A Survey on Spatial-Keyword Search

(COMP 6311C Advanced Data Management)

Nikolaos Armenatzoglou

06/03/2012
Outline

• Problem Definition and Motivation

• Query Types

• Query Processing Techniques
  – Indices
  – Algorithms
  – Strengths and Weaknesses

• Conclusion
A Survey on Spatial-Keyword Search

PROBLEM DEFINITION AND MOTIVATION
Problem Definition and Motivation

**Definition: Spatio-Textual Object (STO)**

A STO $o$ is an object that consist of a textual description ($o.text$) and a spatial location ($o.loc$).

**Example**

```
Hong Kong Times Square
```

```
$o_1$ ("Chinese Restaurant")
$o_2$ ("Italian Ristorante")
$o_3$ ("Restaurante de France")
$o_4$ ("Italian Ristorante")
$o_5$ ("Greek Restaurant")
$o_6$ ("Chinese Restaurant")
$o_7$ ("Shoes")
$o_8$ ("Athletic Shoes")
$o_9$ ("Cinema")
$o_{10}$ ("bookstore")
```


Problem Definition and Motivation

**Definition: Spatial-Keyword Search (SKQ)**

Given a spatial constraint $Q.loc$ and a set of keywords $Q.text$, a SKQ $Q$ returns a list (ranked or not) of the STOs whose location satisfies $Q.loc$ and whose textual description is relevant to $Q.text$. 

![Diagram showing STOs with keywords and location]

*Hong Kong Times Square*
Definition: Spatial-Keyword Search (SKQ)

Given a spatial constraint $Q.loc$ and a set of keywords $Q.text$, a SKQ $Q$ returns a list (ranked or not) of the STOs whose location satisfies $Q.loc$ and whose textual description is relevant to $Q.text$. 

Hong Kong Times Square
A Survey on Spatial-Keyword Search

QUERY TYPES
Query Types

• Basic
  – Spatial-Keyword Range Query (SKR)
  – Spatial-Keyword Nearest Neighbor Query (SKNN)
  – Top-k Spatial Keyword Query (Top-k SK)

• Variations
  – Approximation Spatial-Keyword Range Query (ASK Range)
  – Top-k Spatial Boolean query (kSB)
  – Spatial-Keyword Auto-Completion (SK AC)
  – ...

Query Types (Basic)
Spatial-Keyword Range Query (SKR)

Given a region $Q.loc$ and a set of keywords $Q.text$, a SKR query $Q$ returns all STOs within $Q.loc$ that contain all keywords in $Q.text$.

Example

$Q.loc = [(200, 0), (500, 300)]$ and $Q.text = \{“Chinese”, “Restaurant”\}.
Ans(Q) = \{o_1, o_6\}$
Query Types (Basic)
Spatial-Keyword Range Query (SKR)

Given a region $Q.loc$ and a set of keywords $Q.text$, a SKR query $Q$ returns all STOs within $Q.loc$ that contain all keywords in $Q.text$.

**Example**

$Q.loc = [(200, 0), (500, 300)]$ and $Q.text = \{"Chinese", "Restaurant"\}$.

$Ans(Q) = \{o_1, o_6\}$
Query Types (Basic)
Spatial-Keyword Nearest Neighbor Query (SKNN)

Given a query point $Q.loc$, a set of keywords $Q.text$ and a positive integer $k$, a SKNN query $Q$ returns the $k$ nearest STOs to $Q.loc$, whose textual description contains all keywords in $Q.text$.

Example

$Q.loc = (500, 200)$, $Q.text = \{"Restaurant", "Chinese"\}$, $k = 1$.
Ans($Q$): $\{o_1\}$. 
Query Types (Basic)
Spatial-Keyword Nearest Neighbor Query (SKNN)

Given a query point $Q.loc$, a set of keywords $Q.text$ and a positive integer $k$, a SKNN query $Q$ returns the $k$ nearest STOs to $Q.loc$, whose textual description contains all keywords in $Q.text$.

**Example**

$Q.loc = (500, 200)$, $Q.text = \{“Restaurant”, “Chinese”\}$, $k = 1$.

Ans($Q$): $\{o_1\}$. 
Query Types (Basic)
Top-k Spatial Keyword Query (Top-k SK)

Given a query point $Q.loc$, a set of keywords $Q.text$ and a positive integer $k$, a Top-k SK query $Q$ returns a list of $k$ STOs ranked according to a ranking function $f$ based on their spatial proximity to $Q.loc$ and on their textual similarity to $Q.text$.

Example

$$f = a f_{IR} + (1-a) f_{SP},$$ where $a \in (0, 1)$, $f_{IR}$ : text similarity and $f_{SP}$: spatial proximity.

$Q.loc = (500, 200)$, $Q.text = \{“restaurant”\}$, $k = 2$ and $a = 0.5$.

$Ans(Q) = \{o_2, o_1\}$. 
Query Types (Basic)
Top-$k$ Spatial Keyword Query ($\text{Top}-k \text{ SK}$)

Given a query point $Q.loc$, a set of keywords $Q.text$ and a positive integer $k$, a $\text{Top}-k \text{ SK}$ query $Q$ returns a list of $k$ STOs ranked according to a ranking function $f$ based on their spatial proximity to $Q.loc$ and on their textual similarity to $Q.text$.

Example

$$f = a f_{IR} + (1-a) f_{SP},$$
where $a \in (0, 1)$, $f_{IR}$: text similarity and $f_{SP}$: spatial proximity.

$Q.loc = (500, 200)$, $Q.text = \{\text{“restaurant”}\}$, $k = 2$ and $a = 0.5$.

$\text{Ans}(Q) = \{o_2, o_1\}$. 

Query Types (Variations)
Approximate Spatial-Keyword Range Query (ASK Range)

Given a region $Q.loc$, a set of keywords $Q.text$ and a positive integer $t$, an ASK Range query $Q$ returns the STOs in $Q.loc$ that contain similar strings to all keywords in $Q.text$ with edit-distance threshold $t$.

Example

$Q.loc = [(200, 0), (500, 300)]$, $Q.text = \{“Restaurant”\}$ and $t = 1$. 
$Ans(Q) = \{o_1, o_3, o_6\}$
Query Types (Variations)

Approximate Spatial-Keyword Range Query (ASK Range)

Given a region \(Q.loc\), a set of keywords \(Q.text\) and a positive integer \(t\), an ASK Range query \(Q\) returns the STOs in \(Q.loc\) that contain similar strings to all keywords in \(Q.text\) with edit-distance threshold \(t\).

**Example**

\(Q.loc = [(200, 0), (500, 300)]\), \(Q.text = \{\text{“Restaurant”}\}\) and \(t = 1\).

\(Ans(Q) = \{o_1, o_3, o_6\}\)
Query Types (Variations)

Top-\(k\) Spatial Boolean query (\(kSB\))

Given a query point \(Q.loc\), a Boolean expression \(e\) on some query keywords and a positive integer \(k\), a \(kSB\) query \(Q\) returns the \(k\) nearest STOs whose textual description satisfies \(e\).

Example

\[Q.loc = (400, 300), \ e = \text{“restaurant”} \land \neg \text{“Chinese”} \text{ and } k = 2.\]

\(Ans(Q)\): \(\{o_5, o_2\}\).
Query Types (Variations)
Top-k Spatial Boolean query (kSB)

Given a query point $Q.loc$, a Boolean expression $e$ on some query keywords and a positive integer $k$, a $kSB$ query $Q$ returns the $k$ nearest STOs whose textual description satisfies $e$.

Example

$Q.loc = (400, 300)$, $e = \text{“restaurant”} \land \neg \text{“Chinese”}$ and $k = 2$. 
Ans($Q$): $\{o_5, o_2\}$. 

$0 \quad 100 \quad 200 \quad 300 \quad 400 \quad 500 \quad 600 \quad 700$

$0 \quad 100 \quad 200 \quad 300$ 

$\bullet o_7(\text{“Shoes”})$ 
$\bullet o_9(\text{“Cinema”})$ 
$\bullet o_{10}(\text{“bookstore”})$ 
$\bullet o_6(\text{“Chinese Restaurant”})$ 
$\bullet o_4(\text{“Restaurante de France”})$ 
$\bullet o_3(\text{“Chinese Restaurant”})$ 
$\bullet o_8(\text{“Athletic Shoes”})$ 
$\bullet o_3(\text{“Italian Restaurant”})$ 
$\bullet o_2(\text{“Italian Ristorante”})$ 
$Q.loc$
Query Types (Variations)
Spatial-Keyword Auto-Completion (SK AC)

Given a point $Q.loc$, a string $Q.text$ and a positive integer $k$, a SK AC query $Q$ returns a list of $k$ STOs, where $Q.text$ is a prefix of their textual description, ranked according to a ranking function $f$ that is based on their spatial proximity to $p$ and several other static scores like popularity and ratings.

Example

Static scores are 1 for all STOs. $Q.loc = (300, 0)$, $Q.text = “rest”, k = 3$. $Ans(Q): \{o_6, o_3, o_1\}$
Query Types (Variations)
Spatial-Keyword Auto-Completion (SK AC)

Given a point $Q.loc$, a string $Q.text$ and a positive integer $k$, a SK AC query $Q$ returns a list of $k$ STOs, where $Q.text$ is a prefix of their textual description, ranked according to a ranking function $f$ that is based on their spatial proximity to $p$ and several other static scores like popularity and ratings.

Example

Static scores are 1 for all STOs.
$Q.loc = (300, 0)$, $Q.text = “rest”, k = 3.$
Ans($Q$): $\{o_6, o_3, o_1\}$
## Query Types

<table>
<thead>
<tr>
<th>Query Type</th>
<th>Spatial Categorization</th>
<th>IR Categorization</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASK Range</td>
<td>Range</td>
<td>Proximity Query</td>
</tr>
<tr>
<td>SKR</td>
<td></td>
<td>Contains all keywords</td>
</tr>
<tr>
<td>SKNN</td>
<td></td>
<td>Proximity Query</td>
</tr>
<tr>
<td>Joint SKNN</td>
<td></td>
<td>Boolean Query Model</td>
</tr>
<tr>
<td>ASK kNN</td>
<td>kNN</td>
<td>Auto-completion</td>
</tr>
<tr>
<td>kSB</td>
<td></td>
<td>Ranked Query Model</td>
</tr>
<tr>
<td>SK AC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Top-k SK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LkPT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MkSK</td>
<td>Moving kNN</td>
<td></td>
</tr>
<tr>
<td>RSTkNN</td>
<td>Reverse kNN</td>
<td></td>
</tr>
<tr>
<td>mCK</td>
<td>Aggregate</td>
<td>Contains at least one keyword</td>
</tr>
</tbody>
</table>
A Survey on Spatial-Keyword Search

QUERY PROCESSING TECHNIQUES
Query Processing Techniques
On Answering SKR Query

Indexing Approach

Separate indices for spatial and textual attributes

[ Martins et al. GIR, 2005 ]
Query Processing Techniques
On Answering SKR Query

SI: Separate Indices Approach

**1\(^\text{st}\) Approach**
- \(\text{Ans}_{SP}(Q.loc)\) = Answer to spatial constrains.
- \(\text{Ans}_{IR}(Q.text)\) = Answer to textual constrains.
- \(\text{Ans}(Q) = \text{Ans}_{SP}(Q.loc) \cap \text{Ans}_{IR}(Q.text)\)

**2\(^\text{nd}\) Approach**
- \(\text{Ans}_{SP}(Q.loc)\) = Answer to spatial constrains.
- For each \(o \in \text{Ans}_{SP}(Q.loc)\) check if \(o\) satisfies the textual constraints.

**Query Processing**

[Martins et al. GIR, 2005]

**Pros**
- it is easy to maintain the two indices,
- it supports both textual, spatial and SKQ queries and
- updates and optimizations are performed in each index independently.

**Cons**
- the pruning power of query processing algorithms is poor and
- leads to high I/O cost.
Query Processing Techniques
On Answering SKR Query

Indexing Approach

Inverted Files + Grids: words’ occurrences contains spatial information.

[TS: Textual - Spatial Approach]

[Vaid et al. SSTD, 2005]

<table>
<thead>
<tr>
<th>Word</th>
<th>Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Athletic</td>
<td>$(c_{3,4}, {o_8})$</td>
</tr>
<tr>
<td>Bookstore</td>
<td>$(c_{1,2}, {o_{10}})$</td>
</tr>
<tr>
<td>Chinese</td>
<td>$(c_{4,3}, {o_1}), (c_{3,1}, {o_6})$</td>
</tr>
<tr>
<td>Cinema</td>
<td>$(c_{1,3}, {o_9})$</td>
</tr>
<tr>
<td>France</td>
<td>$(c_{4,2}, {o_3})$</td>
</tr>
<tr>
<td>Greek</td>
<td>$(c_{3,4}, {o_5})$</td>
</tr>
<tr>
<td>Italian</td>
<td>$(c_{6,3}, {o_2}), (c_{5,1}, {o_4})$</td>
</tr>
<tr>
<td>Restaurant</td>
<td>$(c_{4,3}, {o_1}), (c_{6,3}, {o_2}), (c_{3,4}, {o_5}), (c_{3,1}, {o_6})$</td>
</tr>
<tr>
<td>Restaurante</td>
<td>$(c_{4,2}, {o_3})$</td>
</tr>
<tr>
<td>Ristorante</td>
<td>$(c_{5,1}, {o_4})$</td>
</tr>
<tr>
<td>Shoes</td>
<td>$(c_{1,4}, {o_7}), (c_{3,4}, {o_8})$</td>
</tr>
</tbody>
</table>
Query Processing Techniques
On Answering SKR Query

Indexing Approach

Inverted Files + Grids: words’ occurrences contains spatial information.

TS: Textual - Spatial Approach

[Vaid et al. SSTD, 2005]

<table>
<thead>
<tr>
<th>Word</th>
<th>Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Athletic</td>
<td>(c_{3,4}, {o_8})</td>
</tr>
<tr>
<td>Bookstore</td>
<td>(c_{1,2}, {o_{10}})</td>
</tr>
<tr>
<td>Chinese</td>
<td>(c_{4,3}, {o_1}), (c_{3,1}, {o_6})</td>
</tr>
<tr>
<td>Cinema</td>
<td>(c_{1,3}, {o_9})</td>
</tr>
<tr>
<td>France</td>
<td>(c_{4,2}, {o_3})</td>
</tr>
<tr>
<td>Greek</td>
<td>(c_{3,4}, {o_5})</td>
</tr>
<tr>
<td>Italian</td>
<td>(c_{6,3}, {o_2}), (c_{5,1}, {o_4})</td>
</tr>
<tr>
<td>Restaurant</td>
<td>(c_{4,3}, {o_1}), (c_{6,3}, {o_2}), (c_{3,4}, {o_5}), (c_{3,1}, {o_6})</td>
</tr>
<tr>
<td>Restaurante</td>
<td>(c_{4,2}, {o_3})</td>
</tr>
<tr>
<td>Ristorante</td>
<td>(c_{5,1}, {o_4})</td>
</tr>
<tr>
<td>Shoes</td>
<td>(c_{1,4}, {o_7}), (c_{3,4}, {o_8})</td>
</tr>
</tbody>
</table>
Query Processing Techniques
On Answering SKR Query

**TS: Textual - Spatial Approach**

**Query Processing**

- **Approach**
  - \( A = \{c_{cid} \mid R_{cid} \cap Q.loc \neq \emptyset\} \)
  - For each keyword \( t \in Q.text \) get the STOs such that their cell id is in \( A \).
  - Finally, get the intersection.

**Example:** \( Q.loc = [(200, 0), (400, 200)] \) and \( Q.text = \{“Chinese”, “Restaurant”\} \).

\[ A = \{c_{3,1}, c_{3,2}, c_{4,1}, c_{4,2}\}. \]

**Evaluation:** Better performance than SI approach.

<table>
<thead>
<tr>
<th>Word</th>
<th>Occurrences</th>
</tr>
</thead>
<tbody>
<tr>
<td>Athletic</td>
<td>((c_{3,4}, {o_8}))</td>
</tr>
<tr>
<td>Bookstore</td>
<td>((c_{1,2}, {o_{10}}))</td>
</tr>
<tr>
<td><strong>Chinese</strong></td>
<td>((c_{4,3}, {o_1}), (c_{3,4}, {o_6}))</td>
</tr>
<tr>
<td>Cinema</td>
<td>((c_{1,3}, {o_9}))</td>
</tr>
<tr>
<td>France</td>
<td>((c_{4,2}, {o_3}))</td>
</tr>
<tr>
<td>Greek</td>
<td>((c_{3,4}, {o_5}))</td>
</tr>
<tr>
<td>Italian</td>
<td>((c_{6,3}, {o_2}), (c_{5,1}, {o_4}))</td>
</tr>
<tr>
<td><strong>Restaurant</strong></td>
<td>((c_{4,3}, {o_1}), (c_{6,3}, {o_2}), (c_{3,4}, {o_5}), (c_{3,4}, {o_6}))</td>
</tr>
<tr>
<td>Restaurante</td>
<td>((c_{4,2}, {o_3}))</td>
</tr>
<tr>
<td>Ristorante</td>
<td>((c_{5,1}, {o_4}))</td>
</tr>
<tr>
<td>Shoes</td>
<td>((c_{1,4}, {o_7}), (c_{3,4}, {o_8}))</td>
</tr>
</tbody>
</table>
Query Processing Techniques
On Answering SKR Query

Indexing Approach

Hybrid Index: Inverted File $\rightarrow$ R*-Trees, R*-Tree $\rightarrow$ Inverted Files

[Zhou et al. CIKM, 2005]
Query Processing Techniques
On Answering SKR Query

**Query Processing**

**Example:** \( Q.loc = [(200, 0), (500, 300)] \) and \( Q.text = \{ \text{“Chinese”, “Restaurant”} \} \).
Query Processing Techniques
On Answering SKR Query

R*-Tree → Inverted Files & Inverted File → R*-Trees

• **Pros**
  • They reduce I/O cost and query processing.
  • The Inverted File → R*-Tree outperforms the R*-Tree → Inverted File.

• **Cons**
  • Inverted File → R*-Tree: It does not exploit the spatial distribution of keywords.
  • R*-Tree → Inverted File: The pruning process is not powerful as we can get a lot of candidates from spatial restrictions.
Query Processing Techniques
On Answering SKR Query

Indexing Approach

R*-Tree where the internal nodes are virtually augmented with textual information

KR*-Tree: Keyword R*-Tree Approach

[Hariharan et al. SSDBM, 2007]

<table>
<thead>
<tr>
<th>Word</th>
<th>Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Athletic</td>
<td>$n_0, n_1, n_4$</td>
</tr>
<tr>
<td>Bookstore</td>
<td>$n_0, n_1, n_3$</td>
</tr>
<tr>
<td>Chinese</td>
<td>$n_0, n_2, n_5, n_6$</td>
</tr>
<tr>
<td>Cinema</td>
<td>$n_0, n_3, n_3$</td>
</tr>
<tr>
<td>France</td>
<td>$n_0, n_2, n_5$</td>
</tr>
<tr>
<td>Greek</td>
<td>$n_0, n_1, n_4$</td>
</tr>
<tr>
<td>Italian</td>
<td>$n_0, n_2, n_5, n_6$</td>
</tr>
<tr>
<td>Restaurant</td>
<td>$n_0, n_2, n_4, n_5, n_6$</td>
</tr>
<tr>
<td>Restaurante</td>
<td>$n_0, n_2, n_5$</td>
</tr>
<tr>
<td>Ristorante</td>
<td>$n_0, n_2, n_6$</td>
</tr>
<tr>
<td>Shoes</td>
<td>$n_0, n_1, n_3, n_4$</td>
</tr>
</tbody>
</table>
Query Processing Techniques
On Answering SKR Query

**Example:** $Q.\text{loc} = [(200, 0), (500, 300)]$ and $Q.\text{text} = \{\text{“Chinese”, “Restaurant”}\}$.

<table>
<thead>
<tr>
<th>Word</th>
<th>Nodes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Athletic</td>
<td>$n_0, n_1, n_4$</td>
</tr>
<tr>
<td>Bookstore</td>
<td>$n_0, n_1, n_3$</td>
</tr>
<tr>
<td>Chinese</td>
<td>$n_0, n_2, n_5, n_6$</td>
</tr>
<tr>
<td>Cinema</td>
<td>$n_0, n_1, n_3$</td>
</tr>
<tr>
<td>France</td>
<td>$n_0, n_2, n_5$</td>
</tr>
<tr>
<td>Greek</td>
<td>$n_0, n_1, n_4$</td>
</tr>
<tr>
<td>Italian</td>
<td>$n_0, n_2, n_5, n_6$</td>
</tr>
<tr>
<td>Restaurant</td>
<td>$n_0, n_1, n_2, n_4, n_5, n_6$</td>
</tr>
<tr>
<td>Restaurante</td>
<td>$n_0, n_2, n_5$</td>
</tr>
<tr>
<td>Ristorante</td>
<td>$n_0, n_2, n_6$</td>
</tr>
<tr>
<td>Shoes</td>
<td>$n_0, n_1, n_3, n_4$</td>
</tr>
</tbody>
</table>

Experimental evaluation: More efficient than the previous approaches.
Query Processing Techniques
On Answering SKNN Query

Indexing Approach

R-Tree & Signature Files

IR²-Tree: Information Retrieval R-Tree Approach

[De Felipe et al. ICDE, 2008]

Signature Files

Block1: A text has many words

• Hash(text) = 000101
• Hash(many) = 110000
• Hash(words) = 100100

OR

Signature (Block1) = 110101

False Positives:

Block2: A text has many

Signature (Block2) = 110101

“words” ∈ Block2 ✗
Query Processing Techniques
On Answering \( SKNN \) Query

**IR^2-Tree Approach**

**Query Processing**

Example: \( Q.loc = (500, 200) \), \( Q.text = \{"restaurant", "Chinese"\} \) and \( k = 1 \).

Signature(\( Q.text \)): 000110000000

Priority Queue: \( N_0, 0.0 \)
Query Processing Techniques

On Answering SKNN Query

**IR²-Tree Approach**

---

**Example:** $Q.loc = (500, 200)$, $Q.text = \{"restaurant","Chinese"\}$ and $k = 1$.

**Signature**($Q.text$): $000110000000$

Priority Queue: $N_2, 0.0$
Query Processing Techniques
On Answering SKNN Query

IR²-Tree Approach

Query Processing

Example: $Q.loc = (500, 200)$, $Q.text = \{“restaurant”, “Chinese”\}$ and $k = 1$.

Signature($Q.text$): 000110000000

Priority Queue: $N_5, 0.0$  $N_6, 161.2$
Query Processing Techniques
On Answering SKNN Query

IR²-Tree Approach

Example: $Q.loc = (500, 200)$, $Q.text = \{“restaurant”, “Chinese”\}$ and $k = 1$.

Signature($Q.text$): 000110000000

Priority Queue: $N_6, 161.2$ $o_1, 174.6$
Query Processing Techniques
On Answering SKNN Query

**IR²-Tree Approach**

**Query Processing**

**Example:** $Q.loc = (500, 200)$, $Q.text = \{\text{“restaurant”}, \text{“Chinese”}\}$ and $k = 1$.

**Signature($Q.text$):** 000110000000

**Priority Queue:** $o_1, 174.6$ $o_6, 258.1$
Query Processing Techniques
On Answering SKNN Query

• **Multi-level IR²-Tree (MIR²-Tree)**
  – Uses different signature lengths for different levels.
  – More complex update operations
  – Fewer False Positives

• **Experimental evaluation:** MIR²-Tree is more efficient than IR²-Tree.

• False Positives.

• No comparison with other indices.
Query Processing Techniques
On Answering Top-k SK Query

Indexing Approach

R-Tree where the internal nodes are augmented with textual & scoring information

IR-Tree: Inverted file R-Tree Approach

[Cong et al. PVLDB, 2009]
Query Processing Techniques

On Answering Top-\(k\) SK Query

IR-Tree Approach

- Similar to the IR\(^2\)-Tree query processing algorithm (Branch-and-Bound)
- They propose two metrics for scoring rectangles and objects.

\[
MIND_{ST}(Q, N) \quad D_{ST}(Q, O)
\]

Both metrics are based on Euclidean distances and IR scores.

Shortcoming

Inverted file \(I_1\)

- \(o_1\): “restaurant”, weight 5
- \(o_2\): “Chinese”, weight 3
- \(o_3\): “restaurant”, weight 4
- \(o_3\): “Chinese”, weight 2
Query Processing Techniques
On Answering Top-k SK Query

Indexing Approach

Enhanced indexing: At construction time, take into account both the area parameter and the textual similarity between the indexed objects.

DIR-Tree: Document Inverted file R-Tree Approach [Cong et al. PVLDB, 2009]

Area cost of inserting STO o into rectangle R:

IR-Tree (similar to R-Tree)

\[ \text{AreaCost}(R) = \text{area}(R') - \text{area}(R) \]

where \( \text{area}(R') \) is the area of \( R \) after inserting \( o \)

DIR-Tree

\[ \text{SimAreaCost}(R, o) = a \frac{\text{AreaCost}(R)}{\max\text{Area}} + (1 - a)(1 - \text{IRsim}(R.\text{text}, o.\text{text})) \]

where \( a \in (0, 1) \)

Query Processing: Similar to IR-Tree.
Experimental Evaluation: More efficient.
Query Processing Techniques
On Answering *Top-k SK* Query

**Indexing Approach**

Enhanced indexing: Enhance IR-Tree with clustering.

CIR-Tree: Clustering Inverted file R-Tree Approach

[Cong et al. *PVLDB*, 2009]

**Goal:** Tighter bounds for rectangles i.e., $MIND_{ST}$.

- STO where word $w \in o.text$
- STO where word $w \not\in o.text$

$N$ (node of IR-Tree)

**Query Processing:** Similar to IR-Tree.

**Experimental Evaluation:** Most efficient.
Query Processing Techniques
On Answering \textit{Top-k SK} Query

**Indexing Approach**

Enhanced indexing: Enhance IR-Tree with clustering.

**CIR-Tree: Clustering Inverted file R-Tree Approach**

[Cong et al. \textit{PVLDB}, 2009]

Goal: Tighter bounds for rectangles i.e., \( MIND_{ST} \).

\begin{itemize}
  \item STO where word \( w \in o.text \)
  \item STO where word \( w \notin o.text \)
\end{itemize}

Query Processing: Similar to IR-Tree.
Experimental Evaluation: Most efficient.
Query Processing Techniques
On Answering Top-\(k\) SK Query

Indexing Approach

R-Tree where the internal nodes are augmented with textual & scoring information.

IR-Tree2

[Li et al. TKDE, 2010]
Query Processing Techniques

On Answering Top-$k$ SK Query

IR-Tree2

**Query Processing**

! $Q.loc = \text{region, spatial proximity to the center of } Q.loc$

**Step 1:** Compute $idf = \frac{D_{Q.loc}}{D_{Q.loc,t}}$, where

- $D_{Q.loc}$ is the number of STOs located in $Q.loc$ and
- $D_{Q.loc,t}$ is the number of STOs in $Q.loc$ s.t. $t \in o.text$.

**Step 2:** Similar to IR-Tree.
Query Processing Techniques
On Answering Top-k SK Query

Indexing Approach

Inverted File $\rightarrow$ Aggregate Spatial Index

S2I: Spatial Inverted Index Approach

[Rocha-Junior et al. SSTD, 2011]
Query Processing Techniques
On Answering Top-k SK Query

S2I Approach

Threshold Algorithm – Round Robin Strategy

$|Q.text| = 2, \ k = 1$

Let $S_i$ be a source responsible for the $i^{th}$ query keyword.
Given the $Q.loc$ and a positive integer $j$, $S_i$ returns the STO $o$ with the $j^{th}$ highest score.

<table>
<thead>
<tr>
<th>Iteration $j$</th>
<th>$S_1$</th>
<th>$S_2$</th>
<th>$UB_1$</th>
<th>$UB_2$</th>
<th>$UB_{unseen}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>$o_1$</td>
<td>$o_3$</td>
<td>0.7</td>
<td>0.6</td>
<td>1.3</td>
</tr>
<tr>
<td>2</td>
<td>$o_4$</td>
<td>$o_7$</td>
<td>0.6</td>
<td>0.5</td>
<td>1.1</td>
</tr>
<tr>
<td>3</td>
<td>$o_3$</td>
<td>$o_8$</td>
<td>0.2</td>
<td>0.45</td>
<td>0.65</td>
</tr>
<tr>
<td>4</td>
<td>$o_9$</td>
<td>$o_{10}$</td>
<td>0.15</td>
<td>0.09</td>
<td>0.24</td>
</tr>
</tbody>
</table>

Experimental Evaluation: More efficient than DIR-Tree.
No comparison with CIR-Tree.
CONCLUSION
Conclusion

• Spatial-keyword queries.
• Indexing schemes and the corresponding algorithms for query processing.
• Pros and cons of each proposed framework.

• There are still much left to cover on this topic
  – There is no experimental evaluation that compares the frameworks properly.
  – Spatial-keyword search on road networks.

• Combination of two indices!
Questions