

Performance of Querying Temporal Attributes in Object-Relational Databases

Carsten Kleiner

University of Hannover, Germany

Symposium *TIME-2002*

Manchester, UK

Motivation

- Complex Applications involve time-varying information
- Often **time is feature of attributes**, not objects
 ~> attribute-timestamping required
- Information in an attribute is:
 1. non-temporal information of standard type *plus*
 2. temporal information
 ~> combining temporal and non-temporal information in a **single datatype** is required
- **Object-Relational Databases** facilitate user-defined types

VT_INTEGER: An Example

- Salary information of employees (valid time only)
- Base Type is INTEGER
- Use **temporal elements** for the temporal component
- Conceptual and logical model: see other publication
- Sample definitions:

```
create type tempElement as set of INTERVAL;  
create type VT_INTEGER_BASE(value INTEGER,  
                           time tempElement);  
create type VT_INTEGER as list of VT_INTEGER_BASE;  
create table employee (id INTEGER, salary VT_INTEGER);
```

Query Types

- Possible operators: **conjunctive combinations of** temporal **operators** with base type specific operators
- Any combination of *range*, *point* and $*$ in each dimension as argument; notation cf. [TJS98] (valid time): e. g. *range//point* query
- Efficient querying requires **one index structure** for as many query types as possible
 - ~> **user-defined indexes** recommended
- Many different datatypes possible
 - ~> **indexing framework** required
 - ~> use Generalized Search Trees (GiST)

Index Structures for Temporal Queries

- Combining data from different domains
 - ~> using **user-defined multidimensional index** structures
 - ⇒ classical spatial index structures are well-suited (GiST!)
- alternative: use standard index on a single component
- **VT_INTEGER**: two-dimensional domain
 - ~> traditional **2D spatial indexes** usable
- **BT_INTEGER**: three-dimensional domain
 - ~> use **3D extensions of spatial indexes**
- more complex datatypes: higher multidimensional indexes (VT_GEOMETRY, BT_INTEGER_STRING, ...)

What To See on the Poster?

- Brief **summary of concepts** and design decisions
- **Performance evaluation** of different possible index structures on VT_INTEGER (user-defined and system-provided indexes)
- **Performance evaluation** of different possible index structures on BT_INTEGER (user-defined indexes only)
- **Conclusions** on efficient query support in object-relational database systems for these and other (!) user-defined datatypes (from the evaluation)

Outlook

- Embedding in **temporal query language** and user-friendly environment
- Incorporating special characteristics of *now* and *UC* in indexing (easy due to GiST framework ?!)
- Experiments with more complex base types
~ Higher dimensional index structures
- User-defined cost and selectivity estimation (?)
- Experiments with other/**advanced** temporal/spatial/**spatio-temporal analysis** operators