University of Washington
Department of Computer Science & Engineering

Managing XML and Semistructured Data

Lecture 10: Schema

Prof. Dan Suciu

Spring 2001

In this lecture

- Schema for unordered data
  - conformance and classification
  - using simulation
  - upper bound and lower bound schema

Resources

- MSL: A Model for W3C XML Schema by Brown, Fuchs, Robie, Wadler, in WWW10, 2001.
- Subsumption for XML Types by Kuper and Simeon, ICDT'2001.
- **Data on the Web** Abiteboul, Buneman, Suciu : *chapter 7*

Schemas for Relational v.s. SS Data

Schemas in relational data:

- Defined before the data
- Strictly enforced

Schemas in SS data

- Defined after the data
- May not be enforced (some data doesn't have schema)

When the Schema is Created

- Created by the user
  - Before or after the data
  - Does not need to be unique
- Extracted from the data
  - Unheard of in relational databases
- Extracted from the query
  - Like type inference in programming languages

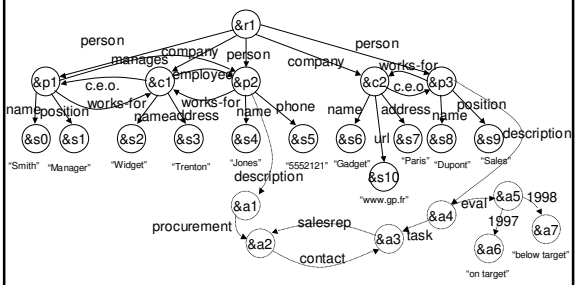
Different schema formalisms for each task !

Schemas for Unordered Data

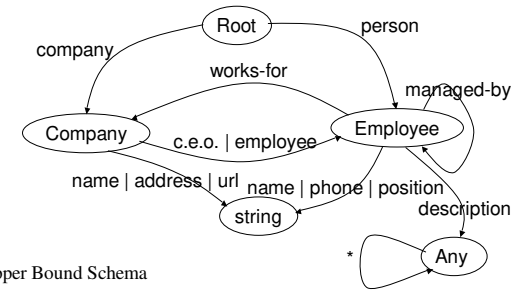
- OEM
  - Cycles
- Schema itself will be a graph

Schemas: An Example

Some database:



Graph Schemas



Upper Bound Schema

The Two Questions to Ask

Conformance: does that data conform to this schema ?

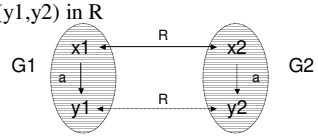
Classification: if so, then which objects belong to what classes ?

Graph Simulation

Definition Two edge-labeled graphs G1, G2

A *simulation* is a relation R between nodes:

- if (x1, x2) in R, and (x1,a,y1) in G1, then exists (x2,a,y2) in G2 (same label) s.t. (y1,y2) in R

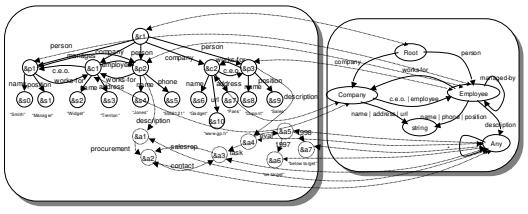


Using Simulation

Data graph D, schema S

- conformance: find maximal simulation R from D to S
  - Notation: $D \leq S$
- classification: check if (x,c) in R
  - Notation: $x \leq c$

Example



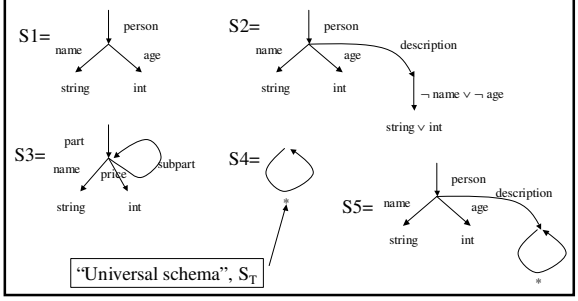
Database Graph Schema

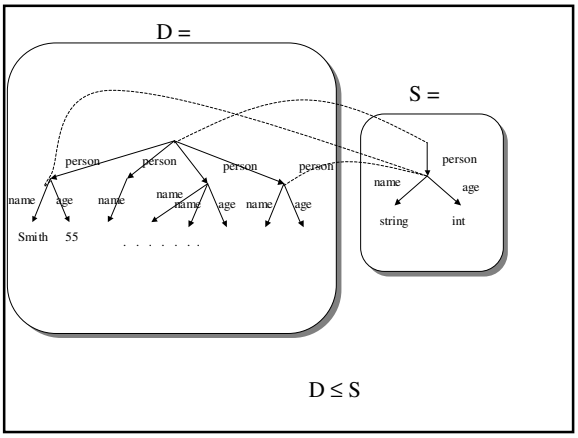
Formally

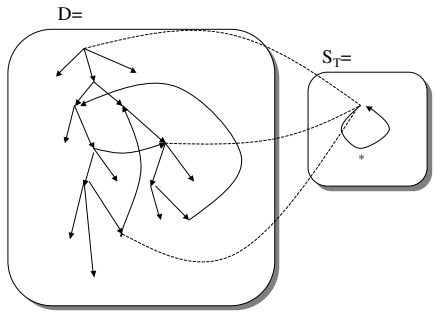
Graph schema S is a *graph*, s.t.:

- Nodes are called *classes*
- Edges are labeled with *unary predicates*, p(x) Examples:
  - person = "x=person"
  - name | address = "x=name ∨ x=address"
  - * = "true"
  - int = "x ∈ int"
  - ¬name = "x ≠ name"

Examples of Graph Schemas (What Do They Mean ?)







Schemas in SS Data v.s. Relational Data

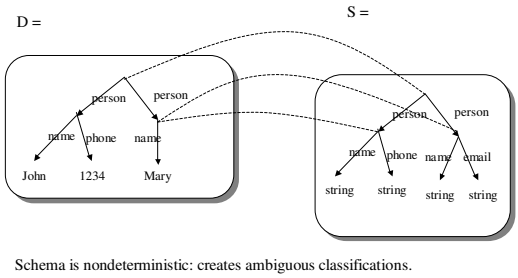
Relational Data:

- Each data instance has exactly one schema

Semistructured data

- One data instance has several schemas

The Classification Problem



The Classification Problem

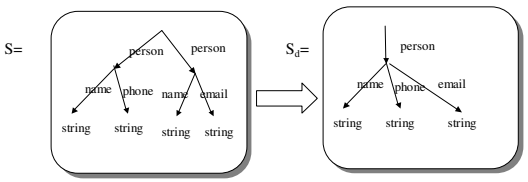
Definition A schema S is *deterministic* if for every class c and every label a, there is at most one outgoing edge labeled a from c

Fact: if S is deterministic and D is a tree, then each node is uniquely classified

(When D is not a tree, then this is not true.)

Deterministic Upper Bound Schemas

- Given a schema S, we can always construct an deterministic approximation, S_d



In general, S_d obtained by powerset constrcut ➔ expensive

Lower Bound Schemas

- Introduced (under a different name and formalism) in:
  - Nestorov, Abiteboul, Motwani, *Extracting Schema from Semistructured Data,* SIGMOD 98
- Goal: extract some "schema" from the data
- We will see later why these are "lower bound schemas"

Lower Bound Schemas

Schema = datalog program with special form
Classes = predicates

```
Company(x) :- link(x,"c.e.o.",y), Employee(y),
              link(x,"name",z), String(z),
              link(x,"address",u), String(u)
Employee(x) :- link(x,"works-for",y), Company(y),
               link(x,"managed-by",z), Employee(z),
               link(x,"name",u), String(u),
               link(x,"address",v), String(v),
Root(x) :- link(x,"company",y), Company(y),
           link(x,"person",z), Employee(z)
```

Maximal fixpoint, rather than minimal fixpoint ! (next)

Datalog Maximal Fixpoint

Standard datalog semantcis. Transform program to:

```
Company(x) ⇐ ∃y. ∃z. ∃u.(link(x,"c.e.o.",y), Employee(y),
                         link(x,"name",z), String(z),
                         link(x,"address",u), String(u)
Employee(x) ⇐ ∃y.∃z.∃u.∃v.(link(x,"works-for",y), Company(y),
                           link(x,"managed-by",z), Employee(z),
                           link(x,"name",u), String(u),
                           link(x,"address",v), String(v))
Root(x) ⇐ ∃y.∃z.(link(x,"company",y), Company(y),
                 link(x,"person",z), Employee(z))
```

Compute minimal model. What is it ? the empty set

Datalog Minimal Fixpoint

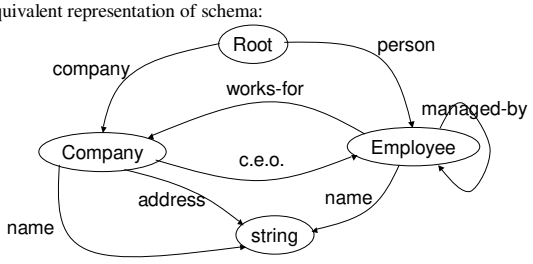
Standard datalog semantcis. Transform program to:

```
Company(x) ⇒ ∃y. ∃z. ∃u.(link(x,"c.e.o.",y), Employee(y),
                         link(x,"name",z), String(z),
                         link(x,"address",u), String(u)
Employee(x) ⇒ ∃y.∃z.∃u.∃v.(link(x,"works-for",y), Company(y),
                           link(x,"managed-by",z), Employee(z),
                           link(x,"name",u), String(u),
                           link(x,"address",v), String(v))
Root(x) ⇒ ∃y.∃z.(link(x,"company",y), Company(y),
                 link(x,"person",z), Employee(z))
```

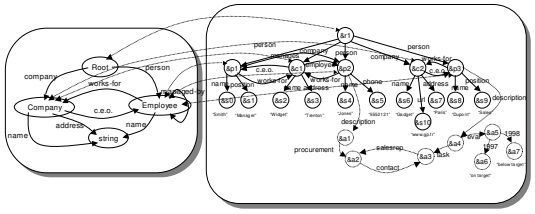
Compute maximal model.

Lower-Bound Schemas

Equivalent representation of schema:



Simulation Strikes Back



A model of program P is precisely a simulation (and vice versa)

Summary: Lower v.s. Upper Bound Schemas

Upper bound schemas

- Tells us what edges are *allowed*

Lower bound schemas

- Tell us what edges are *required*