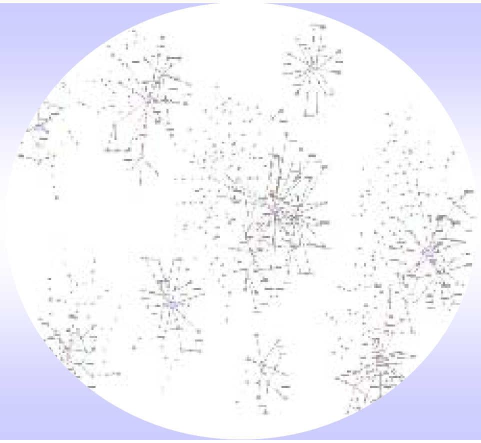


School of Computer and Electrical Engineering

Knowledge and Database Systems Laboratory

National Technical University of Athens

Adaptive Query Formulation to Handle Database Evolution

George Papastefanatos[1], Panos Vassiliadis[2], Yannis Vassiliou[1]

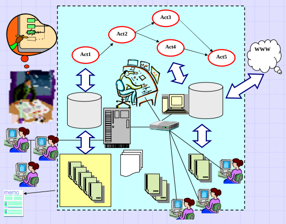
[1] National Technical University of Athens,
Dept. of Electrical and Computer Eng.,
Athens, Hellas
{gpapas,yv}@dbnet.ece.ntua.gr

[2] University of Ioannina,
Dept. of Computer Science,
Ioannina, Hellas
pvassil@cs.uoi.gr

Database Schema Evolution – Query Adaptation

Current database systems are continuously evolving environments, where design constructs are

- added
- removed
- modified



Evolution is not handled by current DBMS with an automatic way, but rather they require great human effort

Existing Queries affected:

Syntactically – i.e., become invalid

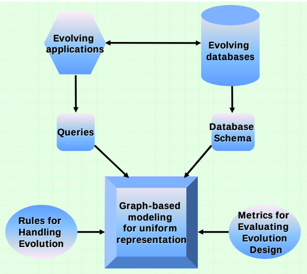
Semantically – i.e., query must conform to the new database semantics

Adaptation of SQL queries and views

- time-consuming task
- treated in most of the cases manually by the administrators/developers

Our Approach

- Graph based representation of database constructs (i.e., relations, views, constraints, queries)
- Mechanism for performing what-if analysis for potential changes of database configurations
- Annotation of graph with rules for adapting queries to database schema evolution


Evolving applications
Evolving databases
Queries
Database Schema
Graph-based modeling for uniform representation
Rules for Handling Evolution
Metrics for Evaluating Evolution Design

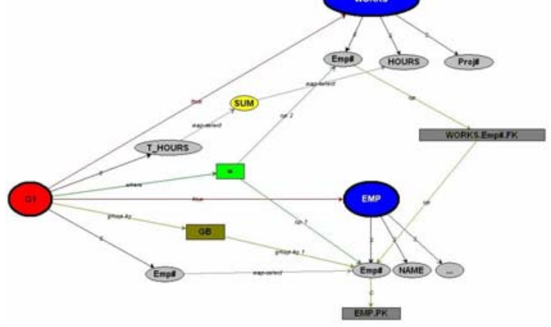
Graph-based modeling

- **Database Constructs mapped to directed graphs**
  - **Relations**
  - **Conditions (covering database constraints and query conditions)**
  - **Queries**
  - **Views**
- **Graph Semantics**

  Nodes Represent Database Constructs, i.e., relation nodes, attribute nodes, query nodes, etc.

  Edges Represent Relationships Between Constructs, i.e., schema edges, map-select edges, operand edges, etc.

```
Q: SELECT EMP.Emp#, Sum(WORKS.Hours) as T_Hours
   FROM EMP, WORKS
   WHERE EMP.Emp# = WORKS.Emp#
   GROUP BY EMP.Emp#
```



Adapting queries and views to Database schema Evolution

1 **Set of evolving database constructs**
- → **relations**
- → **attributes**
- → **constraints**

2 **Set of potential evolution changes**
- → **addition**
- → **deletion**
- → **modification**

3 **Graph elements are annotated with policies**
- → **propagate**

  the graph must be reshaped to adjust to the new semantics incurred by the event
- → **block**

  the old semantics of the graph must be retained and the (hypothetical) event must be blocked or, at least, constrained, through some rewriting that preserves the old semantics

4 **Rule for policies conflict resolution**

When two graph constructs have different policies for the same event

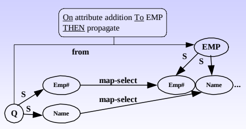
Rule

Policies defined on query graph structures are stronger than policies defined on view graph structures which in turn prevail on policies defined on relation graph structures

5 **According to prevailing policy, the proper action is taken**
- → **query adaptation**

Example

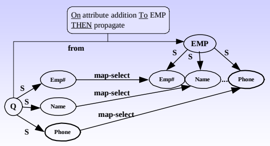
Annotated Query Graph



Event

Add attribute Phone to EMP relation

Transformed Query Graph



Extending SQL With Evolution Semantics

```
ON <event> TO <element> THEN <policy>
```

E.g.

```
SELECT Emp#, NAME, AGE
FROM V
ON condition addition TO V THEN propagate,
ON attribute deletion TO V.AGE THEN block
```

Visualization Tool