Stochastic Driven Relational R-tree

GeoInfo 2003
Campos do Jordão - São Paulo

Hans-Peter Kriegel, Peter Kunath, Martin Pfeifle, Marco Pötke, Matthias Renz, Petra-Maria Strauß

Database Group
Institute for Computer Science
University of Munich, Germany
Outline of the Talk

1. Introduction
2. Relational Access Methods
3. Stochastic Basics
4. Stochastic driven Relational R-tree
5. Experimental Evaluation
6. Conclusions and Future Work
Introduction

New Database Applications:
- Geo Information Systems
- Multimedia Information Systems
- Genome Databases
- CAD Applications
- ...

New Object Types:
- geographic areas
- molecular structures
- ...

→ seamless integration of efficient index structures in commercial ORDBMs needed
Outline of the Talk

1. Introduction
2. Relational Access Methods
3. Stochastic Basics
4. Stochastic driven Relational R-tree
5. Experimental Evaluation
6. Conclusions and Future Work
A simple example for a relational access method: The MBR-list

User table: polygons

<table>
<thead>
<tr>
<th>id</th>
<th>geom</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>POLYGON((10,10), (25,15),..., (10,10))</td>
</tr>
<tr>
<td>B</td>
<td>POLYGON((30,15), (30,45),..., (30,15))</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Index table: polygons_mbr

<table>
<thead>
<tr>
<th>id</th>
<th>mbr</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>BOX((5,10),(13,15))</td>
</tr>
<tr>
<td>B</td>
<td>BOX((30,5),(40,50))</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Meta table: mbr_index_metadata

<table>
<thead>
<tr>
<th>index_name</th>
<th>user_table</th>
<th>index_table</th>
</tr>
</thead>
<tbody>
<tr>
<td>'polygons_idx'</td>
<td>'polygons'</td>
<td>'polygons_mbr'</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Relational mapping of an R-tree directory

Hierarchical directory

Relational index-table: \textit{polygons\_rtree}

\begin{tabular}{|c|c|c|c|}
\hline
\textbf{page\_id} & \textbf{page\_lev} & \textbf{son\_id} & \textbf{son\_mbr} \\
\hline
ROOT & 3 & 1 & BOX((0,0), (200,120)) \\
\hline
1 & 2 & 2 & BOX((0,0), (80,60)) \\
\hline
1 & 2 & 3 & BOX((60,20), (100,120)) \\
\hline
1 & 2 & 4 & BOX((140,120), (200,120)) \\
\hline
2 & 1 & 5 & \ldots \\
\hline
2 & 1 & 6 & \ldots \\
\hline
5 & 0 & A & \ldots \\
\hline
6 & 0 & B & \ldots \\
\hline
\ldots & \ldots & \ldots & \ldots \\
\hline
\end{tabular}
Recursive window query on a RR-tree

Relational index-table: *polygons_rtree*

<table>
<thead>
<tr>
<th>page_id</th>
<th>page_lev</th>
<th>son_id</th>
<th>son_mbr</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROOT</td>
<td>3</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>2</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>...</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>4</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>5</td>
<td>...</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>6</td>
<td>...</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>A</td>
<td>...</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>B</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

```sql
SELECT son_id AS id FROM polygons_rtree
WHERE page_lev = 0
START WITH page_id = ROOT
CONNECT BY
  PRIOR son_mbr INTERSECTS BOX((0,0),(100,100))
  AND PRIOR son_id = page_id;
```
Advantages of the relational mapping

Relational mapping:

- ORDBMS: user has no access to data organization (SQL)
- frees us from physical factors such as block size (virtual pages)
- max / min filling factors less important
- improvement in terms of clustering and similarity of data:
  → considering also quality of data, not only quantity
  → allowing individual fanout for each node
Outline of the Talk

1. Introduction
2. Relational Access Methods
3. Stochastic Basics
4. Stochastic driven Relational R-tree
5. Experimental Evaluation
6. Conclusions and Future Work
point transformation of interval $I = [a, b]$ into upper triangle $\Delta$ of the unit hyper rectangle

$\rightarrow$ the interval $I = [a, b]$ corresponds to the point $P = (a, b)$
The point transformation of interval $I$ into upper triangle $\Delta$ of the unit hyper rectangle is given by:

$$A(I) = b \cdot (1 - a) - \frac{1}{2} \cdot (b - a)^2$$

The probability of any query $x$ hitting interval $I$ is:

$$P(I) = \frac{A(a,b)}{\Delta}$$
probability of any query hitting two intervals $I_1$ and $I_2$:

$$P(I_1, I_2) = \frac{A(\max\{a_1, a_2\}, \min\{b_1, b_2\})}{\Delta}$$

**general case**: probability, that $n$ intervals $I_1...I_n$ are intersected by a query

$$P(I_1, .. ,I_n) = \frac{A(\max_{i=1..n}\{a_i\}, \min_{i=1..n}\{b_i\})}{\Delta}$$
Stochastics III

\[ P(I_1 | I_2) = \frac{P(I_1, I_2)}{P(I_2)} \]

**conditional probability**: \( I_1 \) is intersected by a query that already intersects \( I_2 \)

\[ P(I_1, \ldots, I_L | I_R) = \frac{P(I_1, \ldots, I_L, I_R)}{P(I_R)} \]

**conditional probability**: \( I_1 \) .. \( I_L \) is intersected by a query that already intersects \( I_R \)

\[ P(R_1, \ldots, R_L | R_R) = \prod_{i=1}^{d} P(I_{i_1}, \ldots, I_{L_i} | I_{R_i}) \]

**d-dimensional case** with \( R_j = \prod_{i=1}^{d} I_i, \)

\( j \in \{1, \ldots, L, R\} \) and \( L, R \in IN \)
Outline of the Talk

1. Introduction
2. Relational Access Methods
3. Stochastic Basics
4. Stochastic driven Relational R-tree
5. Experimental Evaluation
6. Conclusions and Future Work
Update operations

Insert:
- Insert Position
- Decision on when to split
- Partitioning of entries when splitting

Update: delete and reinsert

Delete: can lead to underfilled nodes
→ global optimization needed
(idea of merging nodes)
PROCEDURE Insert\((R, Page)\)

\[
\begin{align*}
\text{ActualSimilarity} & \quad \text{REAL}; \\
\text{MaxSimilarity} & \quad \text{REAL}; \\
\text{BestEntry} & \quad \text{PageEntry};
\end{align*}
\]

BEGIN

IF Page IS DirectoryPage THEN

\[
\begin{align*}
\text{MaxSimilarity} & := 0; \\
\text{FOR EACH E IN Page} \\
\text{ActualSimilarity} & := P(R, E); \\
\text{IF ActualSimilarity} \ > \ \text{MaxSimilarity} \ \text{THEN} \\
\text{BestEntry} & := E; \\
\text{MaxSimilarity} & := \text{ActualSimilarity};
\end{align*}
\]

END IF;
END FOR;

Insert\((R, \text{BestEntry})\);

ELSE

\[
\begin{align*}
\ldots
\end{align*}
\]

END IF;
END;

\[\Rightarrow \text{insert into most similar entry (using access probability)}\]
Insert II: when to split

PROCEDURE Split(Page)
    L INTEGER;
    Threshold REAL;
BEGIN
    L := NumberOfEntries (Page);
    // Determine bounding boxes of page B and of all elements Eᵢ
    Similarity := P(E₁,...,Eₖ | B);
    IF L <= CommonEntriesPerPage OR
        Similarity > Threshold THEN // Don’t split
        RETURN;
    ELSE
        // Split
        
        END IF;
END;

split only if entries are relatively dissimilar
common filling factor prevents sparsely filled nodes
Insert III: partitioning algorithm

Split algorithm:

- Reinsert, split axis and initial pair as in R*-tree algorithm
- Difference: using projections and stochastic heuristics instead of geometric distance measure

a) Choosing split axis and initial pair (as in R*-tree algorithm)

b) Contemplate projections onto split axis:
assign according to conditional probabilities

I₁ I I₂

c) Use the partitioning with least overlap
Insert III: partitioning algorithm

using projection for split minimizes the overlap

node to be split, initial pair: R₁, R₆ → high overlap

without projection

with projection → no overlap
Outline of the Talk

1. Introduction
2. Relational Access Methods
3. Stochastic Basics
4. Stochastic driven Relational R-tree
5. Experimental Evaluation
6. Conclusions and Future Work
Experiments: insertion times

Insertion times for rectangles, side length between 0 and 100, center between 0 and 100000
Experiments: highly selective window queries

Response times for queries with high selectivity

- R*-split
- Stochastic split
Response times for queries with decreasing selectivity on 50,000 rectangles

- R*-split
- stochastic split
Experiments: window queries with varying dimensions

Selective window queries with different dimensions (values normalized to 1)

comparison of real-times

R*-split
stochastic split

dimension
Outline of the Talk

1. Introduction
2. Relational Access Methods
3. Stochastic Basics
4. Stochastic driven Relational R-tree
5. Experimental Evaluation
6. Conclusions and Future Work
Conclusions and Future work

Our Approach:

- the relational mapping frees us from physical factors
- the quality of the entries within a node is taken into account
- especially useful for highly clustered data

Future Work:

- optimized query on the stochastic driven RR-tree
- recoverable version of the RR-tree