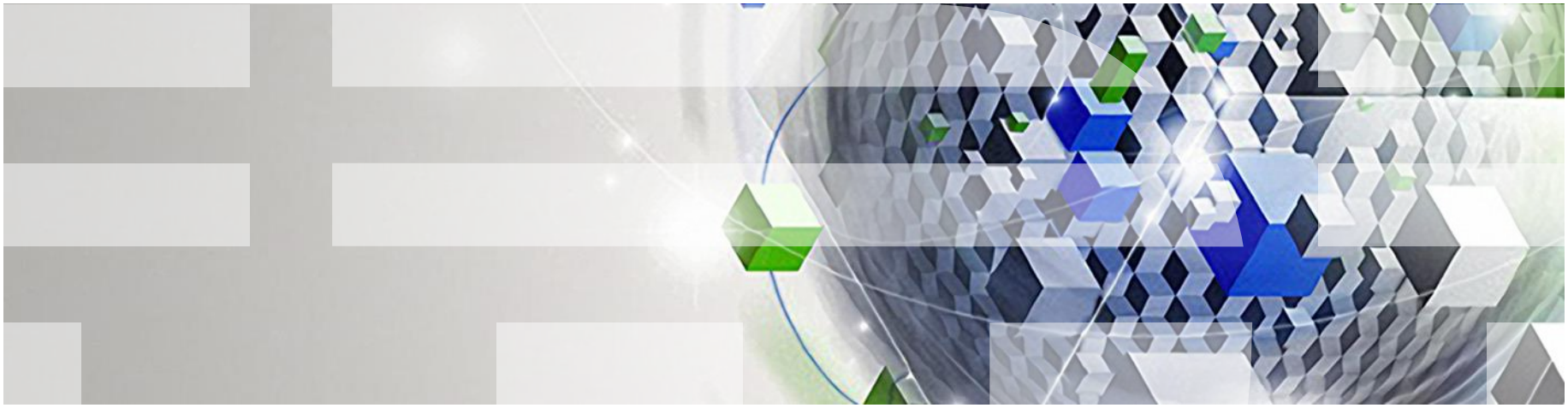
Bringing Order to Big Data

Jarek Szlichta

Conducted research was partially supported by IBM CAS



Data, Data Everywhere…

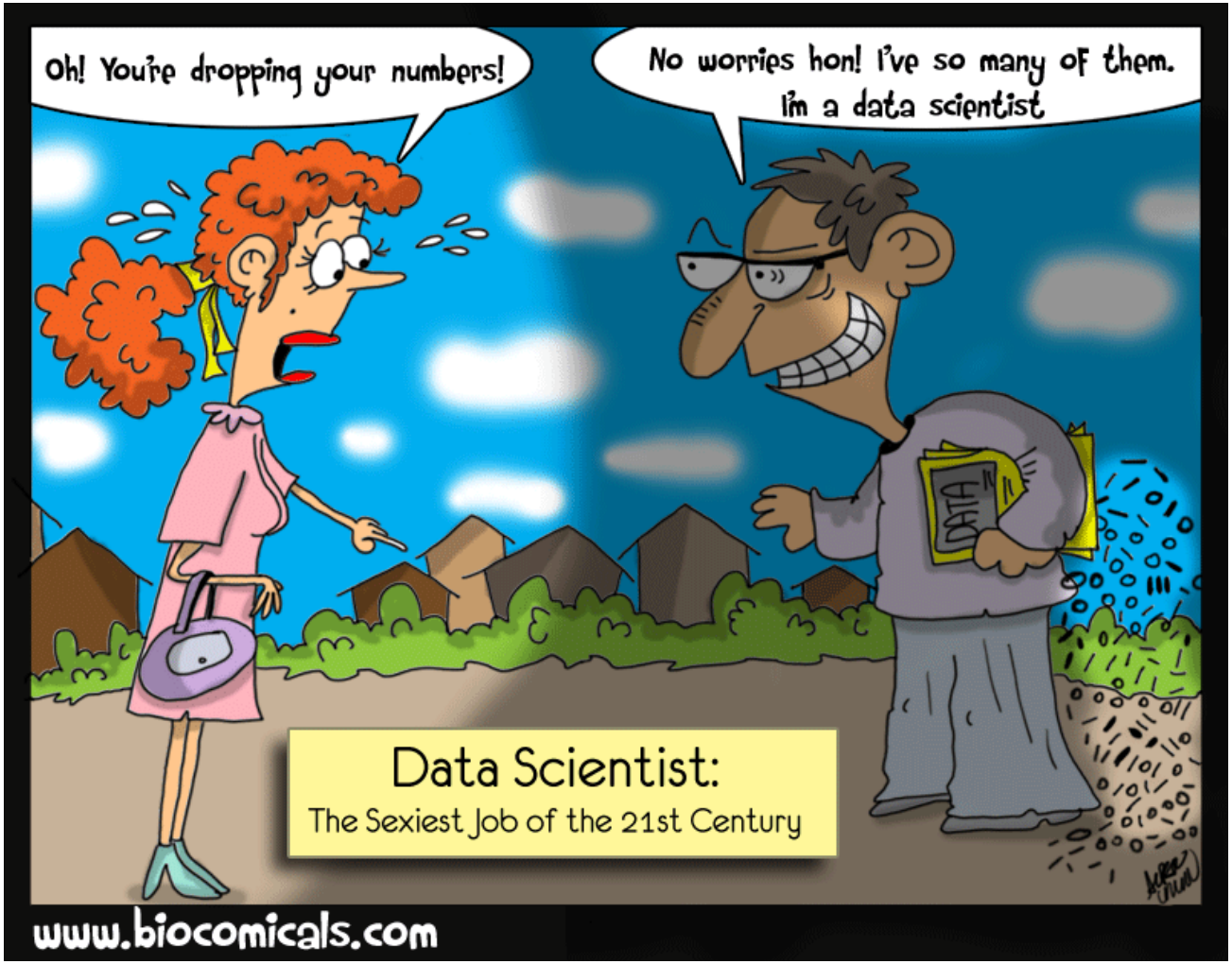
- Open data

- Business Data

- Web Data

- Available at different

Big Data to Data Science



Data Scientist: The Sexiest Job of the 21st Century

Harvard Business Review Oct. 2012

(c) 2012 Biocomicals by Dr. Alper Uzon

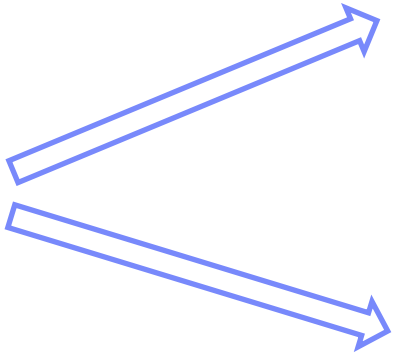
Can we take data science out of the realm of geekdom and make it a trusted, respected profession?

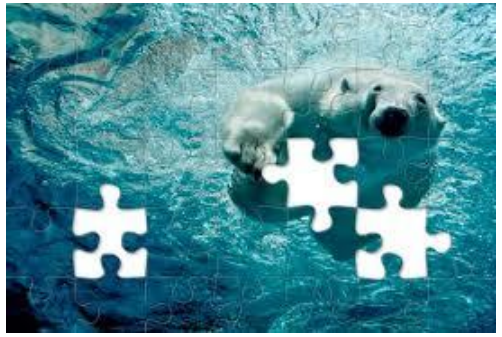
What is Big Data Integration?

- Big data integration (BDI) = big data + data integration.
- Data Integration – easy access to multiple data sources
  - Warehouse – materialized data, easier querying, consistency issues.
- Big data.. all about V's..
  - Heterogeneity – huge **variety** of data, of questionable **veracity**,
  - Size – large **volume** of data,
  - Utility – data of considerable **value**.
- Big data in the context of data integration – still about V's.

BDI – Why is it Challenging?

- Data integration = solving lots of *jigsaw puzzles*
  - Each piece of a puzzle comes from some **source** we want to integrate,
  - Big data integration – **big**, **complex**, **messy** puzzles, e.g., missing, duplicate.





Outline

I. Continuous Data Cleaning [ICDE2014]

II. Business Intelligence (BI)

- Fundamentals [VLDB'12, AMW'12, AMW'13]
- Optimization of Queries for BI [VLDB'13, EDBT'11, EDBT'14]

III. Future Themes

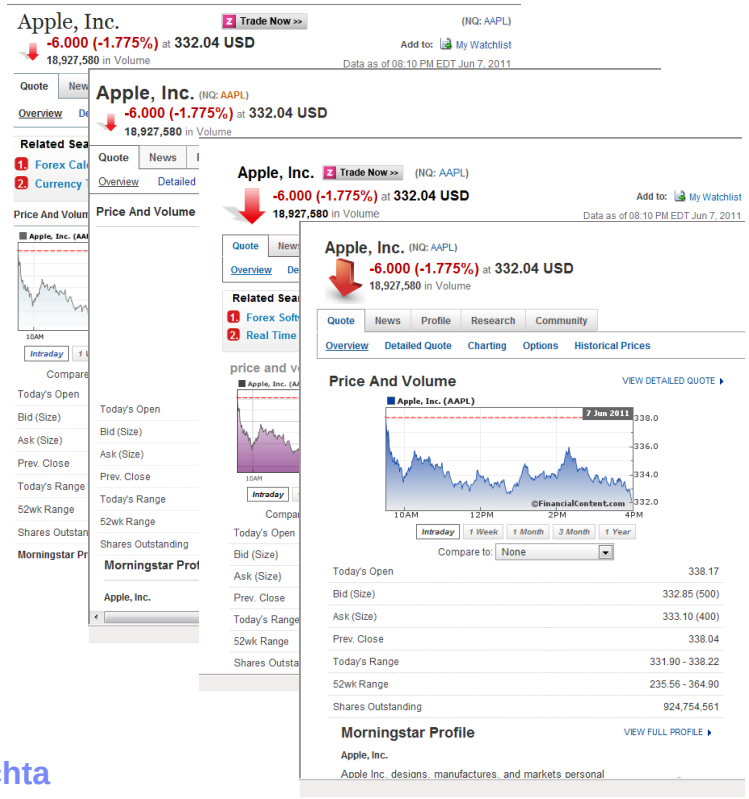
IV. Conclusions

II. Poor Data Quality

- Data quality is an increasingly pervasive problem for organizations.
- Operational inefficiencies and mistakes.
  - Major Canadian bank faxed sensitive customer data to US junk yard (incorrect extra digit in fax number).
- Waste of money and time [Gartner Research Report `09]
  - Organizations in Fortune 1,000 companies lose, on average, $8M annually due to poor quality data,
  - 25% of critical data in the Fortune 1,000 companies is inaccurate.

Sample Sources of Inconsistences

- Pure (human) error.
- Lack or violations of constraints (rules).
- Data integration.
- Changes in business rules.
- Copying on erroneous data
  - "A lie told often enough becomes the truth." – V. Lenin,
  - Copying or Data Sharing Between Deep-Web Sources.



Integrity Constraints

- Integrity constraints are the primary means for preserving data integrity
  - If [Salary < 80K] then [credit rate = 7%].
- Data may become inconsistent with the constraints.
- Options:
  - Assume the constraints are correct, and repair the data:
    - Bohannon et al. [SIGMOD05], Cong et al. [VLDB07], Kolahi et al. [ICDT09], Dallachiesa et al., (NADEEF) [SIGMOD13], Geerts et al. (LLUNATIC) [VLDB13], Yakout et al. [SIGMOD13].
  - Repair the data and constraints to fit:
    - Miller et al. [ICDE11],
    - Effective for static data,
    - Start cleaning process from scratch,
    - Manually setting up parameters.

Motivating Example

- Consider table *Employee_salary* and the constraints:

#	First_name	Surname	BranchID	Salary	Zip	City
t1	James	Brown	107	70K	33101	Miami
t2	Craig	Rosberg	107	50K	33101	Miami
…	…	…	….	…	…	…

FD: [*Zip*] → [City].

- **Functional Dependency** (FD) states that whenever two tuples agree on values for *Zip*, they must agree on values for attribute *City*.
- Similarly, one can define FD: [*Surname, First_name*] → [*Salary*].

Motivating Example

- Consider table *Employee_salary* and the constraints:

FD1: [Surname, First_name] → [Salary].

- If the company decides to consolidate all employee salaries across all the branches to a central data warehouse (ETL), ambiguity among employee salaries can arise.

#	First_name	Surname	BranchID	Salary	Zip	City
t1	**James**	**Brown**	107	**70K**	33101	Miami
t2	Craig	Rosberg	107	50K	33101	Miami
t3	**James**	**Brown**	308	**60K**	10003	NY
t4	**Monica**	**Johnson**	308	**60K**	10003	Boston
t5	**James**	**Brown**	401	**80K**	10003	NY
t6	**Monica**	**Johnson**	401	**100K**	10003	NY

Constraint Repair

- Consider table *Employee_salary* and the constraints:

FD1: [Surname, First_name] → [Salary].

FD2: [Surname, First_name, BranchID] → [Salary]

- A possible constraint repair would be to add BranchID to the left side of FD1.

#	First_name	Surname	BranchID	Salary	Zip	City
t1	James	Brown	107	70K	33101	Miami
t2	Craig	Rosberg	107	50K	33101	Miami
t3	James	Brown	308	60K	10003	NY
t4	Monica	Johnson	308	60K	10003	Boston
t5	James	Brown	401	80K	10003	NY
t6	Monica	Johnson	401	100K	10003	NY

Data Repair

- Consider table *Employee_salary* and the constraint:

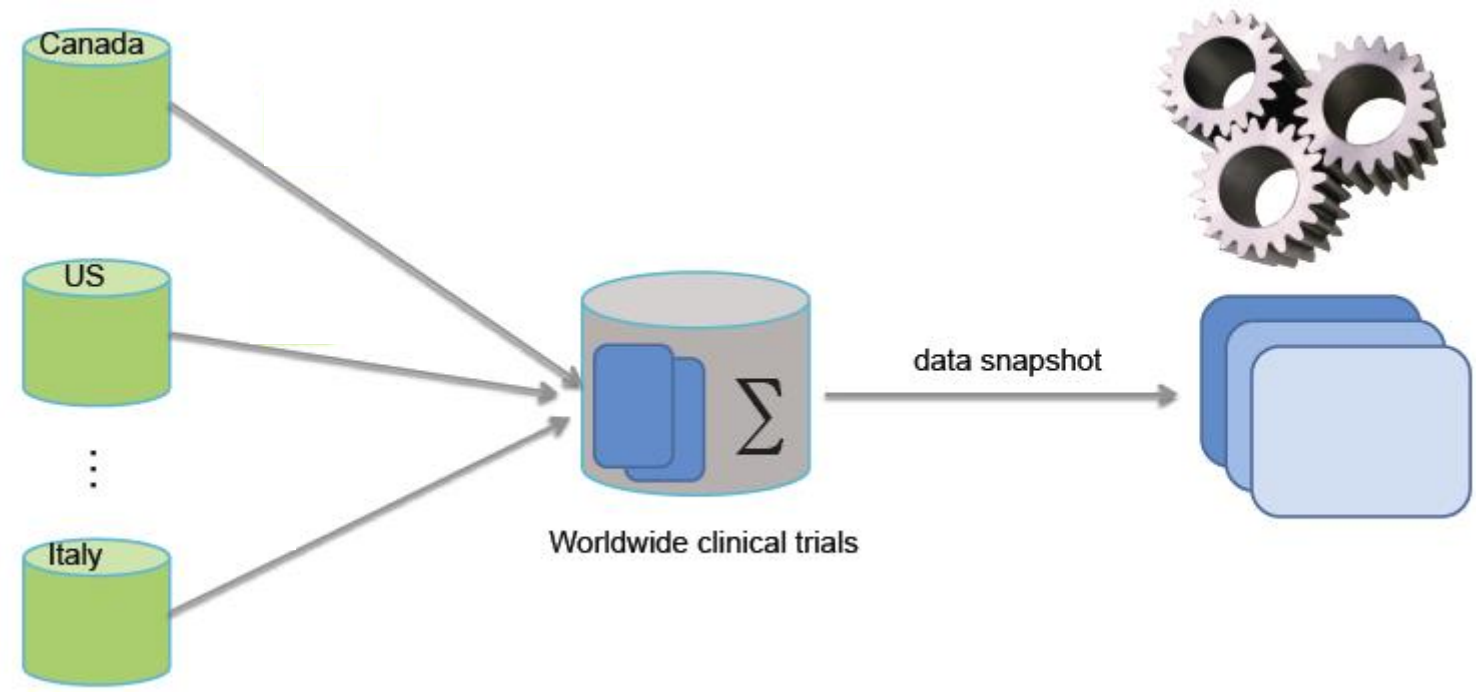
  FD3: [Zip] → [City] 

  - There is more support for "NY" in the data, since t3, t4 and t6 are all in agreement.

#	First_name	Surname	BranchID	Salary	Zip	City
t1	James	Brown	107	70K	33101	Miami
t2	Craig	Rosberg	107	50K	33101	Miami
t3	James	Brown	308	60K	10001	NY
t4	Monica	Johnson	308	60K	10001	**NY**
t5	James	Brown	401	80K	10001	NY
t6	Monica	Johnson	401	100K	10001	NY

Changed from Boston to NY

Dynamic Data Environments

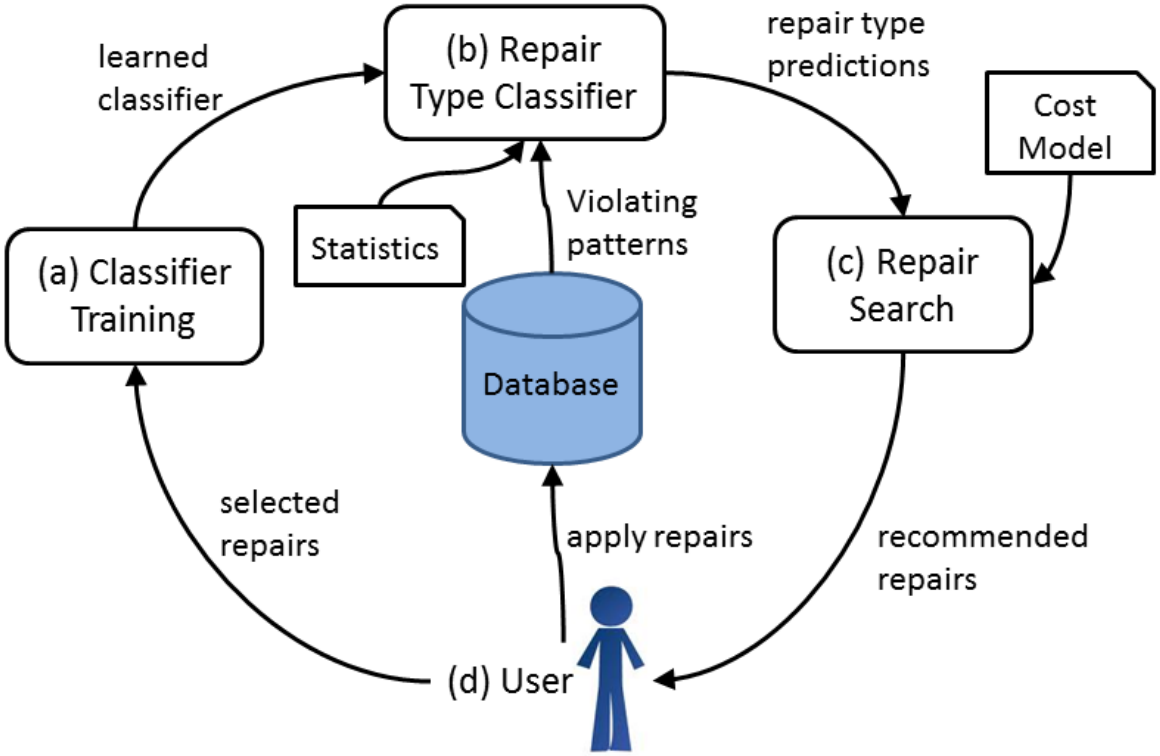


Continuous Data Cleaning Framework

- Considers evolutionary changes in the data and in the constraints.
- Leverage a set of statistics computed over the data.
- Repair **classifier** adapts to user feedback and repair preferences.
- Considers repairs to:
  - FDs,
  - Data,
  - Data and FDs (hybrid repairs)

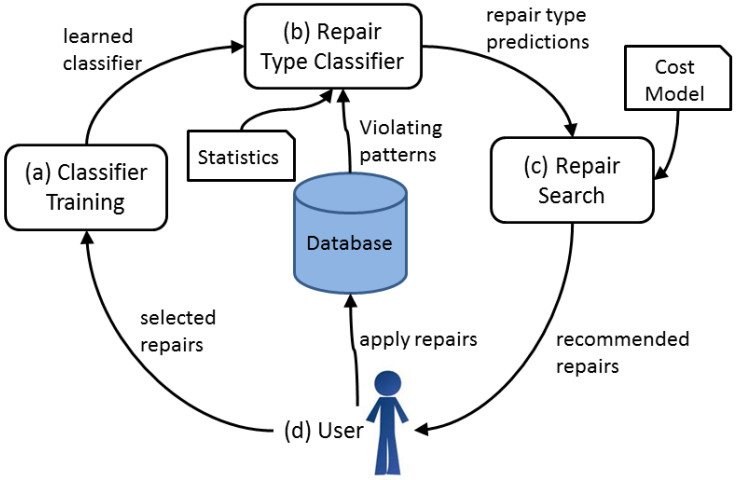
IEEE International Conference on Data Engineering (ICDE) 2014
Continuous Data Cleaning for Dynamic Data Environments.
M. Volkovs, F. Chiang, J. Szlichta, R. Miller.

Framework Overview



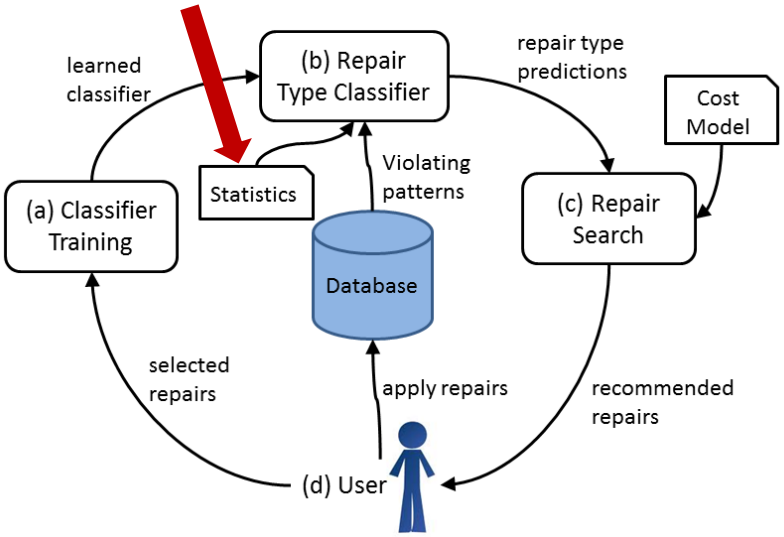
Key Ideas

- Formulate prediction problem as a multi-class classification task.
- Work with tuple patterns wrt FD
  - Assume FD: [zip] → [city],
  - E.g., (*10001*, *NY*) is a pattern.
- Extract violating patterns for each table.
- Predict the likelihood of repair classes for each pattern
  - Compute statistics (for each pattern)
  - Train on a set of target repairs (user applied, repair algorithm).



Sample Statistics

- Stay up-to-date with evolving data distributions and changing constraints.
- Number of violating tuples of FDs relative to size of dataset *N*.
- Stats on pattern *p* across possible data repairs
  - Entropy of (un-) satisfying tuples.
- Stats on *p* across possible FD repairs
  - Fraction of tuples that are not repaired.
- …



Repair Classes

- Class 1 – not repaired (if <50% *p* tuples are repaired)
- Repaired completely by (if all *p* tuples are repaired)

  Class 2 – FD repair,

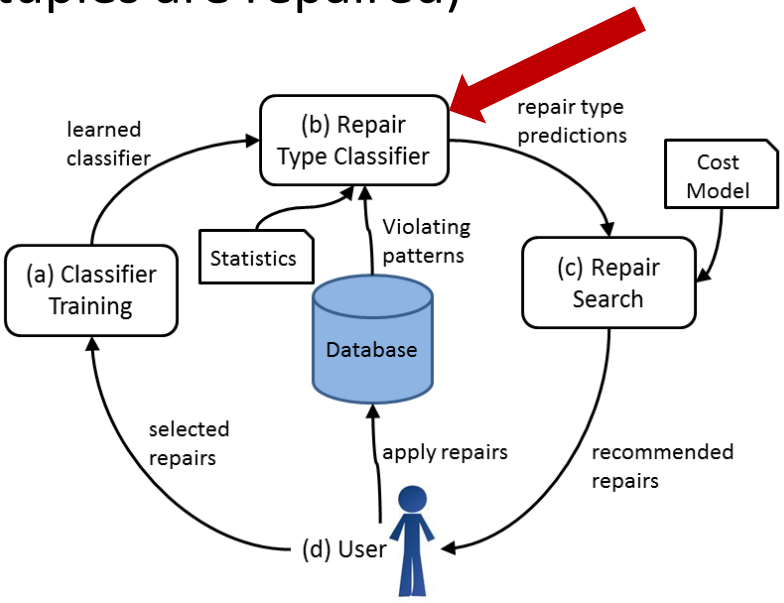
  Class 3 – Data repairs,

  Class 4 – Data and FD repair
- Repaired partially by (if >50% *p* tuples are repaired)

  Class 5 – FD repair,

  Class 6 – data repairs,

  Class 7 – data and FD repair.

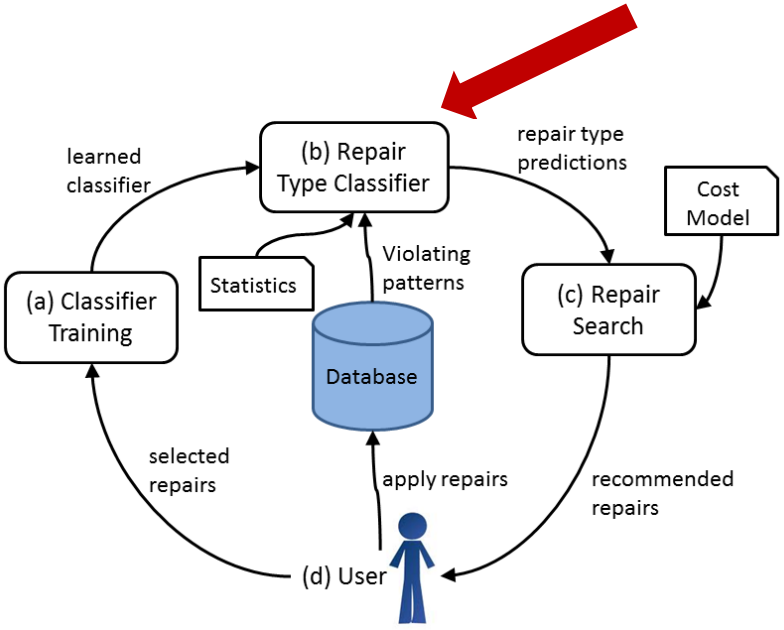


Logistic Regression

- Probability of a class given by:

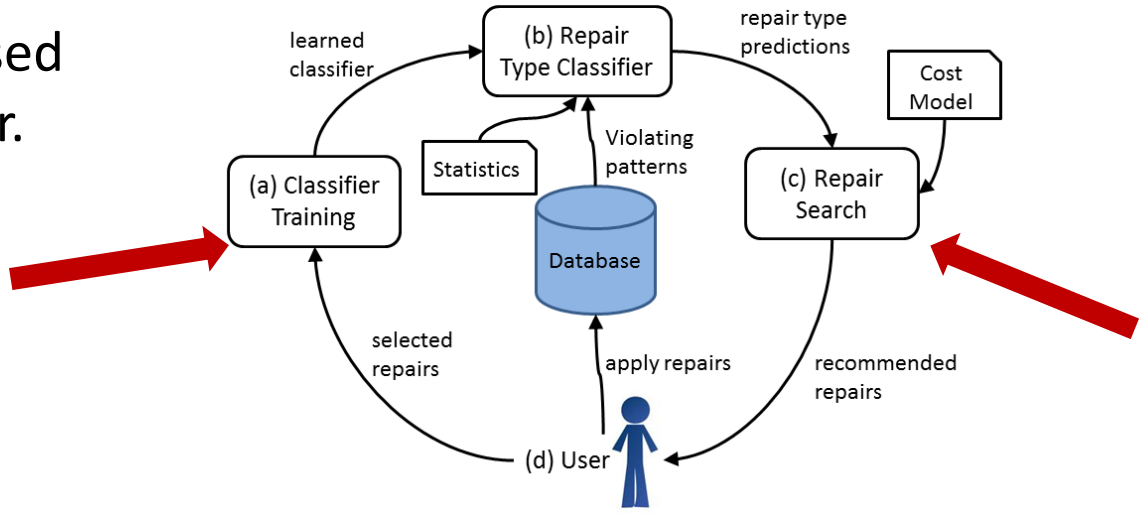
$$Q(p = class_i) = \frac{\exp(W_i * G(p) + b_i)}{\sum_{j=1}^{7} \exp(W_j * G(p) + b_j)}$$

- Minimize the cross entropy objective function between Q and target probability function P
  - Use standard stochastic gradient descent.



Generating Repairs

- Classifier provides a set of class probabilities.
- Repair algorithm filters its search space based on the recommended repair types.
- Can incorporate any repair algorithm supporting data and constraint repairs.
- Recommended repairs are presented to the user.
- Selected repairs are used to re-train the classifier.



Experiments

- Two real data collections (financial data and medical trials: *data.gov*) and TPC-H benchmark.
- An overall average of around 80% *classification accuracy* on real data and synthetic data set.
- After *re-training* phase, our average *classification accuracy* increases by approximately 11%.

	CLASS 1	CLASS 2	CLASS 3	CLASS 4	CLASS 5	CLASS 6	CLASS 7	AVG
FVV-A, CL-A	74.32 ± 2.90	62.74 ± 7.05	96.77 ± 1.74	·	69.18 ± 6.76	·	·	·
FVV-B, CL-A	62.46 ± 2.71	77.78 ± 6.43	97.45 ± 2.01	·	45.00 ± 3.25	·	·	70.67
FVV-B, CL-B	82.34 ± 1.84	80.13 ± 6.95	96.03 ± 1.92	·	68.75 ± 3.47	·	·	81.81
Clinical-A, CL-A	80.26 ± 1.25	89.47 ± 2.13	70.31 ± 5.71	·	73.89 ± 4.89	·	·	·
Clinical-B, CL-A	72.14 ± 1.52	81.63 ± 1.09	52.31 ± 4.43	·	62.71 ± 5.11	·	·	67.19
Clinical-B, CL-B	79.36 ± 1.30	90.01 ± 1.97	71.64 ± 6.22	·	77.47 ± 5.03	·	·	79.62

Evaluation

- Our dynamic framework is able to achieve more accurate repairs than static solutions.
  - We are able to achieve an average 20% improvement in repair accuracy over non-learning approaches.
- We provide performance improvement by 20%.
- Linear dependence allows our system to scale well to large datasets.

Collaborators

- **University of Toronto:**
  - Renée J. Miller
  - Maksims Volkovs
  - Javed Siddique
  - Natasha Prokoshyna
- **McMaster University:**
  - Fei Chiang
- **University of Waterloo**
  - Ihab F. Ilyas
  - Anup Chalamalla
  - Lukasz Golab
- **AT&T Labs**
  - Divesh Srivastava
- **York University & IBM CAS:**
  - Parke Godfrey, Jarek Gryz.
- **IBM LAB:**
  - Calisto Zuzarte,
  - Anastasios Kementsietsidis
  - Wenbin Ma,
  - Weinan Qiu.
- **Technische Universität Wien**
  - Reinhard Pichler

II. Business-Intelligence

- Data warehouses are designed to assist in analysis of business data over a historical period.
- Business-Intelligence (BI) applications have become more complex and data volume have grown.
- The increasing complexity raises performance issues and numerous challenges for query optimization.
- Traditional optimization methods often fail to apply when logical subtleties in database schemas and in queries circumvent them.

Motivation

- Order dependencies (ODs) capture a monotonicity property between attributes.
- FDs play important roles in query optimization [*SIGMOD, Simmen et al., 1996*]. We introduce ODs to use it in similar manner.
- Order plays pivotal roles in the query optimization. Data is often stored sorted by a clustered (tree) index's key.
- An OD can be declared as **integrity constraint** to prescribe which instances are admissible.

Date in TPC-DS Schema

date_dim table:

d_date_sk	d_date	d_year	d_month	d_day
…	…	…	…	…
2000	2010-08-30	2010	08	30
2001	2010-09-31	2010	09	31
2401	2011-01-05	2011	01	05
2402	2011-01-06	2011	01	06
2487	2011-04-01	2011	04	01
…	…	…	…	…

Unidirectional ODs

d_date_sk	d_date	d_year	d_month	d_day
2000	2010-08-30	2010	08	30
2001	2010-09-31	2010	09	31
2401	2011-01-05	2011	01	05
2402	2011-01-06	2011	01	06
2487	2011-04-01	2011	04	01
2488	2011-04-01	2011	04	01

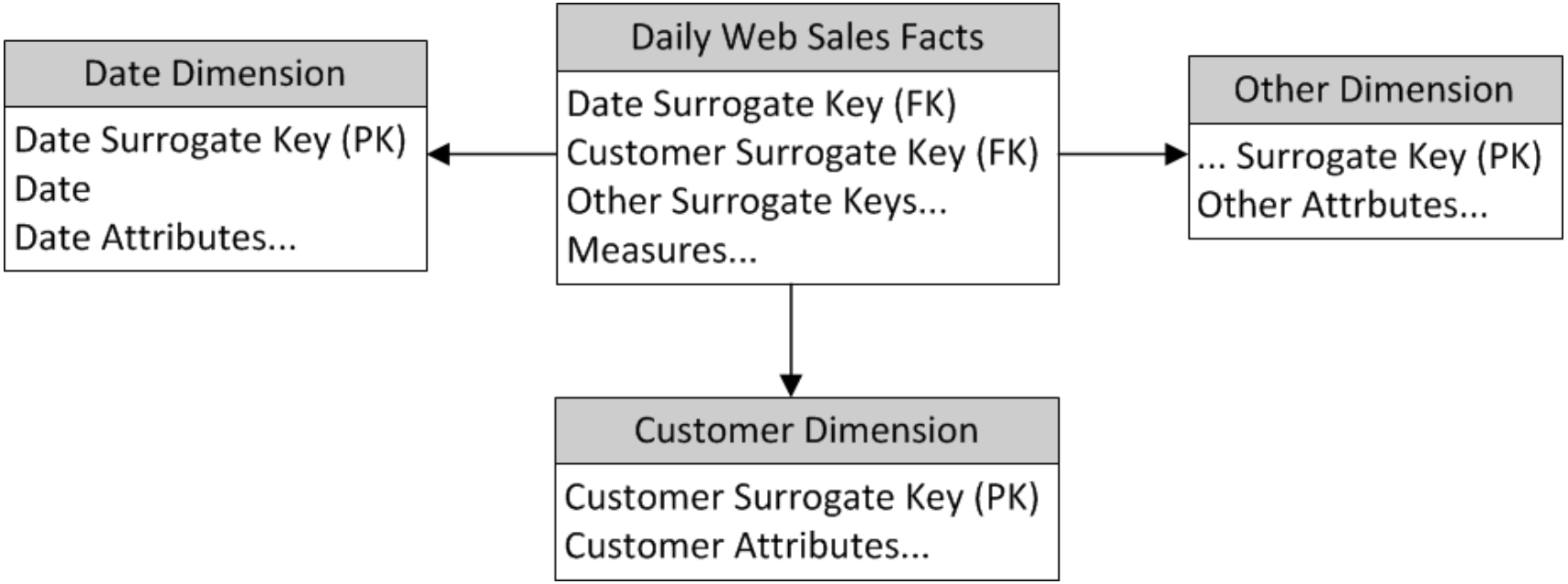
- **dates** $\models [\text{d\_date}] \mapsto [\text{d\_year}, \text{d\_month}, \text{d\_day}]$ (UOD).
- **Unidirectional order dependency** (UOD) states that whenever two tuples grow or agree on values for *d_date*, they must lexicographically grow or agree on values for attributes *d_year, d_month, d_year*.
- However, **dates** $\not\models [\text{d\_date}] \mapsto [\text{d\_year}, \text{d\_day}, \text{month}]$ (UOD).

Example of ODs

date_sk	d_date	d_year	d_month	d_day	d_bucket
2000	2010-08-30	2010	08	30	E
2001	2010-09-31	2010	09	31	E
2401	2011-01-05	2011	01	05	D
2402	2011-01-06	2011	01	06	D
2487	2011-04-01	2011	04	01	D

- 2014: A; …2011: bucket D; 2010: bucket E, …
- **dates** $\models$ [d_year $\downarrow$] $\mapsto$ [d_bucket $\uparrow$] (OD).

Sample of TPC-DS Schema



Query with an Expensive Join

```
select ...
from web_sales W, item I, date_dim D
where W.ws_item_sk = I.i_item_sk and
      I.i_category
            in ('Sports', 'Books', 'Home') and
      W.ws_sold_date_sk = D.d_date_sk and
      D.d_date between
            cast('1999-02-22' as date) and
            (cast('1999-02-22' as date)
                  + 30 days)
...;
```

Eliminating Join in Data Warehouses

- This requires a potentially expensive join between sales fact table and the date dimension table.
- We optimize such queries by removing the join.
- There is an OD which can be declared as integrity constraint, $[\text{d\_date\_sk}] \mapsto [\text{d\_date}]$.
- Two probes can be made into the dimension table to calculate the range of the surrogate keys in the fact table.

Extending Database Technology (EDBT) 2011
Queries on Dates: Fast Yet not Blind.
J. Szlichta, P. Godfrey, J. Gryz, W. Ma, Pawluk, P., W. Qiu, C. Zuzarte

Rewrite of the Query

```
select ...
from web_sales W, item I,
(select min(d_date_sk) as mindate
      from date_dim
      where d_date >=
            cast('1999-02-22' as date))
      as A,
(select max(d_date_sk) as maxdate
      from date_dim
      where d_date <=
            cast('1999-02-22' as date)
                  + 30 days)
      as Z
where ... and
W.ws_sold_date_sk between
      A.mindate and Z.maxdate
...;
```

Implicit ODs

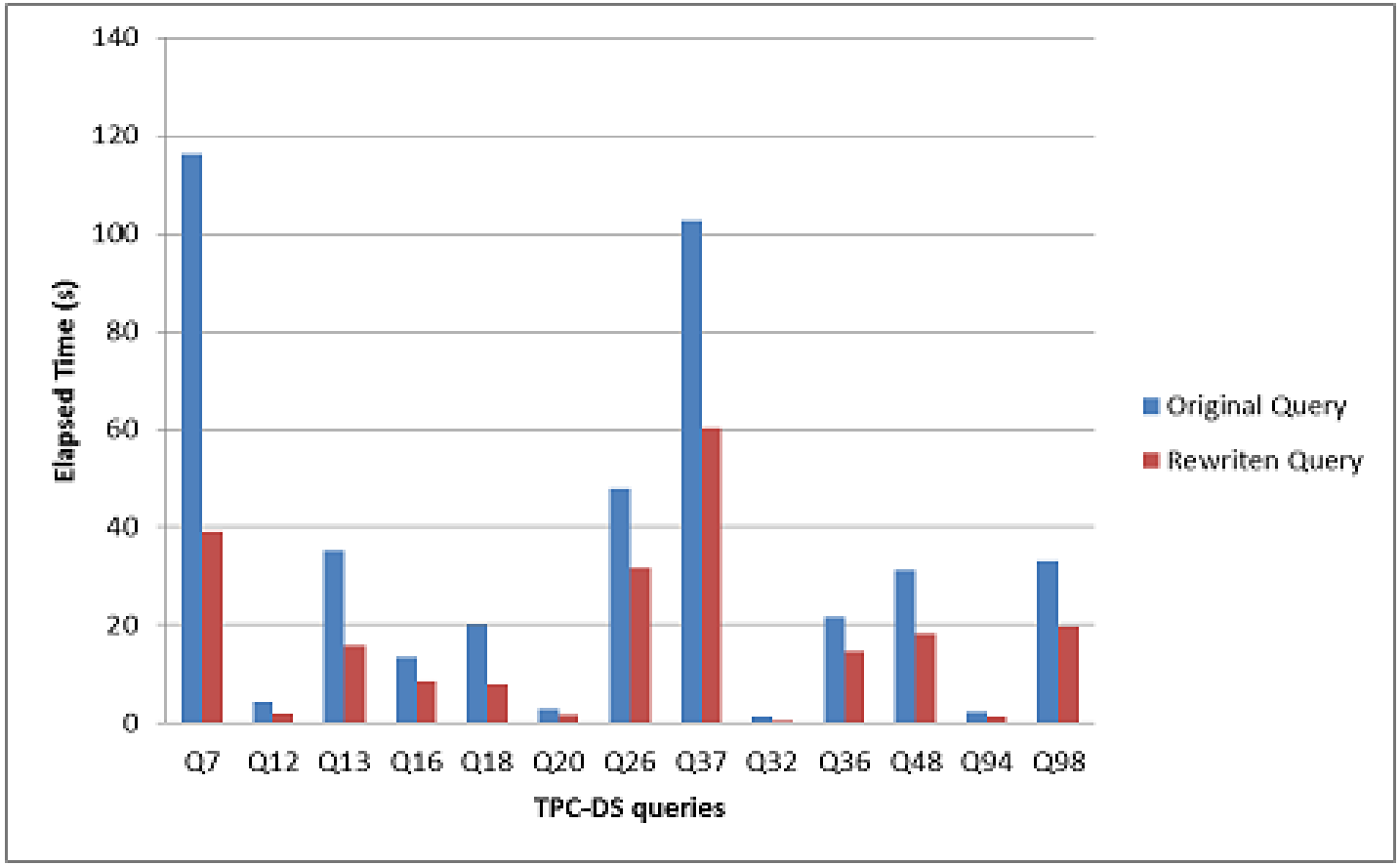
```
select substr(s_zip, 1, 2) as area,
       count(distinct s_zip) as cnt,
       sum(ss_net_profit) as net
from store_sales, store
where ss_store_sk = s_store_sk
group by substr(s_zip, 1, 2);
```

- Let there be a B-tree index on *s_zip* in table *store*.
- [*s_zip*] ↦ [substr(*s_zip*, 1, 2)].
- Optimizer can accomplish group-by *on-the-fly* (partial group by).

Very Large Data Bases (PVLDB) 2013
Expressiveness and Complexity of Order Dependencies.
J. Szlichta, P. Godfrey, J. Gryz, C. Zuzarte

Extending Database Technology (EDBT) 2014
Business-Intelligence Queries with Order Dependencies in IBM DB2.
J. Szlichta, P. Godfrey, J. Gryz, W. Ma, W. Qiu, C. Zuzarte

Evaluation


140
120
100
80
60
40
20
0
Elapsed Time (s)
Q7
Q12
Q13
Q16
Q18
Q20
Q26
Q37
Q32
Q36
Q48
Q94
Q98
TPC-DS queries
Original Query
Rewriten Query

ODs and FDs

- Lists $\mathbf{X}$ and $\mathbf{Y}$ are *order equivalent iff*

  $\mathbf{X} \mapsto \mathbf{Y}$ and $\mathbf{Y} \mapsto \mathbf{X}$

  We denote this by $\mathbf{X} \leftrightarrow \mathbf{Y}$.

- (FDs are subsumed in ODs)

  $\mathcal{X} \rightarrow \mathcal{Y}$ *iff* $\mathbf{X} \mapsto \mathbf{XY}$,
  for any list $\mathbf{X}$ *over the set of attributes of* $\mathcal{X}$,
  and any list $\mathbf{Y}$ *likewise for* $\mathcal{Y}$.

- (Decomposition)

  $\mathbf{X} \mapsto \mathbf{Y}$ *iff* $\mathbf{X} \mapsto \mathbf{XY}$ & $\mathbf{XY} \leftrightarrow \mathbf{YX}$.

Axiomatization

1: Reflexivity

$$\mathbf{XY} \mapsto \mathbf{X}$$

2: Prefix

$$\frac{\mathbf{X} \mapsto \mathbf{Y}}{\mathbf{ZX} \mapsto \mathbf{ZY}}$$

3: Normalization

$$\mathbf{WXYXV} \leftrightarrow \mathbf{WXYV}$$

4: Transitivity

$$\frac{\mathbf{X} \mapsto \mathbf{Y} \quad \mathbf{Y} \mapsto \mathbf{Z}}{\mathbf{X} \mapsto \mathbf{Z}}$$

5: Suffix

$$\frac{\mathbf{X} \mapsto \mathbf{Y}}{\mathbf{X} \leftrightarrow \mathbf{YX}}$$

6: Chain

$$\frac{\mathbf{X} \sim \mathbf{Y}_1 \quad \forall_{i \in [1,n-1]} \mathbf{Y}_i \sim \mathbf{Y}_{i+1} \quad \mathbf{Y}_n \sim \mathbf{Z} \quad \forall_{i \in [1,n]} \mathbf{Y}_i\mathbf{X} \sim \mathbf{Y}_i\mathbf{Z}}{\mathbf{X} \sim \mathbf{Z}}$$

- (Order compatibility) $\mathbf{X} \sim \mathbf{Y}$ *iff* $\mathbf{XY} \leftrightarrow \mathbf{YX}$.
- We proved the set of axioms is **sound** and **complete**.

Very Large Data Bases (PVLDB) 2011
Fundamentals of Order Dependencies.
J. Szlichta, P. Godfrey, J. Gryz

Complexity of OD Inference

- Inference problem is to decide whether a dependency is satisfied based on a given set of dependencies.
  - Assume $\boldsymbol{M} = \{AB \mapsto C, C \mapsto D\}$. Is it true that $\boldsymbol{M} \vDash \{AB \mapsto D\}$?
- We show how to infer ODs with *chase* procedure (exponential time schema complexity).
- We establish the following:
  - A proof of *co-NP-completeness* for the inference problem for UODs.
  - A proof of *co-NP-completeness* for the inference problem of functional dependencies from ODs.
  - *Linear* time complexity algorithm for the inference of FDs from UODs.

Very Large Data Bases (PVLDB) 2013
Expressiveness and Complexity of Order Dependencies.
J. Szlichta, P. Godfrey, J. Gryz, C. Zuzarte

Restricted Domain

- We introduced a ***restricted* domain**.
- It makes reasoning over ODs “simpler”.
- A domain is restricted if an additional order property is guaranteed over the schema.
- We propose a restricted set of axioms.

Restricted Domain

Complexity over restricted axiom based approach is polynomial.

1: Reflexivity

$$\mathbf{XY} \mapsto \mathbf{X}$$

2: Prefix

$$\frac{\mathbf{X} \mapsto \mathbf{Y}}{\mathbf{ZX} \mapsto \mathbf{ZY}}$$

3: Normalization

$$\mathbf{WXYXV} \leftrightarrow \mathbf{WXYV}$$

4: Transitivity

$$\frac{\mathbf{X} \mapsto \mathbf{Y} \quad \mathbf{Y} \mapsto \mathbf{Z}}{\mathbf{X} \mapsto \mathbf{Z}}$$

5: Suffix

$$\frac{\mathbf{X} \mapsto \mathbf{Y}}{\mathbf{X} \leftrightarrow \mathbf{YX}}$$

Replacing Chain Axiom

6: **Downward Closure**

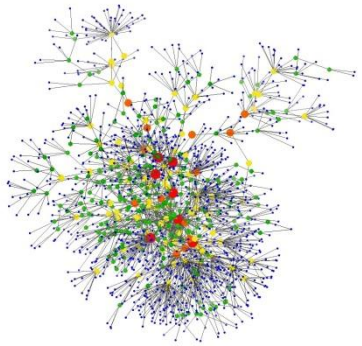
$$\frac{\mathbf{XZ} \sim \mathbf{YV}}{\mathbf{X} \sim \mathbf{Y}}$$

Collaborators

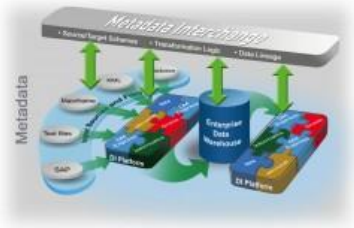
- University of Toronto:
  - **Renée J. Miller**
  - **Maksims Volkovs**
  - Javed Siddique
  - Natasha Prokoshyna
- McMaster University:
  - **Fei Chiang**
- University of Waterloo
  - Ihab F. Ilyas
  - Anup Chalamalla
  - Lukasz Golab
- AT&T Labs
  - Divesh Srivastava
- York University & IBM CAS:
  - **Parke Godfrey, Jarek Gryz.**
- IBM LAB:
  - **Calisto Zuzarte,**
  - Anastasios Kementsietsidis
  - **Wenbin Ma,**
  - **Weinan Qiu**.
- Technische Universität Wien
  - Reinhard Pichler

III. Current & Future Projects

- Big Data Curation
  - Divesh Srivastava (AT&T Labs),
  - Fei Chiang (McMaster),
  - Renée Miller, Nataliya Prokoshyna (UofT).



- Provenance Aware Data Cleaning
  - Ihab Ilyas, Anup Challamala (University of Waterloo),
  - Fei Chiang (McMaster),
  - Renée Miller, Javed Siddique (UofT).



- Big Data Analytics
  - Calisto Zuzarte, Anastasios Kementsietsidis (IBM Research).
  - Lukasz Golab (University of Waterloo).

IV. Conclusions

- Big data integration is an important area of research.
- We have done some (hopefully) interesting work in this area
  - Data Cleaning,
  - Business Intelligence,
  - Challenges due to **volume**, **velocity**, **variety**, **veracity**.
- A lot more research needs to be done!
- Contact: jaroslaw.szlichta@uoit.ca