NeT & CoT: Translating Relational Schemas to XML Schemas

Dongwon Lee, Murali Mani,
Frank Chiu, Wesley. W. Chu

Outline

- Problem Definition and Motivation
- NeT – Translation using the nest operator in Nested Relational Algebra
- CoT – Translation using foreign key constraints between tables
- Conclusions and Future directions
- Important References

What are we studying?

Pname	Age
Muntz	61

Sname	Advisor	Since
MM	Muntz	1998
YC	Muntz	2000

```
<professor Pname="Muntz" Age="61">
  <student Sname="MM" Since="1998"/>
  <student Sname="YC" Since="2000"/>
</professor>
```


Why do we study this?

- Data exchange - XML has established itself as the standard format for data exchange between applications.
- Data integration and "better" data model - XML views of relational data are oftentimes easier to use, and also helps in data integration of multiple data sources.

NeT: Using the Nest operator

course → (Cname, Prof, Text)

Cname	Prof	Text
Algorithms	Gafni	Udi Manber
Algorithms	Gafni	CLR
Algorithms	Majid	Udi Manber
Algorithms	Majid	CLR

NeT (contd…)

course → (Cname, Prof $^{+}$, Text)

Cname	Prof	Text
Algorithms	{Gafni, Majid}	Udi Manber
Algorithms	{Gafni, Majid}	CLR

course → (Cname, Prof $^{+}$, Text $^{+}$)

Cname	Prof	Text
Algorithms	{Gafni, Majid}	{Udi Manber, CLR}

NeT (contd…)

person → (Name, City, State, Zip)

Name	City	State	Zip
MM	Los Angeles	CA	90034
AN	Los Angeles	CA	90034

person → (Name^{+}, City, State, Zip)

{MM, AN}	Los Angeles	CA	90034

NeT: Formal Definition

- Consider Table *t* with column set *C*. Nesting on column *X* is defined as:
  Any two tuples with the same values for (*C* – *X*) will be combined to one tuple
- **Observation: We need to nest only on key columns**

CoT: Using foreign key constraints

Pname	Age
Muntz	61

Sname	Advisor	CName
MM	Muntz	DBs
YC	Muntz	DBs
YC	Muntz	QSs

professor → (Pname, Age, student*)
student → (Sname, Cname)

```
<professor Pname="Muntz" Age="61">
  <student Sname="MM" Cname="DBs"/>
  <student Sname="YC" Cname="DBs"/>
  <student Sname="YC" Cname="QSs"/>
</professor>
```


CoT (contd…)

Pname	Age
Muntz	61

Cname	Since
DBs	1979
QSs	1962

Sname	Advisor	Cname
MM	Muntz	DBs
YC	Muntz	DBs
YC	Muntz	QSs

professor → (Pname, Age, student*)
student → (Sname, Ref_course)
course → (Cname, Since, ID_course)

CoT (contd…)

<professor Pname="Muntz" Age="61">

<student Sname="MM" Ref_course="DBs"/>

<student Sname="YC" Ref_course="DBs"/>

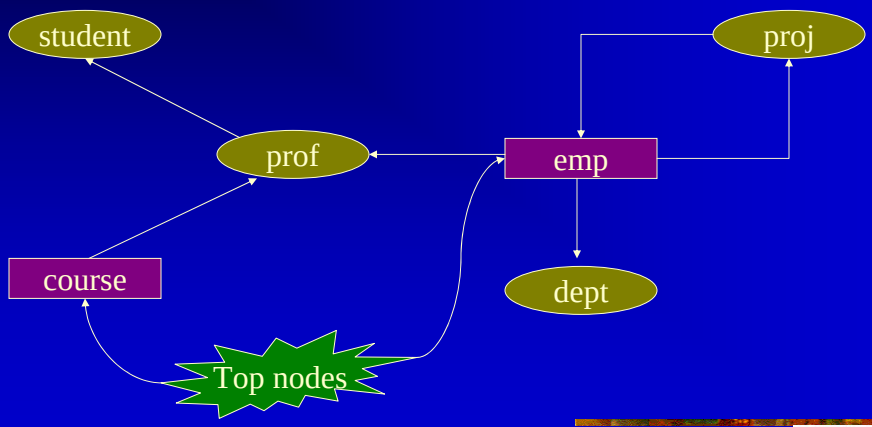
<student Sname="YC" Ref_course="QSs"/>

</professor>

<course Cname="DBs" Since="1979" ID_course="DBs"/>

<course Cname="QSs" Since="1962" ID_course="QSs"/>

CoT algorithm



Conclusions

- "Better" data model for the user to work with.
- Constraints are maintained.
- Reduced data redundancy from NeT and CoT – backed by experiments on UCI-KDD repository and TPC-H data.
- Implemented and available for use.

Lessons Learnt

- Helped us learn key issues in XML data modeling:
  - Local Tree Grammars (such as DTD) sufficient for these applications
  - The resulting XML Schema did not have any recursion

Future Work

Btitle	Year
B1	1994
B2	1997

Atitle	Year
A1	2002

Name	Age	Book	Article
P1	64	B1	Null
P2	52	B2	Null
P3	31	Null	A1
P4	25	Null	A1

book → (Btitle, Year)
article → (Atitle, Year)
person → (Name, Age, Book, Article)

Future Work (contd…)

If we knew
book.person ∩ article.person = ∅
book.person ∪ article.person = person

book → (Btitle, Year, person*)
article → (Atitle, Year, person*)
person → (Name, Age)

Future Work (contd…)

Name	Age	Mgr
A	25	M
B	31	M
M	61	Null

```
<person Name="M" Age="61">
   <person Name="A" Age="25"/>
   <person Name="B" Age="31"/>
</person>
```


References

- *Taxonomy of XML Schema Languages using Formal Language Theory*, M. Murata, D. Lee, M. Mani, Extreme Markup Languages 2001
- *Semantic Data Modeling using XML Schemas*, M. Mani, D. Lee, R. R. Muntz, ER 2001