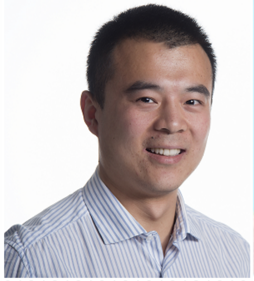
School of Computing Science
Simon Fraser University
Vancouver, Canada

FactorBase: Multi-Relational Structure Learning with SQL All the Way

Zhensong Qian	Oliver Schulte

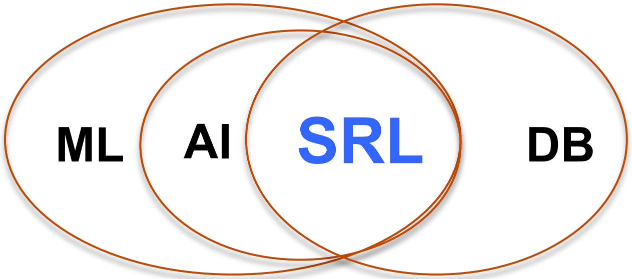


Introduction

Statistical-Relational Learning (SRL)

- Recent growing field.
- Intersection of Machine Learning, Artificial Intelligence and Database Systems.
- Applications for structured/linked/relational data.
  - Link-based Classification
  - Relational Query Optimization.
  - Information Extraction
  - Outlier Detection

  ….

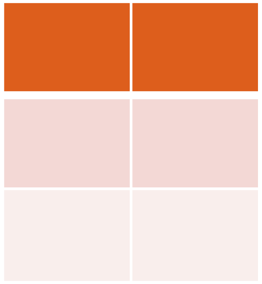


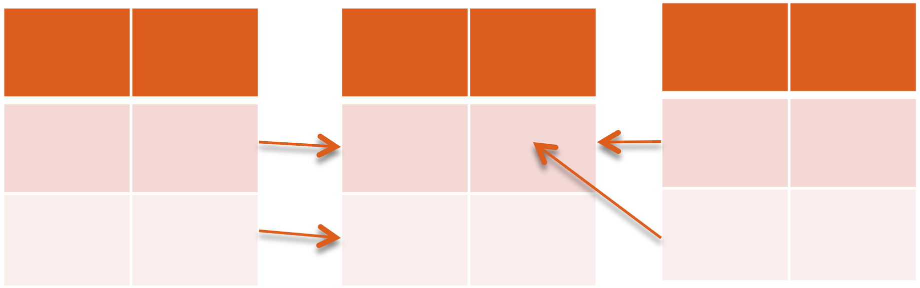
Statistical-Relational Model Learning

- Extends traditional machine learning from ***single-table*** to ***multiple interrelated tables***.
- Provides ***integrated statistical analysis of data sources***.
- Typically a ***log-linear model*** = product of factors.
- Our work:
  provides ***database system support*** for learning a generative log-linear model of the **entire input database**.

Traditional Machine Learning

Relational Learning

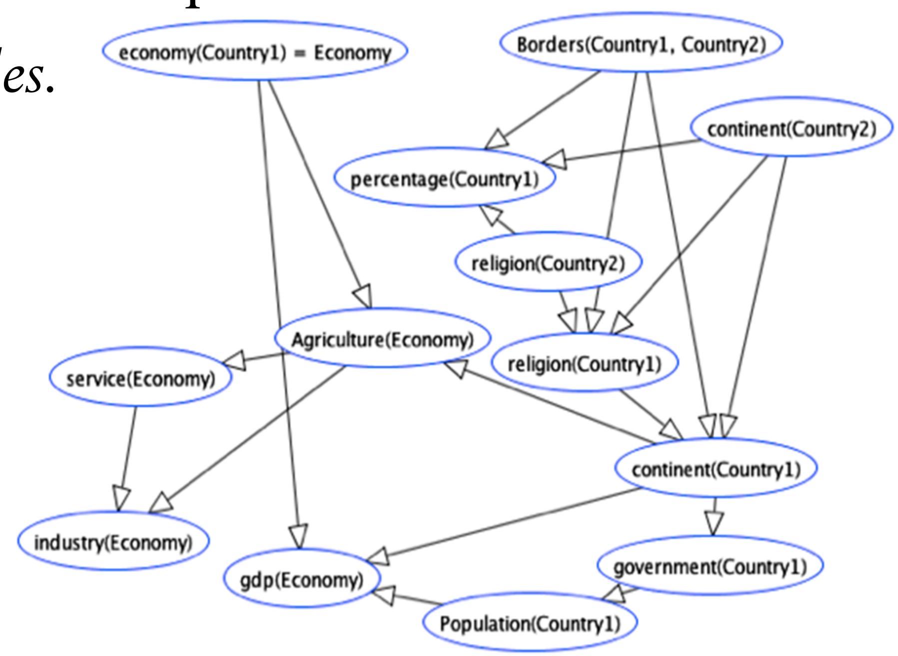




Challenges for Programming Model Structure Learning

Programming graphical model learning for relational data is hard.

- Multi-relational data is *NOT self-describing*.
  - Need to query metadata.
- *Structured* models with *structured* components.
- Event *counts across multiple tables.*
  - expensive and error-prone.
- Large parameter space.
  - > 1M sometimes.



The Solution: SQL Scripts All the Way

- Store relational ***model*** inside the database*.
  [As well as relational data.]
- SQL for creating, transforming, storing sets of models.
- SQL for querying metadata from DB catalog.
- Native SQL support for complex counts [count(*)].
- SQL for computing and storing parameter values.
  - >1M parameters no problem.
- SQL is *standardized*
  - system is portable, works out of the box.

*Daisy W, Eirinaios M, Garofalakis, M and Hellerstein, M. **BayesStore**: Managing Large, Uncertain Data Repositories with Probabilistic Graphical Models, Proc. VLDB 2008

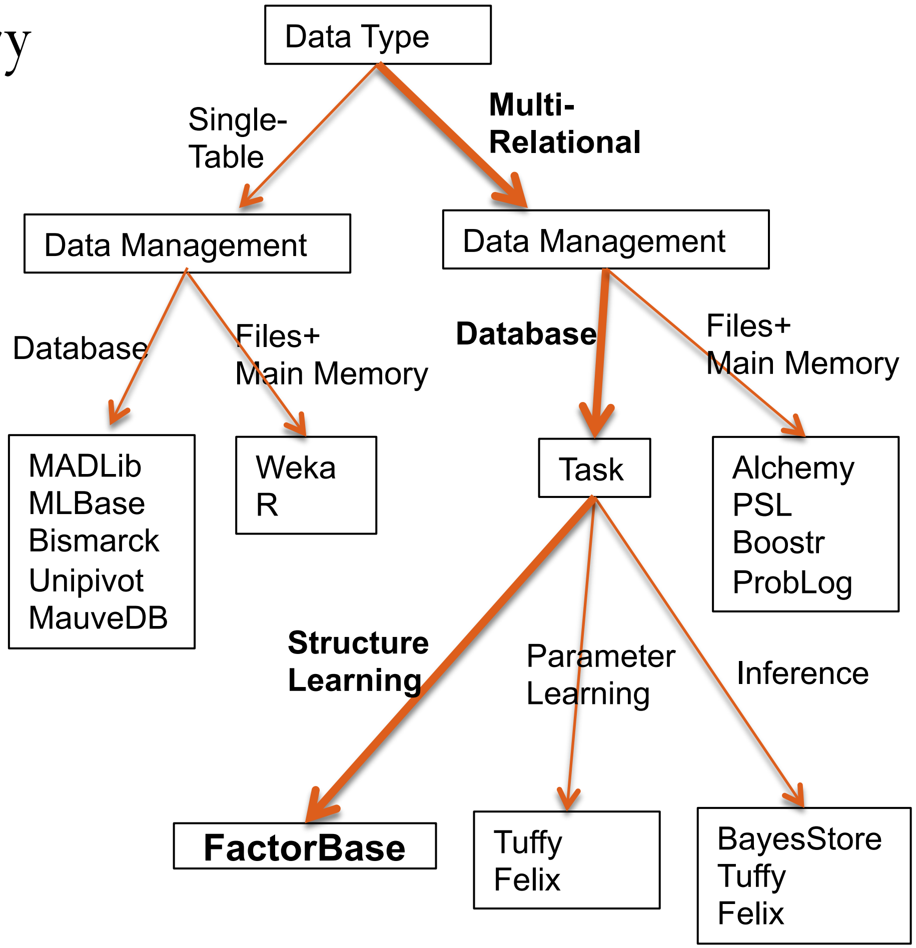
Contributions

- Identifying ***new system requirements*** for multi-relational model learning that go beyond single table machine learning.
- An integrated set of ***SQL-based solutions*** for providing these system capabilities.
- SQL can do more than we think!
- All code and datasets are available online*.

*Qian, Z.; Schulte, O. The BayesBase System. http://www.cs.sfu.ca/~oschulte/BayesBase

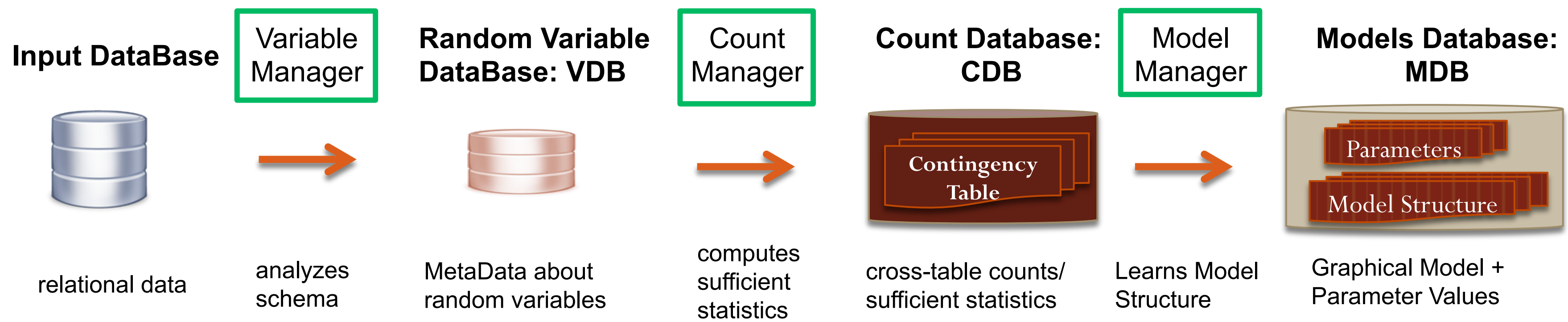
Related Works

- BayesStore, Tuffy: complementary
  - push model inside the database too
  - leverage database techniques for ***inference/parameter*** learning, not model learning.
- Madlib, MLBase, Bismarck, MauveDB, Unipivot…
  - leverage database techniques for ***single-table*** learning.

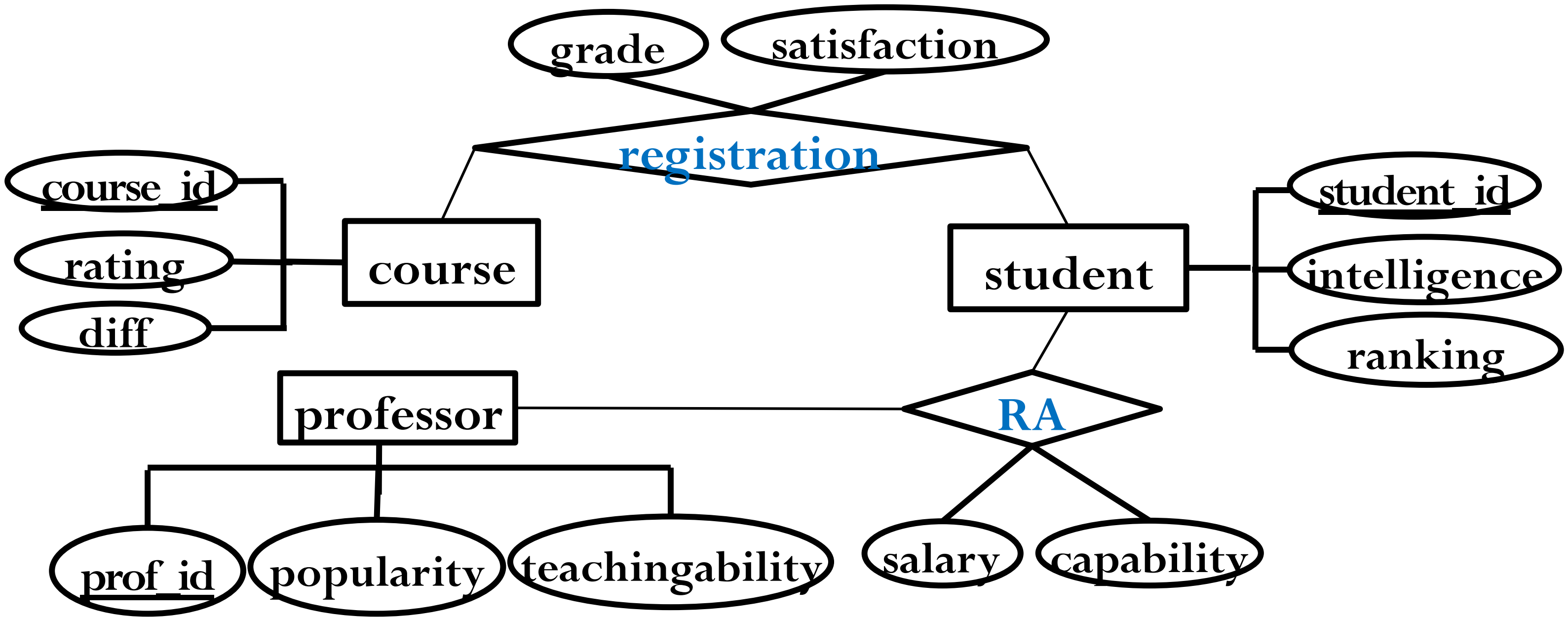


System Overview

- Each component is ***stored***, ***constructed*** and ***managed*** using database tables and SQL.
- Components are ***integrated*** using SQL as well.



Running Example

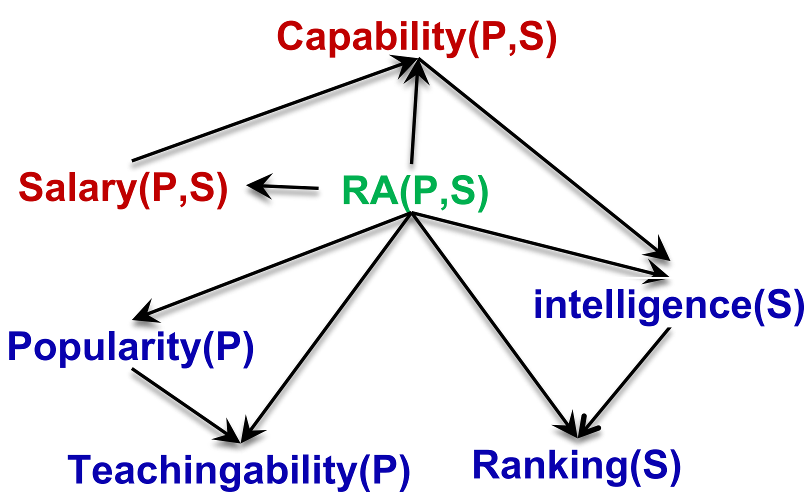


Entity-Relationship Diagram for University Domain

The Model Manager

Goal: Learn First-Order Bayesian Network Structure*.

- Nodes = Random Variables
- Edges are stored in Database tables
- Model selection scores are also stored
  - not shown (BIC, AIC, BDeu)



Child	Parent
Capability(P,S)	RA(P,S)
Capability(P,S)	Salary(P,S)
Teachingability(P)	Popularity(P)
Teachingability(P)	RA(P,S)
…	…

*Russell, S. & Norvig, P. Artificial Intelligence: A Modern Approach Prentice Hall, 2010.

The Random Variable Database

Metadata about random variables stored in database tables.

- Domain of possible values.
- Pointer to corresponding data table/column.

  …

Table Name	Column Headers in Random Variable Database					
Pvariables	**Pvid**	**TABLE_NAME**				
	C	course				
	P	prof				
	S	student				
1Variables	**1VarID**	**COLUMN_NAME**			**Pvid**	
	diff(C)	diff			C	
	intelligence(S)	intelligence			S	
	popularity(P)	popularity			P	
2Variables	**2VarID**	**COLUMN_NAME1**	**COLUMN_NAME2**		**Pvid1**	**Pvid2**
	capability(P,S)	p_id	s_id		P	S
	grade(C,S)	c_id	s_id		C	S
Relationship	**RVarID**	**TABLE_NAME**	**COLUMN_NAME1**	**COLUMN_NAME2**	**Pvid1**	**Pvid2**
	RA(P,S)	RA	p_id	s_id	P	S
	Registered(C,S)	Registered	c_id	s_id	C	S

The Count Manager

Goal: for a conjunctive query, compute the instantiation count = result set size.

- Stored in ***Contingency Table***.
- Main computational cost in learning.

Problem: need to generate SQL queries for **arbitrary variable lists**.

Solution: use Meta Data + **Meta Queries**

General Form of Count Query:

```
SELECT COUNT(*) AS Count, <VARIABLE-LIST>  FROM <TABLE-LIST>
GROUP BY <VARIABLE-LIST>   WHERE <Join-Conditions>
```

<table>
<tr><th>Meta-Queries</th><th>Entries</th></tr>
<tr><td rowspan="5">CREATE VIEW Select_List AS
SELECT RVarID, CONCAT(‘COUNT(*)',' as "count"') AS Entries
FROM VDB.Relationship
UNION DISTINCT
SELECT RVarID, AVarID AS Entries
FROM VDB.Relationship_Attributes;</td><td>COUNT(*) as “count"</td></tr>
<tr><td>`Popularity(P)`</td></tr>
<tr><td>`Teachingability(P)`</td></tr>
<tr><td>`Intelligence(S)`</td></tr>
<tr><td>`Ranking(S)`</td></tr>
<tr><td rowspan="3">CREATE VIEW From_List AS
SELECT RVarID, CONCAT('@database@.',TABLE_NAME) AS Entries
FROM VDB.Relationship_FOvariables
UNION DISTINCT
SELECT RVarID, CONCAT('@database@.',TABLE_NAME) AS Entries
FROM VDB.Relationship;</td><td>@database@.prof AS P</td></tr>
<tr><td>@database@.student AS S</td></tr>
<tr><td>@database@.RA AS `RA`</td></tr>
</table>

The Parameter Manager

Goal: Learn Bayesian Network Parameters

- Stored in Conditional Probability (CP) database table.
- Maximum Likelihood Estimates: easy SQL given Contingency Table.

Count	Capa(P,S)	RA(P,S)	Salary(P,S)
5	4	T	high
4	5	T	high
2	3	T	high
1	3	T	low
2	2	T	low
2	1	T	low
2	2	T	med
4	3	T	med
3	1	T	med

CT Table
(from Count Manager)

1. CT_Table
   JOIN
   CT_Table
2. Group By

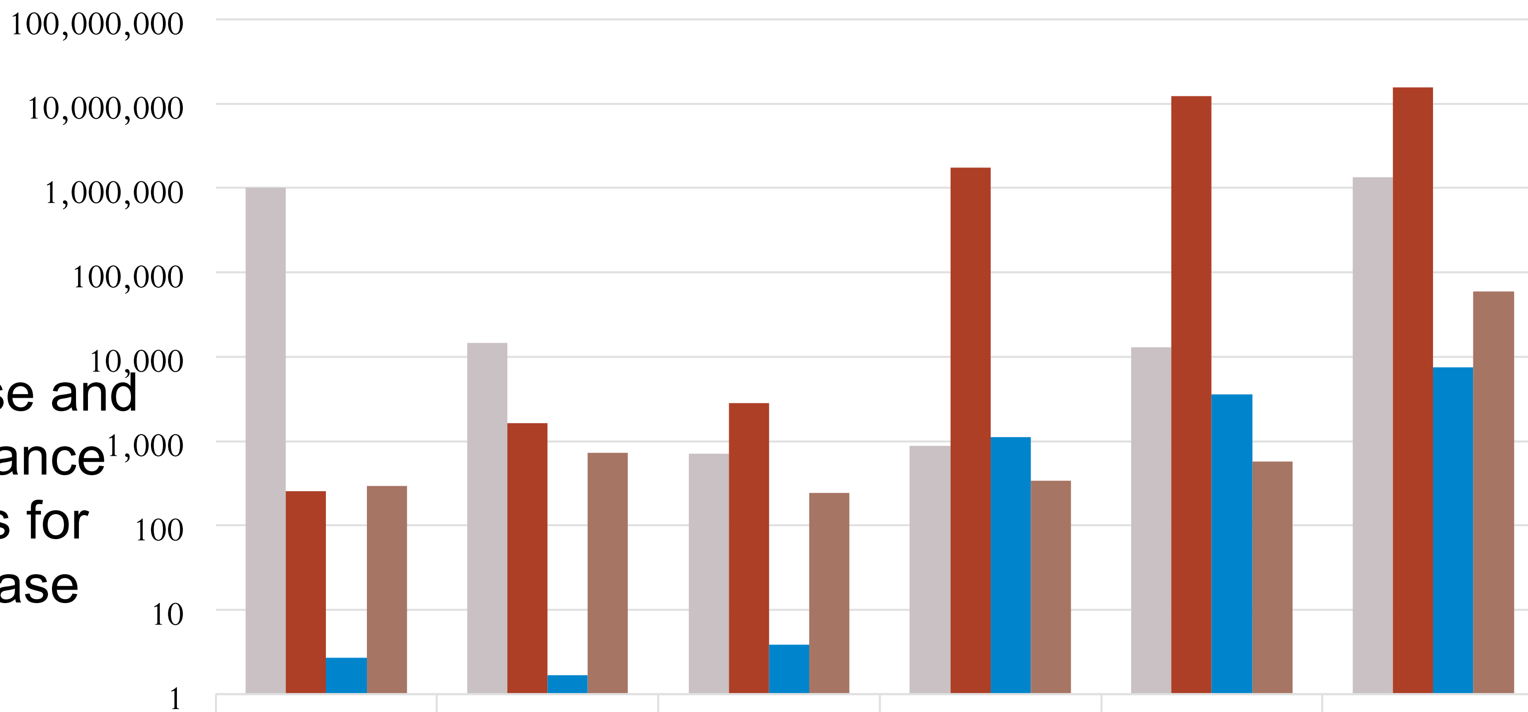
Capa(P,S)	RA(P,S)	Salary(P,S)	CP
4	T	high	0.45
5	T	high	0.36
3	T	high	0.18
3	T	low	0.20
2	T	low	0.40
1	T	low	0.40
2	T	med	0.22
3	T	med	0.44
1	T	med	0.33

CP table

Results and Summary

Results

Task: learning a multi-relational Bayesian network

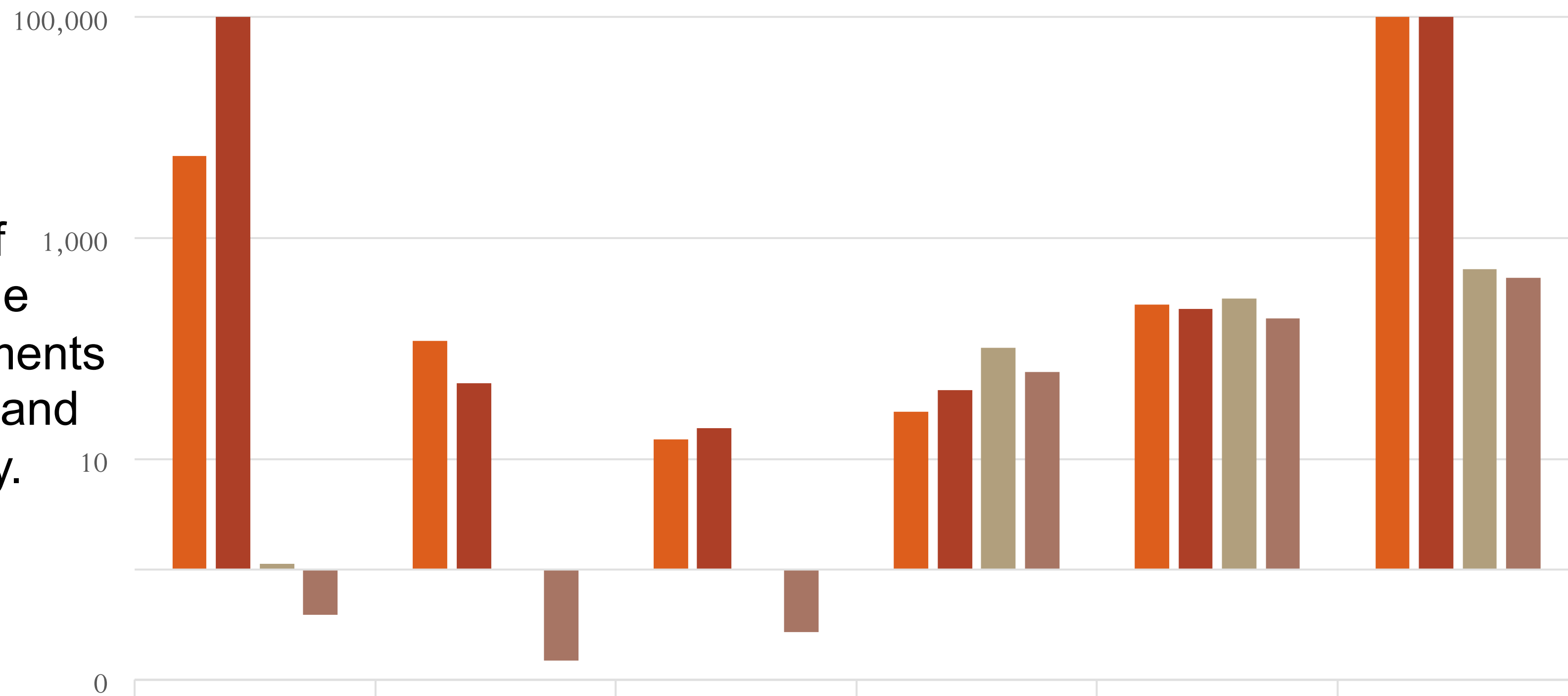


Database and performance statistics for FactorBase

	Movielens	Mutagenesis	UW-CSE	Mondial	Hepatitis	IMDB
# Database Tuples	1,010,051	14,540	712	870	12,927	1,354,134
# Sufficient Statistics	252	1,631	2,828	1,746,870	12,374,892	15,538,430
SS Computing Time (s)	2.7	1.67	3.84	1,112.84	3,536.76	7,467.85
#BN Parameters	292	721	241	339	569	60,059

Comparison with other statistical-relational learning
(Markov Logic Network learning using gradient boosting)

Orders of magnitude improvements in speed and scalability.



	Movielens	Mutagenesis	UW-CSE	Mondial	Hepatitis	IMDB
RDN_Boost	5,562	118	15	27	251	100,000
MLN_Boost	100,000	49	19	42	230	100,000
FactorBase-Total	1.12	1	1	102	286	524.25
FactorBase-Count	0.39	0.15	0.27	61.82	186.15	439.29

Other Tasks

- Use Natural Join + Group By for ***evaluating log-linear expressions***.
- Compute Log-likelihood, compute Class Label Distribution.
- ***Block Access*** for Test Instance Predictions ➔ scales to >1M instances.

Counts

Count	Capa(P,S)	RA(P,S)	Salary(P,S)
5	4	T	high
4	5	T	high
2	3	T	high
1	3	T	low
2	2	T	low
2	1	T	low
2	2	T	med
4	3	T	med
3	1	T	med

Parameters

Capa(P,S)	RA(P,S)	Salary(P,S)	CP
4	T	high	0.45
5	T	high	0.36
3	T	high	0.18
3	T	low	0.20
2	T	low	0.40
1	T	low	0.40
2	T	med	0.22
3	T	med	0.44
1	T	med	0.33

Test Set

jack_Capability_(P,S)_CT

sid	Count	Cap.(P,S)	RA(P,S)	Salary(P,S)
Jack	5	N/A	N/A	F
Jack	5	4	high	T
….	….	….	….	….

jill_Capability_(P,S)_CT

sid	Count	Cap.(P,S)	RA(P,S)	Salary(P,S)
Jill	3	N/A	N/A	F
Jill	7	4	high	T
…	….	….	….	….

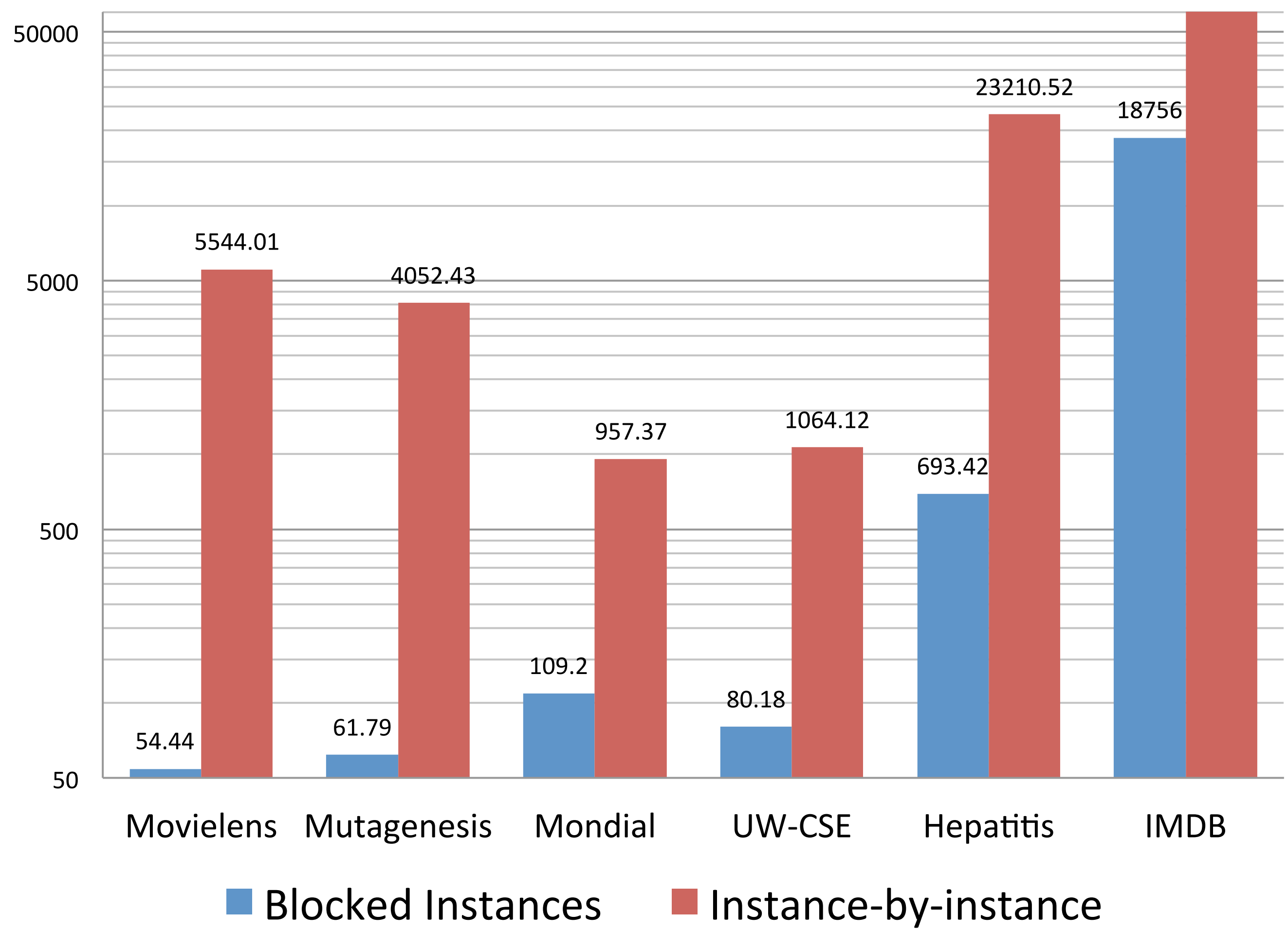
Log-likelihood

Predictive Accuracy

Other Tasks

Task: ***Block Access*** for Test Instance Predictions ➔ scales to >1M instances.

??



Summary and Conclusions

- Multi-relational *model discovery* requires new system capabilities.
- BayesStore Design Philosophy: Store data **and** models inside the database system.
- *SQL* is used to build and transform statistical objects inside the database
  - Structured Graphical Model.
  - Parameter Estimates.
  - Sufficient Statistics (counts).
- Empirical evaluation: leveraging the RDBMS capabilities achieves scalable learning and fast model testing.
- Future Direction:

  Integrate with relational inference systems (BayesStore, Tuffy).

  Distributed processing, in-memory computing (SparkSQL)

Thanks for your attention.

Multi-Relational Sufficient Statistics

Why

- Find **correlations involving relationships**. e.g.

user searches for item → user watches video about item.

?

user does not search for item → user does not watch video about item

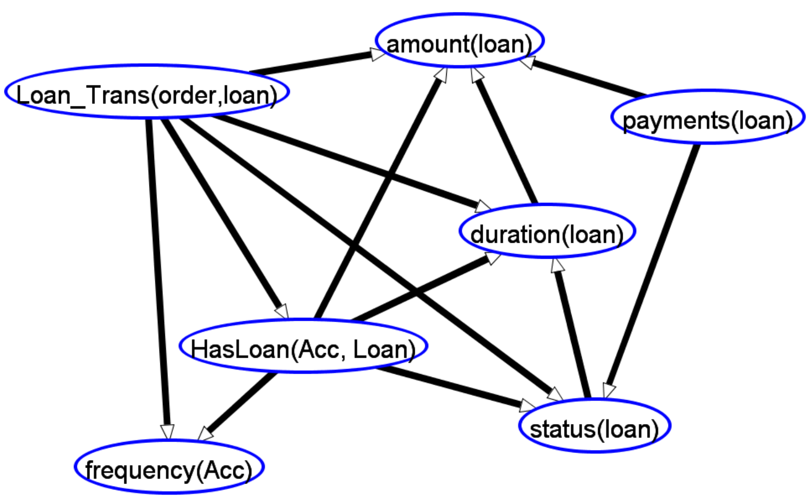
- Compactness: summarize original data by counts.

Previous Approaches

- Single-table data: row counts (σ selection only).
- Multiple tables: Table joins $\bowtie$.

Applications

- Feature Selection.
  - Does **frequency of bank statement** predict whether **customer has loan**?
- Association Rules.
  - statement freq.(Acc) = monthly → HasLoan(Acc, Loan) = **?** .
- Bayesian Network Learning.
- …



Contingency Tables (ct-table)

- Counts for **conjunctive queries**:
  - capability = value1, intelligence = value2.
  - capability = n/a: wasn't RA.
- **Conditional** ct-table :
  - e.g. given capability = 1.

Entity Table	Primary Key	#Tuples
Professor	p_id	6
Student	s_id	38
	Cross Product	228

capability	intelligence	count
1	2	3
1	3	2
2	2	3
2	3	1
3	1	2
3	2	4
3	3	1
4	1	1
4	3	4
5	1	1
5	2	3
N/A	1	80
N/A	2	65
N/A	3	58

Sum(count) : 228

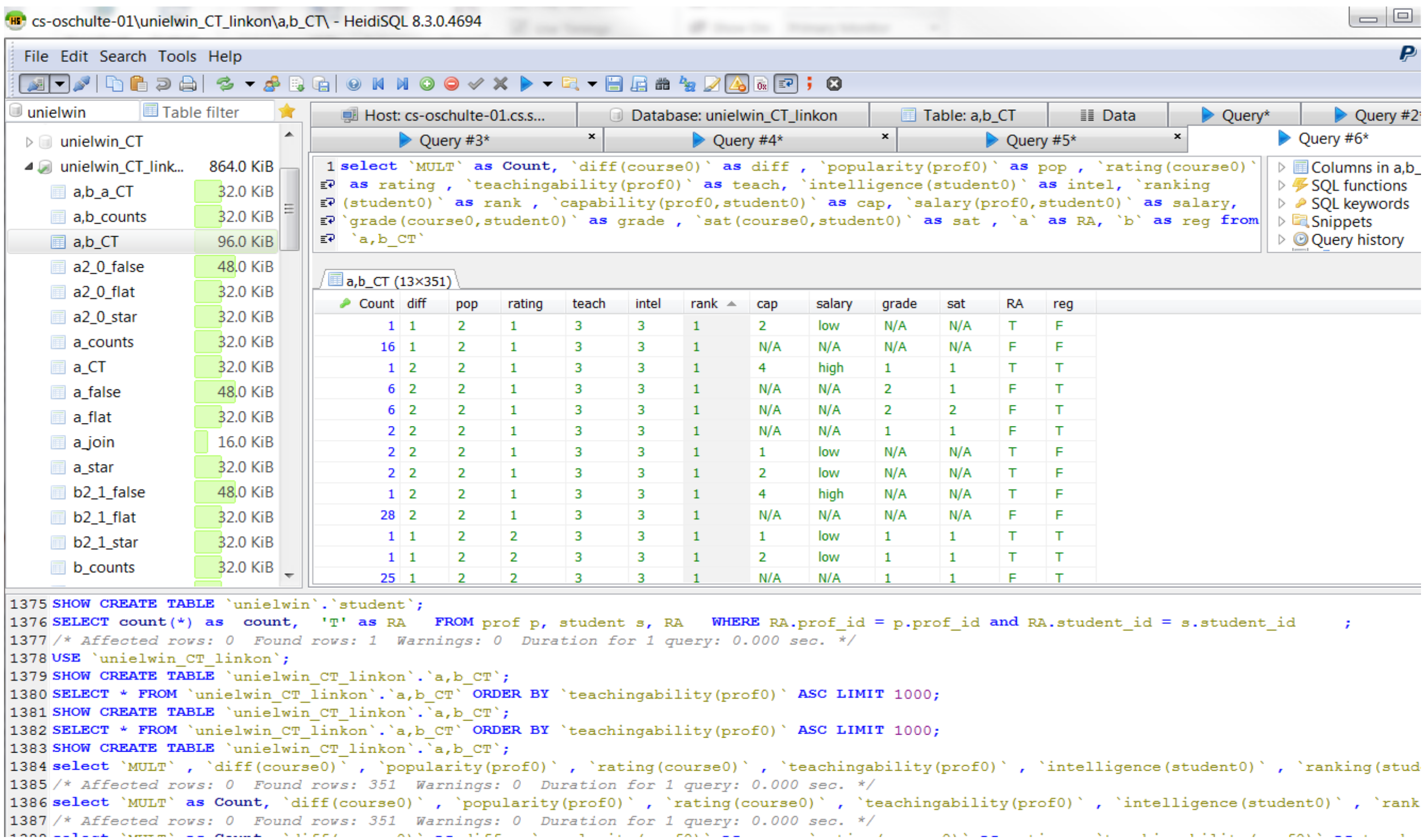
Total Tuples : 14

Count	Diff.	Rat.	Pop.	Teach.	Intel.	Rank.	Cap.	Sal.	Grade	Sat.	RA	Reg.
1	1	1	1	2	3	1	3	High	1	1	T	T
1	1	2	1	2	2	2	n/a	n/a	2	2	F	T
3	1	2	1	2	2	2	1	Med	n/a	n/a	T	F
24	2	1	1	2	1	5	n/a	n/a	n/a	n/a	F	F
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
1	2	1	2	2	1	4	n/a	n/a	3	2	F	T

Storing Sufficient Statistics in Database Tables

- **New**: large contingency table **stored** as database table.

  Manipulate using SQL, Index, …



Computing Sufficient Statistics: positive relationships only (e.g.RA=True)

CREATE TABLE **ct_T(RA)** **AS**

SELECT count(*) as count, pop, teach, intel, rank, cap, salary, 'T' **as** RA

FROM **Professor P, Student S, RA** ← cross-table count

WHERE RA.p_id = P.p_id **AND** RA.s_id = S.s_id

GROUP BY pop, teach, intel, rank, cap, salary

count	pop	teach	intel	rank	cap	salary	RA
2	2	2	3	1	4	high	T
2	2	3	1	4	3	med	T
1	1	2	2	2	1	med	T
1	1	2	2	2	2	med	T
1	1	2	2	2	3	low	T
1	**1**	**2**	**3**	**1**	**3**	**high**	**T**
…	…	…	…	…	…	…	…

Step 1: Contingency Table Cross Product

- Example: ct(Professor) x ct(Student) → **ct$_*$(RA)**

ct(Professor)

Count	pop	teach
2	1	2
1	2	2
3	2	3

x

ct(Student)

Count	intel	rank
6	1	4
8	1	5
5	2	2
7	2	3
1	2	4
8	3	1
3	3	2

ct$_*$(RA)

Count	pop	teach	intel	rank
12	1	2	1	4
16	1	2	1	5
10	1	2	2	2
14	1	2	2	3
2	1	2	2	4
16	1	2	3	1
6	1	2	3	2
6	2	2	1	4
8	2	2	1	5
5	2	2	2	2
7	2	2	2	3
…	…	…	…	…

Step 2: Contingency Table Subtraction

$ct_*(RA)$

Count	Pop	Teach	Intel	Rank
16	1	2	3	1
15	2	3	2	2
18	2	3	1	4
⋮	⋮	⋮	⋮	⋮

$ct_T(RA)$

Count	Pop	Teach	Intel	Rank
1	1	2	3	1
1	1	2	2	2
1	2	2	2	4
⋮	⋮	⋮	⋮	⋮

RA
T

—

Count	Pop	Teach	Intel	Rank
15	1	2	3	1
16	2	3	2	3
21	2	3	3	1
⋮	⋮	⋮	⋮	⋮

RA
F

U

$ct(RA)$

Count	Pop	Teach	Intel	Rank	RA
15	1	2	3	1	F
1	1	2	3	1	T
1	1	2	2	2	T
⋮	⋮	⋮	⋮	⋮	⋮

Final Result: Contingency Table for RA relationship

Datasets for Evaluation

7 Real-world Datasets (over 1M rows).

Dataset	#Relationship Tables/ Total	# Columns	# Rows
UW-CSE	2/4	14	712
Mondial	2/4	18	870
Hepatitis	3/7	19	12,927
Mutagenesis	2/4	11	14,540
Financial	3/7	15	225,932
Movielens	1/3	7	1,010,051
IMDB	3/7	17	1,354,134

Computation Time

- Never enumerates cross product of primary keys.
- Complexity: nearly linear in size of the required output.

(non-trivial) #ct_operation = O(#SS * log (#SS))

Dataset	#Sufficient Statistics (SS)	Cross Product Time	Our Dynamic Program Time
Movielens	252	703.99	**2.70**
Mutagenesis	1,631	1,096.00	**1.67**
UW-CSE	2,828	350.30	**3.84**
Mondial	1,746,870	132.13	**1,112.84**
Financial	3,013,011	N.T.	**1,421.87**
Hepatitis	12,374,892	N.T.	**3,536.76**
IMDB	15,538,430	N.T.	**7,467.85**

(Time in seconds.)