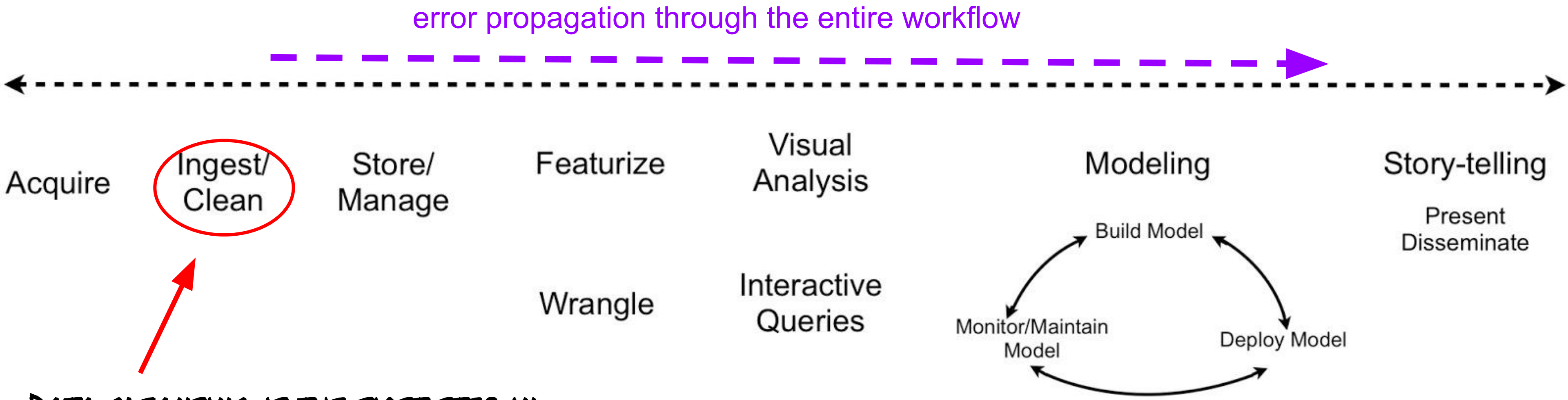
Effective Exploiting of Statistical Relational Learning for Data Cleaning

Larysa Visengeriyeva

Technische Universität Berlin, Germany

Data Science Workflow

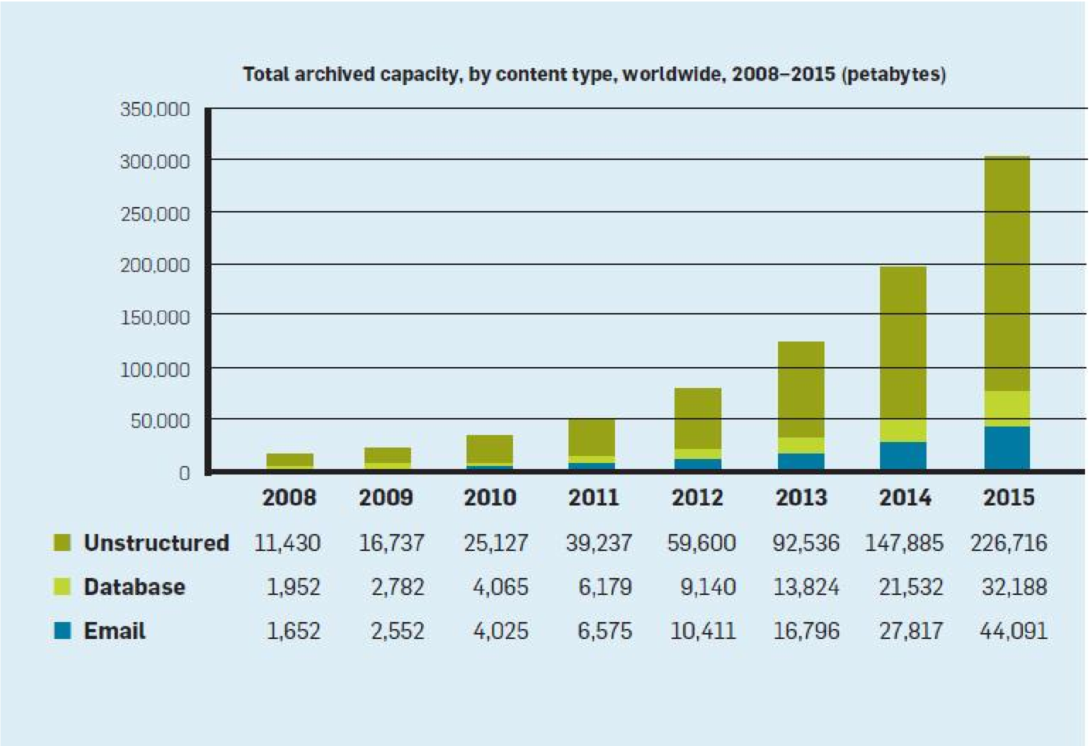


DATA CLEANSING IS THE FIRST STEP IN ANY DATA SCIENCE WORKFLOW, AND OFTEN THE MOST IMPORTANT.

http://radar.oreilly.com/2013/09/data-analysis-just-one-component-of-the-data-science-workflow.html?cmp=ot-strata-na-article-na_62643

Problem Statement

Increasing need for a new approaches in **data cleaning** (for **non-relational** use-cases, e.g Web data)



Vasant Dhar. 2013. Data science and prediction. *Commun. ACM* 56, 12 (December 2013)

Effective Data Cleaning

GOAL

1. Cleaning relational data

2. Cleaning semistructured data

PROBLEM SPACE

SYSTEM REQUIREMENTS:

1 ○ **Joint modeling** for data cleaning rules

2 ○ Defining **“soft” and “hard” data cleaning rules**

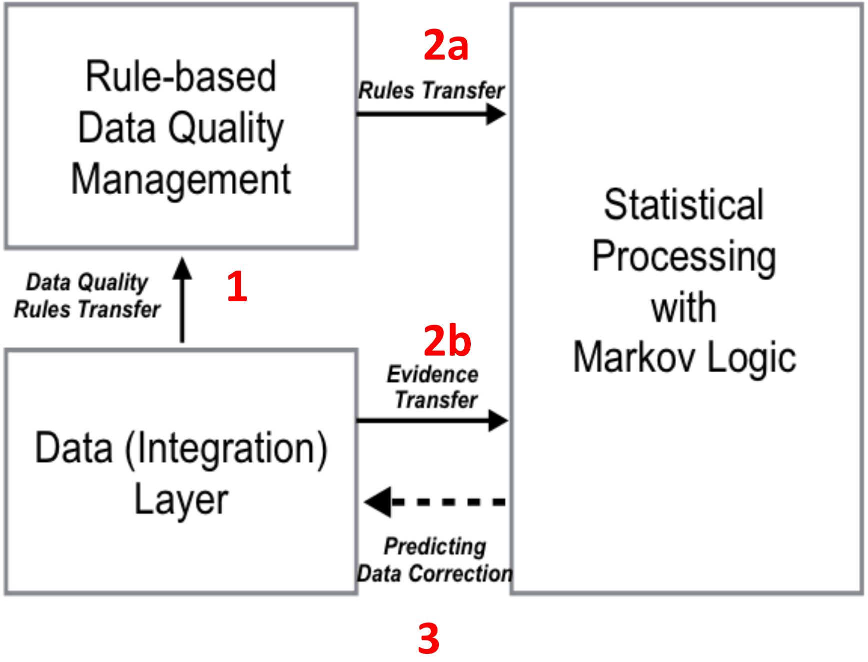
○ **Flexible rules** modeling (non-constraints rules)

Our Idea

... adapt **probabilistic graphical models** to a data cleaning.

The main goal is to translate the constraint-based **data quality rules** into a **predictive model** in order to infer errors in uncertain data and their sources by using **joint inference**.

Method



1. DQ Rules are data agnostic and therefore derived from data. (We assume that constraints are already defined for the data).

2a. Create predictive model: DQ Rules are expressed in first-order logic formulas and then as Markov logic formulas (which is a template for Markov Networks)
2b. Existing data are evidence for the predictive model created in (2a).

3. Running inference on Markov Networks predicts the data correction.

Rules Taxonomie

"HARD" RULES

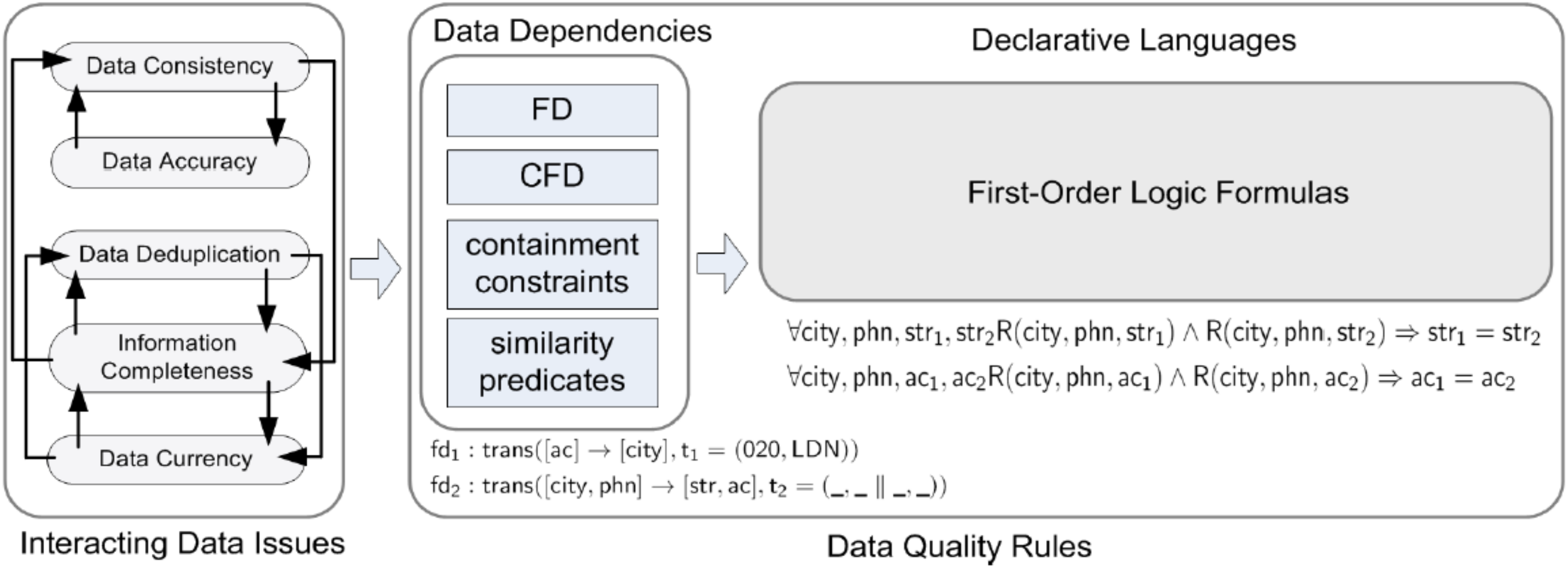
- ❖ Primary Key constraints
- ❖ Constant CFDs

"SOFT" RULES

(FOR SOME RULES MAY NOT EXIST SUFFICIENT KNOWLEDGE)

- ★ non-Primary Key constraints
- ★ CDFs, MDs, CINDs
- ★ "Soft" FDs
- ★ Similarity predicates

How to compile data quality rules into Markov logic?



Why Markov Logic?

Recap Two Requirements:

1
- **Joint modeling** for data cleaning rules

2
- Defining **"soft" and "hard" data cleaning rules**
- Modeling **conflicting rules**
- **Flexible rules** modeling (non-constraints rules)

Markov Logic:

- **Joint inference over hidden predicates** (by using Markov Networks)
- Defining **"soft" and "hard" first-order formulae**
- Modeling **contradicting formulas**
- **Declarative formalism**

Whole Stack in one Example:

INPUT:

Dirty data D_0

id	name	zip	city
1	Dirk	10000	Hamburg
2	Dirk	10000	Berlin
3	Dirk	20000	Berlin

Data cleaning based on conditional functional dependencies (**cfd_1 cfd_2**)and matching rule (**md_3**):

cfd_1: [zip=10000]→[city=Berlin]

cfd_2: [city=Hamburg]→[zip=20000]

md_3: R[name, zip]=R[name,zip]→R[city]⇋R[city]

Scenario I: CFD_1 + MD_3

Dirty data D_0

id	name	zip	city
1	Dirk	10000	Hamburg
2	Dirk	10000	Berlin
3	Dirk	20000	Berlin

Dirty data D_1

cfd_1: [zip=10000]→[city=Berlin]

id	name	zip	city
1	Dirk	10000	Berlin
2	Dirk	10000	Berlin
3	Dirk	20000	Berlin

Cleaned data D_2

md_3: R[name, zip]=R[name,zip]→R[city]⇋R[city]

id	name	zip	city
1+2	Dirk	10000	Berlin
3	Dirk	20000	Berlin

SCENARIO II: $CFD_2 + MD_3$

Dirty data D_0

id	name	zip	city
1	Dirk	10000	Hamburg
2	Dirk	10000	Berlin
3	Dirk	20000	Berlin

Dirty data D_1

cfd$_2$: [city=Hamburg]→[zip=20000]

md$_3$: R[name, zip]=R[name,zip]→R[city]⇋R[city]

id	name	zip	city
1	Dirk	20000	Hamburg
2	Dirk	10000	Berlin
3	Dirk	20000	Berlin

TWO POSSIBLE REPAIRS

$D_{2.1}$

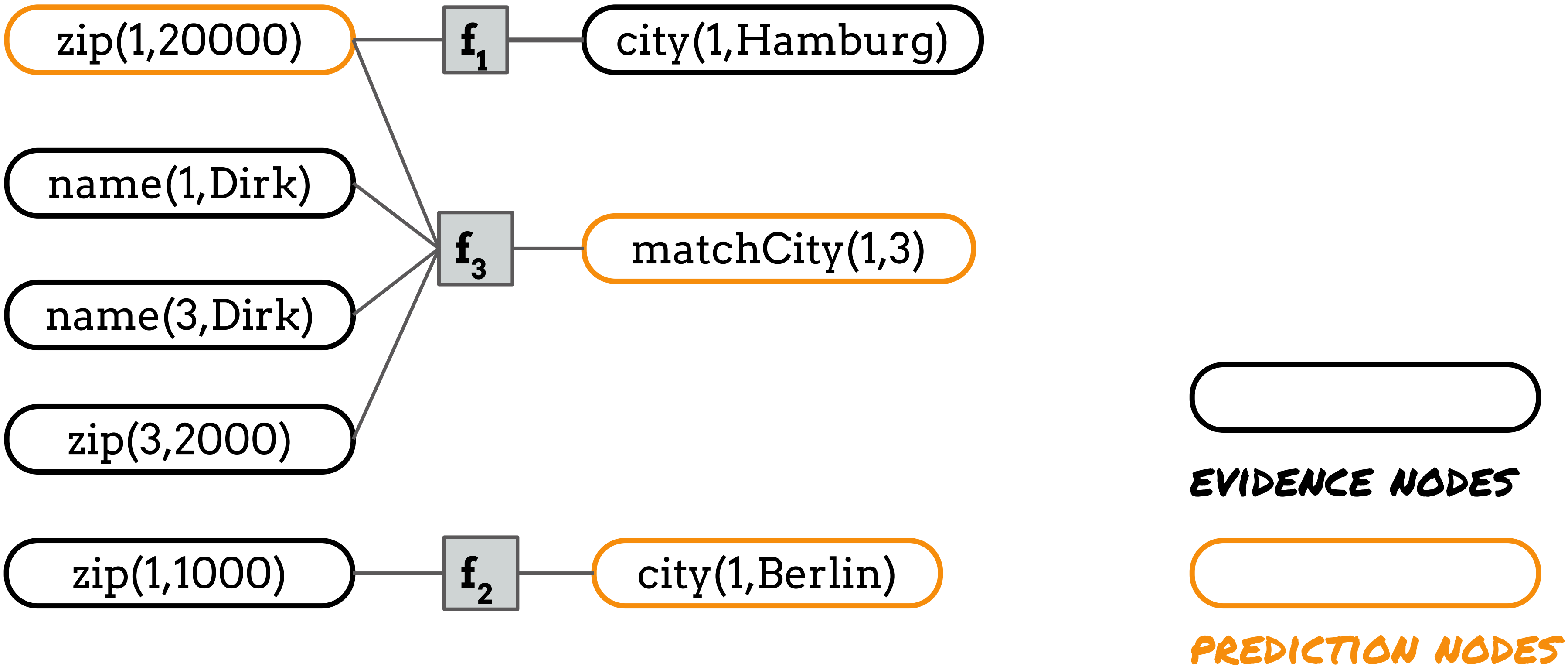
id	name	zip	city
1	Dirk	20000	Hamburg
2	Dirk	10000	Berlin
3	Dirk	20000	Berlin

$D_{2.2}$

id	name	zip	city
1	Dirk	20000	Berlin
2	Dirk	10000	Berlin
3	Dirk	20000	Berlin

Data cleaning rules	First-order logic translation
$\mathbf{cfd_1}$: [zip=10000]$\rightarrow$[city=Berlin]	$\mathbf{f_1}$:zip(id, 10000)$\Rightarrow$city(id, Berlin)
$\mathbf{cfd_2}$: [city=Hamburg]$\rightarrow$ [zip=20000]	$\mathbf{f_2}$:city(id, Hamburg)$\Rightarrow$zip(id, 20000)
$\mathbf{md_3}$: R[name, zip]=R[name,zip]$\rightarrow$R[city]$\leftrightharpoons$R[city]	$\mathbf{f_3}$:name(id_1,n) $\wedge$ zip(id_1,z) $\wedge$ name(id_2,n) $\wedge$ zip(id_2,z) $\Rightarrow$ matchCity(id_1, id_2)

MARKOV LOGIC TO FACTOR-GRAPH TRANSLATION



Semistructured Data Model and Schema

Data Model

```
<value>  ::= <atom> | <array> | <record>
<atom>   ::= <string> | <binary> | <double> |
             <date> | <boolean> | 'null' | ...
<array>  ::= '[' ( <value> (',' <value>)* )? ']'
<record> ::= '{' ( <field> (',' <field>)* )? '}'
<field>  ::= <name> ':' <value>
```

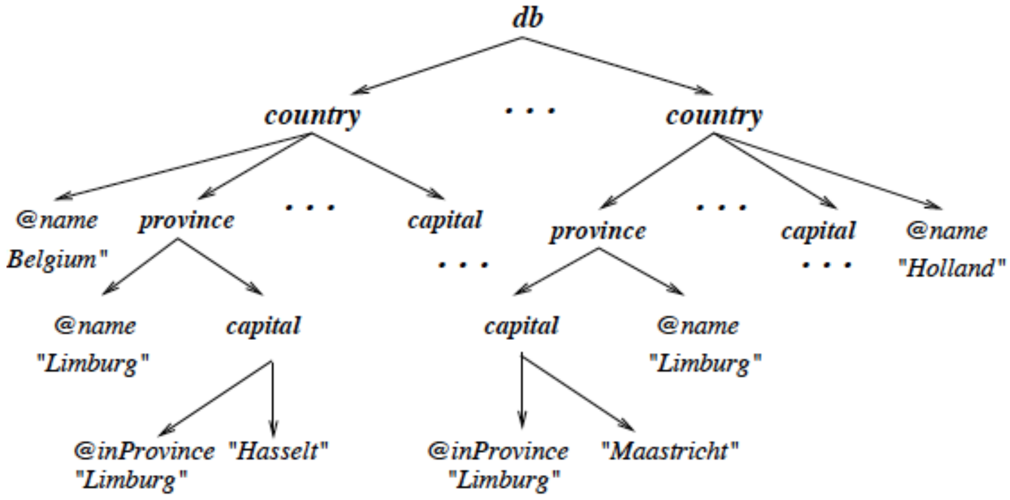
Schema

```
<schema> ::= <basic> '?'? ('|' <schema>)*
<basic>  ::= <atom> | <array> | <record>
             | 'nonnull' | 'any'
<atom>   ::= 'string' | 'double' | 'null' | ...
<array>  ::= '[' ( <schema> (',' <schema>)*
                   '...'? )? ']'
<record> ::= '{' ( <field> (',' <field>)* )? '}'
<field>  ::= (<name> '?'? | '*') (':' <schema>)?
```

From: “Jaql: A scripting language for large scale semistructured data analysis”

Semistructured Data: FD Definition

Functional dependencies on semistructured data [21] is an expression of the form $\varphi: L \rightarrow R$, where L and R are paths (similar to the XPATH notation). Semistructured data satisfies φ iff for any tree tuples t_1, t_2, if $t_1.L = t_2.L$, then $t_1.R = t_2.R$.



FDs in Clausal Form

$$\varphi: L \rightarrow R \xrightarrow{\text{USES}} \omega: P_c(P_t(S \rightarrow \epsilon))$$

RELATIVE KEY DEFINITION

EXPRESSED IN CLAUSAL FORM

$$\forall P_{c1}, P_{t1}, P_{c2}, P_{t2}, L, R_1, R_2 \Big(P_{c1}(P_{t1}(L \rightarrow \epsilon)) \wedge P_{c2}(P_{t2}(L \rightarrow \epsilon)) \\ \wedge P_{c1}(P_{t1}(R_1 \rightarrow \epsilon)) \wedge P_{c2}(P_{t2}(R_2 \rightarrow \epsilon)) \Rightarrow R_1 = R_2 \Big)$$

The *clausal form* of φ will be useful for the later translation into the Markov logic program:

$$\forall P_{c1}, P_{t1}, P_{c2}, P_{t2}, L, R_1, R_2 \Big(\neg P_{c1}(P_{t1}(L \rightarrow \epsilon)) \vee \\ \neg P_{c2}(P_{t2}(L \rightarrow \epsilon)) \vee \neg P_{c1}(P_{t1}(R_1 \rightarrow \epsilon)) \vee \neg P_{c2}(P_{t2}(R_2 \rightarrow \epsilon)) \\ \vee R_1 = R_2 \Big)$$

Compilation into Markov Logic

FD φ: L$\longrightarrow$R

CLAUSAL FORM

$$\forall P_{c1}, P_{t1}, P_{c2}, P_{t2}, L, R_1, R_2 \Big(\neg P_{c1}(P_{t1}(L \rightarrow \epsilon)) \vee \neg P_{c2}(P_{t2}(L \rightarrow \epsilon)) \vee \neg P_{c1}(P_{t1}(R_1 \rightarrow \epsilon)) \vee \neg P_{c2}(P_{t2}(R_2 \rightarrow \epsilon)) \vee R_1 = R_2 \Big)$$

ATOM $P_c(P_t(S \longrightarrow \epsilon))$

MARKOV LOGIC PREDICATE **S(tree-tuple-id(P_c//P_t), value)**

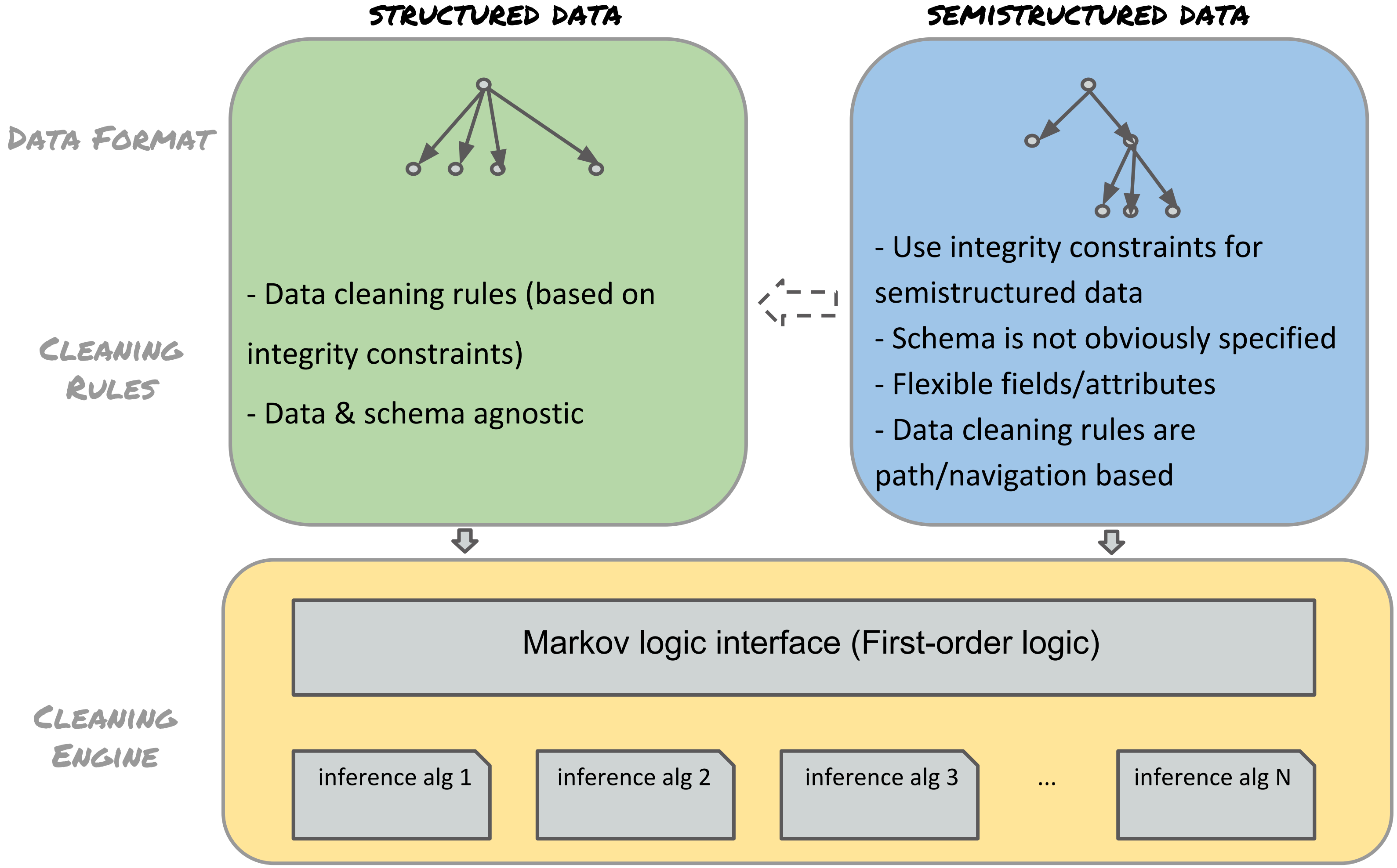
Compilation into Markov Logic

φ: L→R

$$\forall P_{c1}, P_{t1}, P_{c2}, P_{t2}, L, R_1, R_2 \Big(\neg P_{c1}(P_{t1}(L \rightarrow \epsilon)) \vee \neg P_{c2}(P_{t2}(L \rightarrow \epsilon)) \vee \neg P_{c1}(P_{t1}(R_1 \rightarrow \epsilon)) \vee \neg P_{c2}(P_{t2}(R_2 \rightarrow \epsilon)) \vee R_1 = R_2 \Big)$$

$$L(id_1(P_c//P_t), l) \wedge L(id_2(P_c//P_t), l) \wedge R(id_1(P_c//P_t), r_1) \wedge R(id_2(P_c//P_t), r_2) \Rightarrow (r_1 = r_2)$$

$$\mathtt{name(id(country1//province1),\ n)} \wedge \mathtt{name(id(country1//province2),\ n)} \wedge \mathtt{capital(id(country1//province1),\ cname_1)} \wedge \mathtt{capital(id(country1//province2),\ cname_2)} \Rightarrow \mathtt{sameCapital(id(country1//province1),\ id(country1//province2))}$$


STRUCTURED DATA
SEMISTRUCTURED DATA
DATA FORMAT
CLEANING RULES
- Data cleaning rules (based on integrity constraints)
- Data & schema agnostic
- Use integrity constraints for semistructured data
- Schema is not obviously specified
- Flexible fields/attributes
- Data cleaning rules are path/navigation based
CLEANING ENGINE
Markov logic interface (First-order logic)
inference alg 1
inference alg 2
inference alg 3
...
inference alg N

Evaluation Strategy

PARAMETERS

METRICS

	Noise %	Data size	Inference parameters	Various data cleaning rules
Method quality (F_1-score)				✓
Method accuracy	✓		✓	
Efficiency (runtime and memory consumption)		✓		

Initial Experiments

Method quality:

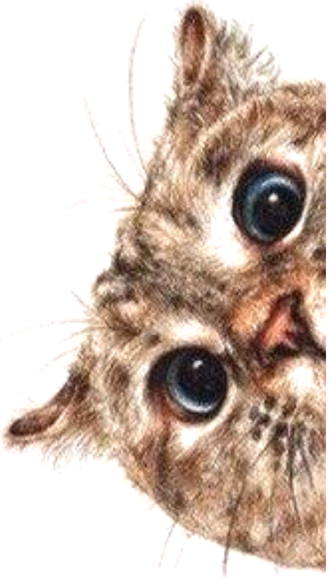
Cleaning relational data: Deduplication (Matching Dependencies) + Improving Accuracy (Conditional Functional Dependencies)

	Data repairing with CFDs									
%noise	ADDR	CITY	COND	COUNT	HOSPNAME	MEASURE	PHONE	STATE	ZIP	**Data repair with CFD and MD**
2	0.9952	0.9944	0.9976	0.9941	1.0	0.9954	1.0	0.9954	1.0	**0.9976**
4	0.9944	0.9988	1.0	0.9978	0.9964	0.9899	0.9963	0.9957	0.9929	**0.9957**
6	0.9964	0.9967	0.9969	0.996	0.9963	0.997	0.9947	0.9943	0.9971	**0.9992**
8	0.9978	0.9931	0.9942	0.9955	0.9979	0.9976	0.997	0.9948	0.9977	**0.9948**
10	0.9949	0.9942	0.9955	0.9995	0.9972	0.9944	0.9951	0.9931	0.9977	**0.9953**
20	0.9963	0.996	0.9943	0.9957	0.995	0.9944	0.9955	0.9965	0.9959	**0.9968**
30	0.9963	0.9966	0.9962	0.9971	0.9954	0.9961	0.9971	0.9944	0.9962	**0.9972**

Table 3: Detailed HOSP evaluation: F_1 measure of each attribute and noise per cent. Repair and matching performed separately from each other.

Outlook

SEE MY PHD REPORT...



Example (how data quality rules influence each other)

Data cleaning based on functional dependencies

fd1: [A] -> [B]
fd2:[B] -> [C]

dirty data D_0

	A	B	C
t1	a1	b1	c1
t2	a2	b2	c3
t3	a1	null	c1

repaired data D_1

	A	B	C
t1	a1	b1	c1
t2	a2	b2	c3
t3	a1	b1	c1

deduplicated data D_2

	A	B	C
t1'=t1+t3	a1	b1	c1
t2	a2	b2	c3

Null value imputation t3[B]

tuples deduplication t1 and t3

Going top-down

1. What is data quality?
2. What are constraint-based data quality rules?
3. How to compile data quality rules into the probabilistic graphical models?
4. What is Markov logic?
5. How we predict data correction with Markov logic?

What is Data Quality?

5 Dimensions of Data Quality:

1. Consistency
2. Currency
3. Accuracy
4. Deduplication
5. Information Completeness

Conditional Functional Dependency

Definition 2. A **conditional functional dependency** (CFD) defined on the relational schema $R(U)$ is a pair $R(X \rightarrow Y, T_p)$, where

1. $X \rightarrow Y$ is a standard FD;
2. T_p is a set of constraints holding on the particular subset of tuples. $\square$

EXAMPLE:

$\mathsf{cfd}_1 : \mathsf{TRANS}([\textsc{ZipCode}] \rightarrow [\textsc{City}], \ \ \mathsf{t}_1 = (90001 \ \| \ \text{Los Angeles}))$

Matching Dependency

Definition 3. A **Matching Dependency** (MD) for schemas R_1 and R_2 is syntacticly defined as:

$$R_1[X_1] \approx R_2[X_2] \rightarrow R_1[Y_1] \rightleftharpoons R_2[Y_2]$$

where X_1 and X_2 are pairwise compatible sets of attributes in R_1 and R_2, respectively; similarly for Y_1 and Y_2; $\approx$ indicates similar attributes and $\rightleftharpoons$ is called the *matching operator*. $\square$

Example:

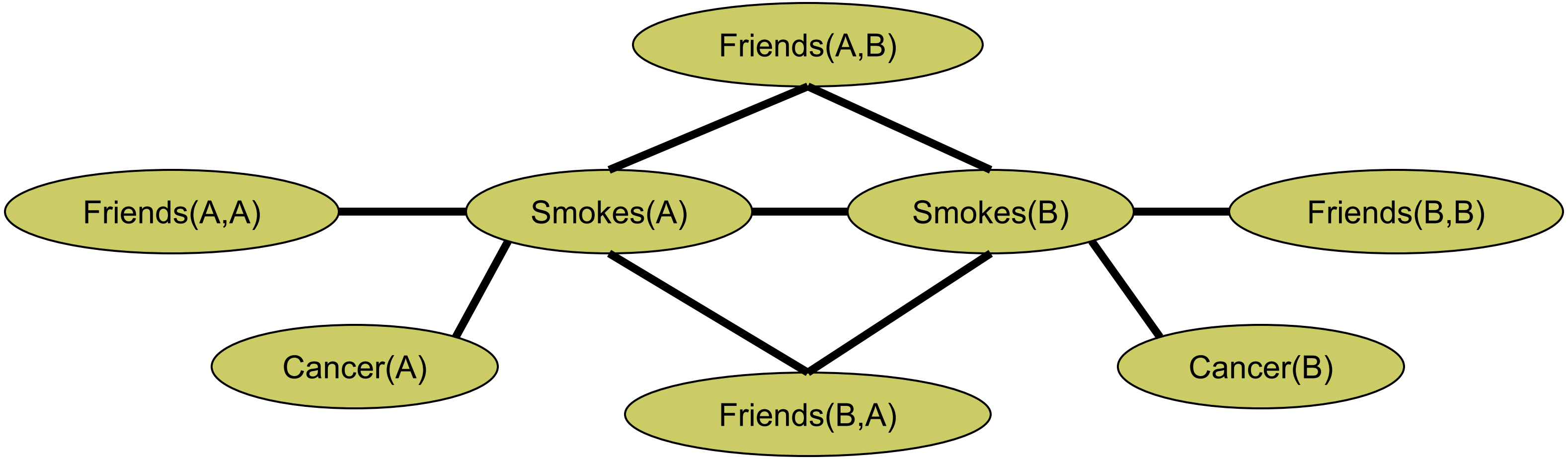
$$\mathsf{md}_1 : \mathsf{TRANS}[\textsc{LastName}, \textsc{City}, \textsc{Street}] = \mathsf{CUST}[\textsc{LastName}, \textsc{City}, \textsc{Street}]$$

$$\wedge \mathsf{TRANS}[\textsc{FirstName}] \approx \mathsf{CUST}[\textsc{FirstName}] \rightarrow \mathsf{TRANS}[\mathsf{FirstName}] \rightleftharpoons \mathsf{CUST}[\mathsf{FirstName}]$$

Example: Friends & Smokers

1.5	$\forall x\ Smokes(x) \Rightarrow Cancer(x)$
1.1	$\forall x, y\ Friends(x,y) \Rightarrow \left(Smokes(x) \Leftrightarrow Smokes(y)\right)$

Two constants: **Anna** (A) and **Bob** (B)



Markov Logic Networks

- MLN (weights+FO formulas) is **template** for ground Markov nets
- Probability of a world *x*:

$$P(x) = \frac{1}{Z} \exp\left(\sum_i \underbrace{w_i}_{\text{Weight of formula } i} \underbrace{n_i(x)}_{\text{No. of true groundings of formula } i \text{ in } x} \right)$$

- **Typed** variables and constants greatly reduce size of ground Markov net
- Functions, existential quantifiers, etc.
- Infinite and continuous domains

FD:[C_CUSTKEY -> C_NAME]

customer 1
(C_NAME contains noisy data and missing values)

customer 28
(C_ADDR contains noisy data)

customer 3
(no errors)

O_ORDERKEY	...	C_CUSTKEY	C_NAME	C_ADDR	C_NATION_KEY	C_PHONE
O_ID_1		C_ID_1	Customer#1	A1	N_ID_15	25-989-741-2988
O_ID_100		C_ID_1	NULL	A1	N_ID_15	25-989-741-2988
O_ID_101		C_ID_1	Customer#1xy	A1	N_ID_15	25-989-741-2988
O_ID_128		C_ID_28	Customer#28	A28	N_ID_8	18-774-241-1462
O_ID_229		C_ID_28	Customer#28	A2XXX	N_ID_8	18-774-241-1462
O_ID_3		C_ID_3	Customer#3	A3	N_ID_1	11-719-748-3364
O_ID_103		C_ID_3	Customer#3	A3	N_ID_1	11-719-748-3364
O_ID_255		C_ID_55	Customer#55	A55	N_ID_3	18-774-241-1462
O_ID_149		C_ID_49	Customer#49	A49	N_ID_3	18-774-241-1462
O_ID_14		C_ID_14	Customer#14	A14	N_ID_5	18-774-241-1462

Modeling integrity constraints: Predicates declaration

Database attributes -> Predicates
Type -> Predicate type
Data -> domain data

observed (present in db,)
evidence

hidden (query) predicates
(we want to reason about)

Cust(orderId, customerId)

CustName(orderId, customerName)

SameName(customerName, customerName)

CustAddr(orderId, addr)

SameAddr(addr, addr)

CustNation(orderId, nationId)

SameNation(nationId, nationId)

CustPhone(orderId, phoneNr)

SamePhone(phoneNr, phoneNr)

MayBeSimilar(orderId, orderId)

Modeling integrity constraints: First-order logic formulas

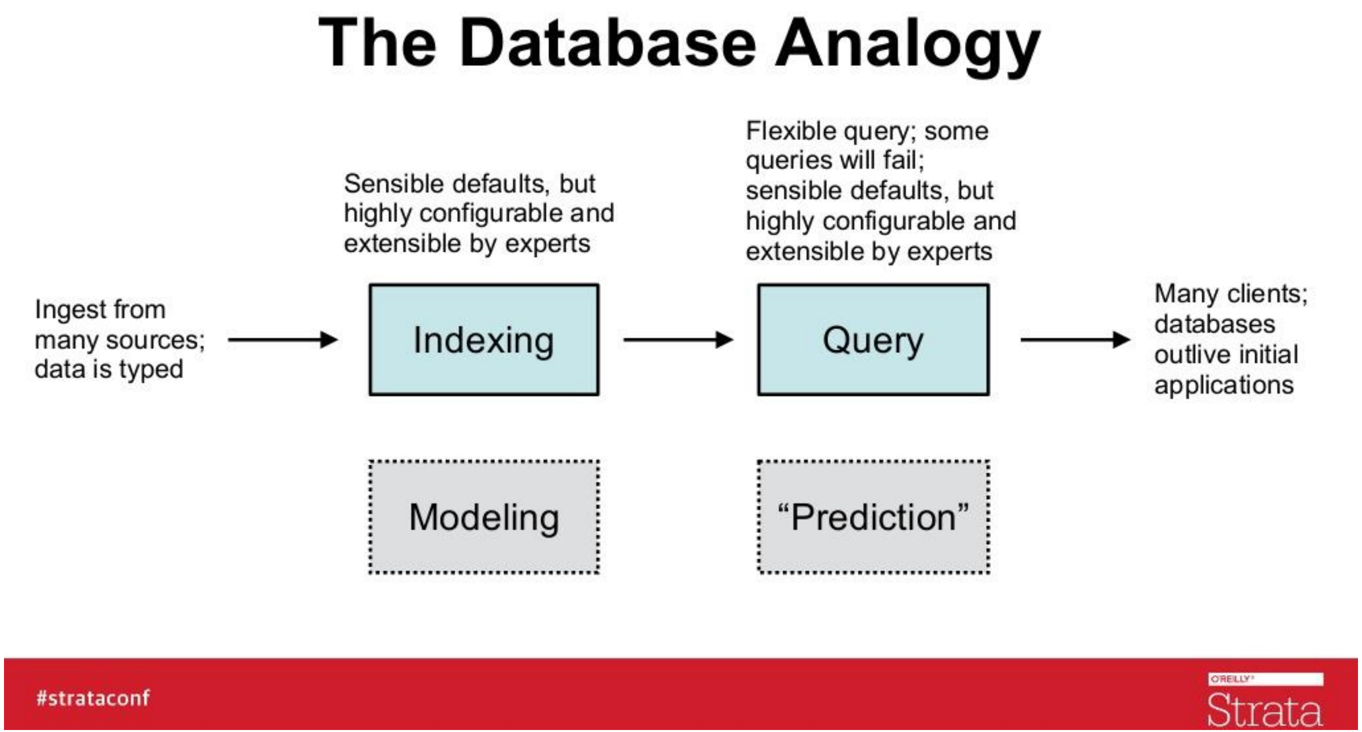
1.0 Cust(id1, x) ^ Cust(id2, x) ^ CustName(id1, n1) ^ CustName(id2, n2) => (MayBeSimilar(id1, id2) ^ SameName(n1, n2))

0.9 Cust(id1, x) ^ Cust(id2, x) ^ MayBeSimilar(id1, id2) ^ (CustName(id1, n) v CustName(id2, n)) => (MayHaveName(id1, n) ^ MayHaveName(id2, n))

Modeling equality

Our trick

How we predict data correction with Markov logic?



https://www.slideshare.net/slideshow/embed_code/31310497

MAP/MPE Inference

- **Problem:** Find most likely state of world given evidence

$$\arg\max_{y} \; P(y \mid x)$$

Query

Evidence

HIDDEN PREDICATE THAT DENOTES ERROR CORRECTION

DATA FROM THE DATA LAYER

MAP/MPE Inference

- **Problem:** Find most likely state of world given evidence

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_i w_i n_i(x)\right) \Rightarrow \arg\max_y \sum_i w_i n_i(x, y)$$

Dirty Data:

CUSTOMERS NAME IS MISSING OR INCORRECT

O_ORDERKEY	...	C_CUSTKEY	C_NAME	C_ADDR	C_NATION_KEY	C_PHONE
O_ID_1		C_ID_1	Customer#1	A1	N_ID_15	25-989-741-2988
O_ID_100		C_ID_1	NULL	A1	N_ID_15	25-989-741-2988
O_ID_101		C_ID_1	Customer#1xy	A1	N_ID_15	25-989-741-2988

DATA (EVIDENCE) FROM DATABASE

Customer(O_ID_1, C_ID_1)
CustName(O_ID_1, Customer#1)
CustAddr(O_ID_1, A1)
CustNation(O_ID_1, N_ID_15)
CustPhone(O_ID_1, PHONE_25-989-741-2988)

Customer(**O_ID_100**, C_ID_1)
/* name is missing */
CustAddr(O_ID_100, A1)
CustNation(O_ID_100, N_ID_15)
CustPhone(O_ID_100, PHONE_25-989-741-2988)

Customer(O_ID_101, C_ID_1)
CustName(O_ID_101, Customer#1xy)
CustAddr(O_ID_101, A1)
CustNation(O_ID_101, N_ID_15)
CustPhone(O_ID_101, PHONE_25-989-741-2988)

INFERENCE (HIDDEN PREDICATES) RESULTS

MayBeSimilar(O_ID_101,**O_ID_100**) 0.817574
MayBeSimilar(**O_ID_100**,O_ID_1) 0.817574

SameName(Customer#1xy,Customer#1) 0.622459

MayHaveName(**O_ID_100**,Customer'000000001) 0.588274
MayHaveName(**O_ID_100**,Customer'000000001xy) 0.588274

JOINT DEDUPLICATION AND NULL VALUE COMPUTATION

Markov Logic Advantages for Data Cleaning

1. Defining “soft” and “hard” data cleaning rules
2. Joint inference of the hidden predicates (where are errors?)
3. Ability to model contradicting rules

Conflicting rules

CFD: **$R([A]\text{->}[B], T_0)$**

where T_0 consists of the two pattern tuples ($\_ || b$) and ($\_ || c$) with $b \neq c$.

Meaning: The first pattern requires t[B]=*b*, at the same time the second pattern says that t[B] must be *c*.

Can we model such conflicting rules with Markov Logic?

Modeling *extended* conditional dependencies (I)

CFDs extended allowing disjunction, negation and pattern tuples:

eCFD: **R([city]->[AC], (NY||{212, 718, 646}))**

"for tuples t such that t[city]=NY, their area code should come from a set {212, 718, 646}"

Can we model such extended constraints with Markov Logic?

Modeling *extended* conditional dependencies (II)

CFDs extended allowing predicate-patterns:

CFD: **R([book]->[price], (_||>0))**

“for every book, its price should be greater than zero”

Can we model such extended constraints with Markov Logic?

Modeling *extended* conditional dependencies (III)

CFDs with cardinality constraints and synonym rules:

CFD^{c}: **R([country, zip]->[street], (UK, _| |_), c) c$\subseteq\mathbb{N}$**

"For all tuples t such that t[country, zip]$\asymp$(UK,_), all tuples that are synonymous to t in county and zip can only have at most c distinct street values."

Can we model such extended constraints with Markov Logic?

$\asymp$ *is an operator, defining that either [country, zip]=(UK,_) or [country, zip]=(_,_)*

Denial Constraints

¬(G(g,f,n,r,c,a,s), G(g',f',n',r',c',a',s'), (r=r'), (c='NYC'), (c'≠'NYC'), (s'>s))

states: *Every time there are two employees with the same rank, one in NYC and one in a different city, there is a violation if the salary of the second is greater than the salary of the first.*

Can we model denial constraints with Markov Logic?

Denial constraints are the first-order formulae:

$\forall \mathbf{x} \neg(\mathbf{R_1}(\mathbf{x_1}) \wedge \ldots \wedge \mathbf{R_n}(\mathbf{x_n}) \wedge \mathbf{P_1} \wedge \ldots \wedge \mathbf{P_m})$ where $\mathbf{R_i}$ is a relational atom and $\mathbf{P_j}$ is a predicate.

Quality Rules with Operators

(Build-in) predicates containing the following operators: **=,<, >,≈,≤,≥,≠**

CAN BE EXPRESSED AS DENIAL CONSTRAINTS

Quality improvement for schemaless data

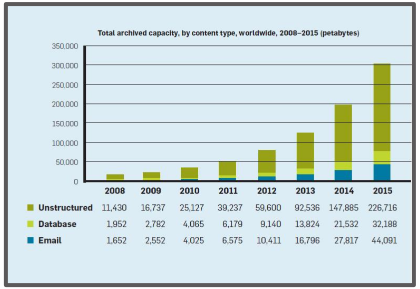
ONGOING WORK...

Cleaning semi-structured data

data analyzed have fundamental quality problems

"garbage in, garbage out."

data is in semi-structured form



need for a new approaches in **data cleaning** for **non-relational** use-cases

Schema vs. Schemaless

Database Schema	**Schemaless data**
Describes a logical structure and semantic of data	Denotes data where the logical structure and semantic are (yet) unknown.

	Title	Year	Length	Filmtype
r_0	Amelie	2001	NULL	color
r_1	Babe	1995	89	color
r_2	Cocktail	1988	NULL	color
r_4	NULL	1996	134	color
r_5	Frenzy	NULL	116	color
r_6	Gaslight	1944	114	bw
r_7	Hamlet	1990	242	NULL
r_8	NULL	1958	100	color
r_9	John Q	2002	116	color

	Title	Year	Filmtype
r_0	Amelie	2001	color

	Title	Year	Length	Filmtype
r_1	Babe	1995	89	color

	Title	Year	Filmtype	Noise-4
r_2	Cocktail	1988	color	noiseval

	Length
r_3	75

	Year	Length	Filmtype	Noise-2
r_4	1996	134	color	noiseval

	Title	Filmtype	Length	Noise-4	Noise-5
r_5	Frenzy	116	color	noiseval	noiseval

	Title	Year	Length	Filmtype	Noise-1
r_6	Gaslight	1944	114	bw	noiseval

	Title	Year	Length
r_7	Hamlet	1990	242

	Year	Filmtype	Length	Noise-5
r_8	1958	color	100	noiseval

	Title	Year	Length	Filmtype	Noise-2	Noise-3
r_9	John Q	2002	116	color	noiseval	noiseval

Cafarella, Michael J., Dan Suciu, and Oren Etzioni. "Navigating Extracted Data with Schema Discovery." *WebDB*. 2007.

What is Markov Logic?

Markov Logic is a knowledge representation system that combines:

- → Weighted First-Order Logic
- → Markov Networks (undirected prob. graph. models)

What are constraint-based data quality rules?

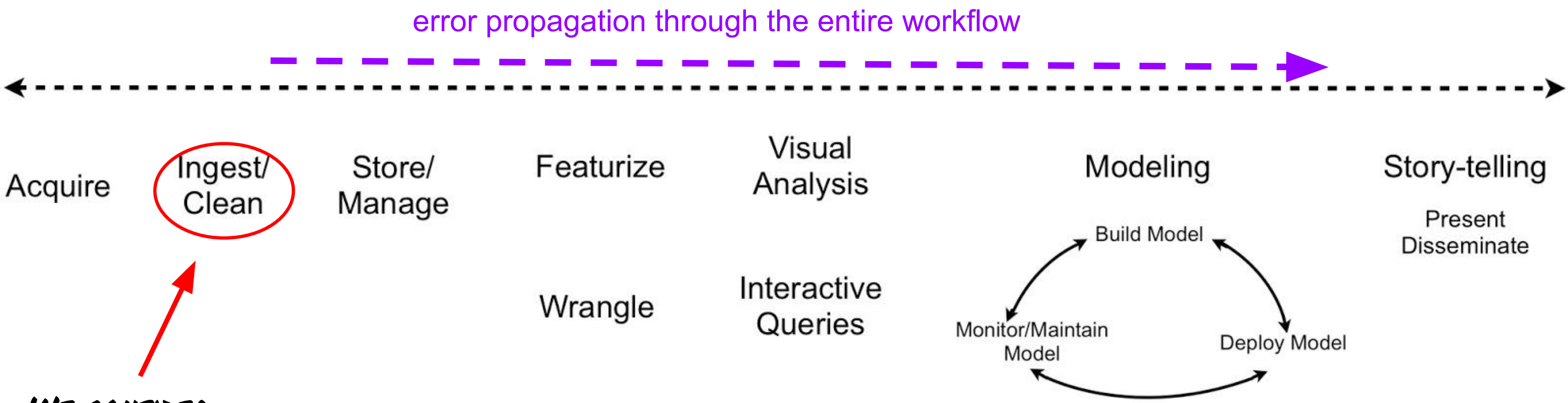
We use data dependencies to formulate data quality rules.

ADVANTAGE:

Data dependencies specify **semantic** of data in a **declarative way**

Example: Functional Dependencies: [A -> B_1, B_2,...]

Two Challenges



We consider:

- ★ Structured data
- ★ Semistructured data

http://radar.oreilly.com/2013/09/data-analysis-just-one-component-of-the-data-science-workflow.html?cmp=ot-strata-na-article-na_62643

Two Requirements on Data Cleaning Systems:

1
- **Joint modeling** for data cleaning rules

2
- Defining **“soft” and “hard” data cleaning rules**
- Modeling **conflicting rules**
- **Flexible rules** modeling (non-constraints rules)