MISTRAL

Processing Relational Queries Using the Multidimensional Access Method

UB-Tree

Prof. R. Bayer, Ph.D.

Dr. Volker Markl

Dept. of Computer Science, Technical University Munich

and Bavarian Research Center for Knowledgebased Systems

(FORWISS)

Agenda

1. **Concept of the UB-Tree**
2. **Range Query Algorithm**
3. **Tetris Algorithm**

Next Generation DB-Applications

- huge databases
- multidimensional
- complex queries

Examples:

- Datawarehouses
- geographic informationsystems
- time series

B-Tree and UB-Tree

B-Tree

- invented in 1969, published in 1970/72
- enabling technology for all commercial DBMS for 25 years

UB-Tree

- invented in 1996
- enabling technology for next generation DB applications

UB-Tree Access method (TransBase HC)

enable next generations DB-applications!!

- **buying patterns of consumers**
- **geographic data and time series**
- **datamining mobile phone calls**
  - **10 million users * 10 calls/day**
  - **= 100 million records**
  - caller callee time
  - duration geographic location
  - **~ 100 Bytes/call**

⇒ **10 GB/day = 3.6 TeraByte/year**

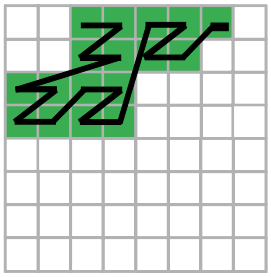
Design Goals

- **clustering tuples on disk pages while preserving spatial proximity**
- **efficient incremental organization**
- **logarithmic worst-case guarantees for insertion, deletion and point queries**
- **efficient handling of range queries**
- **good average memory utilization**

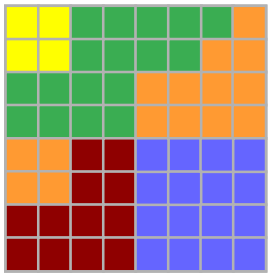
⇒ ***revolution in DB-applications***

Z-regions/UB-Trees

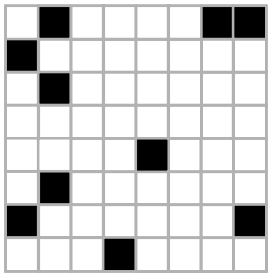
A *Z-region* [α : β] is the space covered by an interval on the Z-curve and is defined by two Z-addresses α and β.



Z-region
[0.1 : 1.1.1]

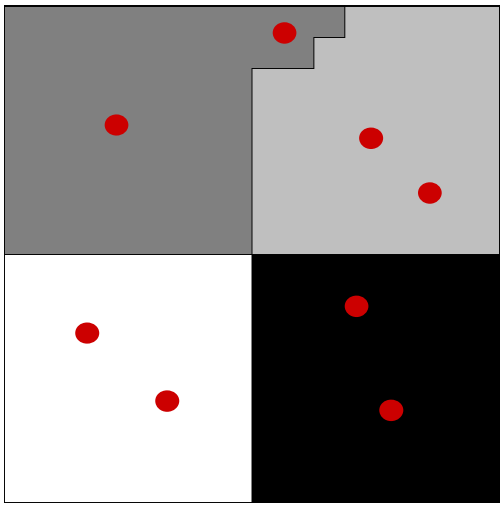


UB-Tree partitioning:
0.1] 1.1.1]
2.1] 3] 4]



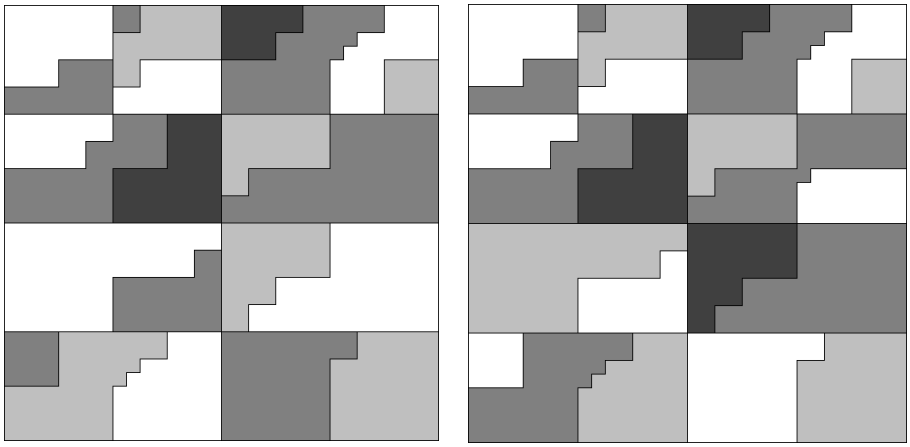
point data creating the UB-Tree on the left for a page capacity of 2 points

UB-Tree Insertion 1/2/3/4



Note:
partioning is not unique!!

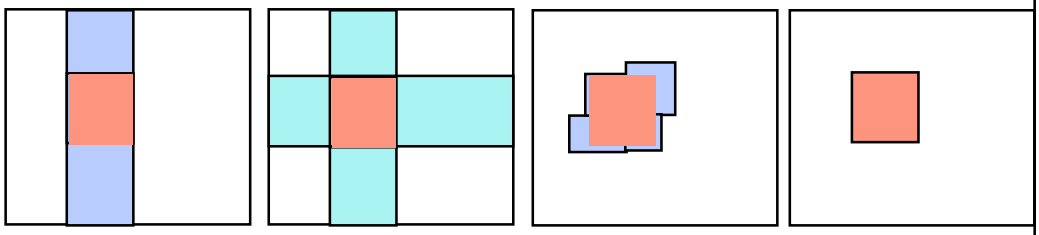
UB-Tree Insertion 18/19



Multidimensional Range Query

SELECT * FROM table
WHERE (A_1 BETWEEN a_1 AND b_1) AND
(A_2 BETWEEN a_2 AND b_2) AND
.....
(A_n BETWEEN a_n AND b_n)

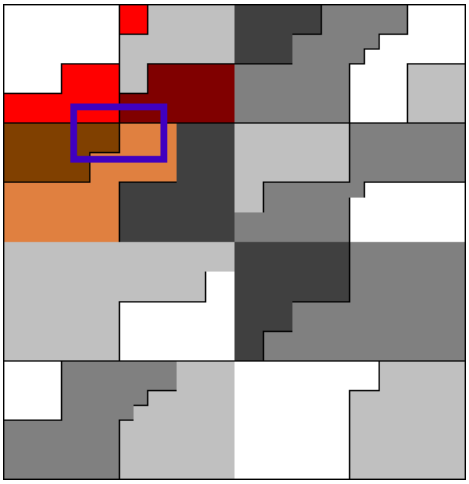
Comparison of the Rangequery Performance



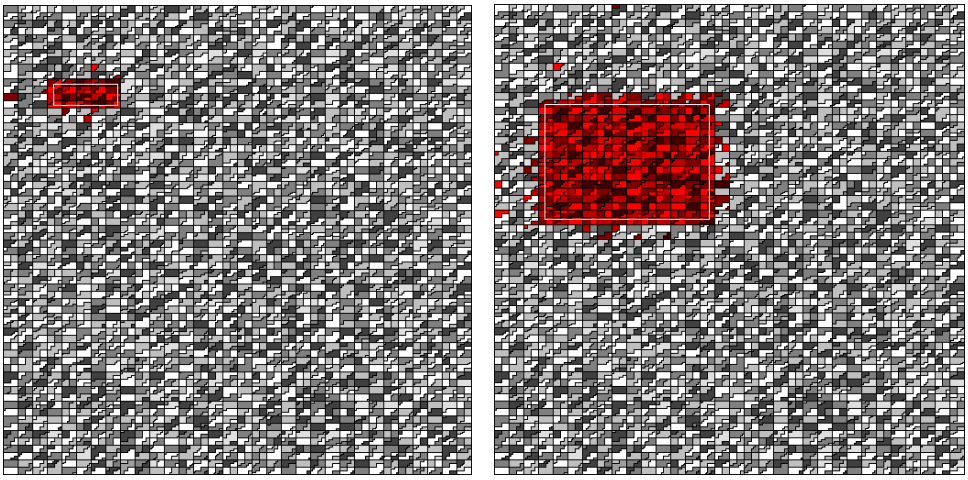
compound primary B-Tree	multiple B-Trees, bitmap indexes	multidimensional index	ideal case
$s_1 * P$	$s_1 * l_1 + s_2 * l_2 + s_1 * s_2 * T$	$s_1^{\uparrow} * s_2^{\uparrow} * P$	$s_1 * s_2 * P$

Example of a Range Query

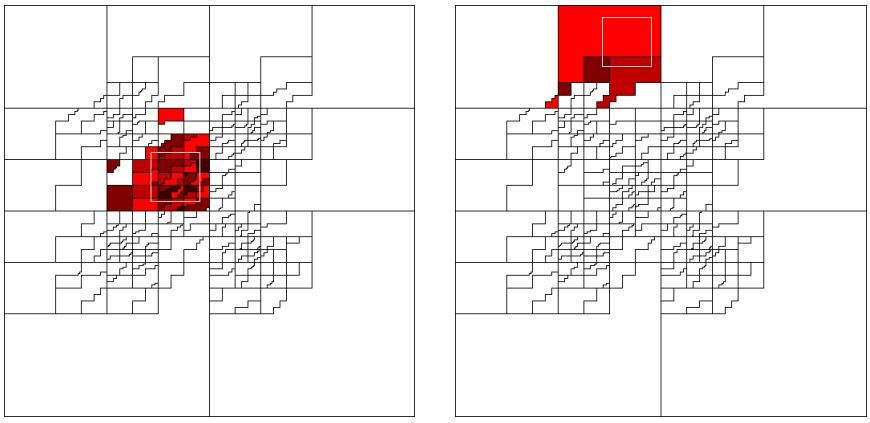
```
rangequery(ql,qh) =
    α := Z(ql); ω := Z(qh)
    while α < ω
        p := read page(α)
        output intersection(p,ql,qh)
        α := getnext(α,ql,qh)

getnext(α,ql,qh) =
    l := last-intersection-in-region(α,ql,qh)
    f := first-intersection-in-
    brother/father(l,ql,qh))
    return alpha(f)
```



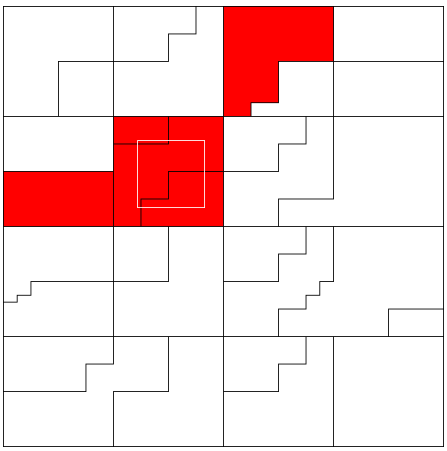
Two Visualized Range-Queries



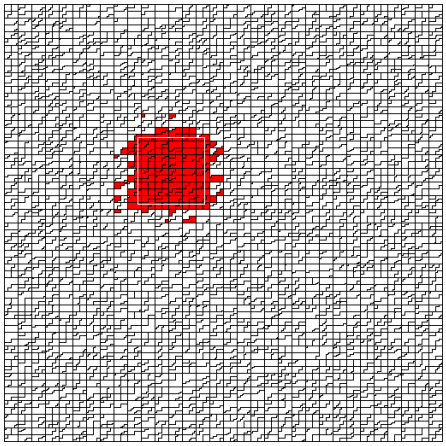
Range Queries in sparsely and densely populated parts of the Universe



Range Queries and Growing Databases



1000 tuples



50 000 tuples

Summary UB-Trees

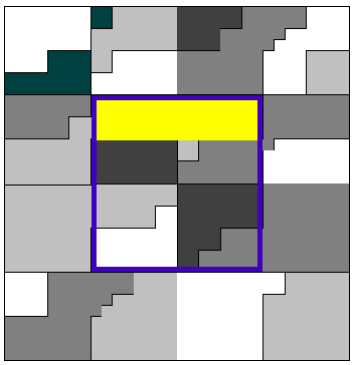
- 50%-83% storage utilization, dynamic updates
- Efficient Z-address calculation (bit-interleaving)
- Logarithmic performance for the basic operations
- Efficient range query algorithm (bit-operations)
- Prototype UB/API above RDBMS (Oracle 8, Informix, DB2 UDB, TransBase, soon: MS SQL 7.0) using ESQL/C
- **Patent application**
- ***most DBMS applications benefit***

Sorting Query Boxes

SELECT * FROM table
WHERE (A_1 BETWEEN a_1 AND b_1) AND
(A_2 BETWEEN a_2 AND b_2) AND
.....
(A_n BETWEEN a_n AND b_n)
ORDER BY A_i, A_j, A_k, ...
(GROUP BY A_i, A_j, A_k, ...)

The Tetris Algorithm

sort direction



Summary Tetris

- Combines sort process and evaluation of multi-attribute restrictions in one processing step
- I/O-time linear w.r. to result set size
- temporary storage sublinear w.r. to result set size
- Sorting no longer a “blocking operation”

✍ **Patent application**

✍⇒ ***speedup for all DB operations***