Updating Relational Databases Through Views

Zhenjiang Hu

June 29, 2010

References

* Arthur M. Keller: Algorithms for Translating View Updates to Database Updates for Views Involving Selections, Projections, and Joins. PODS 1985: 154-163.

* Arthur M. Keller: Choosing a View Update Translator by Dialog at View Definition Time. VLDB 1986: 467-474.

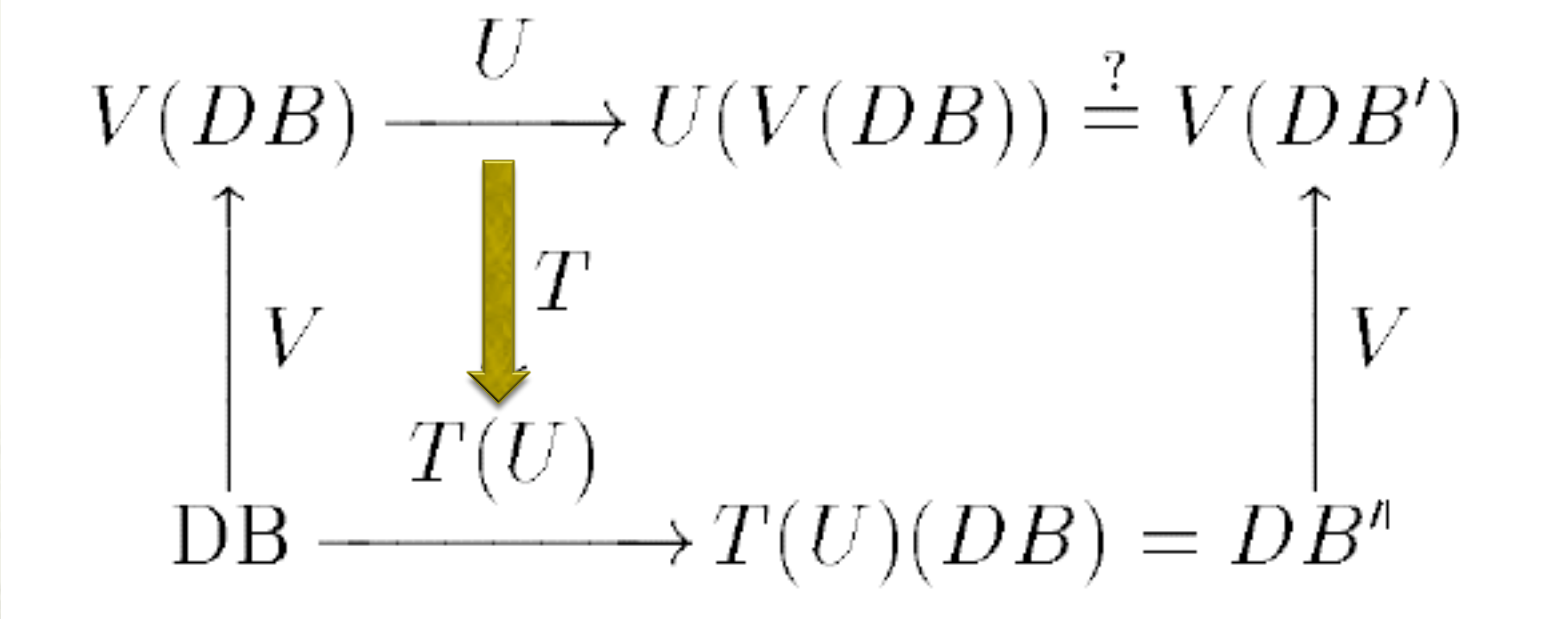
* Arthur M. Keller: Updating Relational databases Through Views, PhD thesis, Stanford University, 1985: 111 pages.

My Motivation of Reading these Papers

* How to formally define a translation from view updates to source updates in the context of graph transformation? What is the criteria for view update translation?

* When the translation is not unique, how can we specify our intension statically?

View Update Translation



Define T such that

- V(DB') is as close to U(V(DB)) as possible
- T(U) is as simple as possible

Relation (DB, View)

* key, attribute,tuple

Employee

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

Define Relation Employee (
  name :> dept, loc, onbbteam);

Relation (DB, View)

* key, attribute,tuple

Dept

deptname	namager	division
software	Tanaka	system
content	Suzuki	application

Define Relation Dept (
deptname :> manager, division);

Updates

- Deletion
  - Removal of a single tuple from a relation
- Insertion
  - Addition of a single tuple into a relation
- Replacement
  - Combination of a deletion and an insertion into the same relation into a single atomic action that does not require an intermediate consistent state

    (This can avoid losing information by deletion)

View Definition

* Selection

```
SELECT *
FROM Employee
WHERE
   Employee.loc IN {15}
```

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES

View Definition

* Projection

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

```
SELECT
   Employee.name,
   Employee.dept
FROM Employee
```

name	dept
Tanaka	software
Sato	software
Suzuki	content

View Definition

* Join (extension join)

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

deptname	namager	division
software	Tanaka	system
content	Suzuki	application

```
SELECT Employee.name, Dept.deptname,
       Dept.manager, Dept.division
FROM Employee, Dept
WHERE
  Employee.dept = Dept.deptname
```

name	deptname	manager	division
Tanaka	software	Tanaka	system
Sato	software	Tanaka	system
Suzuki	content	Suzuki	application

Through key

SPJ Normal Form

- Here, we only consider views defined by SPJNF
- Normalization Rules:

  project . join → join . (project x project)

  select . join → join . (select x select)

  select . project → project . select

  select . select → select

  project . project → project

NB: Project . join: project should keep keys

Criteria for View Update Translation

- No database side effects
- Only “one step” change (on any tuple)
- Minimal change
  - No unnecessary changes
  - Replacements cannot be simplified
  - No delete-insert pairs

Permit all possible changes, but only in their simplest forms.

Update Translation: SP View

* Insertion

If the new tuple has no conflict with any tuple in DB
then insert it to DB
else replace the conflict tuple by it

Theorem: The insertion translation satisfies the five criteria.

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

```
SELECT *
FROM Employee
WHERE
  Employee.loc IN {15}
```

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Kato	content	15	No

inserted Valid tuple

Update Translation: SP View

* Deletion

D1: Delete the tuple in DB

D2: Replace the tuple in DB
by changing loc to an arbitrary excluded value

Theorem: Both D1 and D2 satisfy the five criteria.

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

```
SELECT *
FROM Employee
WHERE
  Employee.loc IN {15}
```

name	dept	loc	onbbteam
Tanaka	software	15	NO
~~Sato~~	~~software~~	~~15~~	~~YES~~

Update Translation: SP View

- Replacement

If the key does not change in the view tuple replacement, then replace the tuple in DB

otherwise,
if there is no same key in DB,
then (a) replace the tuple, or
(b) delete; insert
else delete; insert

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

```
SELECT *
FROM Employee
WHERE
  Employee.loc IN {15}
```

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	NO

Update Translation: Join View

* Deletion — Delete the tuple in one of the two DBs

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

deptname	namager	division
software	Tanaka	system
content	Suzuki	application

```
SELECT Employee.name, Dept.deptname,
            Dept.manager, Dept.division
FROM Employee, Dept
WHERE
   Employee.dept = Dept.deptname
```

name	deptname	manager	division
Tanaka	software	Tanaka	system
~~Sato~~	~~software~~	~~Tanaka~~	~~system~~
Suzuki	content	Suzuki	application

Update Translation: Join View

* Insertion — De-join the inserted tuple, and insert them to the SP views resp.

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

deptname	namager	division
software	Tanaka	system

```
SELECT Employee.name, Dept.deptname,
            Dept.manager, Dept.division
FROM Employee, Dept
WHERE
   Employee.dept = Dept.deptname
```

name	deptname	manager	division
Tanaka	software	Tanaka	system
Sato	software	Tanaka	system
Suzuki	content	Suzuki	application

Update Translation: Join View

* Replacement

De-join the replaced tuple, and reflect insert them to the SP views

name	dept	loc	onbbteam
Tanaka	software	15	NO
Sato	software	15	YES
Suzuki	content	18	YES

deptname	namager	division
software	Tanaka	system

```
SELECT Employee.name, Dept.deptname,
            Dept.manager, Dept.division
FROM Employee, Dept
WHERE
   Employee.dept = Dept.deptname
```

name	deptname	manager	division
Tanaka	software	Tanaka	system
Sato	software	Sato	system

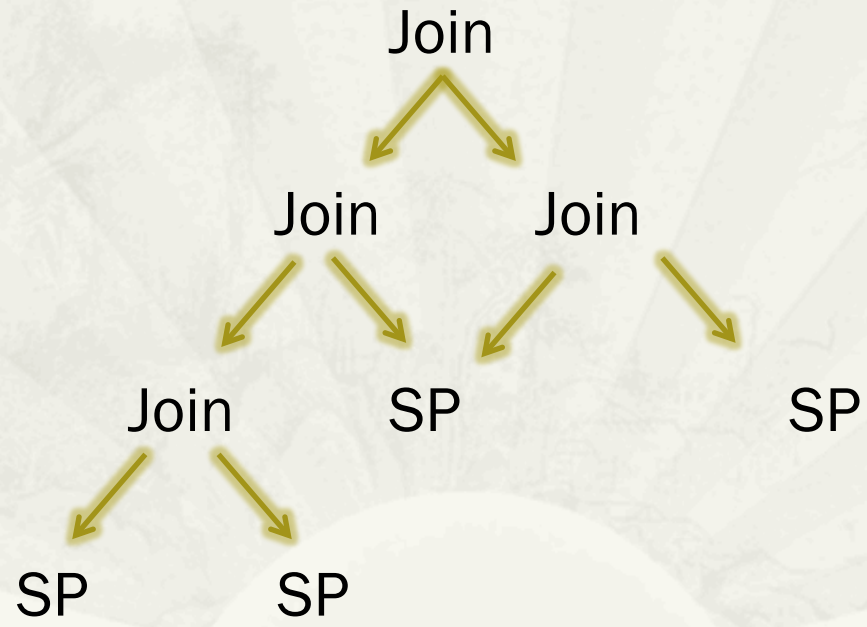
replace ment

Disambiguating View Update Translation Statically

- Choices
  - SP View
    - Deletion: 2 (delete or replace)
    - Replacement: 2 (replace or delete+insert)
  - Join View
    - Deletion: 2 (left or right)

Disambiguating View Update Translation Statically

* Building a decision tree according to the query tree



Configuration File

```
DIALOG PART = DELETION
ALLOWED = YES
RELATION = Employee
METHOD = REPLACE
RSET onbbteam = NO
DIALOG PART = INSERTION
ALLOWED = YES
RELATION = Employee
METHOD = CHANGE_TUPLE
CHANGE_NON_SATISFYING_SELECT_COND_ALLOWED = YES
SSET onbbteam   = YES
RELATION = Dept
METHOD = NO_CHANGE
DIALOG PART = REPLACEMENT
ALLOWED = YES
```

My Conclusion

- SPJ normal form helps a lot in view update translation
  - Graph is more complex (but it has no keys): selection, projection, join, unnest, and structured recursion
- It is possible to choose a update translator statically
  - We may do better if we describe such intension with types
- How to deal with practical queries remains unknown
  - Our idea is to introduce primitives with bidirectional semantics