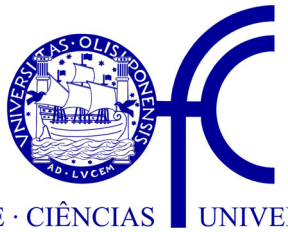


Distributed Indexes Creation for Large Scale Web Collections in the Sidra System

Miguel Costa, Mário J. Silva
Universidade de Lisboa, Faculdade de Ciências,
Departamento de Informática
XLDB Research Group @ LASIGE
mjs@di.fc.ul.pt

Indexing on the Web

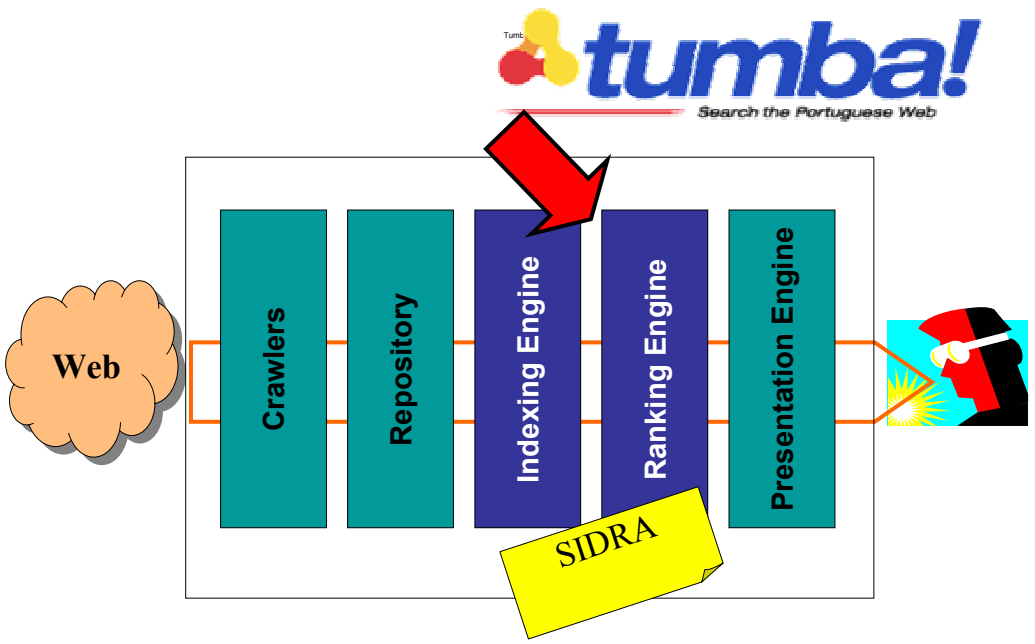
- Web size is huge, largest index only covers a tiny part.
  - Fast and unpredictable growth rates
- Web indexing requirements:
  - High scalability
  - Inexpensive hardware
  - Multiple dimensions

Tumba!

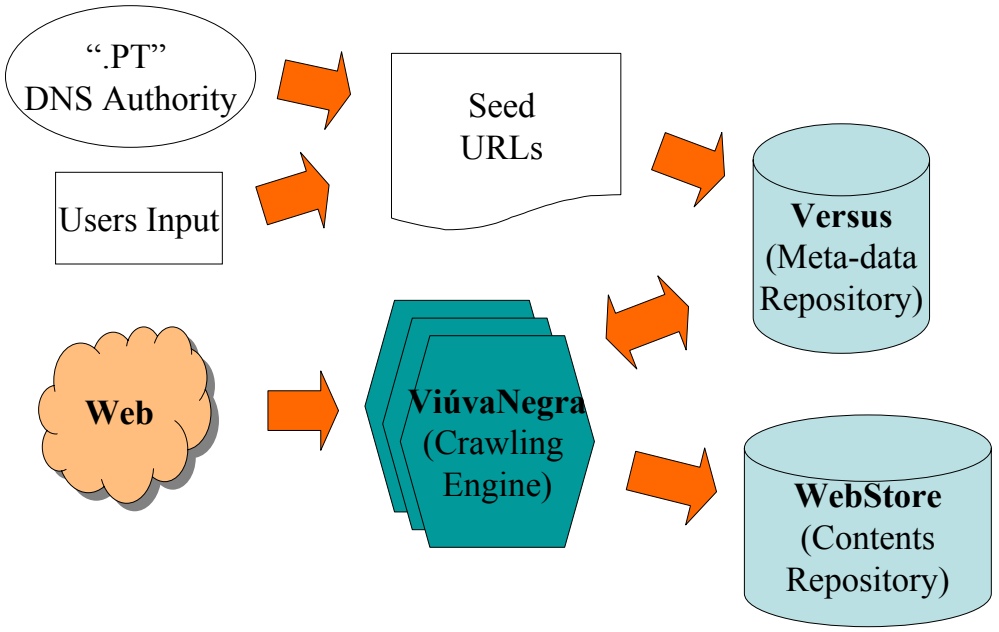
(Temos um Motor de Busca Alternativo!)

- Portuguese Web – The Web of the people related to Portugal.
- Public service
  - Community Web Search Engine
  - Web Archive
  - Research infrastructure
- See it in action at **http://tumba.pt**

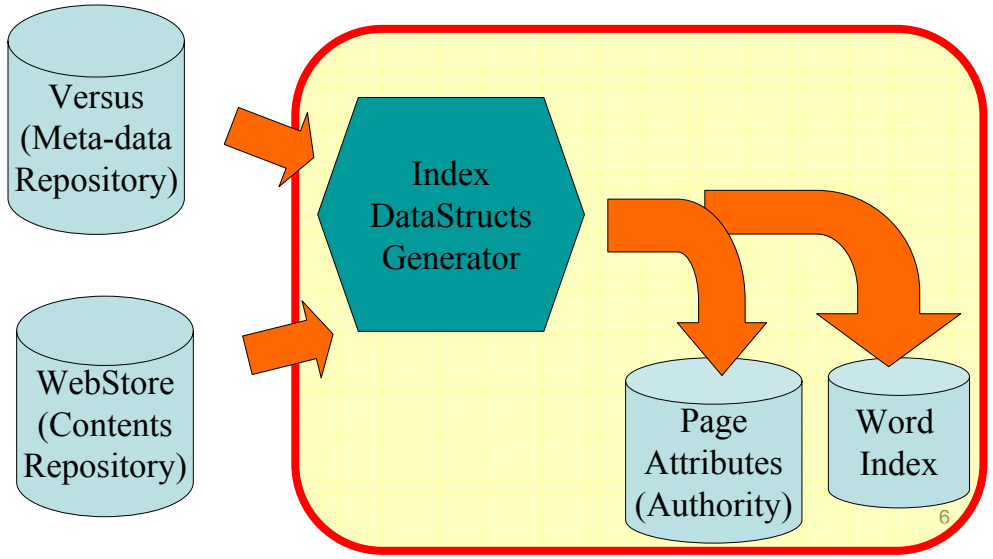
tumba!
Search the Portuguese Web



Crawling+Archiving



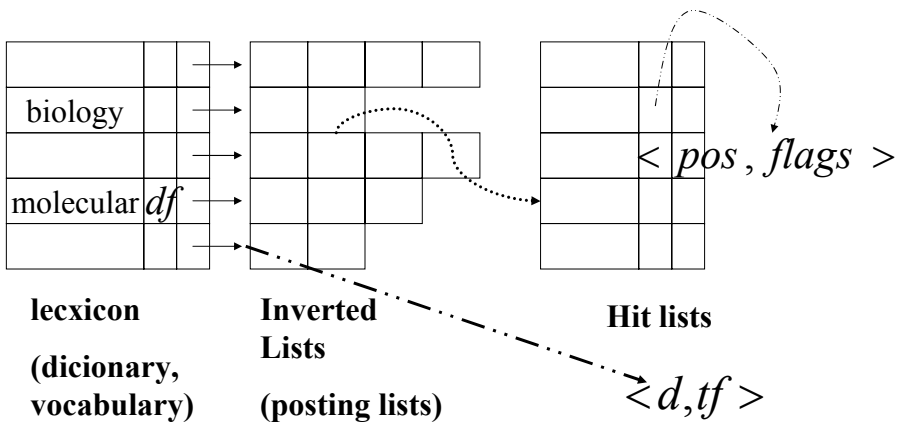
Query Processing Architecture (indexing phase)



Presentation Outline

- Index Data Structure
- Index Creation Algorithm
- Distributed Index design
- Evaluation
- Conclusions

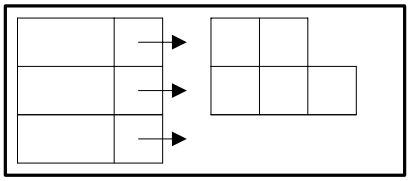
Term-Frequency Matrix (Inverted File Index)

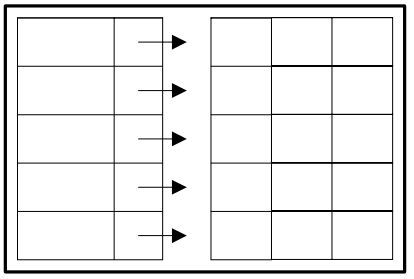


Efficient Ranked Retrieval for all IR Models – Fast and highly compressible

SIDRA - Word Index Data Structure

- 2 files
  Term → {docID}
  <Term,docID> → {hit}
- Hit = position + attrib
- Document IDs (sid) assigned in Static Rank order
- Posting lists ordered by Document ID.

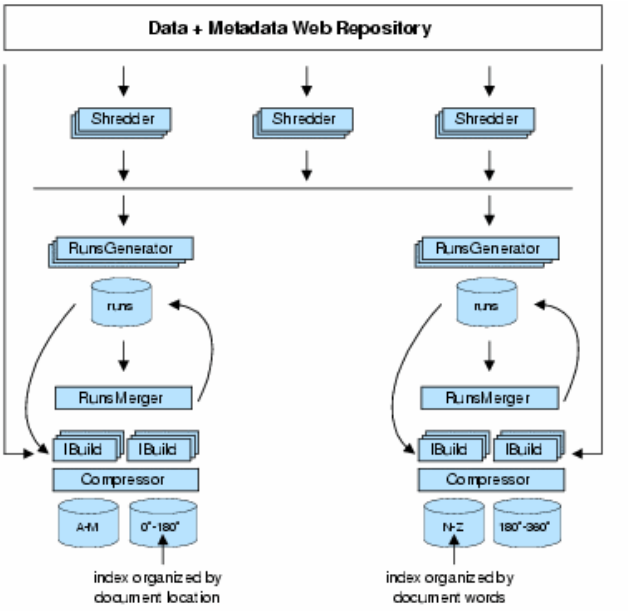




Index Generation Algorithm

1. Parse the documents
2. Generate (sorted) runs } (Shredding)
   <term, sid, hit>
3. Sort-Merge Runs
4. Generate Inverted File
   <term, [sid, [hit]]>

Index Generation Architecture

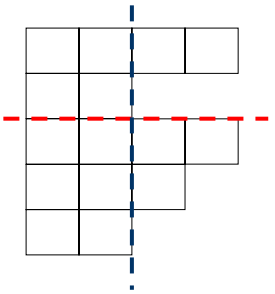


Presentation Outline

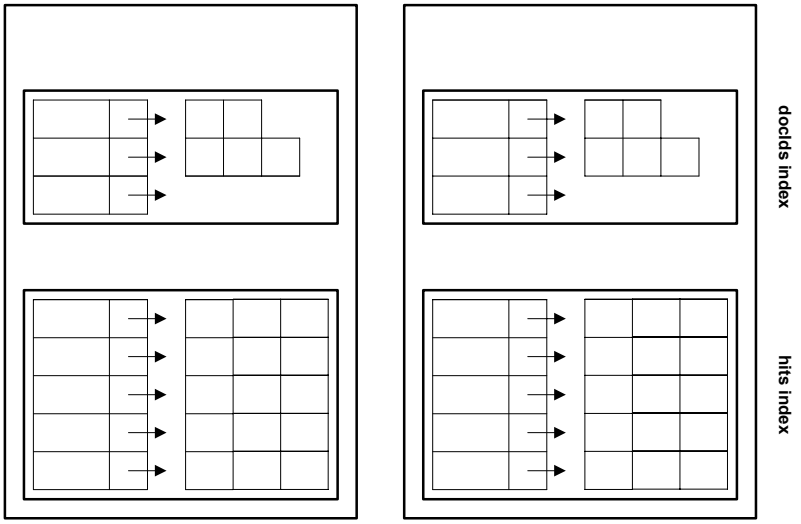
- Index Data Structure
- Index Creation Algorithm
- **Distributed Index design**
- Evaluation
- Conclusions

Data Distribution

- Replication (mirrors)
- Partitioning
  - Local (or document-based) (**vertical**)
  - Global (or vocabulary-based) (**horizontal**)

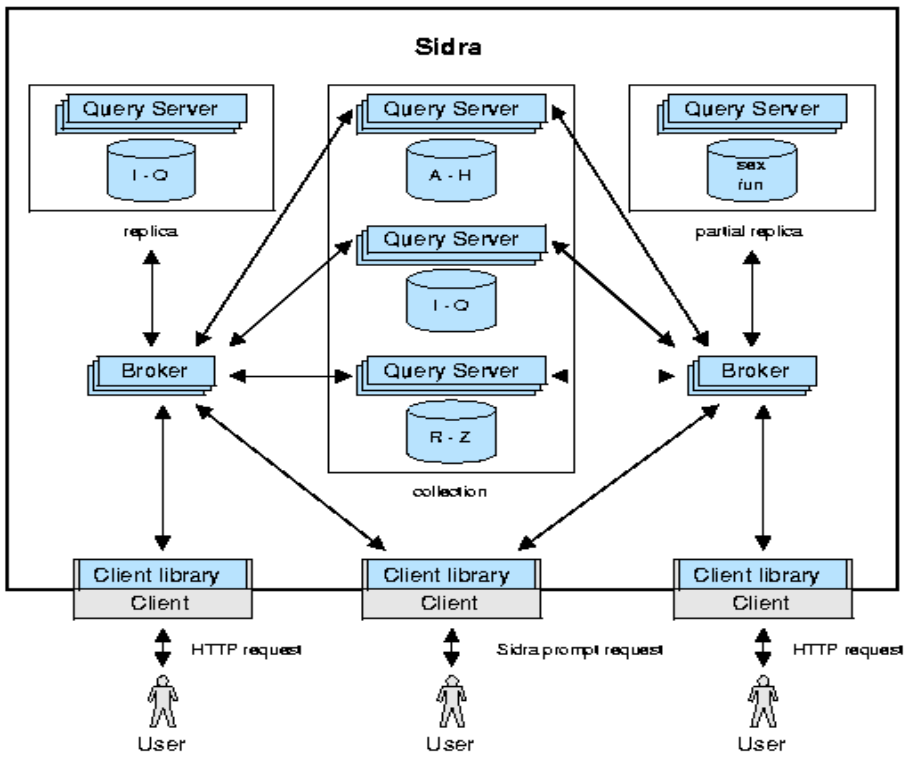


SIDRA – Global Index Range Partitioning

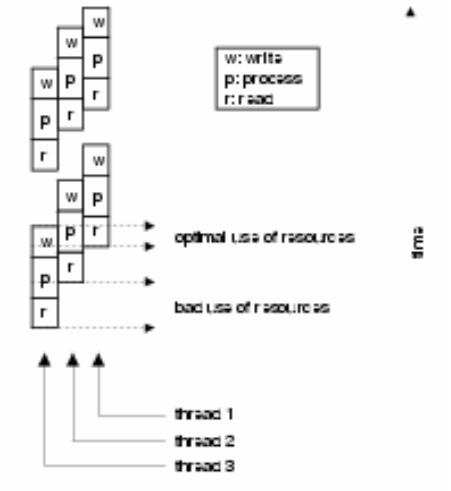


Distributed Index design

- Horizontal/global partition
  - Each QueryServer contains all the documents of a restriction (e.g., a keyword)
- Allow searches by different criteria to run in parallel (partition parallelism)
- Brokers merge results as they are being produced (pipelline parallelism)



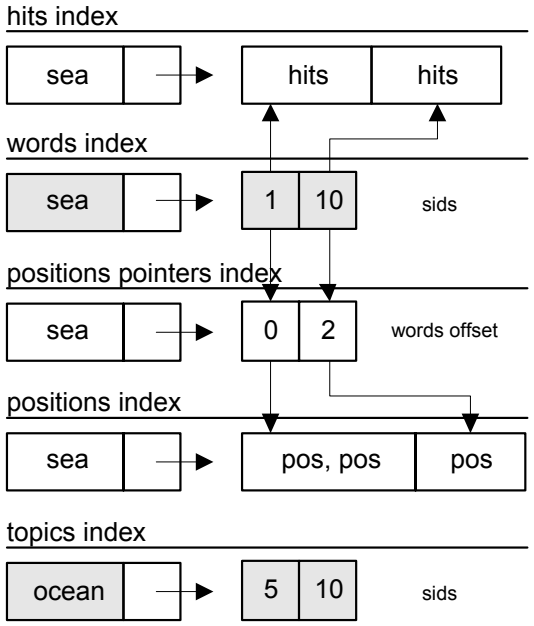
Shredding with double buffering



Addressing Multi-dimensionality

- Generalization: page-rank (page importance measure) isn´t but one of the possible ranking contexts.
- Query Servers may index data according to other dimensions
  - time
  - Location
  - ...
- Query Brokers perform the results “fusion”

Physical Index design in Sidra



- Goals:
  - Minimize IO
  - Minimize #seeks

Presentation Outline

- Index Data Structure
- Index Creation Algorithm
- Distributed Index design
- **Evaluation**
- Conclusions

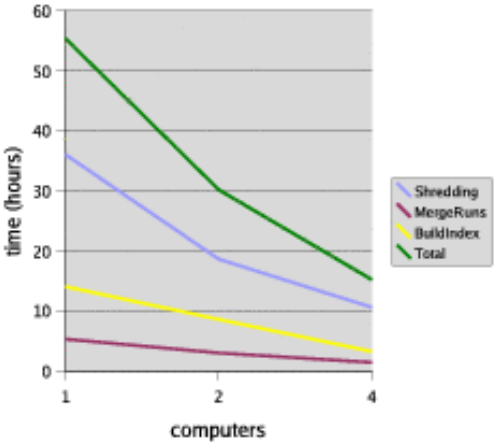
Testbed

- 4 ASUS AP140R servers
  - 2.4 GHz CPU/1GB RAM
  - 2 7200 rpm / 699 Mbit/sec mirrored disks
  - Linux (RedHat9)
  - 100Mb/s ethernet
- Web data collection
  - 78.4 GBytes
  - 3.2 Mdocs

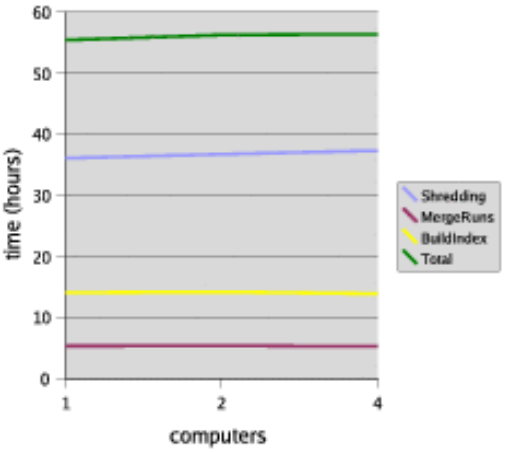
Index Size Statistics

- sids: 814 M (compressed) – 99%
- hits: 4.9 G
- posPointers: 3.4 G
- pos: 6.1 G
- site: 60 M (compressed)
- rankingValues: 17 M
- metadata: 641M
- total: 15.9 GB

Scalability



Speed-up



Scale-up

Comparative Evaluation

System	# CPUs	Collection			Index	
		Size (GB)	type	compressed	Index	comp.
Sidra	4	313.6	Web	yes	global	yes
RR	16	100.0	text	?	global	yes
Google	4	147.8	Web	yes	local	yes
CobWeb	312	500,000.0	Web	?	?	?

Comparative Evaluation

System	# CPUs	Collectn Size (GB)	Gen. Time (hrs.)	Speed	
				Gb/CPU/hr	#pages/CPU/hr
Sidra	4	313.6	56,4	1,39	16,2
RR	16	100.0	6	1,04	?
Google	4	147,8	147,4	0,25	13,5
CobWeb	312	500000	130,6	12,27	75

Index Update Strategies

- Incremental indexing is complex, needs extra free space allocation
- Cho and Molina: for monthly rebuilds, freshness is almost identical
- **Google**: full monthly rebuilds + full rebuilds on some partitions
- **SIDRA**: full rebuilds

Final Conclusions

- Web indexes grow at a fast rate
  - High scalability is essential
- Sidra:
  - supports multiple indexes that enable contextualization of queries on multiple search dimensions
  - uses a global partitioning scheme
- Implementation based on known IR and distributed database techniques.