Hybrid Index Structures for Location-Based Web Search

Y. Zhou, X. Xie, C. Wang, Y. Gong, and W. Ma. *In Proceedings of the 14th ACM international conference on Information and knowledge management*

Processing Spatial-Keyword Queries in Geographic Information Retrieval Systems

R. Hariharan, B. Hore, C. Li and S. Mehrotra. *19th International Conference on Scientific and Statistical Database Management (SSDBM 2007)*

Presented by: Thomadakis Polykarpos

CS 834 – Introduction to Information Retrieval

Fall 2017

Old Dominion University

Location-based web search

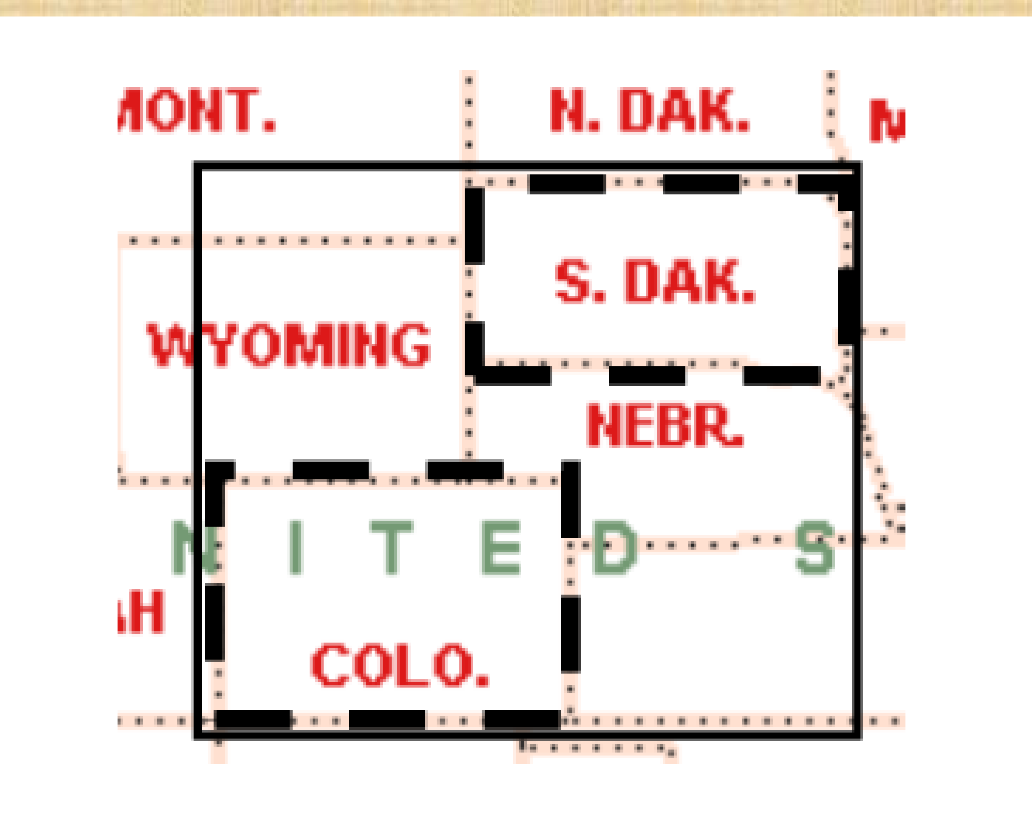
- Location-specific information common on Web
  - Nearly one fifth of web search related to place/region
  - Mostly useful for mobile users
- Efficiently indexing and searching location-specific information is a key problem
- Naïve approach: treat location information as common keywords
  - Neglects spatial relationships

Solution

- Design an indexing structure that considers both spatial and textual features
- Three approaches tested
  - Inverted file and R*-tree double index
  - First index file then R*-tree
  - First R*-tree then inverted file

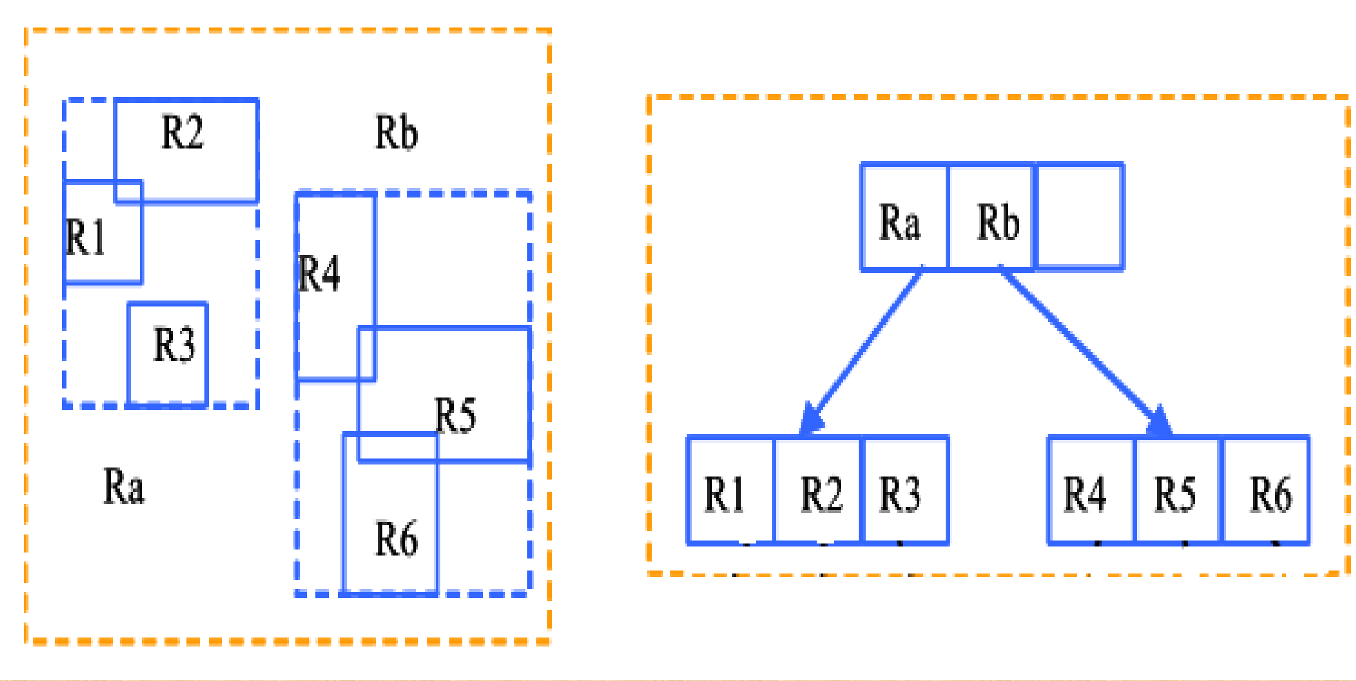
Preliminaries: MBR

- Minimum Bounding Rectangles (MBRs) can be used for a simple approximation to region's shape
- Only two diagonal points needed to represent location
- Memory efficient and simple for computations
- Multiple MBRs per web page can be used

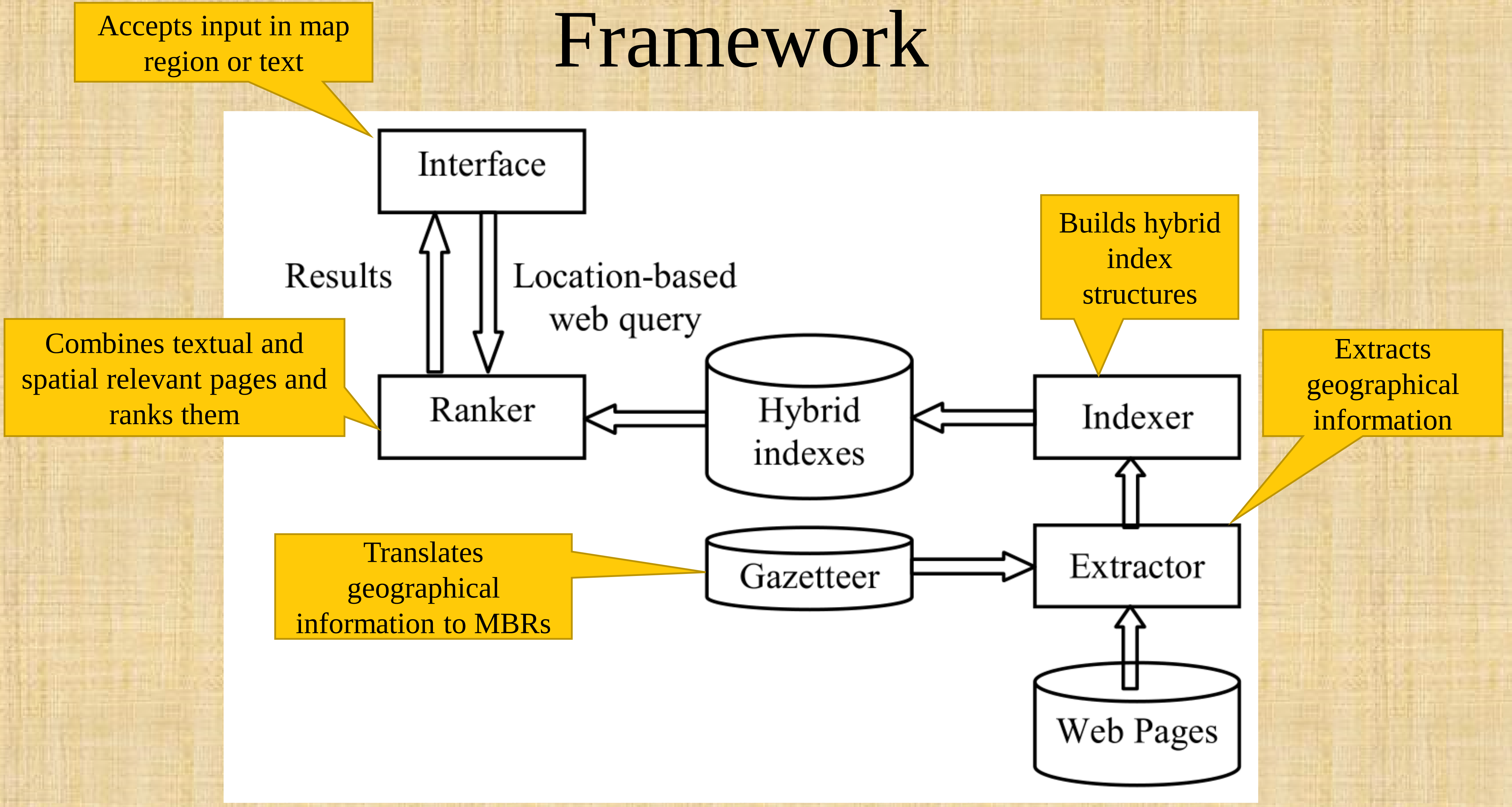


Preliminaries: R*-tree

- MBR structure where the parent node contains all sub-MBR regions.
- Useful to represent part-of relationships
- Used to extract the geographical scope of web pages

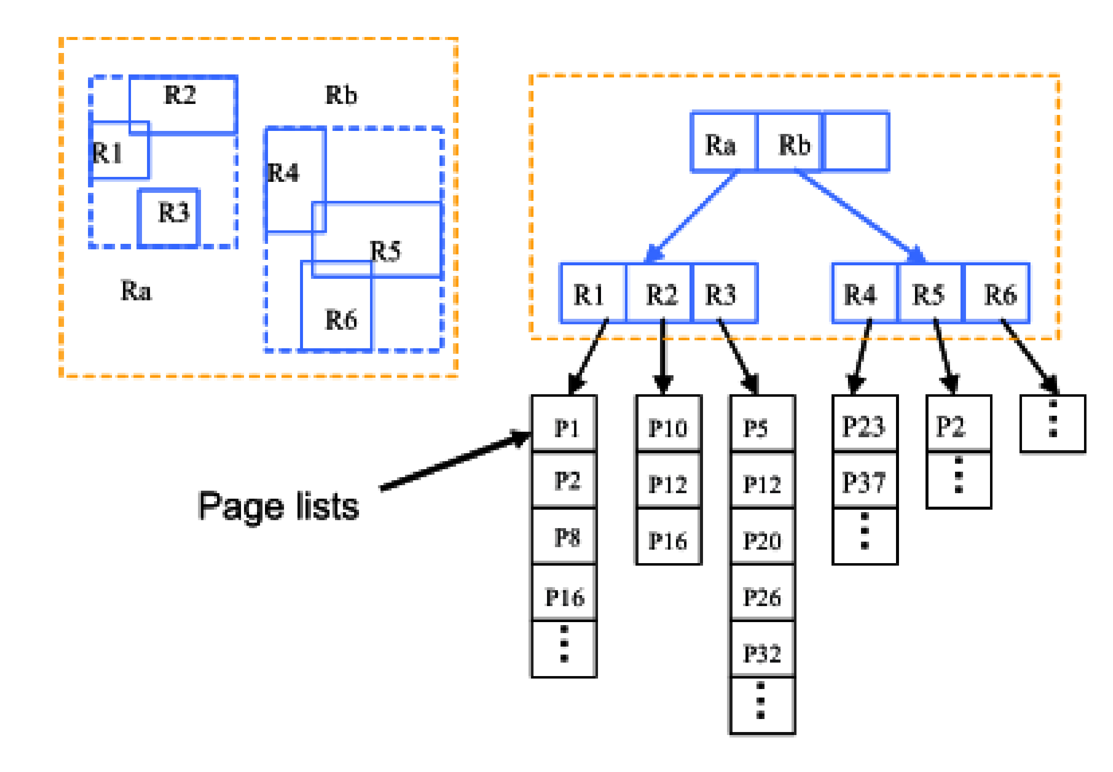


Framework


Accepts input in map region or text
Interface
Results
Location-based web query
Builds hybrid index structures
Combines textual and spatial relevant pages and ranks them
Ranker
Hybrid indexes
Indexer
Extracts geographical information
Translates geographical information to MBRs
Gazetteer
Extractor
Web Pages

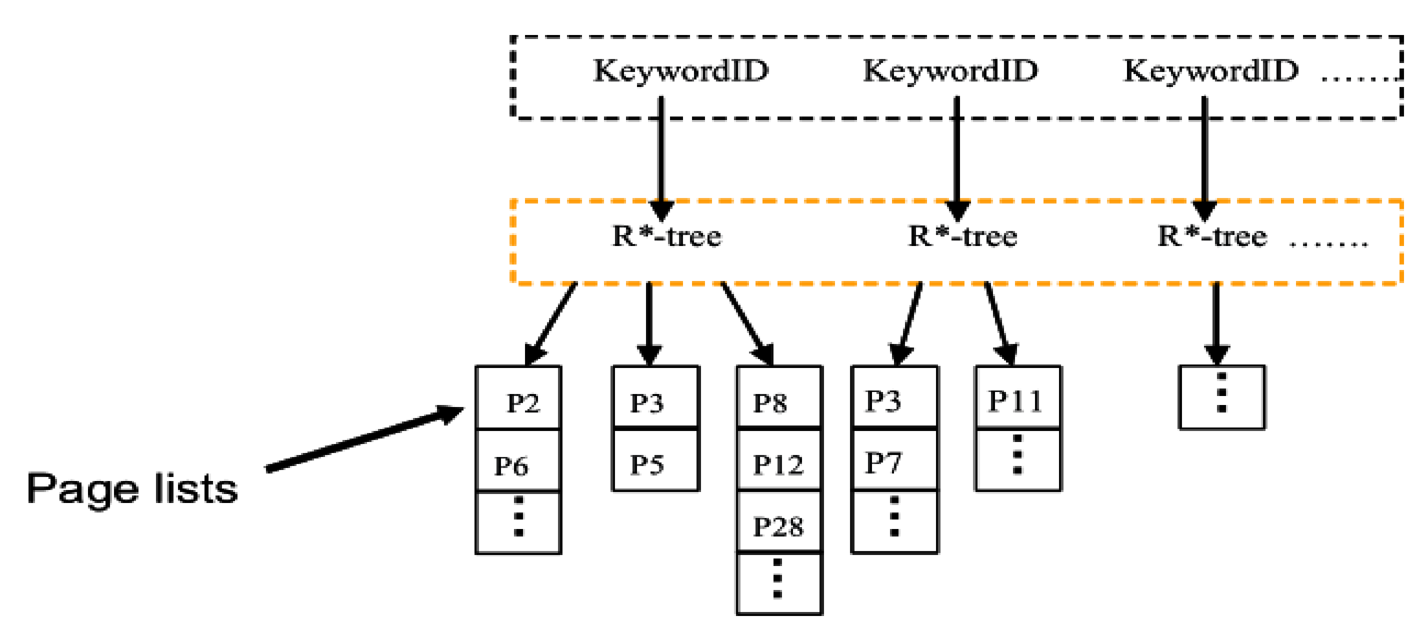
Approach 1: Inverted file and R*-tree double index

- Web pages indexed separately twice
  - By R*-tree for MBRs
  - By inverted file for keywords
- R*-tree leaves contain points to a page list
- Query answering:
  - Retrieve page lists with the keywords from inv. Index
  - Retrieve MBRs for R*-tree and the corresponding page lists
  - Merge the two lists



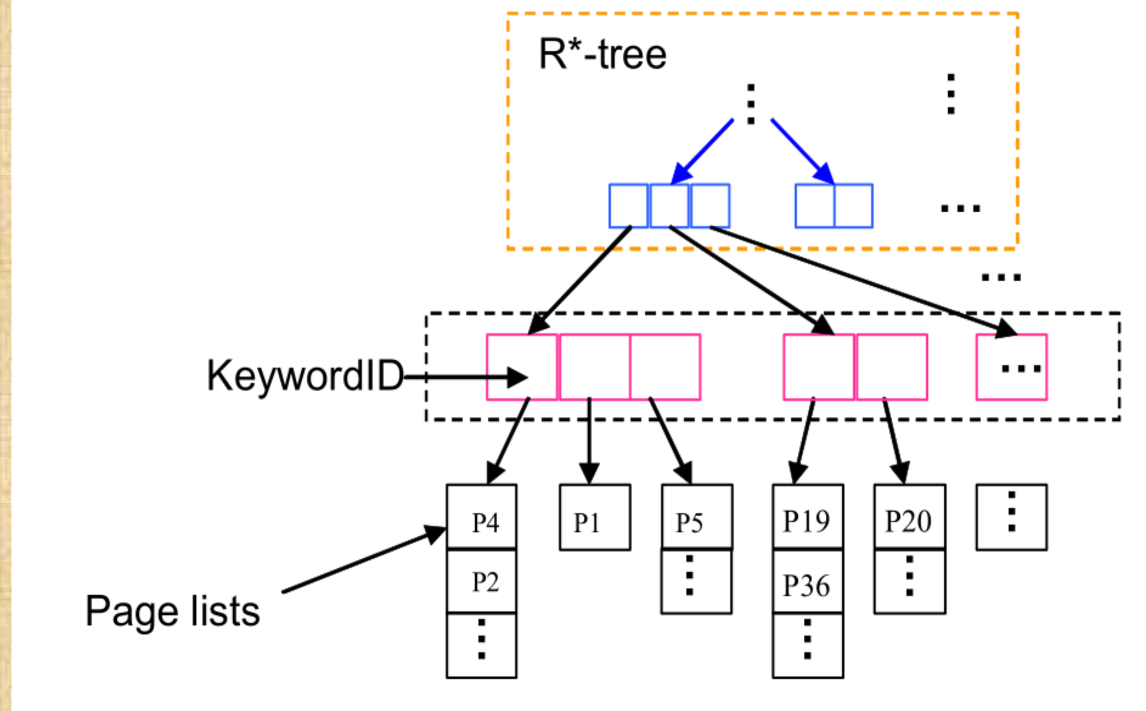
Approach 2: First Inverted File Then R*-tree

- Each keyword points to an R*-tree
- Built on the different MBRs of the pages of the in its page list according to their geographic scope
- Answering queries:
  - Retrieve the keywords
  - Search in the corresponding R*-trees and get their page lists
  - Merge the page lists returned by the R*-trees



Approach 3: First R*-tree Then Inverted File

- R*-tree is built on all MBRs included in scopes of all web pages
- Web pages assigned to MBRs according to their scopes
- All pages of each MBR are textually indexed by keywords
- Answering queries
  - Get the MRBs from the R*-tree
  - For each MRB, retrieve the keywords' page lists
  - Merge the page lists



Results

Table 4. Disk storage for three hybrid index structures.

	Page lists	The number of lists	Total length of page lists	Average length	Physical size (Mbytes)
The 1st structure	entry is a keyword	758,717	33,481,669	44.13	140.00
	entry is an MBR	4,246	197,988	46.63	0.83
The 2nd and 3rd structures	entry is a geo-keyword	3,535,505	27,666,384	7.83	138.95

Table 5. Average query time for three hybrid index structures.

	The 1st structure	The 2nd structure	The 3rd structure
Total length of page lists per query	38,868.91	122.42	122.42
Number of page lists per query	72.04	4.36	4.36
$T_R(ms)$	2.34	0.16	2.34
$T_{I/O}(ms)$	30.83	7.91	7.91
$T_{mg}(ms)$	17.01	0.73	0.73
Query time(ms)	50.18	8.80	10.98

Why Processing Queries in GIS?

- GIS databases contain vital location-based information for applications such as:
  - disaster management
  - national infrastructure protection
  - crime analysis
- Crucial to be able to answer geographic based queries e.g:
  - Find shelters with emergency medical facilities in OrangeCounty.
  - Find earthquake-prone zones in Southern California.

GIS databases

- Comprise of two components:
  - Spatial information
  - Textual information
- Queries also contain spatial and textual components referred as spatio-keyword queries (SK)
  - Defined as {spatial part as MBR, set of keywords}
  - Answer is a set of objects which MBRs has a non-empty intersection with the query and contains ALL keywords in the query
- Retrieval of such information referred as Geographic Information Retrieval (GIR)

Framework for GIR Systems

- GIR Database
  - Can come from structured or unstructured data sources
- Indexer
  - Indexes both textual and spatial information
- Ranker
  - Ranks GIS objects based on their relevance to the SK query
- Interface
  - Input using map and a text interface

Indexing Mechanisms

- Separate index for spatial and textual attributes
  - Index of spatial data: Grid, Quadtree, R*-tree, etc.
  - Index of textual data: Inverted index
- Pros
  - Ease of maintaining two separate indices
- Cons
  - Too much traffic on the disk

Indexing Mechanisms (cont.)

- Hybrid index combining spatial and text attributes
  - Inverted File – R*-tree
  - R*-tree – Inverted File
- Cons
  - 1st case: Does not take advantage of the association of keywords in space. Paying extra costs to access closely related SKs from different R*-tress
  - 2nd case: Leverages the above disadvantage but spatial filtering generates many candidate objects

Proposed method: KR*-Tree

- Prune text and space simultaneously, merging the two steps into one
  - Instead of pruning them separately or one after the other
- Capture the joint distribution of keywords
  - Instead of maintaining them separately
- Thus objects containing the query words are directly obtained without merging any lists
- Similar to R*-tree – Inverted File but
  - Internal and leaf nodes augmented with a set of keywords appearing in the space covered by them

Answering SK Queries

- First the children node ids of the current node that contain all the keywords in the query are obtained
- Next, each of these children that has a non-empty intersection with the Q’s MBR is chosen for further traversal
- Leaf children are added to the results, otherwise they are traversed recursively.
- Finally, the set of objects that satisfy the query is returned

Results

keywords	Dataset Size	*KR*-tree* vs. *Inv.R*-tree* (d_{gain})	*KR*-Tree* vs. R*-tree.Inv (d_{gain})
2-keywords	small	37%	66%
	medium	24%	70%
	large	33%	67%
3-keywords	small	43%	61%
	medium	26%	68%
	large	36%	60%

Table 7. Average Disk I/Os