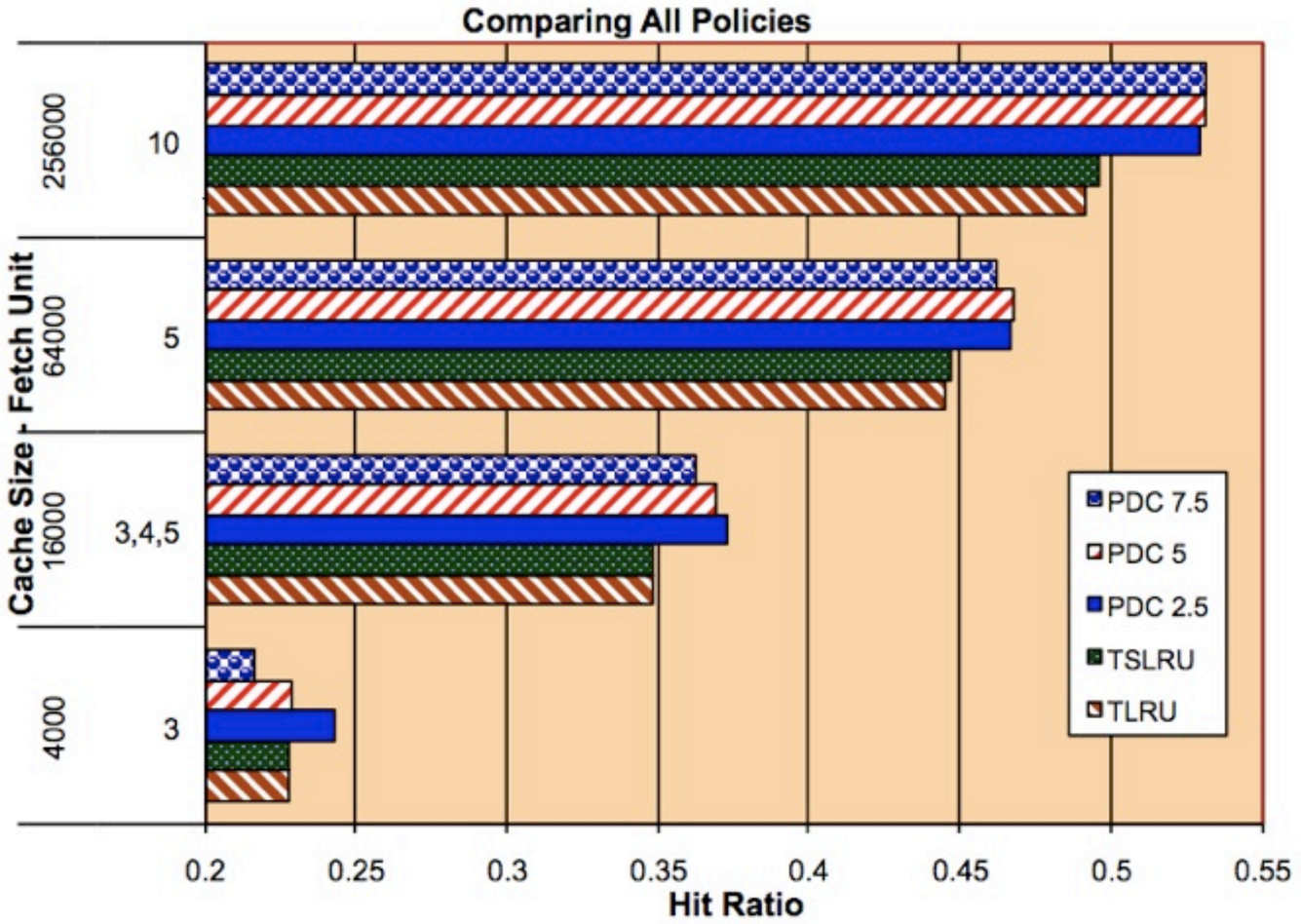
Lempel, R. and Moran, S. 2003. **Predictive caching and prefetching of query results in search engines.** In Proceedings of WWW '03. 19-28.

PDC and Prefetching

- in PDC results are organized according to "*Fetch Units*"
- When SERP i is requested for a query Q, the cache is first looked up
- If i is not cached, SERPs i, i + 1, ..., i + f are requested to the back-end
  - That is we prefetch f SERPs.
  - The fetch unit is of size f.

Lempel, R. and Moran, S. 2003. **Predictive caching and prefetching of query results in search engines.** In Proceedings of WWW '03. 19-28.

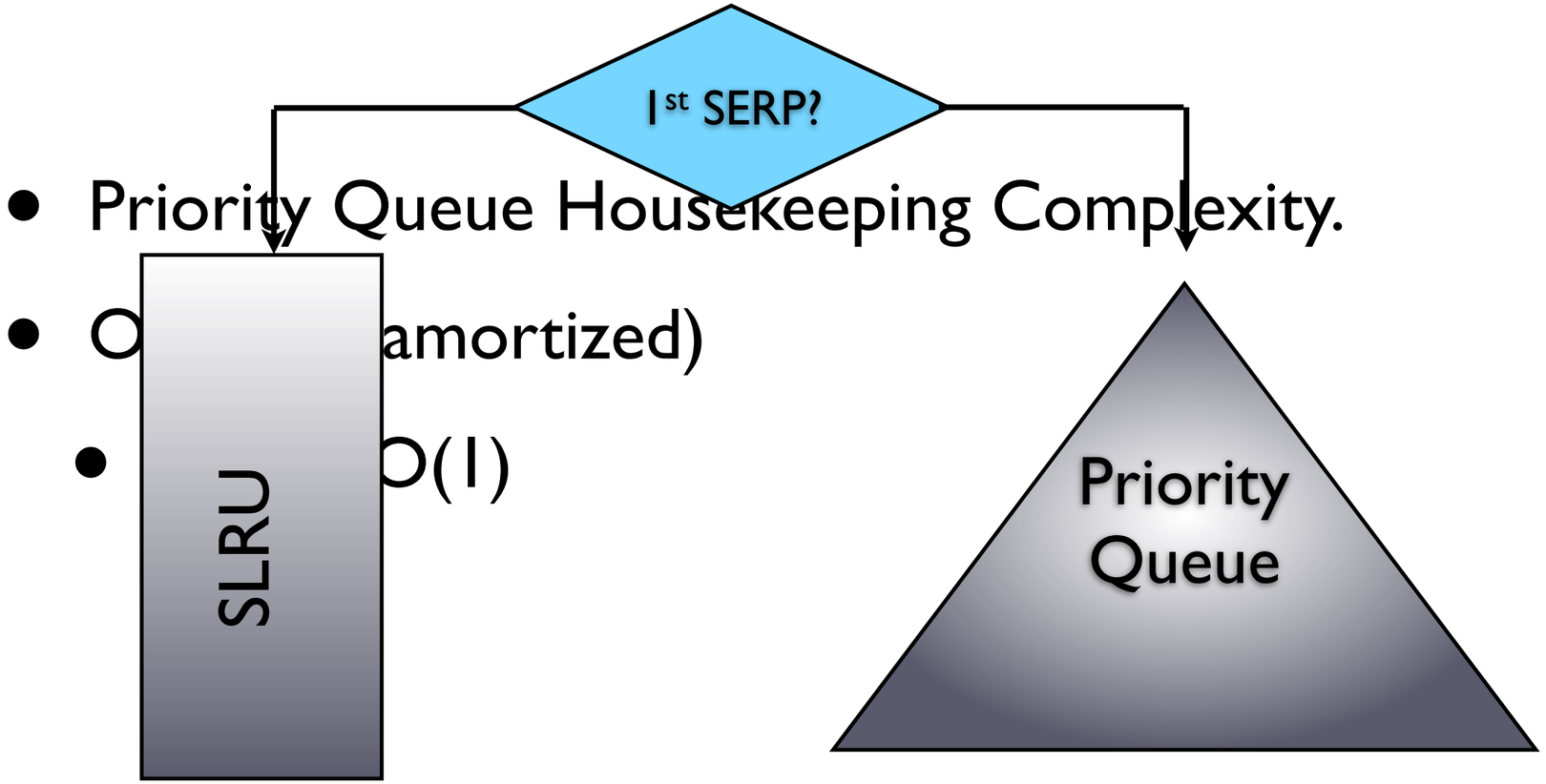
PDC Results



Lempel, R. and Moran, S. 2003. **Predictive caching and prefetching of query results in search engines.** In Proceedings of WWW '03. 19-28.

PDC's Main Drawback

- Priority Queue Housekeeping Complexity.
- O amortized)
  - O(1)



PDC's Main Lessons Learned

- Hit ratio benefits a lot from the use of historical data
- Prefetching helps a lot!
- Differently from previous caching policies, PDC not necessarily caches every submitted queries!!!

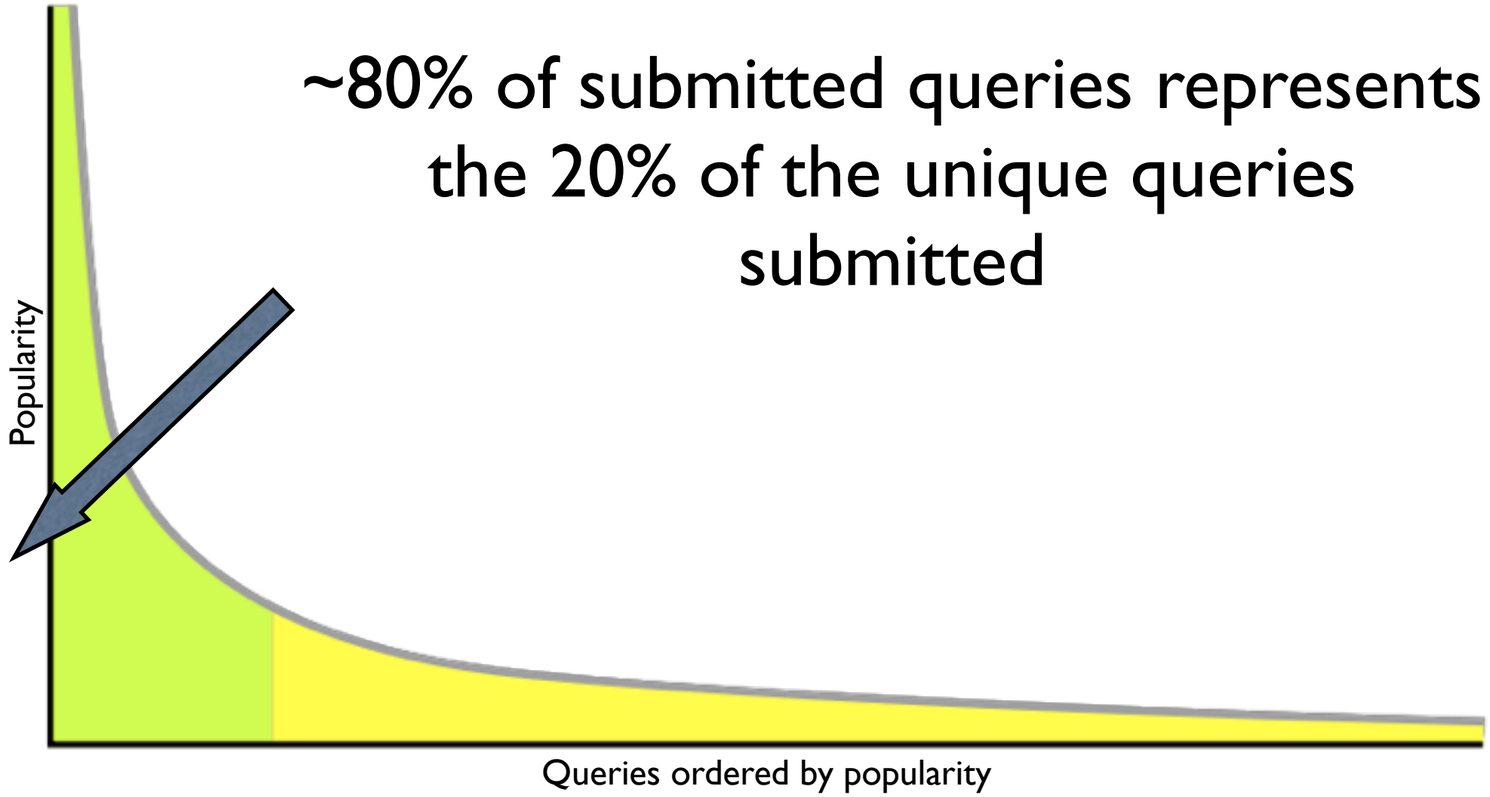
Lempel, R. and Moran, S. 2003. **Predictive caching and prefetching of query results in search engines.** In Proceedings of WWW '03. 19-28.

Overcoming PDC Complexity

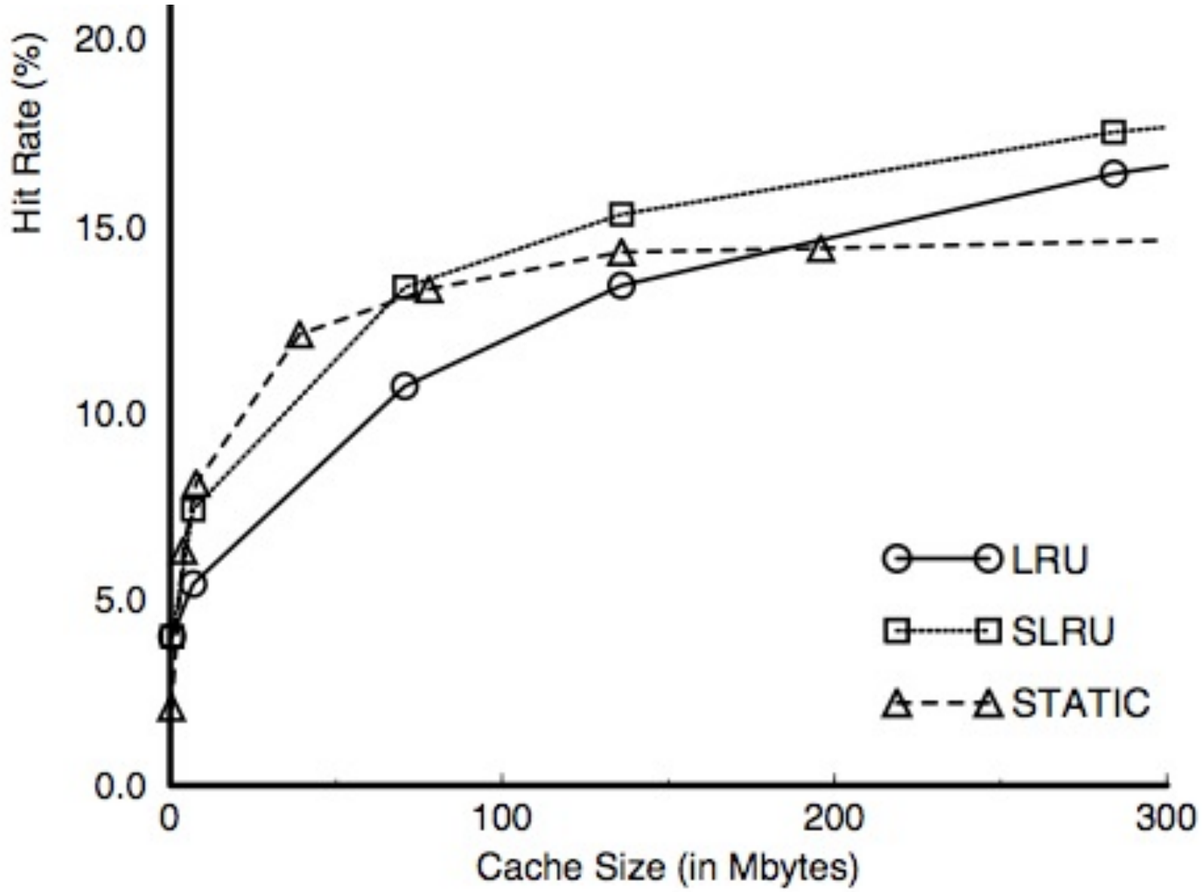
- PDC uses query logs to estimate the likelihood of follow-up queries.
- Why not using query logs to estimate likelihood of resubmitting a query?
- By catching the head of the long tail distribution we might obtain high hit ratios

That is...

~80% of submitted queries represents the 20% of the unique queries submitted



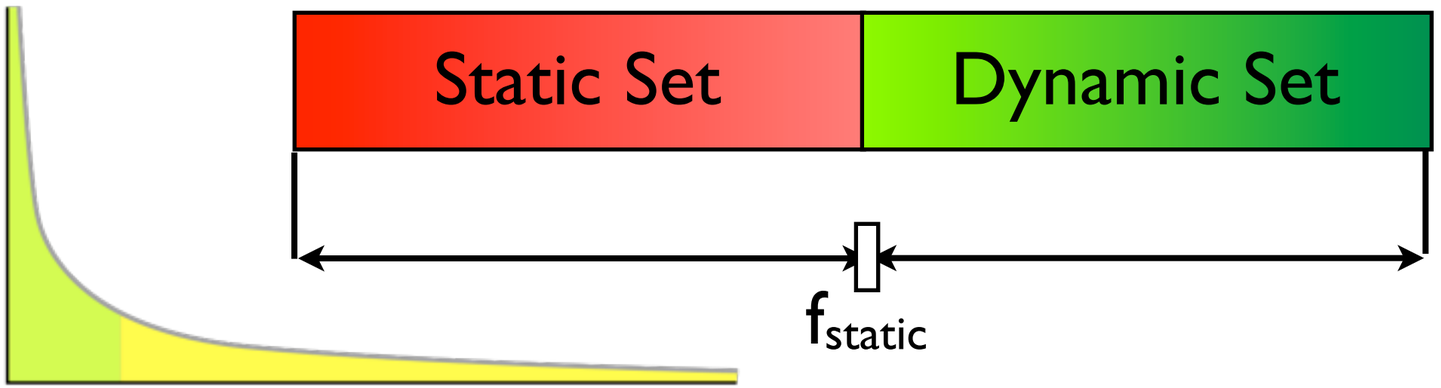
But...



Evangelos P. Markatos: **On caching search engine query results**. Computer Communications 24(2): 137-143 (2001)

Static Dynamic Caching

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

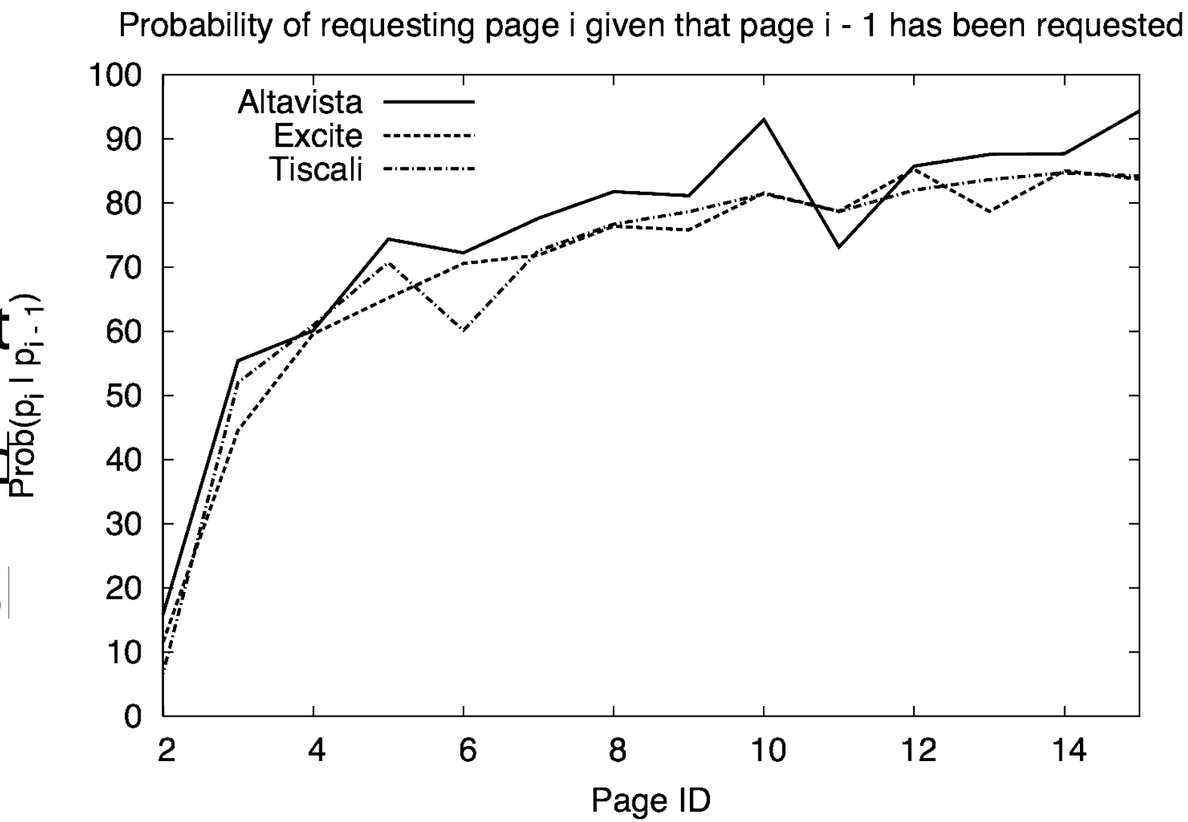


- LRU
- SLRU
- PDC
- ...

T. Fagni, R. Perego, F. Silvestri, and S. Orlando, "**Boosting the performance of web search engines: Caching and prefetching query results by exploiting historical usage data**," ACM Trans. Inf. Syst., vol. 24, no. 1, pp. 51–78, 2006.

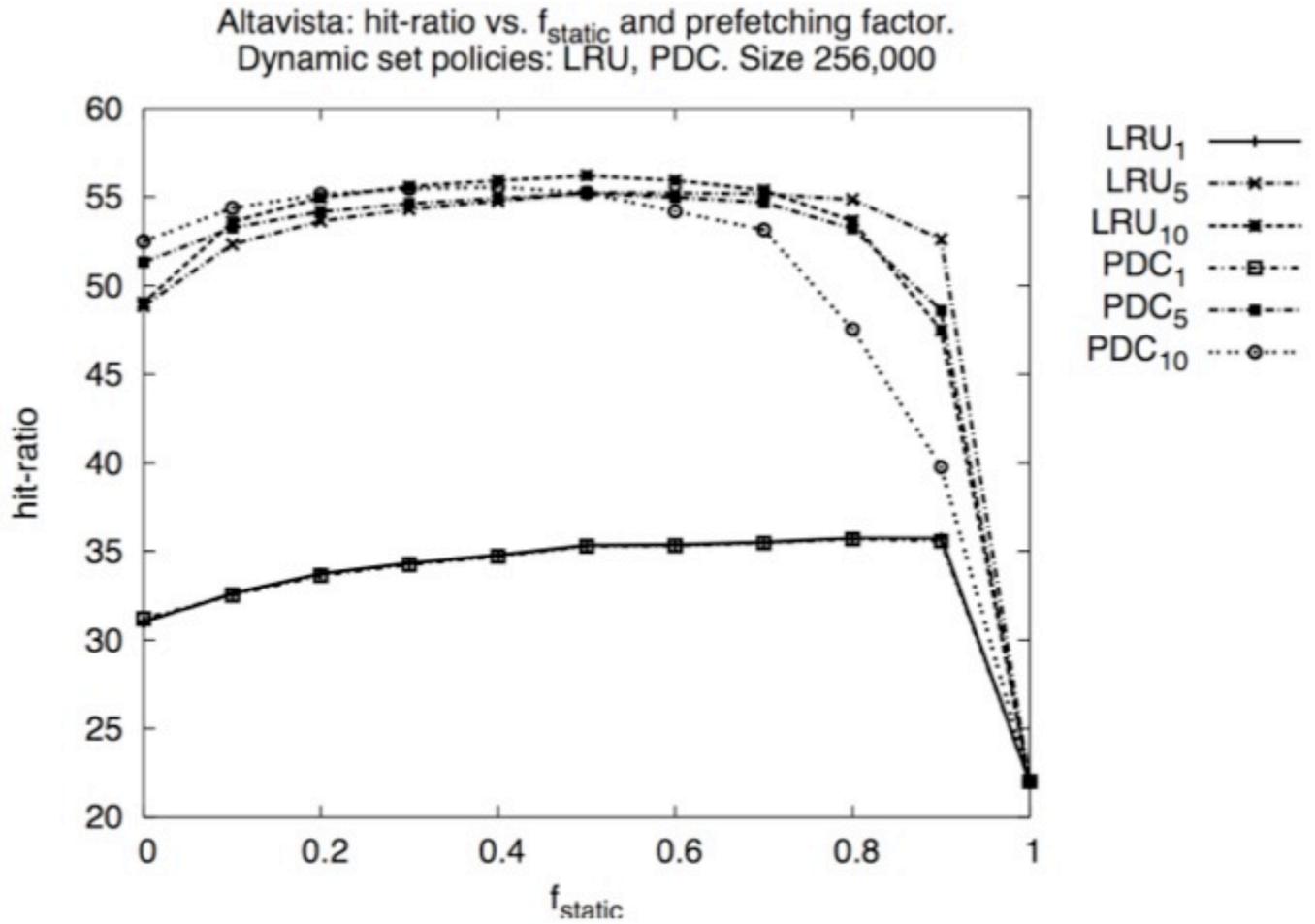
SDC and Prefetching

- SDC adopts an "adapt
  - For the first SERP c
  - For the follow-up S



T. Fagni, R. Perego, F. Silvestri, and S. Orlando, "**Boosting the performance of web search engines: Caching and prefetching query results by exploiting historical usage data**," ACM Trans. Inf. Syst., vol. 24, no. 1, pp. 51–78, 2006.

SDC Hit-Ratios



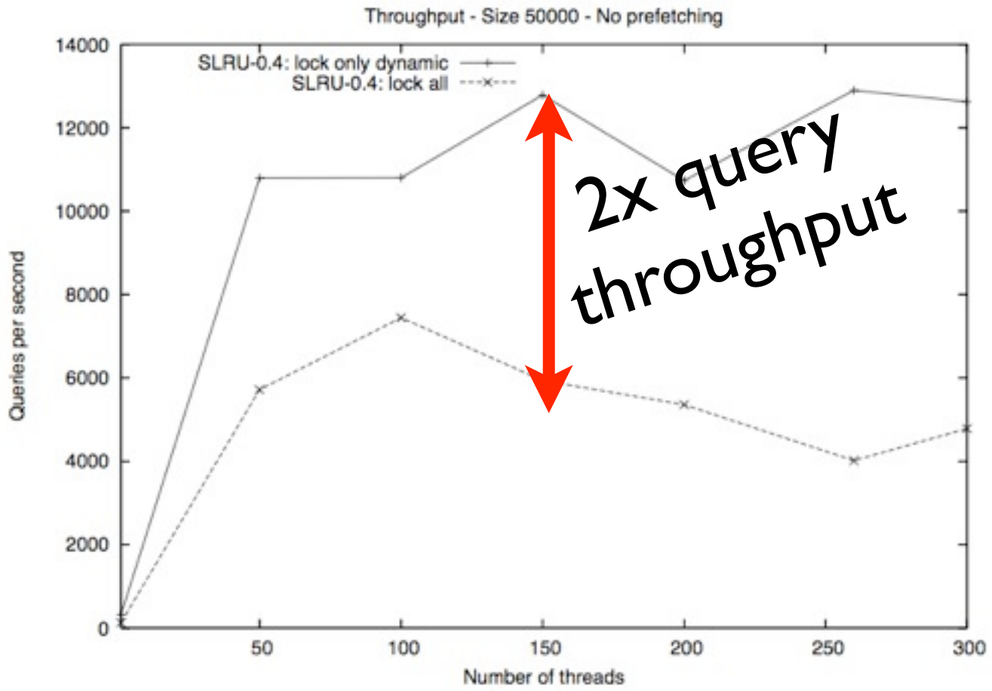
T. Fagni, R. Perego, F. Silvestri, and S. Orlando, "**Boosting the performance of web search engines: Caching and prefetching query results by exploiting historical usage data**," ACM Trans. Inf. Syst., vol. 24, no. 1, pp. 51–78, 2006.

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 combination of static and dynamic caching outperforms any dynamic-only or static-only solution almost independently from the caching policy managing the dynamic section of the cache
  - e.g., Rifat Ozcan, Ismail Sengor Altingovde, and Özgür Ulusoy. **Cost-Aware Strategies for Query Result Caching in Web Search Engines**. *ACM Trans. Web* 5, 2, Article 9 (May 2011)

That's *not* All Folks!

- The static cache alleviates multi-threading contention
- The dynamic cache supports read/write operations



T. Fagni, R. Perego, F. Silvestri, and S. Orlando, "**Boosting the performance of web search engines: Caching and prefetching query results by exploiting historical usage data**," ACM Trans. Inf. Syst., vol. 24, no. 1, pp. 51–78, 2006.

Admission Control

- An interesting idea of SDC: frequent queries are cached permanently
- AC by Baeza-Yates et al. generalizes the idea by using two dynamically updated sets:
  - A Controlled Cache (**CC**)
  - An Uncontrolled Cache (**UC**)
- When a new query arrives an admission policy is applied to steer a query to the CC or to the UC.
- If the query is likely to be seen in the future move it to CC, otherwise send it to UC.

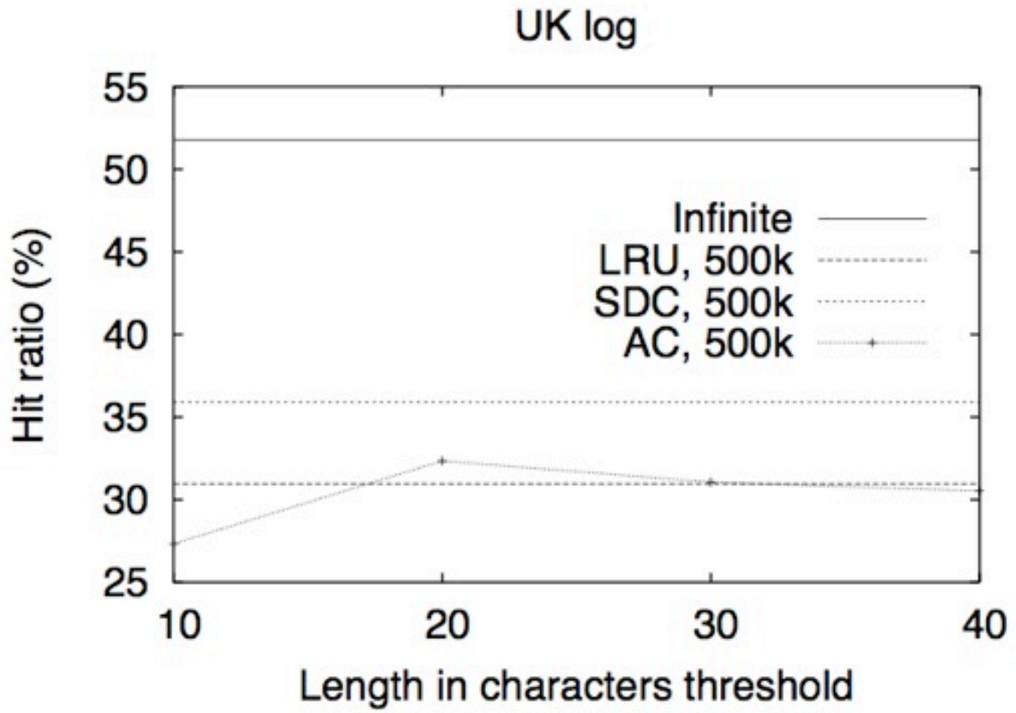
Ricardo Baeza-Yates, Flavio Junqueira, Vassilis Plachouras, and Hans Friedrich Witschel, "**Admission Policies for Caches of Search Engine Results**," SPIRE 2007, 74-85.

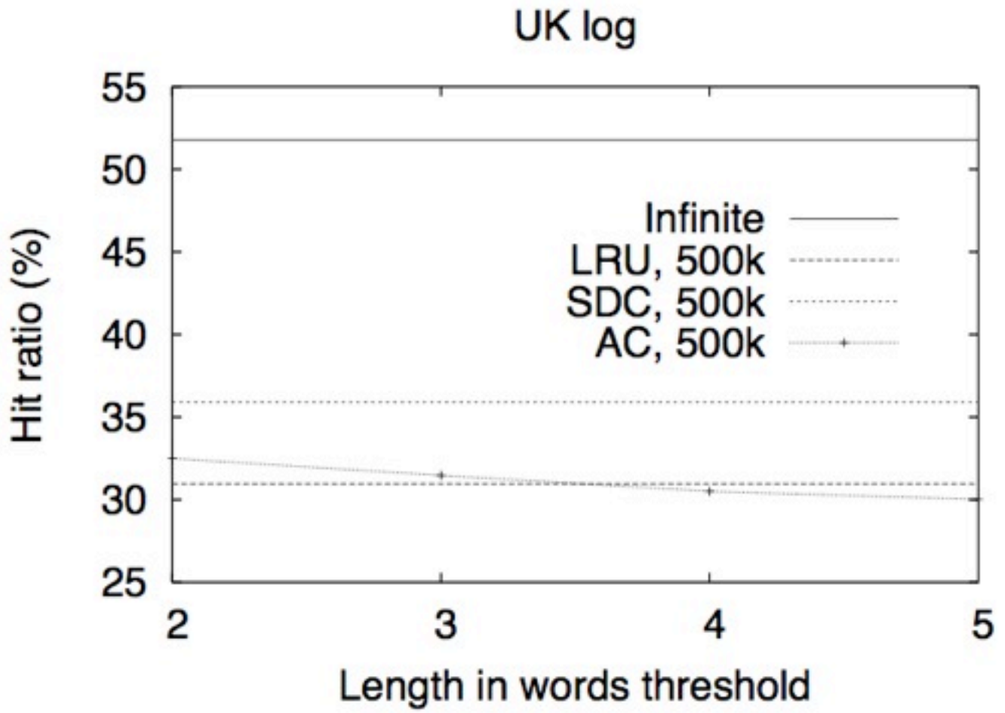
Admission Policy

- Makes use of features, e.g.:
  - Stateless features:
    - *LenC*: the length of the query in characters
    - *LenW*: the length of the query in words
  - Stateful features:
    - *PastF*: the frequency of the query in the (relatively recent) past
    - Uses more memory to enable admission control

Ricardo Baeza-Yates, Flavio Junqueira, Vassilis Plachouras, and Hans Friedrich Witschel, "**Admission Policies for Caches of Search Engine Results**," SPIRE 2007, 74-85.

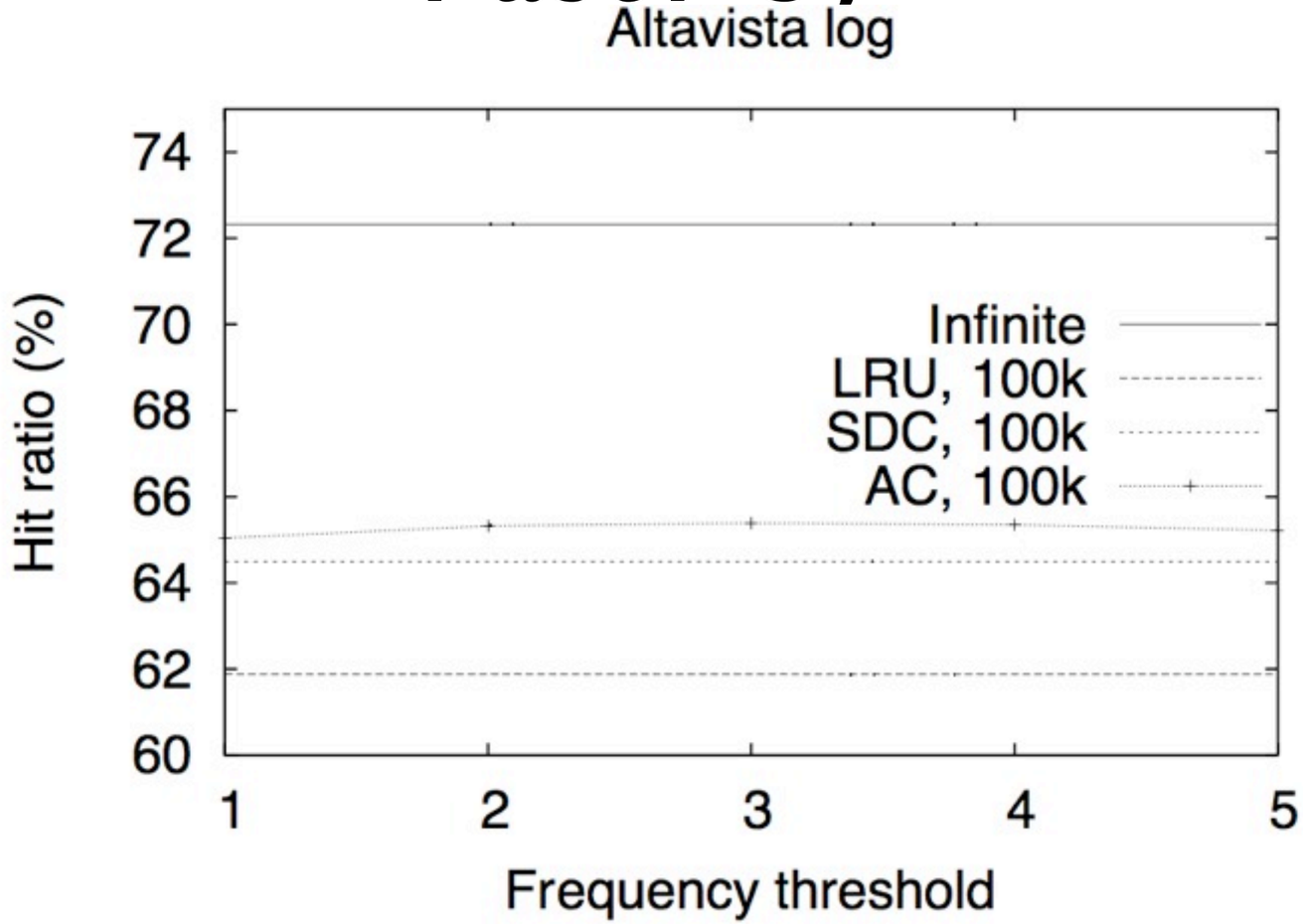
Hit-Ratio Results (stateless features LenC- LenW)





Ricardo Baeza-Yates, Flavio Junqueira, Vassilis Plachouras, and Hans Friedrich Witschel, "**Admission Policies for Caches of Search Engine Results**," SPIRE 2007, 74-85.

Hit-Ratio Results (stateful features Past1-5)



Ricardo Baeza-Yates, Flavio Junqueira, Vassilis Plachouras, and Hans Friedrich Witschel, "**Admission Policies for Caches of Search Engine Results**," SPIRE 2007, 74-85.

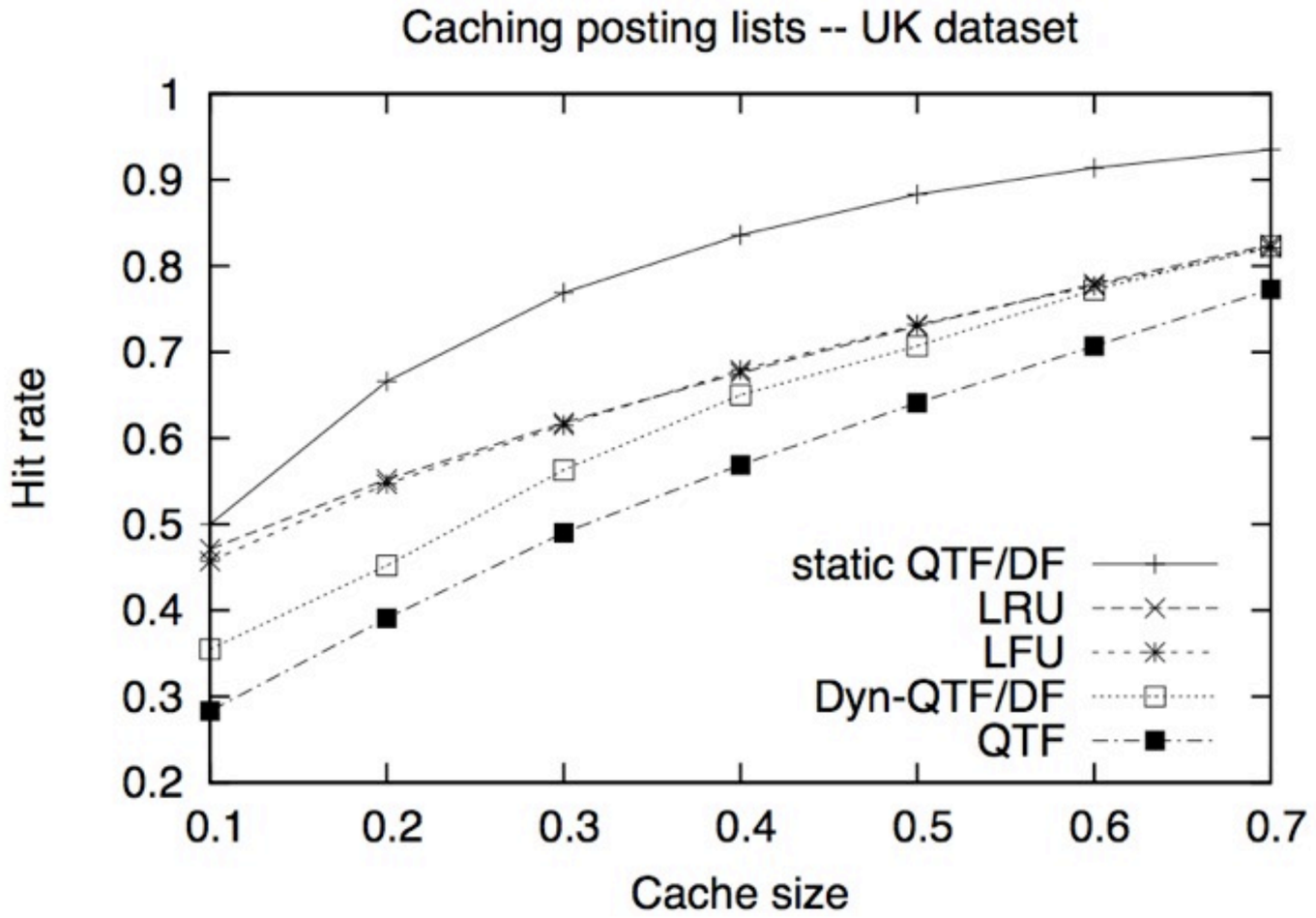
Caching Posting Lists

- SERP size is fixed
- Posting lists have different lengths.
- Posting list caching techniques adopt policies sensitive to list sizes

$Q_{TF}D_F$ Policy

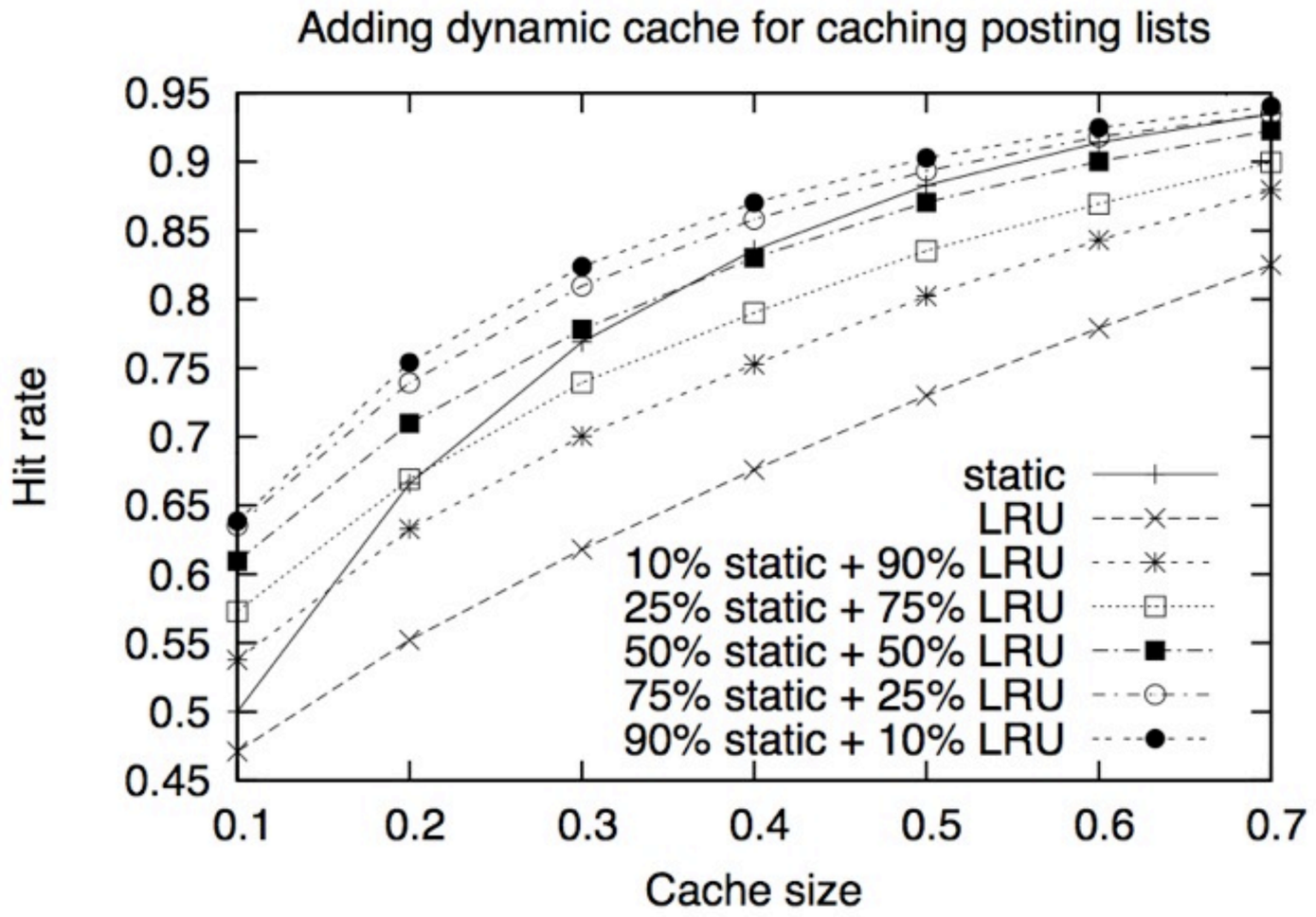
- Idea (derived from several web caching solutions):
  - suppose you have 10 free slots and 3 postings lists to cache l_1, l_2, and l_3. l_1 appears 10 times and it is long 6 postings, l_2 and l_3 appear 6 times each and are long 5 postings.
- Traditional frequency-only-based policies will choose to cache l_1 filling up 6 slots and not leaving space for any of the two other lists.
- $Q_{TF}D_F$ decides to cache l_2 and l_3 since they optimize the ratio frequency/size instead of just frequency
- Results:
  - Traditional static caching results in 10 hits
  - $Q_{TF}D_F$ static policy has 12 hits

$QTFDF$ Results



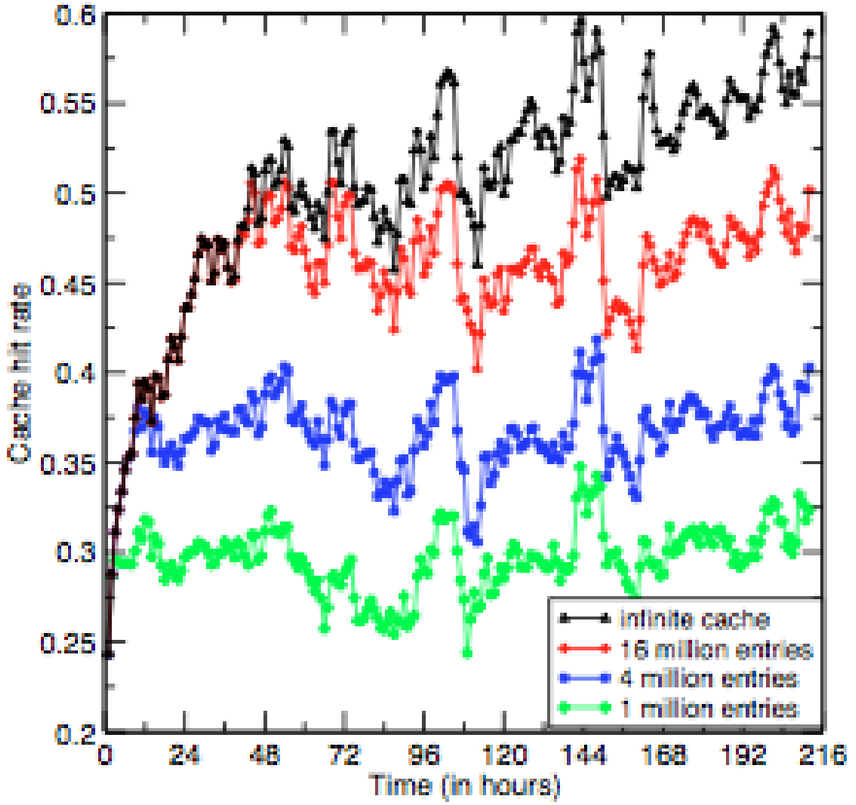
Baeza-Yates, R., Gionis, A., Junqueira, F. P., Murdock, V., Plachouras, V., and Silvestri, F. 2008. **Design trade-offs for search engine caching**. ACM Trans. Web 2, 4 (Oct. 2008), 1-28.

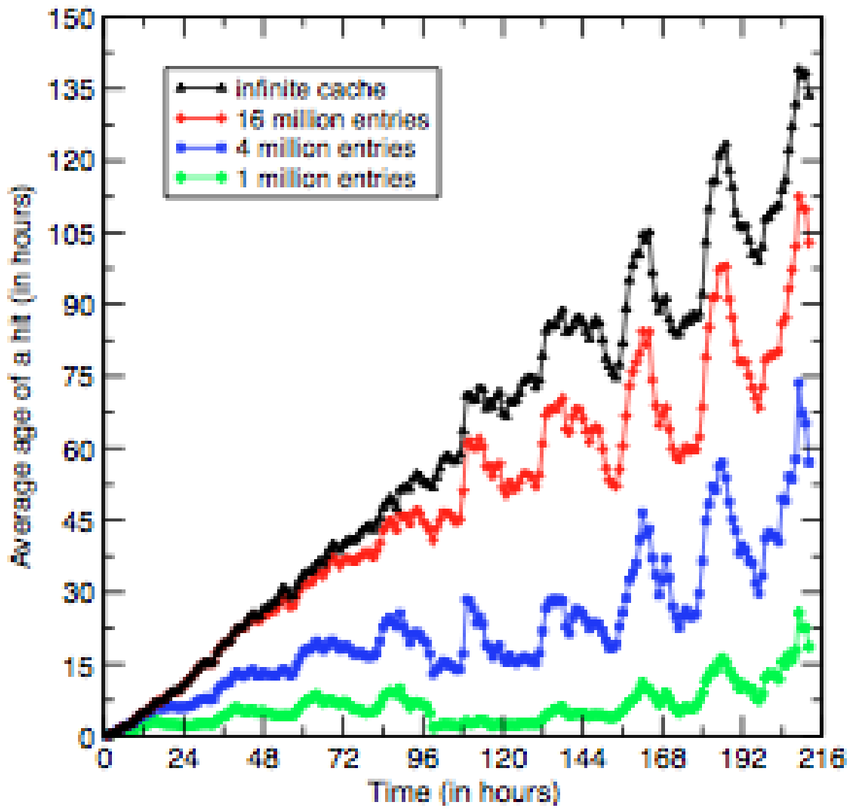
SDC-like $Q_{TF}D_F$



Baeza-Yates, R., Gionis, A., Junqueira, F. P., Murdock, V., Plachouras, V., and Silvestri, F. 2008. **Design trade-offs for search engine caching**. ACM Trans. Web 2, 4 (Oct. 2008), 1-28.

Freshness of cached entries



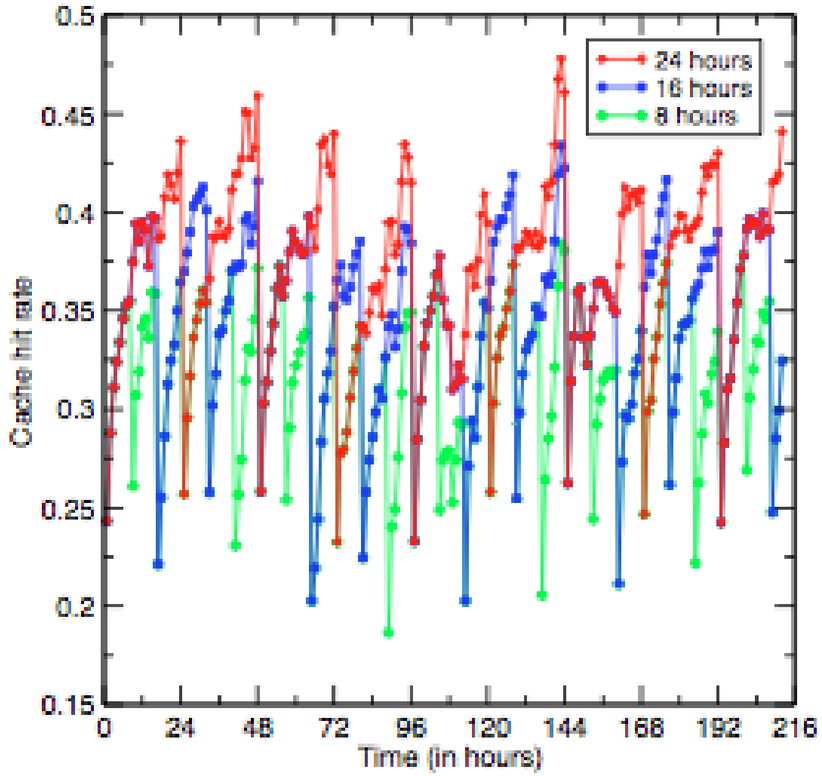


After nine days, the average age of a hit on the infinite cache becomes about 5.6 days

Berkant Barla Cambazoglu, Flavio P. Junqueira, Vassilis Plachouras, Scott Banachowski, Baoqiu Cui, Swee Lim, and Bill Bridge. 2010. **A refreshing perspective of search engine caching**, WWW '10.

Freshness of cached entries

- Flush the content of the cache and re-warm it from scratch
  - effective but expensive....
  - a fixed TTL improves the curves but...



Berkant Barla Cambazoglu, Flavio P. Junqueira, Vassilis Plachouras, Scott Banachowski, Baoqiu Cui, Swee Lim, and Bill Bridge. 2010. **A refreshing perspective of search engine caching**, WWW '10.

Freshness of cached entries

- IDEA: use idle cycles of the back-end to refresh expired cache entries
  - Several possible criteria for selecting entries: frequency, recency, processing cost, probability of a change in the cached results
  - Two (bucketed) parameters considered
    - *Age*, modeling the time effect
    - *Temperature*, lazy adjusted according to the frequency of occurrence
  - Oldest and Hottest entries refreshed earlier

Berkant Barla Cambazoglu, Flavio P. Junqueira, Vassilis Plachouras, Scott Banachowski, Baoqiu Cui, Swee Lim, and Bill Bridge. 2010. **A refreshing perspective of search engine caching**, WWW '10.

Freshness of cached entries

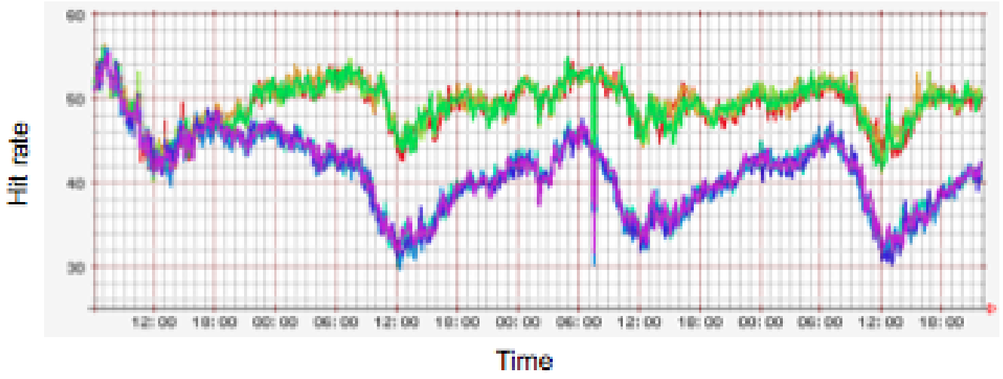


Figure 17: Hit rate in production (%).

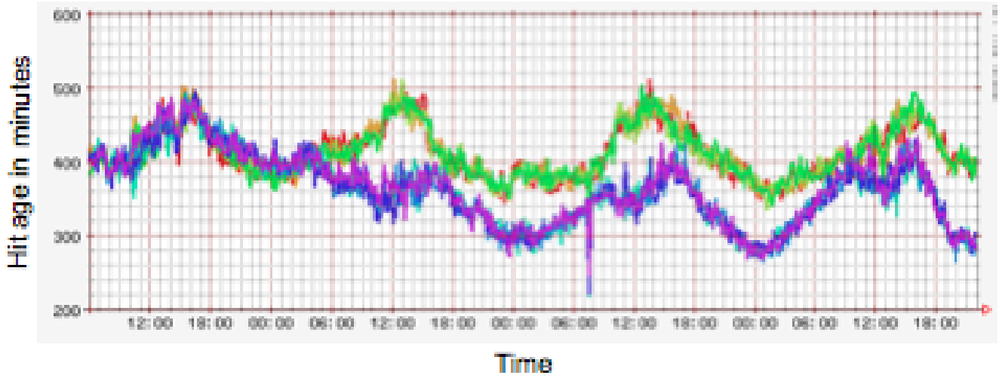


Figure 18: Hit age in production.

- Hit rate with refreshing is about 7-10% lower than with no refreshing

Berkant Barla Cambazoglu, Flavio P. Junqueira, Vassilis Plachouras, Scott Banachowski, Baoqiu Cui, Swee Lim, and Bill Bridge. 2010. **A refreshing perspective of search engine caching**, WWW '10.

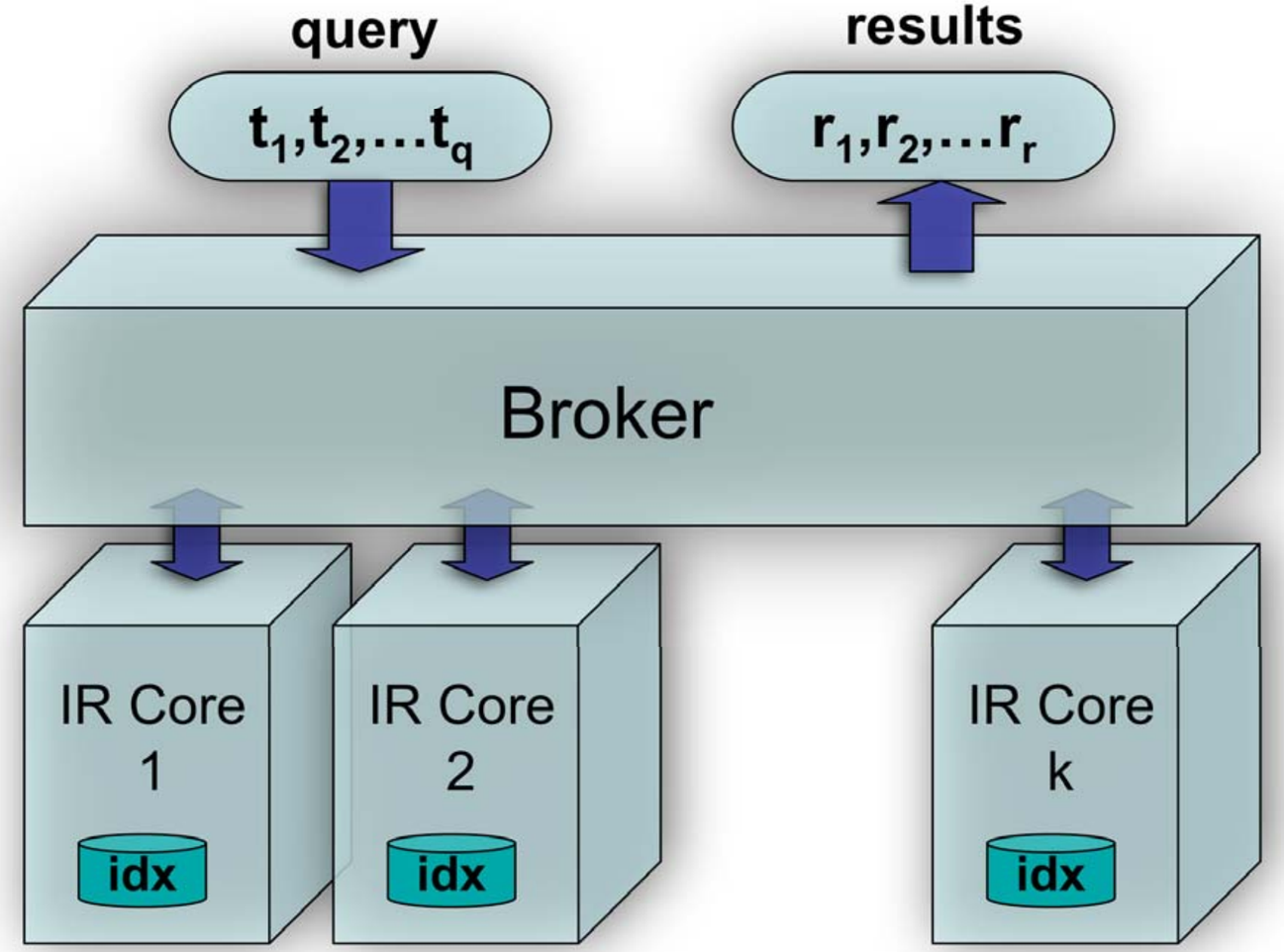
Caching freshness issues

- See also:
  - Blanco, R., Bortnikov, E., Junqueira, F., Lempel, R., Telloli, L., Zaragoza, H.: **Caching Search Engine Results over Incremental Indices**. In: SIGIR 2010, pp. 82–89
  - Edward Bortnikov, Ronny Lempel and Kolman Vornovitsky: **Caching for Realtime Search**. in: ECIR 2011, LNCS 6611

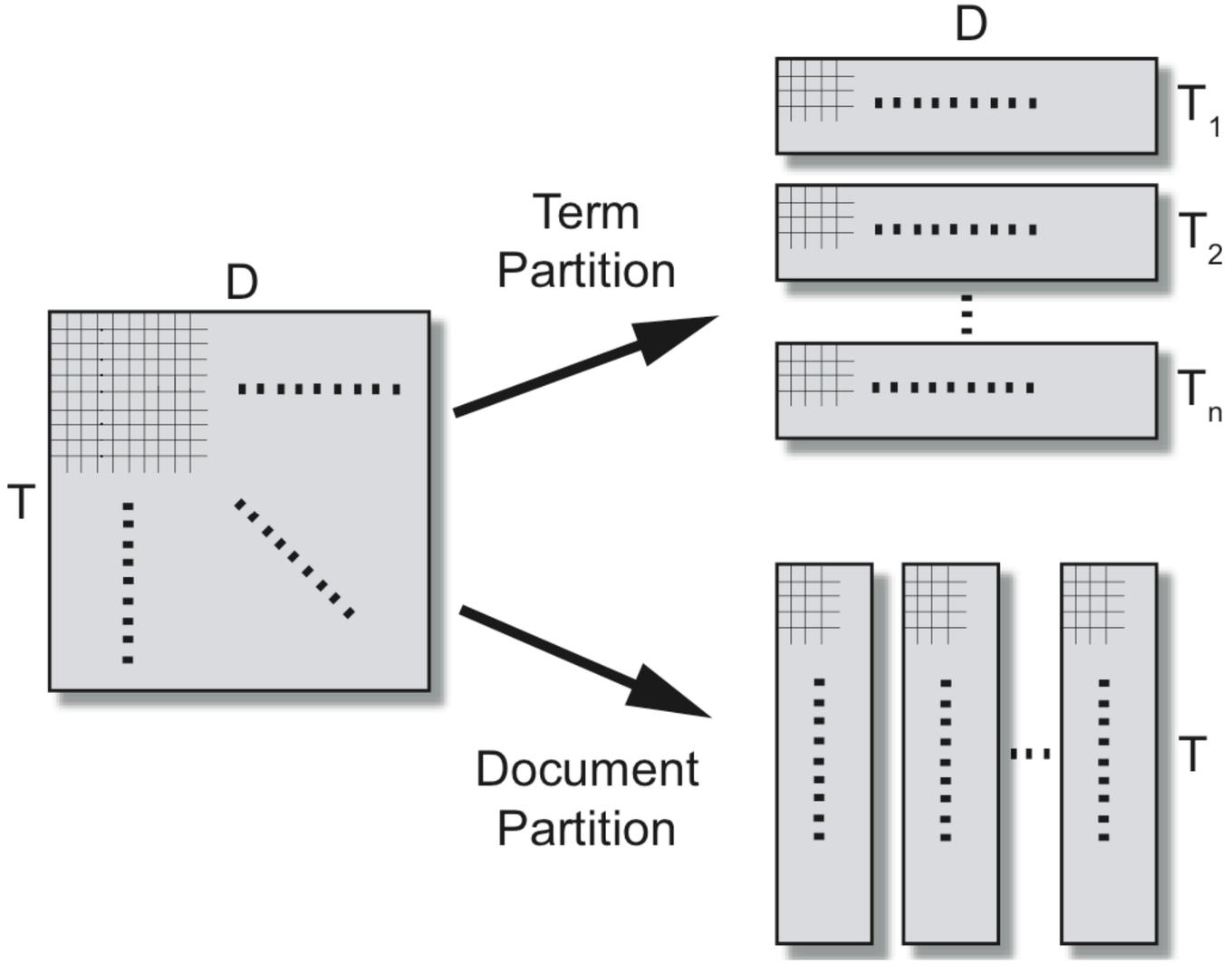
Not Only Caching

- Efficiency improvement by using query logs can be pursued by means of:
  - query routing techniques
  - data/index partitioning strategies

Sketching a Distributed Search Engine



Index Partitioning

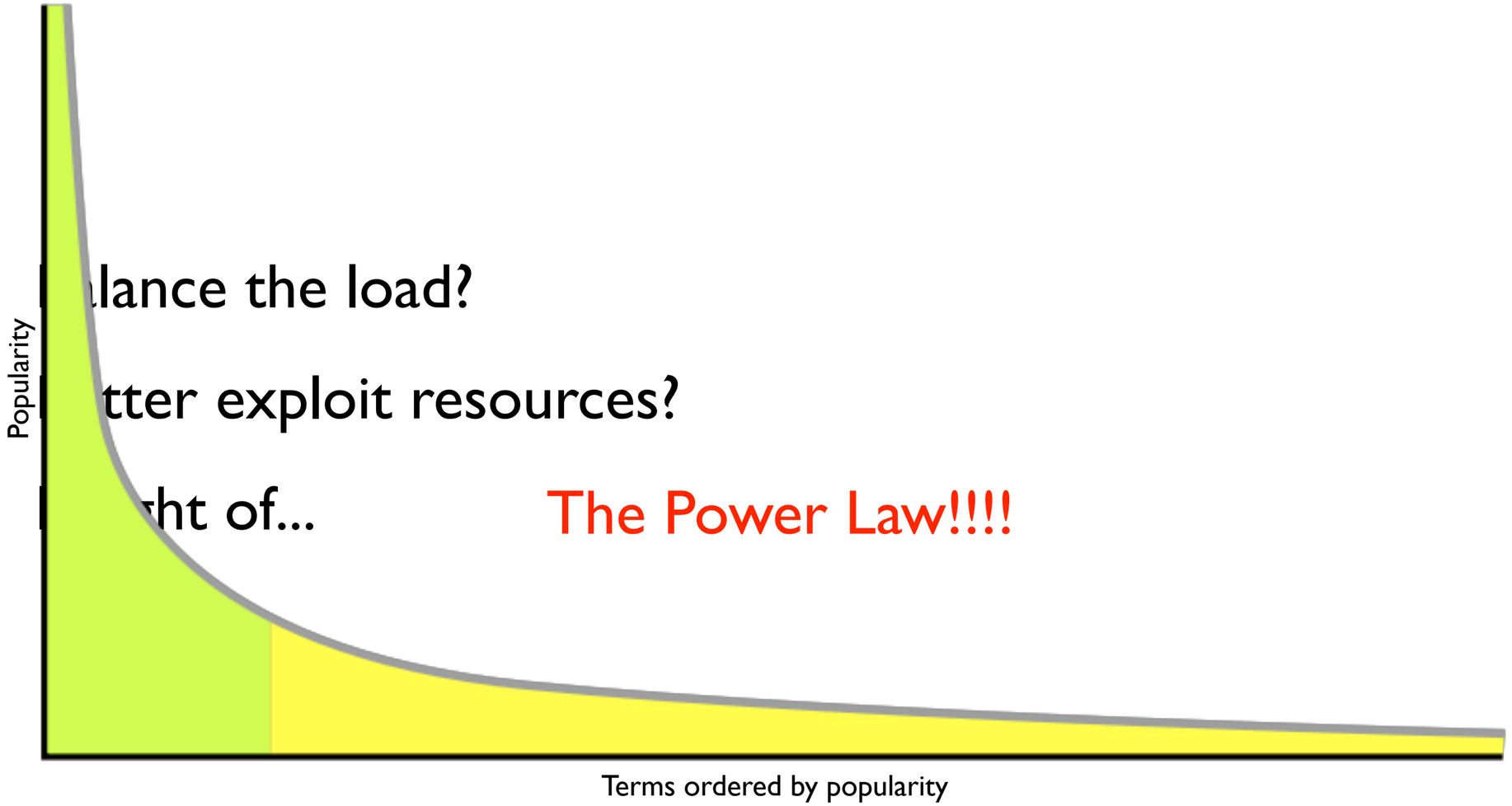


Term Partitioning Systems

- Query routing is "trivial". Whenever $Q=(t_1,t_2, ..., t_n)$ is received route it to the servers containing those terms.
  - less IR Cores queried and less operation performed
  - More available capacity!
- But..
  - *not scalable* (indexing is **n log n**, term partitioning needs reindexing the entire collection from scratch)
  - *load imbalance* among IR cores

How can we...

- [B]alance the load?
- [B]etter exploit resources?
- [K]nowledge of...



The Power Law!!!!

Approaches in Literature

- Moffat, A., Webber, W., and Zobel, J. 2006. **Load balancing for term-distributed parallel retrieval**. In Proceedings of SIGIR 2006. Seattle, Washington, USA, August 06 - 11, 2006.
- Alistair Moffat, William Webber, Justin Zobel, Ricardo A. Baeza-Yates: A **pipelined architecture for distributed text query evaluation**. Inf. Retr. 10(3): 205-231 (2007)
- Lucchese, C., Orlando, S., Perego, R., and Silvestri, F. 2007. **Mining query logs to optimize index partitioning in parallel web search engines**. In Proceedings of Infoscale 2007. Suzhou, China, June 06 - 08, 2007.

The Idea...

- If terms co-occur frequently in past queries...
  - pack them together up in the same IR Core.
- but...
  - Power law prevents load balancing...
    - Knapsack problem can help in balancing the load.
      - Fit in partitions terms with weight $L_t=Q_t x B_t$ (Q_t occurrences of t in the query log, B_t length of t's postings list)

Load Balancing Results

- On .GOV2 collection. Queries adapted from WT10G.

Using frequency in Docs.

Strategy	Batch					Avg
	2	3	4	5	6	
Random	1.45	1.44	1.46	1.50	1.48	1.47
Using f_t	1.43	1.20	1.23	1.40	1.42	1.34
Past L_t	1.14	1.26	1.23	1.19	1.17	1.20
Current L_t	1.00	1.00	1.00	1.00	1.00	1.00

Moffat, A., Webber, W., and Zobel, J. 2006. **Load balancing for term-distributed parallel retrieval.** In Proceedings of SIGIR 2006. Seattle, Washington, USA, August 06 - 11, 2006.

Load Balancing + Replication Results

- On .GOV2 collection. Queries adapted from WT10G.

Strategy	Batch					Avg
	2	3	4	5	6	
Duplicate 1	1.26	1.20	1.10	1.17	1.11	1.17
Duplicate 10	1.06	1.29	1.17	1.18	1.16	1.17
Duplicate 100	1.09	1.14	1.10	1.13	1.15	1.12
Duplicate 1000	1.08	1.09	1.07	1.19	1.09	1.10
Multi-replicate	1.05	1.12	1.09	1.16	1.12	1.11

Moffat, A., Webber, W., and Zobel, J. 2006. **Load balancing for term-distributed parallel retrieval.** In Proceedings of SIGIR 2006. Seattle, Washington, USA, August 06 - 11, 2006.

But...

- Query Throughput.
- On .GOV2 collection. Queries from a real life MSN query log

Strategy	Batch					Avg
	2	3	4	5	6	
Hashed	1.82	1.82	1.86	1.84	1.83	1.83
Duplicate 100	2.21	2.14	2.25	2.17	2.20	2.19
Doc-distributed	2.21	2.25	2.24	2.31	2.27	2.26

Moffat, A., Webber, W., and Zobel, J. 2006. **Load balancing for term-distributed parallel retrieval.** In Proceedings of SIGIR 2006. Seattle, Washington, USA, August 06 - 11, 2006.

Adding a Dimension

- Number of IR Cores used by each query
- Basically, instead of sending lists around try to keep query processing local to a node.
  - More load unbalance
- Trade-off: The Term-Assignment Problem

Lucchese, C., Orlando, S., Perego, R., and Silvestri, F. 2007. **Mining query logs to optimize index partitioning in parallel web search engines**. In Proceedings of Infoscale 2007. Suzhou, China, June 06 - 08, 2007.

The Term-Assignment Problem

The T

a quer

the pa

where

1,

ng

Use a Frequent Itemsets Mining Algorithm to find pairs of co-occurring terms. Then, optimize by considering not only single terms but also term-sets

Lucchese, C., Orlando, S., Perego, R., and Silvestri, F. 2007. **Mining query logs to optimize index partitioning in parallel web search engines**. In Proceedings of Infoscale 2007. Suzhou, China, June 06 - 08, 2007.

Number of Queried IR Cores (Servers)

Servers	Baseline Cases		Term Assignment
	random	bin packing	$\alpha = 0.9$
$\Phi_{test} = TodoBR$			
1	28	28	50
2	31	30	20
3	17	17	14
> 3	24	25	16
$\Phi_{test} = AltaVista$			
1	29	29	41
2	39	39	38
3	21	21	16
> 3	11	11	5

Percentage of queries served by x servers

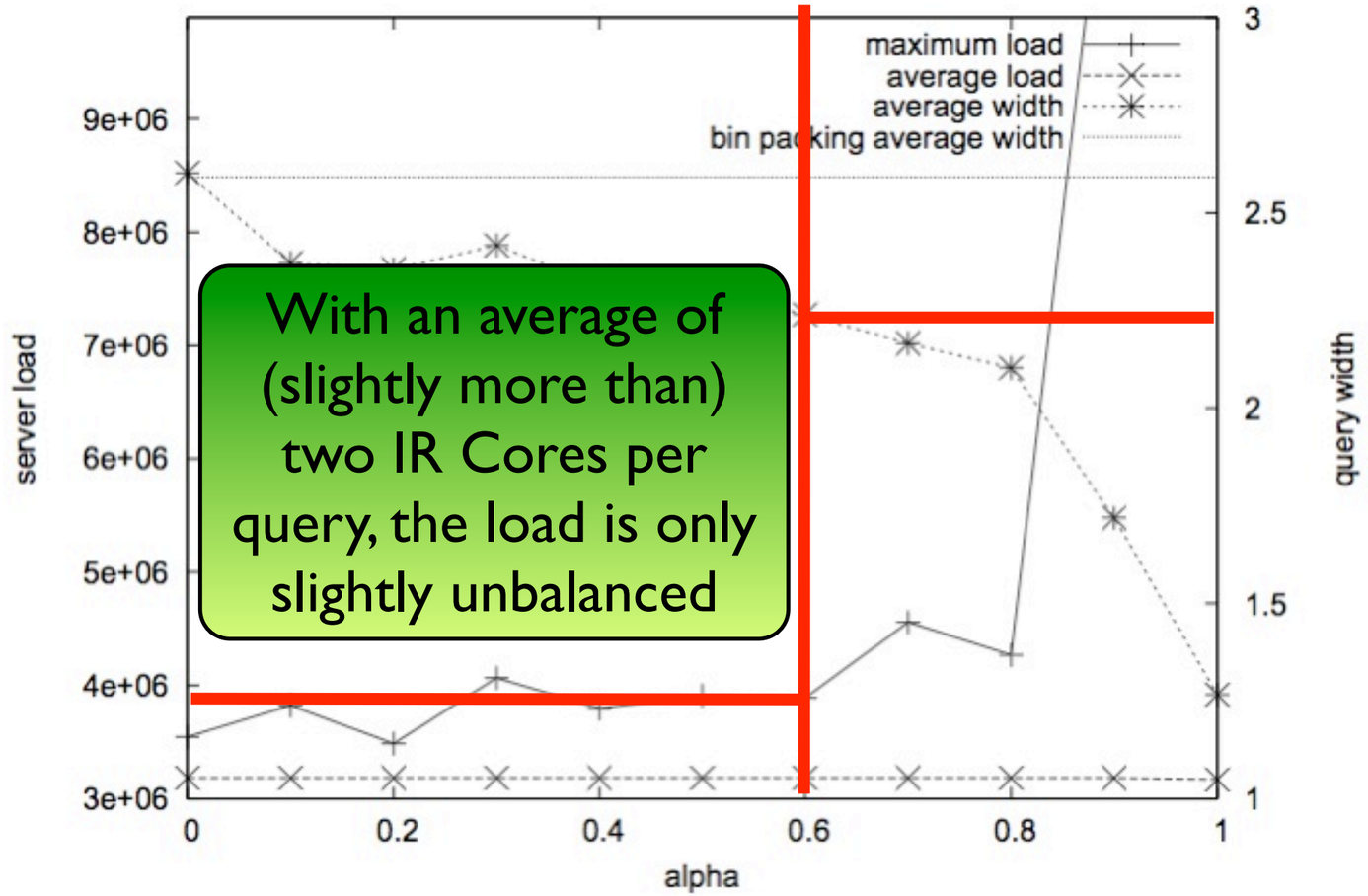
Lucchese, C., Orlando, S., Perego, R., and Silvestri, F. 2007. **Mining query logs to optimize index partitioning in parallel web search engines**. In Proceedings of Infoscale 2007. Suzhou, China, June 06 - 08, 2007.

Impact of Replication on the # of IR Cores

Servers	Replication Factors 0.0001 *bin pack.*	0.0001 *term ass.*	0.0005 *bin pack.*	0.0005 *term ass.*	0.001 *bin pack.*	0.001 *term ass.*
TodoBR						
1	42	54	56	62	63	67
2	31	22	22	18	19	16
3	12	10	9	8	8	8
> 3	15	14	12	11	10	9

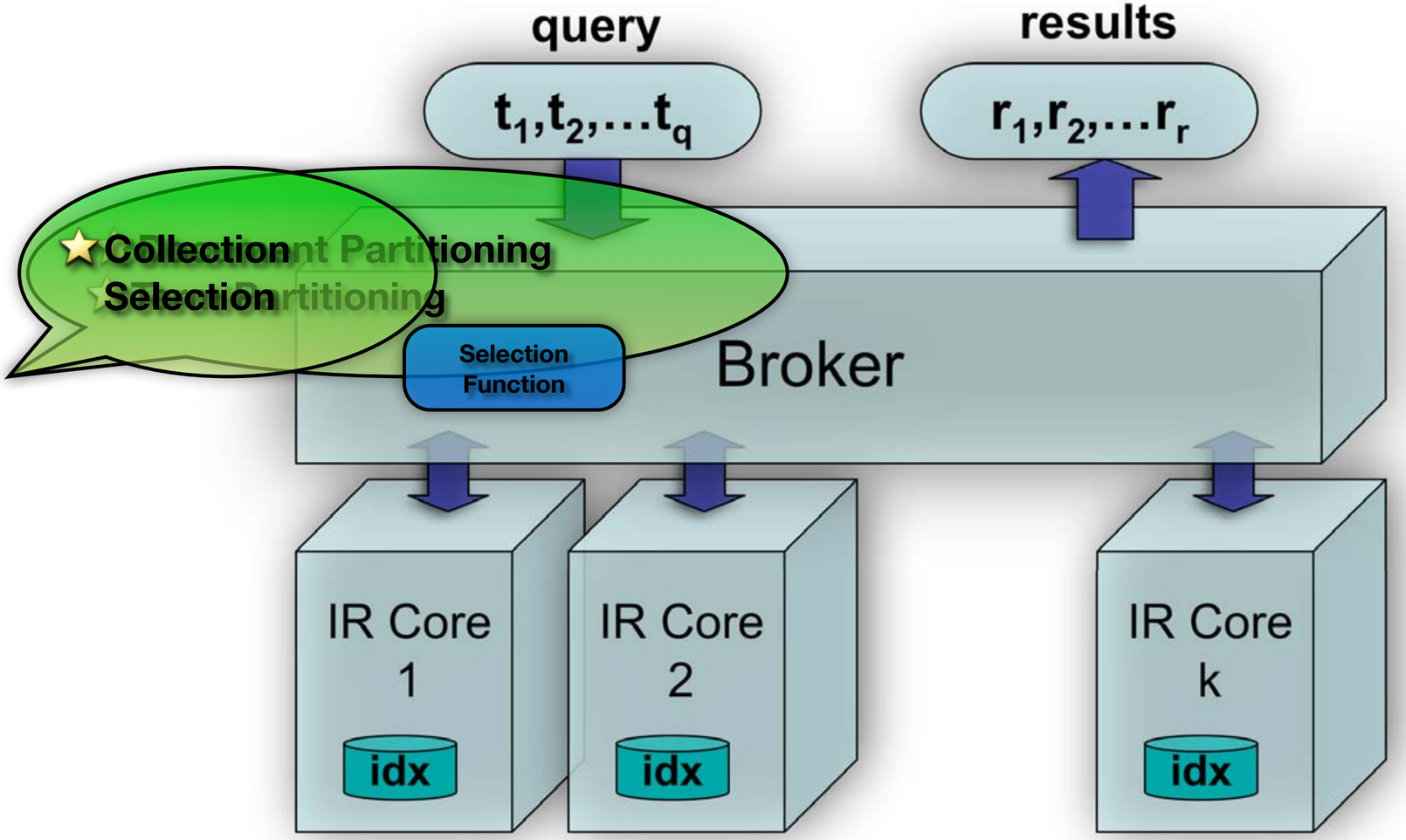
Lucchese, C., Orlando, S., Perego, R., and Silvestri, F. 2007. **Mining query logs to optimize index partitioning in parallel web search engines**. In Proceedings of Infoscale 2007. Suzhou, China, June 06 - 08, 2007.

The Overall Picture



Lucchese, C., Orlando, S., Perego, R., and Silvestri, F. 2007. **Mining query logs to optimize index partitioning in parallel web search engines**. In Proceedings of Infoscale 2007. Suzhou, China, June 06 - 08, 2007.

Document Partitioning



Collection Selection

- Traditionally used in Federated Distributed IR systems to reduce the number of queried servers.
- Rarely (???) used in Web Search Engine systems
  - see Google's MICRO paper on their architecture and recent Yahoo! works on federated search.

To Select or Not To Select?

- Pros
  - Reduced Load on IR Cores
  - Potentially Eliminates Noise due to the presence of non relevant documents w.r.t. a query
- Cons
  - Load Imbalance
  - Reduced Precision

The Curse of Reduced Precision

- The reduction in precision is an issue that have to be taken into consideration.
- In collection selection architectures, it can be enhanced by using:
  - Ad-hoc partitioning strategies
  - Collection Prioritization
  - Incremental Caching

Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

Query-Vector Document Model

- It is a vector-space like model
- Documents are represented by queries they answer.
- Example.
  Query "munch" is answered by documents d1, d2, d7, d3, d4. "munch" is represented by "1 2 3 4 7".

Possible Refinements:
- Consider Ranking Scores
- Consider Clicks

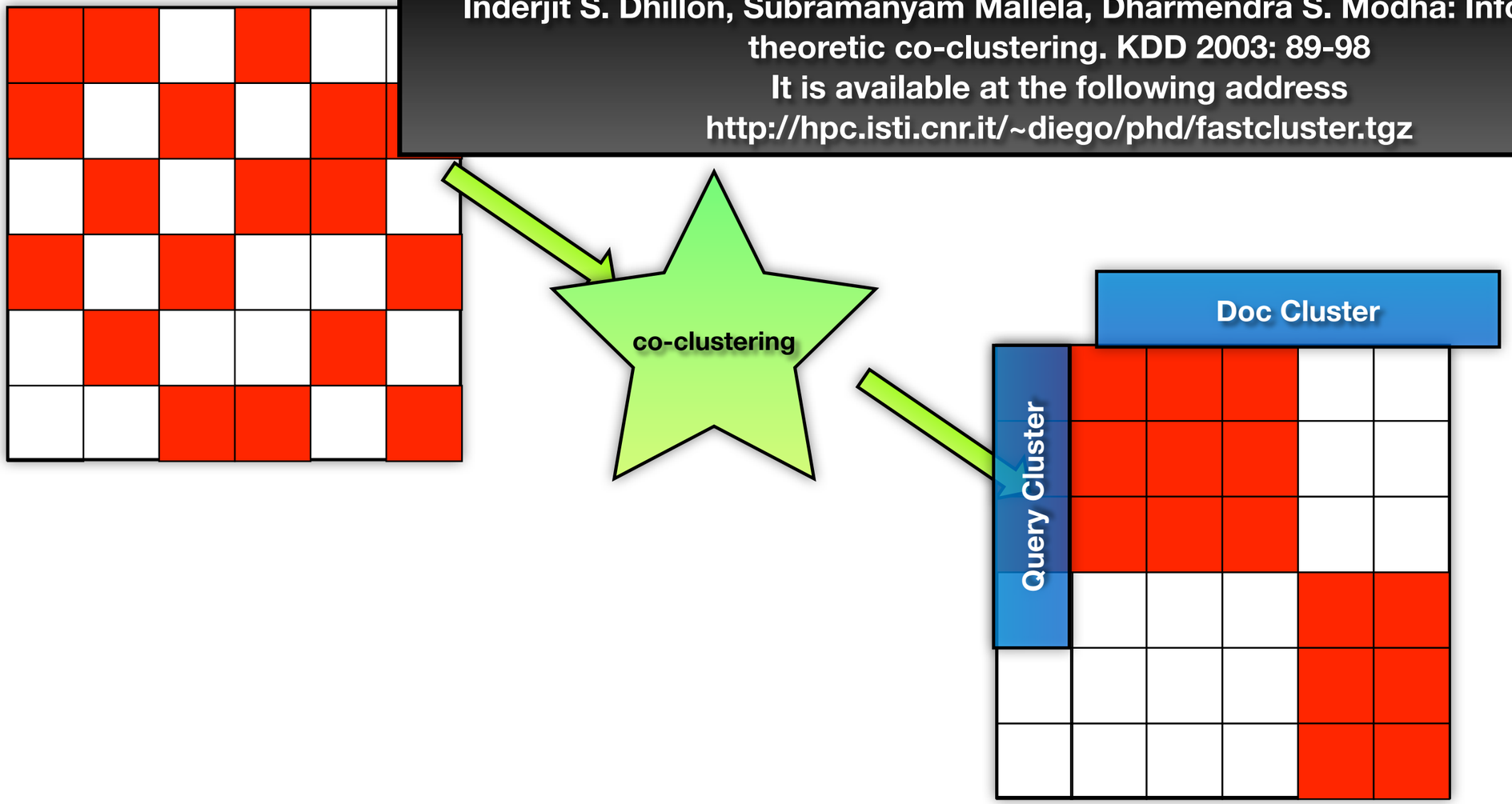
Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

QV-Based Partitioning

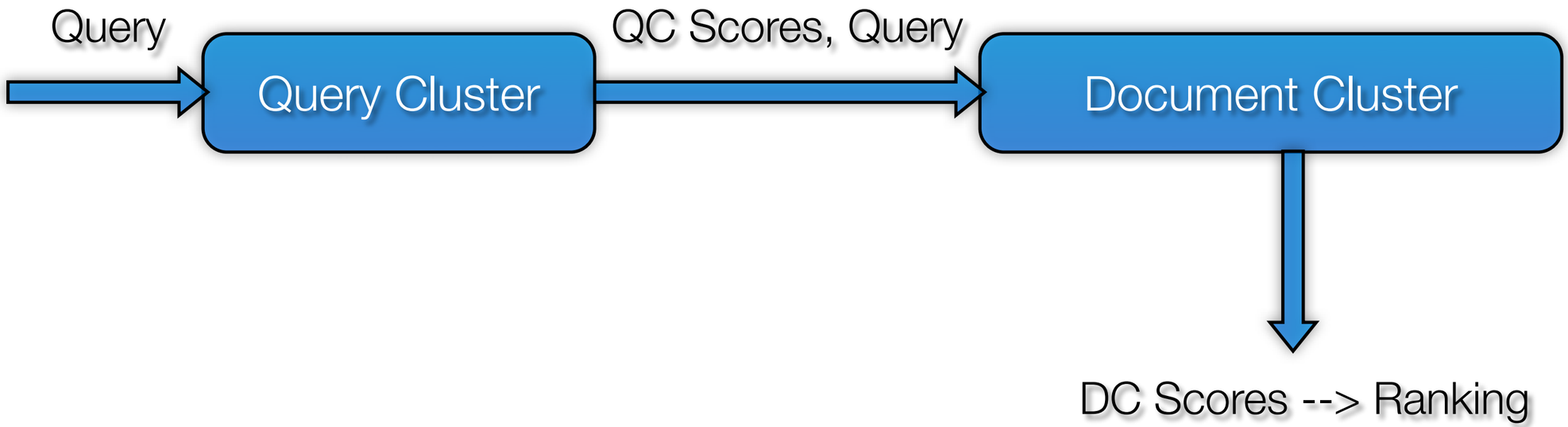
Documents

Queries

The co-clustering implementation is derived from the paper:
Inderjit S. Dhillon, Subramanyam Mallela, Dharmendra S. Modha: Information-theoretic co-clustering. KDD 2003: 89-98
It is available at the following address
http://hpc.isti.cnr.it/~diego/phd/fastcluster.tgz



QV-Based Collection Selection



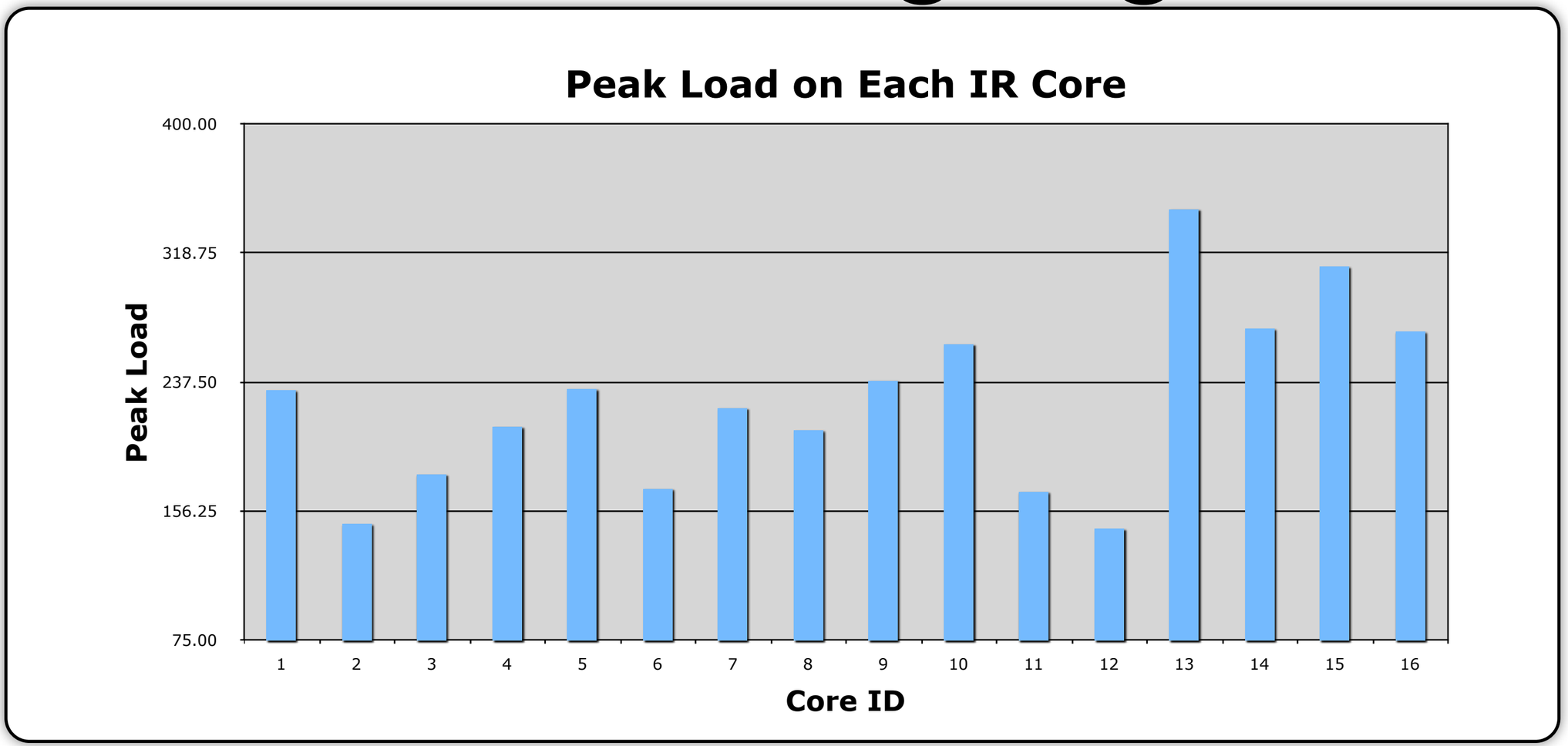
Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

Comparison in Terms of Precise Results

Intersection (%) at 10	1	2	4	8	16	OVR
Random	5	11	25	50	93	100
QV-based	34	45	58	76	96	100
Improvement	**6.8X**	**4.1X**	**2.3X**	**1.5X**	**>1X**	**-**

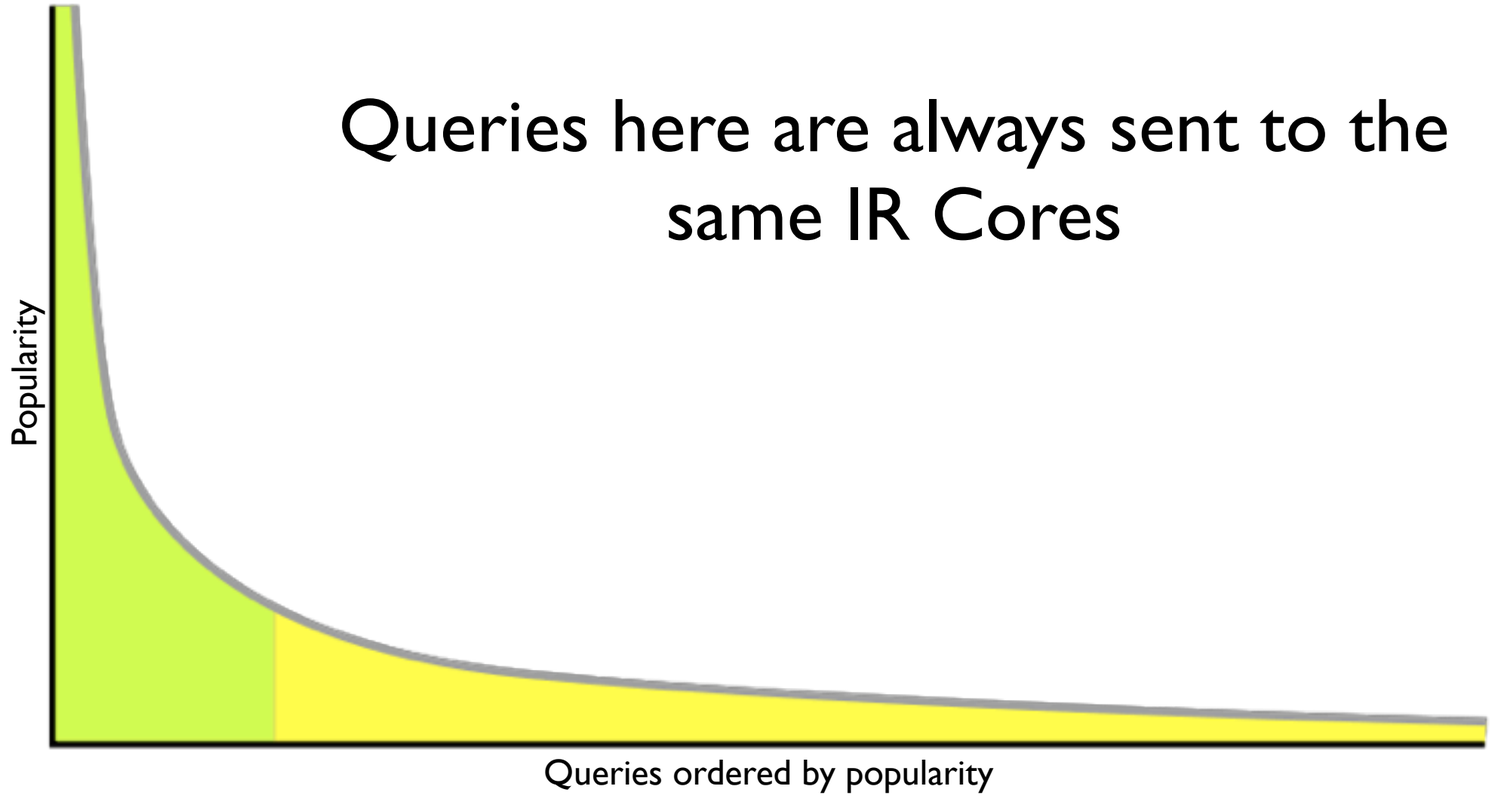
Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

Load Balancing... Again!



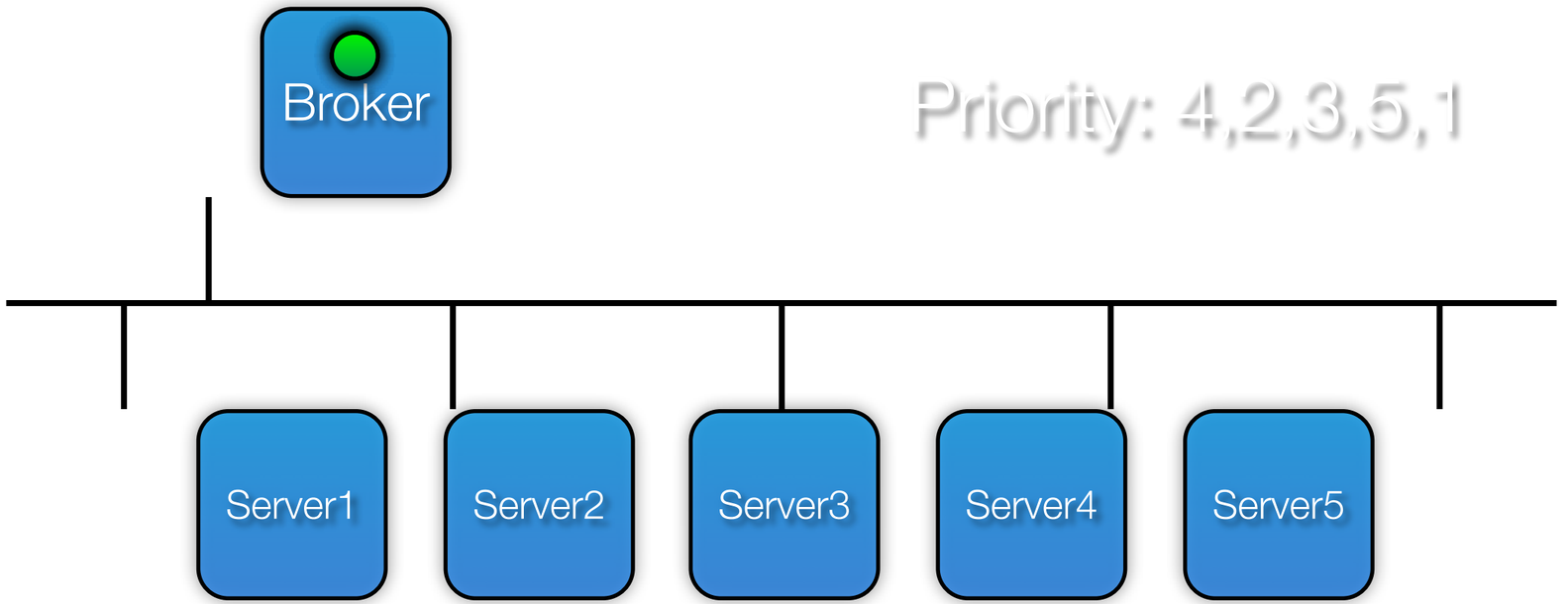
Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

Guess Why...



Queries here are always sent to the same IR Cores

Collection Prioritization



Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

How IR Cores Take Decisions

load-driven basic<*L*>

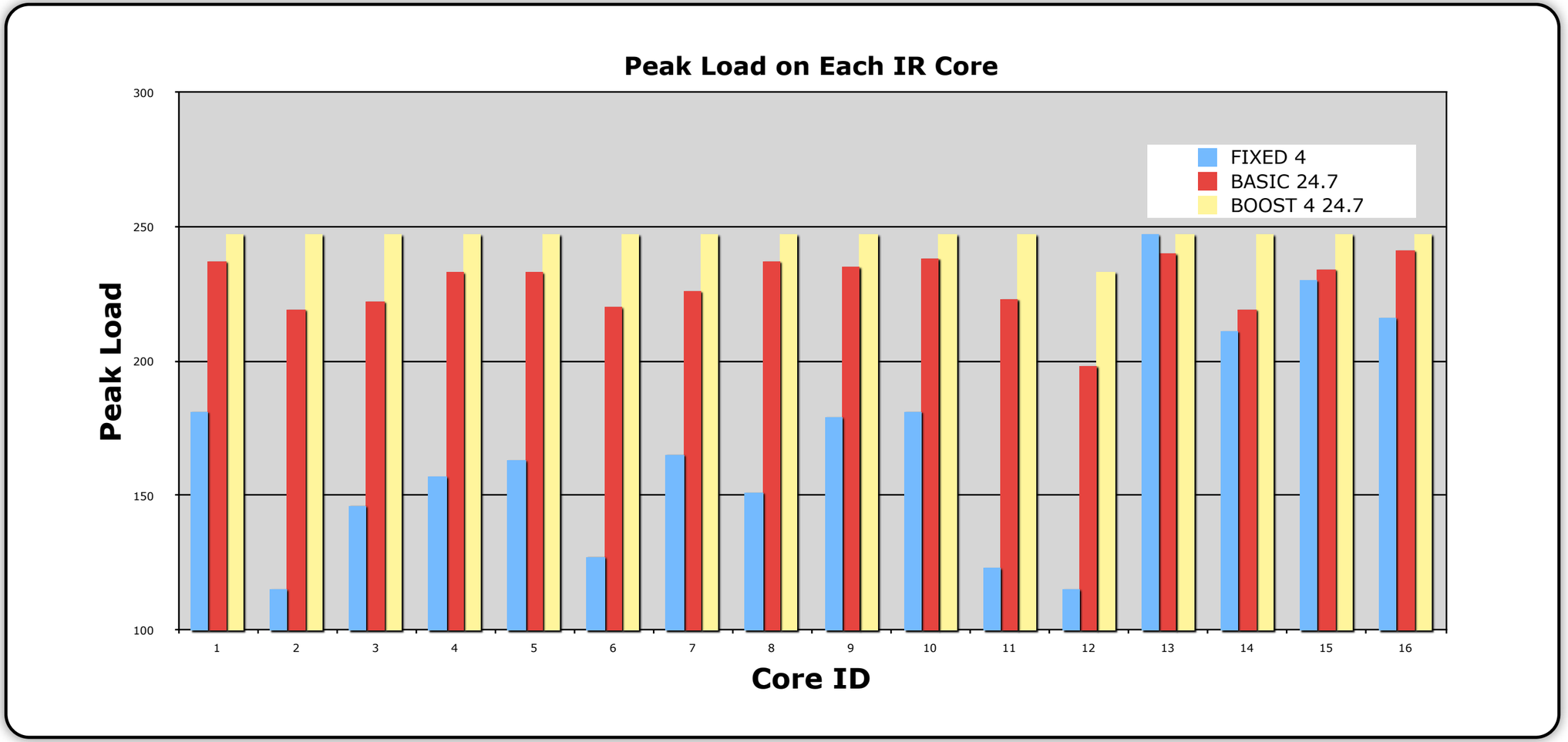
Accept queries only if the load is below *(1-rank/NServers) * L*. E.g. the second best server accepts the query only if its load is below *90%* of *L*.

load-driven boost<*L,T*>

It is the same as the basic strategy except that the first *T* servers' load threshold is always *L* and then it starts to decrease in a linear fashion as in the basic case.

Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

Load... Balanced!



Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

What about Precision?

	FIXED 4	BASIC<25>	BOOST<4, 25>
5	34	56	71
10	34	68	70
20	34	68	70

Intersection (%) at

Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

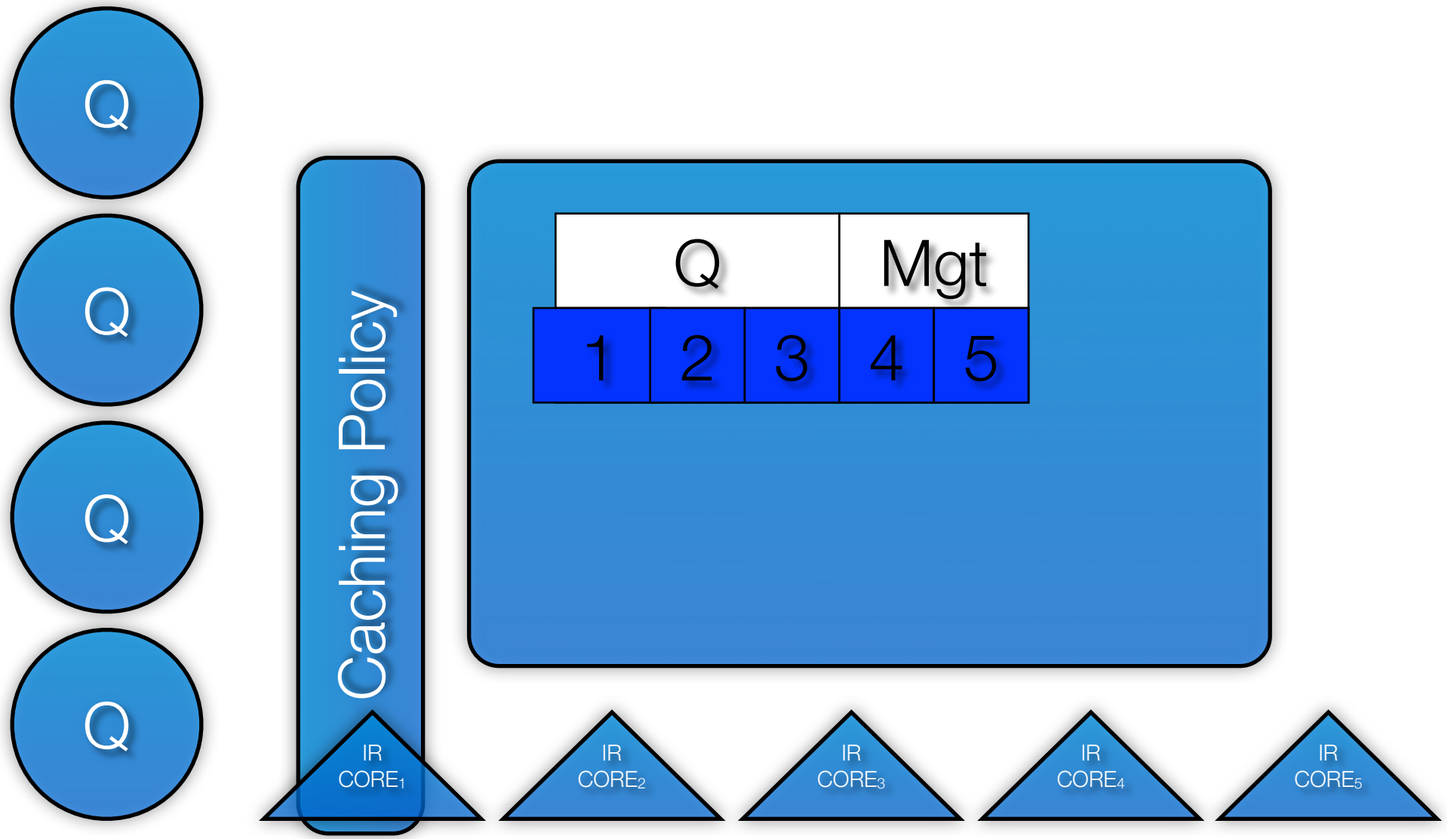
What about Precision?

	FIXED 4	BASIC<25>	BOOST<4, 25>
5	0.88	0.91	0.92
10	0.87	0.90	0.90
20	0.85	0.89	0.90

Competitive Similarity

Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

Incremental Caching



Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

Does it Work?

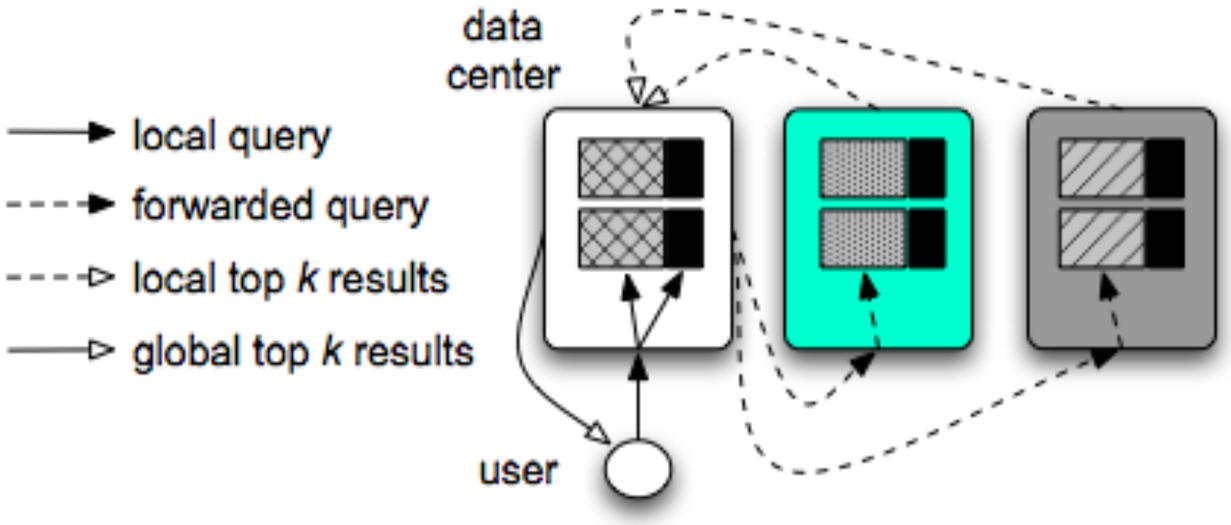
	BASIC<25>	BOOST<4, 25>	BOOST<4, 25> + INC
5	0.91	0.92	0.94
10	0.90	0.90	0.93
20	0.89	0.90	0.93

Competitive Similarity

Diego Puppin, Raffaele Perego, Fabrizio Silvestri, Ricardo Baeza-Yates. **Tuning the Capacity of Search Engines: Load-driven Routing and Incremental Caching to Reduce and Balance the Load**. ACM Transactions on Information Systems (TOIS), 28(2): (2010).

Query forwarding in Federated Search

- multiple, regional data centers (sites), user-to-center assignment, local web crawling, partitioned web index, partial document replication
- query processing with selective forwarding: a query should be forwarded to any site with potential to contribute at least one result to the global top-k



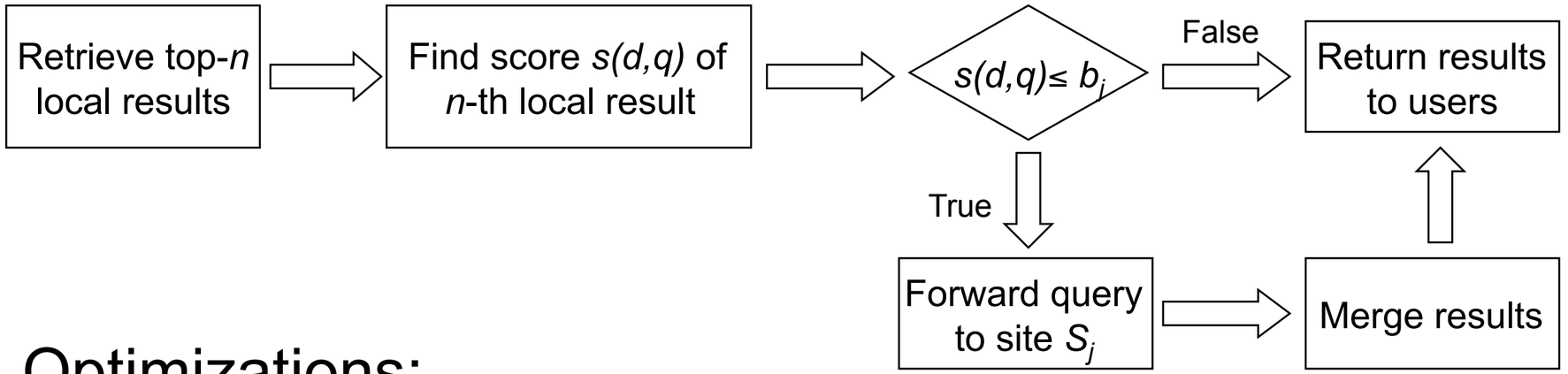
Ricardo A. Baeza-Yates, Aristides Gionis, Flavio Junqueira, Vassilis Plachouras, Luca Telloli: **On the feasibility of multi-site web search engines**. CIKM 2009: 425-434

Query forwarding in Federated Search

- Assumptions:
  - AND mode of query processing and independency of query terms. The document score is computed simply summing query term weights (e.g., BM25)
  - we have the top scores for a set of off-line queries on all non-local sites
- Idea:
  - set an upper bound on the possible top score of a query on non-local sites using the scores computed for off-line queries
  - decide whether a query should be forwarded to a site based on the comparison between the locally computed k-th score and the site's upper bound for the query

Ricardo A. Baeza-Yates, Aristides Gionis, Flavio Junqueira, Vassilis Plachouras, Luca Telloli: **On the feasibility of multi-site web search engines**. CIKM 2009: 425-434

Query forwarding in Federated Search



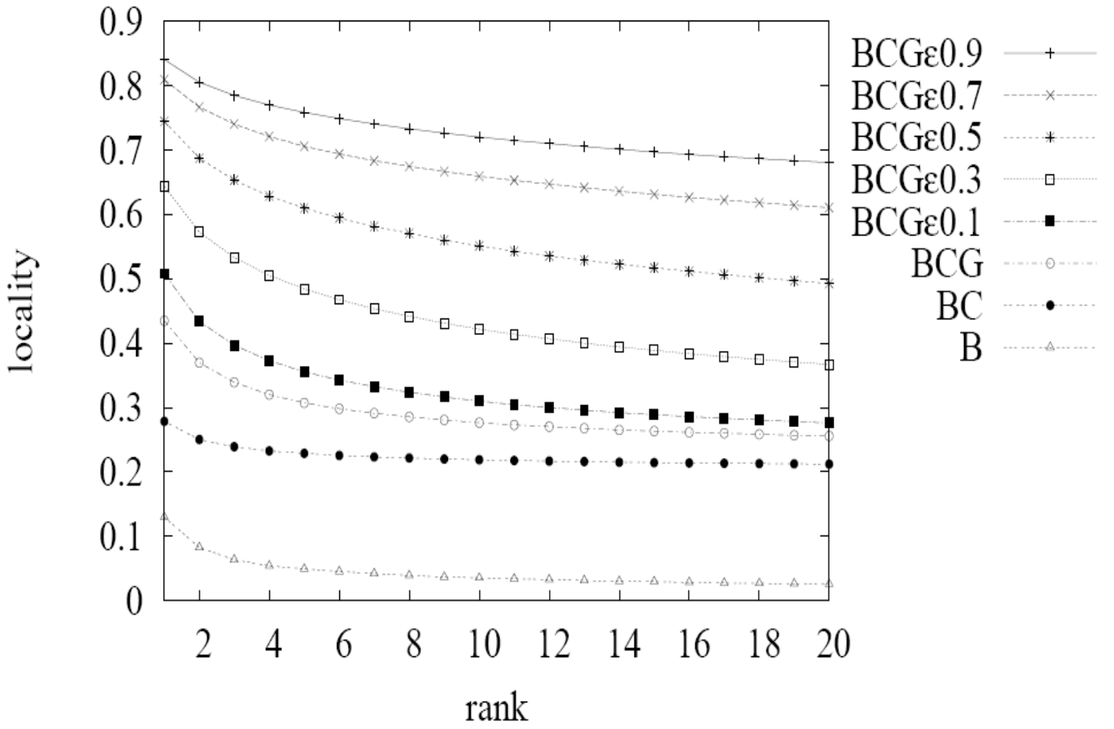
Optimizations:

- Caching SERPs locally
- Global replication of most frequently retrieved documents
- Slackness factor ε replacing bj with $(1-\varepsilon)b_j$

Ricardo A. Baeza-Yates, Aristides Gionis, Flavio Junqueira, Vassilis Plachouras, Luca Telloli: **On the feasibility of multi-site web search engines**. CIKM 2009: 425-434

Query forwarding in Federated Search

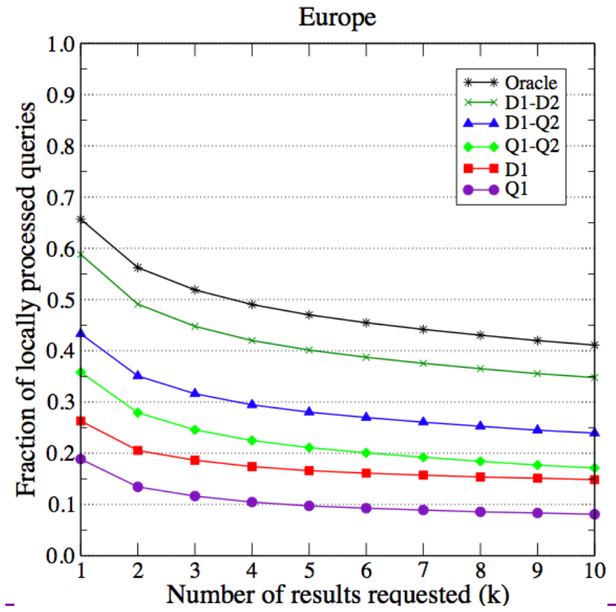
- Locality at rank n for a search engine with 5 sites
  - Percentage of query volume, for which we can return top-n results locally

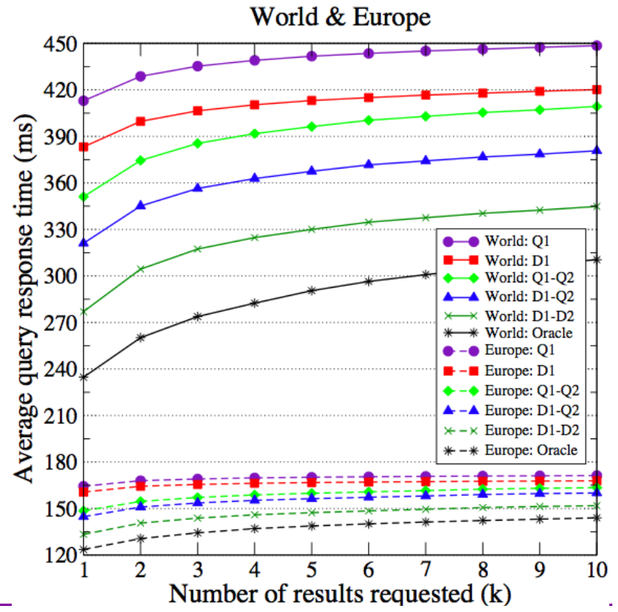


Ricardo A. Baeza-Yates, Aristides Gionis, Flavio Junqueira, Vassilis Plachouras, Luca Telloli: **On the feasibility of multi-site web search engines**. CIKM 2009: 425-434

Query forwarding in Federated Search

- Improvement using a Thresholding algorithm and a LP solution in
  - Berkant Barla Cambazoglu, Emre Varol, Enver Kayaaslan, Cevdet Aykanat, Ricardo A. Baeza-Yates: **Query forwarding in geographically distributed search engines**. SIGIR 2010: 90-97





Questions?