CSEG601 & CSE5601
Spatial Data Management & Application :

Spatial Keyword Search Query

Sungwon Jung

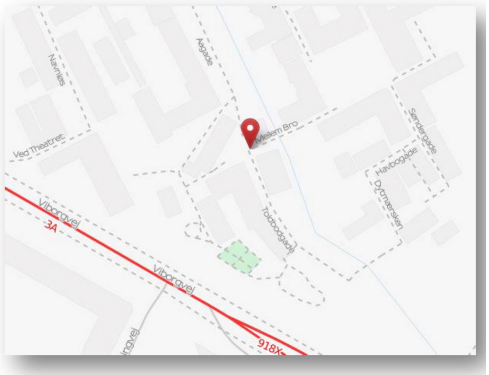
Big Data Processing & Database Lab.
Dept. of Computer Science and Engineering
Sogang University
Seoul, Korea
Tel: +82-2-705-8930
Email : jungsung@sogang.ac.kr

Geospatial and Textual (Geo-Textual) Data

- Spatial Web Objects: $p = \langle \lambda, \psi \rangle$ (location, text description)
- Example:

λ = *(56.158889, 10.191667)*

ϕ = Den Gamle By Open-Air Museum
Den Gamle By - "The Old Town" – was founded in 1909 as the world's first open-air museum of urban history and culture…

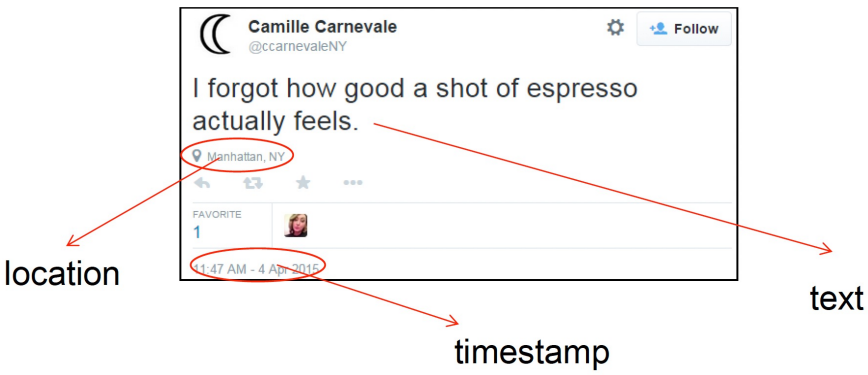


Geo-textual Data – Sources

- Static geo-textual data, POI data
  - Web pages with location
  - Online business directories
    - E.g., Google My Business
  - Location-based social networks
    - E.g., 65 million POIs at Foursquare
- Streaming geo-textual data
  - Geo-tagged micro-blog posts
    - E.g., 10 million geotagged Tweets per day
  - Photos with tags and geo-location in social photo sharing websites
    - E.g., Flickr
  - Check-in information at POIs in location-based social networks (e.g., Foursquare, Facebook places)
    - E.g., Foursquare had 7 million check-in on 3rd Oct 2015

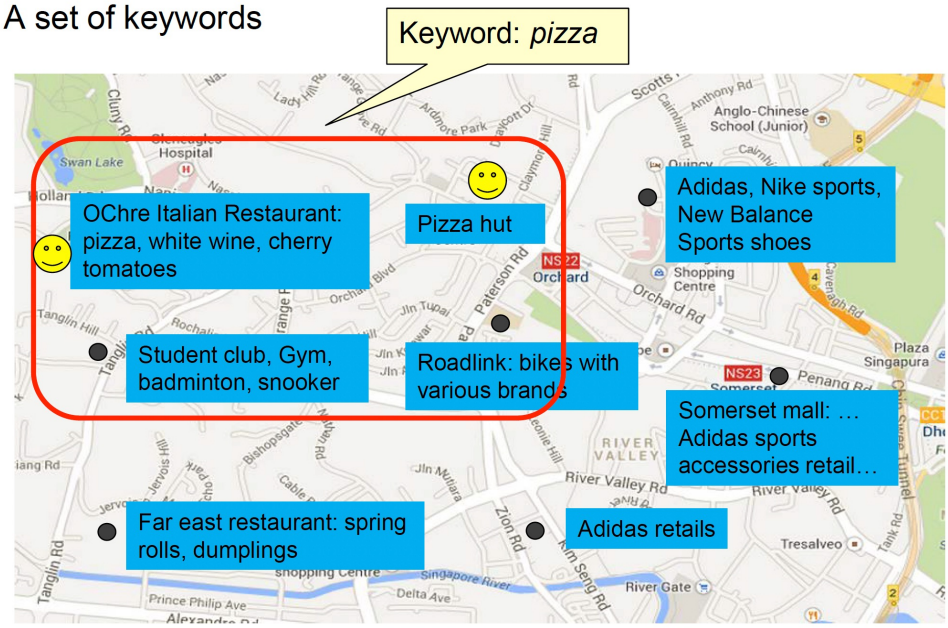
Example of Streaming Geo-Textual Data

- Components of streaming geo-textual data
  - Text
  - Location
  - Time
- Example: Geo-tagged tweets



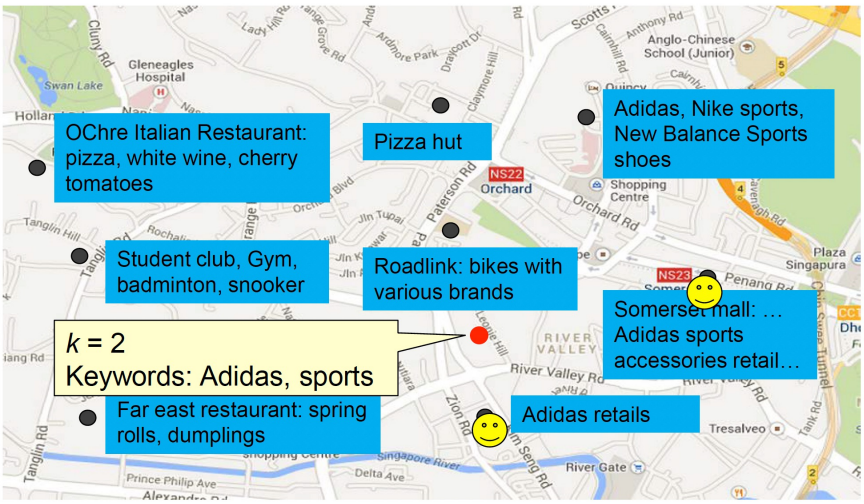
Boolean Range Query

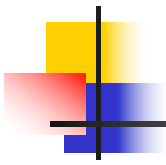
- A query region
- A set of keywords



Top-*k* *k*NN Query (TkQ)

- A query location
- A set of keywords
- Ranking criteria: A combination of spatial proximity and text relevancy





Boolean *k*NN Query

- "Retrieve the k objects nearest to the user's current location (represented by a point) such that each object's text description contains the keywords tasty, pizza, and cappuccino."

Top-*k* Spatial Keyword Query

- Objects: $p = \langle \lambda, \psi \rangle$ (location, text description)
- Query: $q = \langle \lambda, \psi, k \rangle$ (location, keywords, # of objects)

- Ranking function

$$rank_q(p) = \alpha \frac{\| q.\lambda, p.\lambda \|}{\max D} + (1-\alpha)(1 - \frac{tr_{q.\psi}(p.\psi)}{\max P}) \qquad 0 \le \alpha \le 1$$

  - Distance: $\| q.\lambda, p.\lambda \|$
  - Text relevancy: $tr_{q.\psi}(p.\psi)$
    - Probability of generating the keywords in the query from the language models of the documents

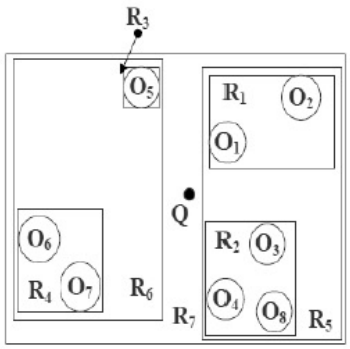
- Generalizes the *k*NN query and text retrieval

R*-IF Index

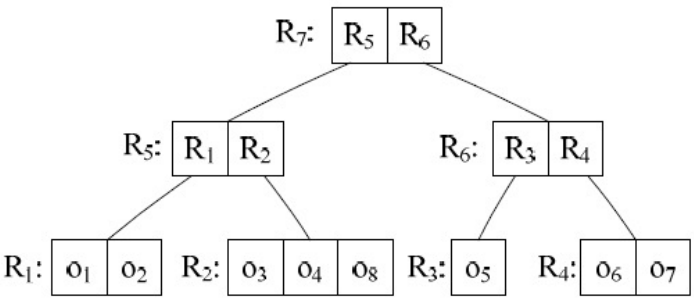
- R*-IF = R*-tree - Inverted File
- R*-IF is a spatial first index.

- R*-IF is first built for indexing all objects in D without considering their text components.
- For each leaf node of the R*-tree, an inverted file is created for indexing the text components.

R*-IF Index



(a) Objects and their bounding rectangles



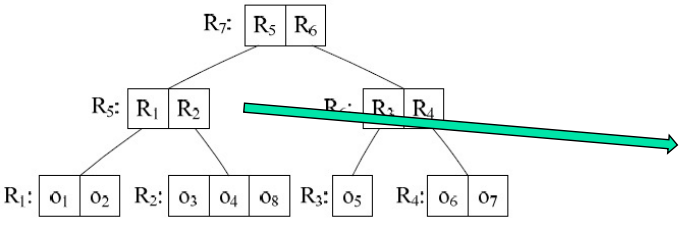
(c) R-tree for objects in Fig. 1(a)

R*-IF Index

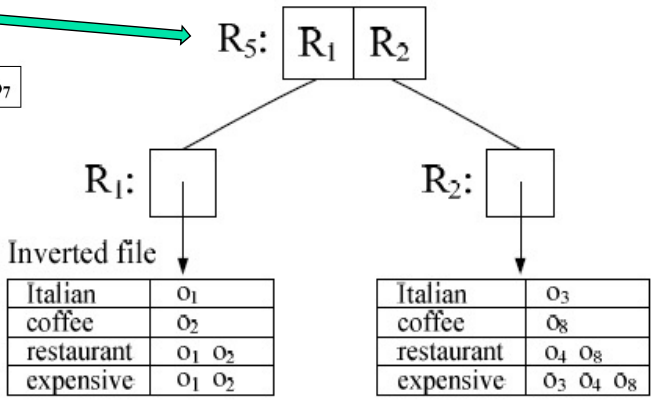
Object	Terms and term frequencies
o_1	(Italian, 5)(restaurant, 5)(expensive, 2)
o_2	(coffee, 5)(restaurant, 5)(expensive, 1)
o_3	(Italian, 7)(pizza, 1)(expensive, 1)
o_4	(restaurant, 7)(pizza, 1)(expensive, 1)
o_5	(Italian, 4)(restaurant, 4)
o_6	(coffee, 4)(restaurant, 3)
o_7	(Italian, 1)(coffee, 1)(restaurant, 4)(pizza, 1)
o_8	(coffee, 3)(restaurant, 3)(expensive, 1)

(b) Text information of objects in (a)

R*-IF Index



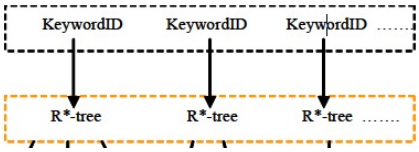
(c) R-tree for objects in Fig. 1(a)



IF-R* Index

- IF-R*= Inverted File - R*-tree
- IF-R* is a text-first geo-textual index.
  (IF-R* is counter part of R*=IF)

- For each distinct term t in D, a separate R*-tree is built for the objects in D containing term t.



IF-R* Index

Object	Terms and term frequencies
o_1	(Italian, 5)(restaurant, 5)(expensive, 2)
o_2	(coffee, 5)(restaurant, 5)(expensive, 1)
o_3	(Italian, 7)(pizza, 1)(expensive, 1)
o_4	(restaurant, 7)(pizza, 1)(expensive, 1)
o_5	(Italian, 4)(restaurant, 4)
o_6	(coffee, 4)(restaurant, 3)
o_7	(Italian, 1)(coffee, 1)(restaurant, 4)(pizza, 1)
o_8	(coffee, 3)(restaurant, 3)(expensive, 1)

(b) Text information of objects in (a)

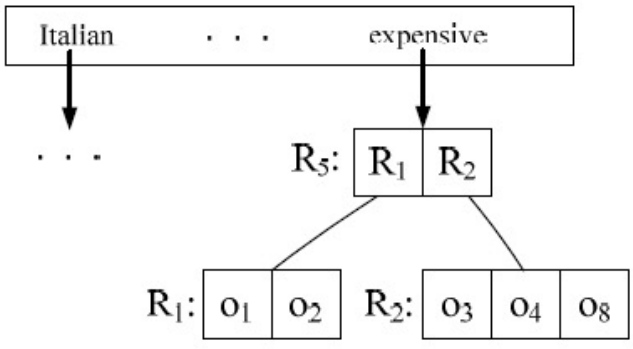
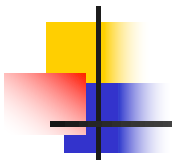
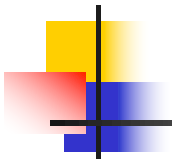


Figure 4: R-tree under the word *expensive*



R*-IF Index & IF-R* Index

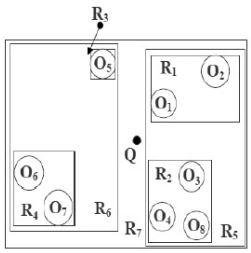
- R*-IF & IF-R*
  - ⇒ BkQ, BRQ
  - ⇒ It is shown that the IF-R* outperforms the R*-IF for the BRQ.
  - ⇒ There exists no sensitive way to use the two indices for the TkQ.



IR^2-Tree

- IR^2-tree uses bitmap.
- The fanout of the tree is dependent on the length of signature file.
- The signature file of a node is the union of all signatures of its entries, each representing a child node.

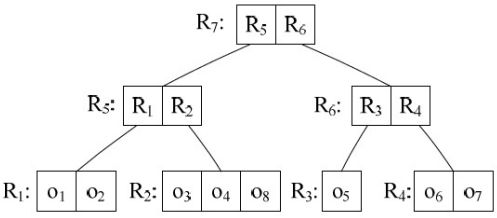
IR^2-Tree



(a) Objects and their bounding rectangles

Object	Terms and term frequencies
o_1	(Italian, 5)(restaurant, 5)(expensive, 2)
o_2	(coffee, 5)(restaurant, 5)(expensive, 1)
o_3	(Italian, 7)(pizza, 1)(expensive, 1)
o_4	(restaurant, 7)(pizza, 1)(expensive, 1)
o_5	(Italian, 4)(restaurant, 4)
o_6	(coffee, 4)(restaurant, 3)
o_7	(Italian, 1)(coffee, 1)(restaurant, 4)(pizza, 1)
o_8	(coffee, 3)(restaurant, 3)(expensive, 1)

(b) Text information of objects in (a)



(c) R-tree for objects in Fig. 1(a)

Figure 1: Example

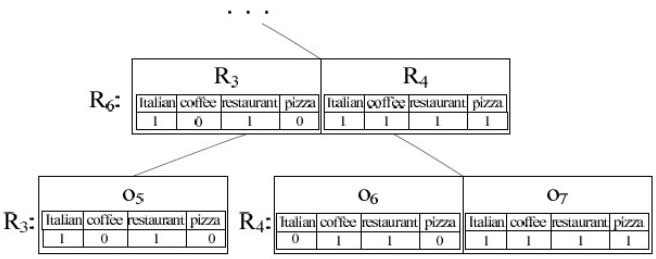


Figure 7: IR^2-tree with bitmaps

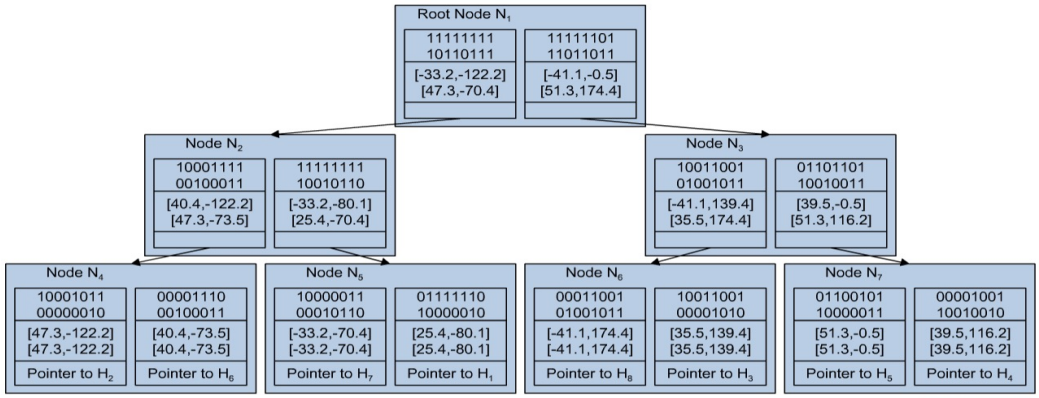
IR^2-tree and Algorithms

	Name	Latitude	Longitude	Amenities
H_1	Hotel A	25.4	-80.1	tennis court, gift shop, spa, Internet
H_2	Hotel B	47.3	-122.2	wireless Internet, pool, golf course
H_3	Hotel C	35.5	139.4	spa, continental suites, pool
H_4	Hotel D	39.5	116.2	sauna, pool, conference rooms
H_5	Hotel E	51.3	-0.5	dry cleaning, free lunch, pets
H_6	Hotel F	40.4	-73.5	safe box, concierge, internet, pets
H_7	Hotel G	-33.2	-70.4	Internet, airport transportation, pool
H_8	Hotel H	-41.1	174.4	wake up service, no pets, pool

Figure 1: Sample dataset of hotel objects.

IR^2-tree and Algorithms

Example: Execution of the IR²-Tree Algorithm on Distance-First Top-2 Spatial Keyword Query [30.5, 100.0] with keyword "internet" and "pool"



First we note that the signature for "internet" is: 00000010 00000000

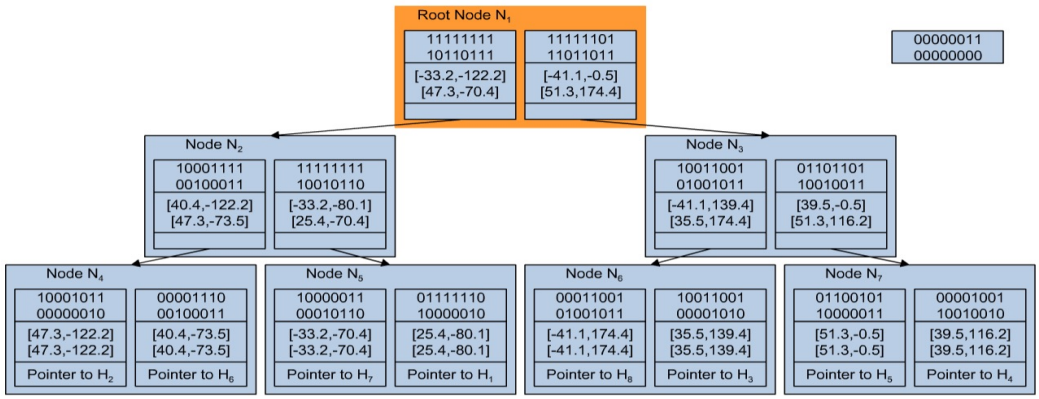
And the signature for "pool" is: 00000001 00000000

Therefore the query signature is: 00000011 00000000

Priority Queue:

IR^2-tree and Algorithms

Example: Execution of the IR²-Tree Algorithm on Distance-First Top-2 Spatial Keyword Query [30.5, 100.0] with keyword "internet" and "pool"

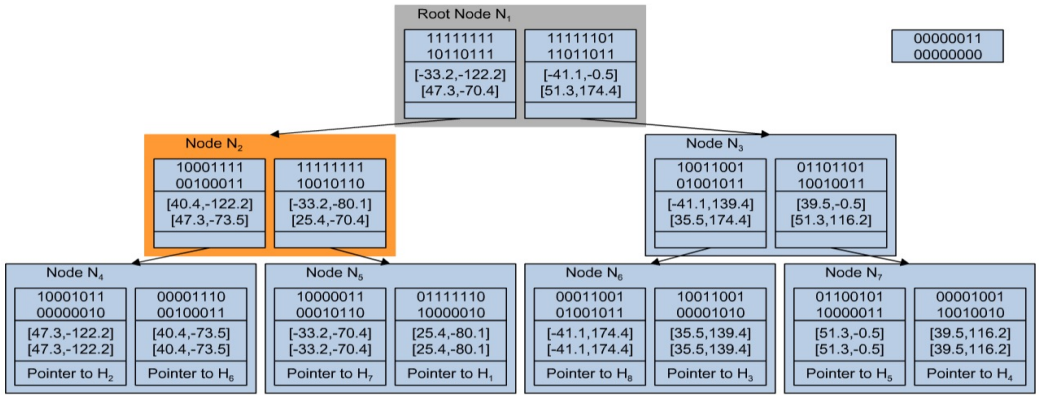


Enqueue N_1

Priority Queue: N_1, 0.0

IR^2-tree and Algorithms

Example: Execution of the IR²-Tree Algorithm on Distance-First Top-2 Spatial Keyword Query [30.5, 100.0] with keyword "internet" and "pool"

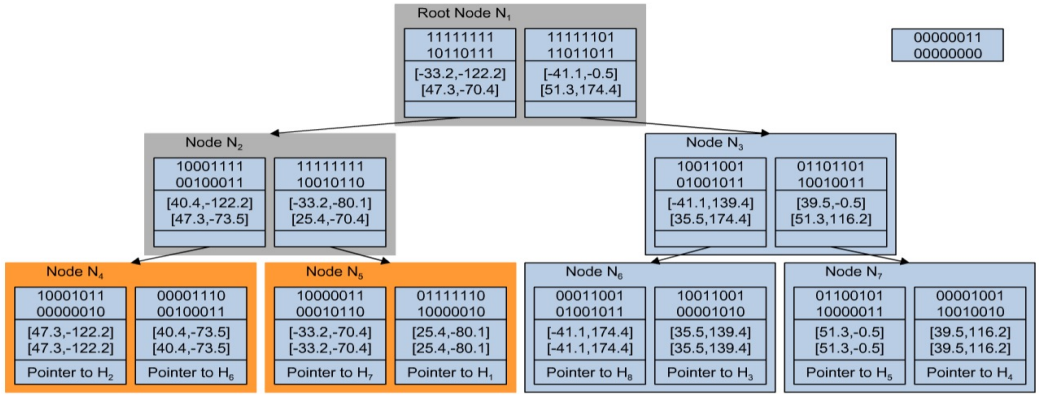


Enqueue N_2; note that N_3 is pruned

Priority Queue: | N_2, 170.4 |

IR^2-tree and Algorithms

Example: Execution of the IR²-Tree Algorithm on Distance-First Top-2 Spatial Keyword Query [30.5, 100.0] with keyword "internet" and "pool"

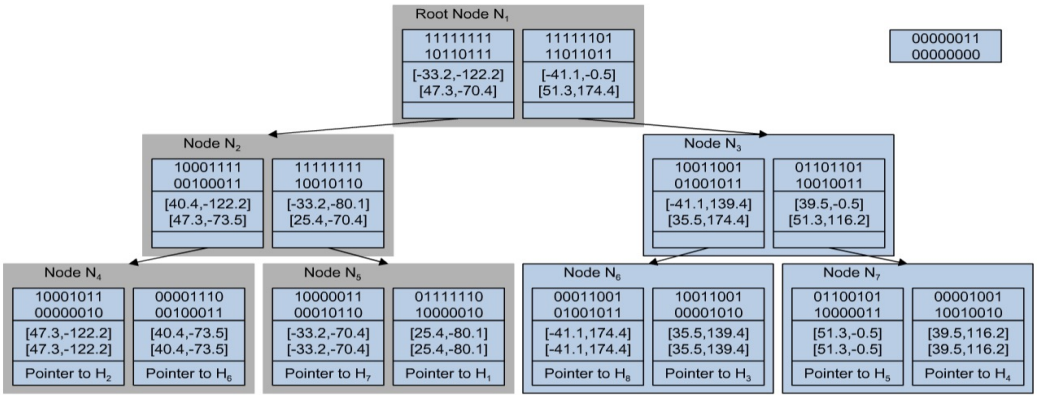


Enqueue N_4 and N_5

Priority Queue: | N_5, 170.5 | N_4, 173.8 |

IR^2-tree and Algorithms

Example: Execution of the IR²-Tree Algorithm on Distance-First Top-2 Spatial Keyword Query [30.5, 100.0] with keyword "internet" and "pool"

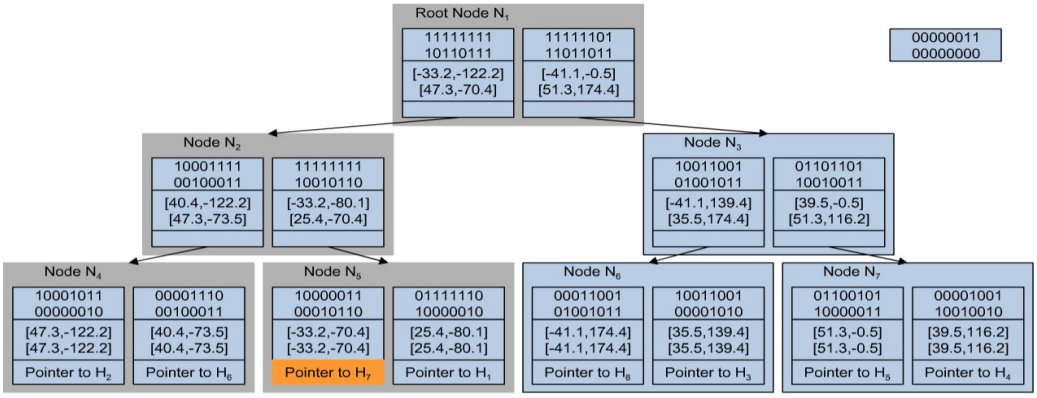


Enqueue H_7; note that H_1 is pruned

Priority Queue: | N_4, 173.8 |

IR^2-tree and Algorithms

Example: Execution of the IR²-Tree Algorithm on Distance-First Top-2 Spatial Keyword Query [30.5, 100.0] with keyword "internet" and "pool"

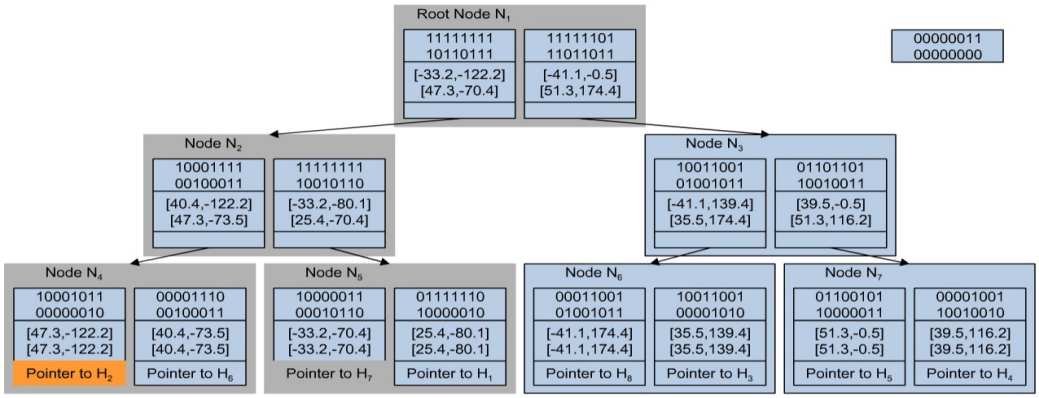


Enqueue H_7; note that H_1 is pruned

Priority Queue: | N_4, 173.8 | H_7, 181.9 |

IR^2-tree and Algorithms

Example: Execution of the IR²-Tree Algorithm on Distance-First Top-2 Spatial Keyword Query [30.5, 100.0] with keyword "internet" and "pool"



Enqueue H_2; note that H_6 is pruned

Priority Queue: | H_7, 181.9 | H_2, 222.8 |

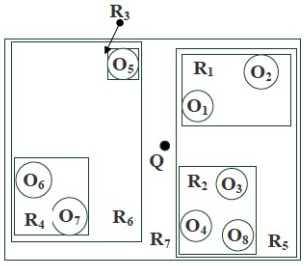
IR²-Tree

- IR²-tree can be used for processing the BkQ and BRQ.
- However, since the signature files do not have the frequency information, it cannot be used to process the TkQ.

IR-tree index

- Each node contains a pointer to an inverted file that describes the objects in the subtree rooted at the node
- The inverted file for a node X contains
  1) A vocabulary of all distinct terms
  2) A set of posting lists, each of which relates to a term t.

IR-tree index



Objects	Dist.	D_{ST}	Rectangles	Dist.	$MIND_{ST}$
O_1	2	0.238	R_1	2	0.238
O_2	5	0.512	R_2	2	0.1048
O_3	6	0.481	R_3	4	0.368
O_4	7	0.517	R_4	5	0.42
O_5	3	0.53	R_5	0.5	0.05119
O_6	9	0.58	R_6	1	0.269
O_7	8	0.55	R_7	0	0
O_8	8	0.686			

Figure 1: Eight Objects and Their Bounding Rectangles

$$M = \begin{array}{c} \\ Chinese \\ Spanish \\ restaurant \\ food \end{array} \begin{array}{c} \begin{array}{cccccccc} O_1.doc & O_2.doc & O_3.doc & O_4.doc & O_5.doc & O_6.doc & O_7.doc & O_8.doc \end{array} \\ \begin{pmatrix} 5 & 0 & 7 & 0 & 4 & 0 & 1 & 0 \\ 0 & 5 & 0 & 0 & 0 & 4 & 1 & 3 \\ 5 & 5 & 0 & 7 & 4 & 3 & 4 & 3 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \end{pmatrix} \end{array}$$

M=

	$O_1.doc$	$O_2.doc$	$O_3.doc$	$O_4.doc$	$O_5.doc$	$O_6.doc$	$O_7.doc$	$O_8.doc$
$Chinese$	5	0	7	0	4	0	1	0
$Spanish$	0	5	0	0	0	4	1	3
$restaurant$	5	5	0	7	4	3	4	3
$food$	0	0	1	1	0	0	1	0

IR-tree index

Objects	Dist.	D_{ST}	Rectangles	Dist.	$MIND_{ST}$
O_1	2	0.238	R_1	2	0.238
O_2	5	0.512	R_2	2	0.1048
O_3	6	0.481	R_3	4	0.368
O_4	7	0.517	R_4	5	0.42
O_5	3	0.53	R_5	0.5	0.05119
O_6	9	0.58	R_6	1	0.269
O_7	8	0.55	R_7	0	0
O_8	8	0.686			

Vocabulary	InvFile 4	InvFile 5	InvFile 6	InvFile 7
Chinese	<O1.doc, 5>	<O3.doc,7>	<O5.doc, 4>	<O7.doc, 1>
Spanish	<O2.doc, 5>	<O8.doc,3>		<O6.doc, 4>, <O7.doc, 1>
restaurant	<O1.doc, 5>, <O2.doc, 5>	<O4.doc, 7>, <O8.doc, 3>	<O5.doc, 4>	<O6.doc, 3>, <O7.doc, 4>
food		<O3.doc, 1>, <O4.doc, 1>		<O7.doc,1>

Vocabulary	InvFile 2	InvFile 3	InFile 1
Chinese	<R1.doc, 5>, <R2.doc, 7>	<R3.doc, 4>, <R4.doc, 1>	<R5.doc, 7>, <R6.doc,4>
Spanish	<R1.doc, 5>, <R2.doc, 3>	<R4.doc, 4>	<R5.doc, 5>, <R6.doc, 4>
restaurant	<R1.doc, 5>, <R2.doc, 7>	<R3.doc, 4>, <R4.doc, 4>	<R5.doc,7>, <R6.doc,4>
food	<R2.doc, 1>	<R4.doc, 1>,	<R5.doc,1>, <R6.doc,1>

$$D_{ST}(Q,O) = \alpha\frac{D_\varepsilon(Q.loc, O.loc)}{maxD} + (1-\alpha)(1 - \frac{\mathrm{P}(Q.keywords|O.doc)}{maxP}),$$

$$MIND_{ST}(Q,N) = \alpha\frac{MIND_\varepsilon(Q.loc, N.rectangle)}{maxD} + (1-\alpha)(1 - \frac{\mathrm{P}(Q.keywords|N.doc)}{maxP}),$$

$$\mathrm{P}(Q.keywords|O.doc) = \prod_{t \in Q.keywords} \hat{p}(t|\theta_{O.doc})$$

$$\hat{p}(t|\theta_{O.doc}) = (1-\lambda)\frac{tf(t,O.doc)}{|O.doc|} + \lambda\frac{tf(t,Coll)}{|Coll|}$$

$$maxP = \prod_{t \in Q.keywords} \max_{O' \in D} \hat{p}(t|O'.doc)$$

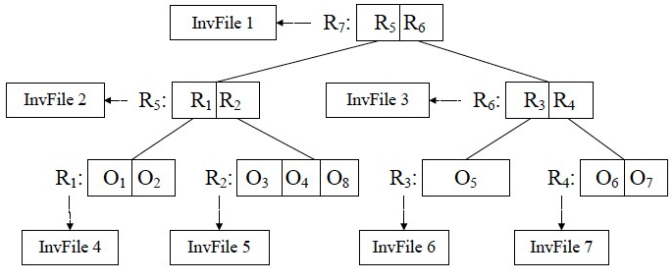


Figure 2: Hybrid Index in the Framework

Algorithm 2 $LkT(Query, Index, k)$

```
1:  Queue ← NewPriorityQueue();
2:  Queue.Enqueue(Index.RootNode, 0);
3:  while not Queue.IsEmpty() do
4:      Element ← Queue.Dequeue();
5:      if Element is an object then
6:          if not Queue.IsEmpty() and D_ST(Query, Object) >
                Queue.First().Key then
7:              Queue.Enqueue(Object, D_ST(Query, Object));
8:          else
9:              Report Element as the next nearest object;
10:             if k nearest objects have been found then
11:                 break;
12:     else if Element is a leaf node then
13:         for each entry(Object) in leaf node Element do
14:             Queue.Enqueue(Object, D_ST(Query, Object));
15:     else
16:         for each entry(Node) in node Element do
17:             Queue.Enqueue(Node, MIND_ST(Query, Node));
```