Leveraging the Structure of Uncertain Data
Tirer parti de la structure des données incertaines

Antoine Amarilli

Télécom ParisTech, DBWeb

March 14th, 2016

Databases

Computers often use databases to store data and query it

Databases

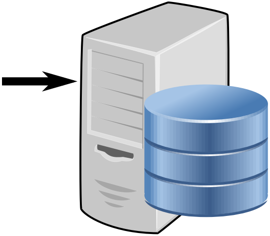
Computers often use databases to store data and query it

Databases

Computers often use databases to store data and query it

Data

time	from	to	message
14:42	Spam	John	Cheap Watch...
14:34	A.	John	Just started
18:20	Mom	John	Hi John!...
14:49	John	Spam	STOP
14:36	John	A.	OK. Good...
⋮	⋮	⋮	⋮



Databases

Computers often use databases to store data and query it

Data

time	from	to	message
14:42	Spam	John	Cheap Watch...
14:34	A.	John	Just started
18:20	Mom	John	Hi John!...
14:49	John	Spam	STOP
14:36	John	A.	OK. Good...
⋮	⋮	⋮	⋮

Query

```
SELECT * FROM Data
WHERE from='A.' OR to='A.'
```

Databases

Computers often use databases to store data and query it

Data

time	from	to	message
14:42	Spam	John	Cheap Watch...
14:34	A.	John	Just started
18:20	Mom	John	Hi John!...
14:49	John	Spam	STOP
14:36	John	A.	OK. Good...
⋮	⋮	⋮	⋮

Query

```
SELECT * FROM Data
WHERE from='A.' OR to='A.'
```

Answer

time	from	to	message
14:32	John	A.	Hi there! :)
14:32	John	A.	Just wanted...
14:33	John	A.	How is it...
14:34	A.	John.	Dunno yet
14:34	A.	John.	Just started
14:36	John	A.	OK. Good...

Databases

Computers often use databases to store data and query it

Data

time	from	to	message
14:42	Spam	John	Cheap Watch...
14:34	A.	John	Just started
18:20	Mom	John	Hi John!...
14:49	John	Spam	STOP
14:36	John	A.	OK. Good...
⋮	⋮	⋮	⋮

Query

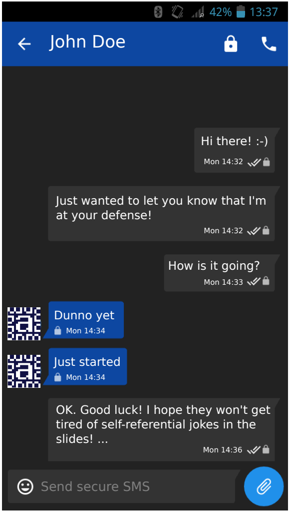
```
SELECT * FROM Data
WHERE from='A.' OR to='A.'
```

Answer

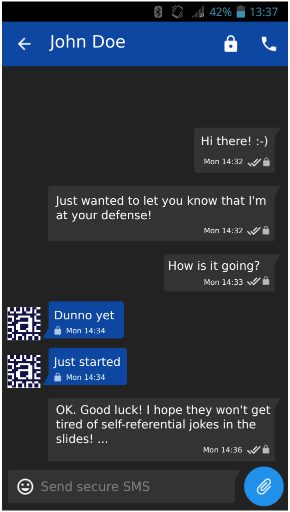
time	from	to	message
14:32	John	A.	Hi there! :)
14:32	John	A.	Just wanted...
14:33	John	A.	How is it...
14:34	A.	John.	Dunno yet
14:34	A.	John.	Just started
14:36	John	A.	OK. Good...

→ Let's see a few examples...

Database example: SMS on Android



Database example: SMS on Android



time	from	to	message
14:32	John	A.	Hi there! :)
14:32	John	A.	Just wanted...
14:33	John	A.	How is it...
14:34	A.	John	Dunno yet
14:34	A.	John	Just started
14:36	John	A.	OK. Good...

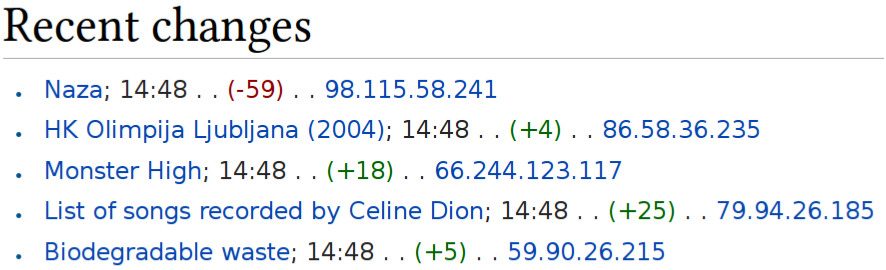
In reality...

```
CREATE TABLE sms (_id INTEGER, thread_id INTEGER,
  address TEXT, address_device_id INTEGER, person INTEGER,
  date INTEGER, date_sent INTEGER, protocol INTEGER,
  read INTEGER, status INTEGER, type INTEGER,
  reply_path_present INTEGER,
  delivery_receipt_count INTEGER, subject TEXT, body TEXT,
  mismatched_identities TEXT, service_center TEXT,
  date_delivery_received INTEGER);
```

In reality...

```
CREATE TABLE sms (_id INTEGER, thread_id INTEGER,
  address TEXT, address_device_id INTEGER, person INTEGER,
  date INTEGER, date_sent INTEGER, protocol INTEGER,
  read INTEGER, status INTEGER, type INTEGER,
  reply_path_present INTEGER,
  delivery_receipt_count INTEGER, subject TEXT, body TEXT,
  mismatched_identities TEXT, service_center TEXT,
  date_delivery_received INTEGER);
INSERT INTO sms VALUES(
  14041,224,'+33611210549',1,NULL,1451921855098,
  1451921849000,0,1,-1,-2147483628,0,0,NULL,
  'Hi there!',NULL,'+33609002960',0);
INSERT INTO sms VALUES(
  14042,224,'+33611210549',1,NULL,1451921945081,
  1451921945081,NULL,1,-1,-2147483561,NULL,0,NULL,
  'Just wanted...',NULL,NULL,0);
```

Database example: Wikipedia



Database example: Wikipedia

Recent changes

- Naza; 14:48 . . (-59) . . 98.115.58.241
- HK Olimpija Ljubljana (2004); 14:48 . . (+4) . . 86.58.36.235
- Monster High; 14:48 . . (+18) . . 66.244.123.117
- List of songs recorded by Celine Dion; 14:48 . . (+25) . . 79.94.26.185
- Biodegradable waste; 14:48 . . (+5) . . 59.90.26.215

title	time	size	user
Naza	14:48	-59	92.115.58.241
HK Olimpija Ljubljana (2004)	14:48	+4	86.58.36.235
Monster High	14:48	+18	66.244.123.117
List of songs recorded by Celine Dion	14:48	+25	79.94.26.185
Biodegradable waste	14:48	+5	59.90.26.215

In reality...

```
CREATE TABLE mw_recentchanges (rc_id INT(8),
  rc_timestamp VARCHAR(14), rc_cur_time VARCHAR(14),
  rc_user INT(10), rc_user_text VARCHAR(255),
  rc_namespace INT(11), rc_title VARCHAR(255),
  rc_comment VARCHAR(255), rc_minor TINYINT(3),
  rc_bot TINYINT(3), rc_new TINYINT(3),
  rc_cur_id INT(10), rc_this_oldid INT(10),
  rc_last_oldid INT(10), rc_type TINYINT(3),
  rc_moved_to_ns TINYINT(3), rc_moved_to_title VARCHAR(255),
  rc_patrolled TINYINT(3), rc_ip CHAR(15),
  rc_old_len INT(10), rc_new_len INT(10),
  rc_deleted TINYINT(1), rc_logid INT(10),
  rc_log_type VARCHAR(255), rc_log_action VARCHAR(255),
  rc_params BLOB,
);
```

In reality...

```
CREATE TABLE mw_recentchanges (rc_id INT(8),
  rc_timestamp VARCHAR(14), rc_cur_time VARCHAR(14),
  rc_user INT(10), rc_user_text VARCHAR(255),
  rc_namespace INT(11), rc_title VARCHAR(255),
  rc_comment VARCHAR(255), rc_minor TINYINT(3),
  rc_bot TINYINT(3), rc_new TINYINT(3),
  rc_cur_id INT(10), rc_this_oldid INT(10),
  rc_last_oldid INT(10), rc_type TINYINT(3),
  rc_moved_to_ns TINYINT(3), rc_moved_to_title VARCHAR(255),
  rc_patrolled TINYINT(3), rc_ip CHAR(15),
  rc_old_len INT(10), rc_new_len INT(10),
  rc_deleted TINYINT(1), rc_logid INT(10),
  rc_log_type VARCHAR(255), rc_log_action VARCHAR(255),
  rc_params BLOB,
);
INSERT INTO mw_recentchanges VALUES
  (1, '20160314144837', '20160314144827', 1, '92.115.58.241', 0,
  'Naza', '', 0, 0, 0, 1, 2, 1, 0, 0, '', 1, '92.115.58.241',
  559, 500, 0, 0, NULL, NULL, '');
INSERT INTO mw_recentchanges VALUES
  (2, '20160314144842', '20160314144842', 1, '66.244.123.117', 2,
  'Monster High', '', 0, 0, 1, 2, 3, 0, 1, 0, '', 1, '66.244.123.117',
  102, 120, 0, 0, NULL, NULL, '');
```

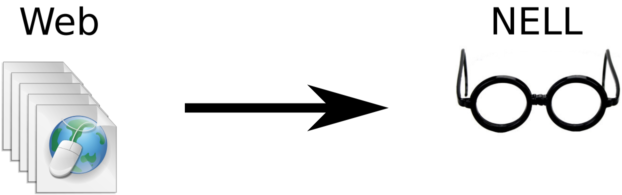
Uncertainty

- Databases usually assume that data is
  - → complete
  - → crisp
  - → certain
  - → correct
- In many situations, this is not the case...

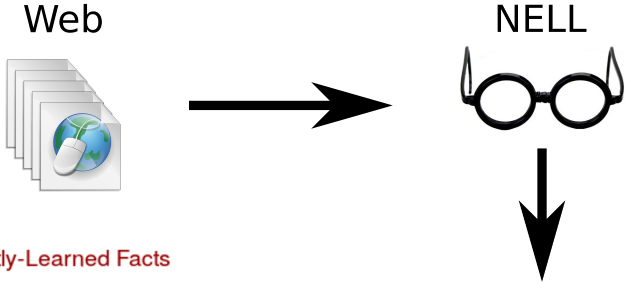
Example: Never-Ending Language Learning

Web

Example: Never-Ending Language Learning



Example: Never-Ending Language Learning



Recently-Learned Facts

Refresh

instance	iteration	date learned	confidence
kampioenschap van zwitserland is a sports race	955	20-oct-2015	95.0
cochran mill nature center is an aquarium	955	20-oct-2015	96.9
kozy shack chocolate pudding is a kind of candy	956	23-oct-2015	90.3
red delicious apple tree is a plant	955	20-oct-2015	92.8
sale miami dade county is a sport	955	20-oct-2015	99.1
chicken001 eat black beans	955	20-oct-2015	100.0
wrigley field is the home venue for the sports team chicago cubs	959	07-nov-2015	100.0
lorena ochoa is a person who has residence in the geopolitical location mexico	958	03-nov-2015	100.0
umass lowell river hawks hired john calipari	955	20-oct-2015	98.4
nuggets participated in the event games	955	20-oct-2015	100.0

Many sources of uncertainty

- Errors in sources:

> This article's **factual accuracy is disputed**. Please help to ensure that disputed statements are reliably sourced. See the relevant discussion on the talk page. *(November 2015)*

Many sources of uncertainty

- Errors in sources:

> This article's **factual accuracy is disputed**. Please help to ensure that disputed statements are reliably sourced. See the relevant discussion on the talk page. *(November 2015)*

- Entity disambiguation:
  "The place and function of Venus in Ovid..."
  "Computed backscattering function of Venus and the moon..."

Many sources of uncertainty

- Errors in sources:

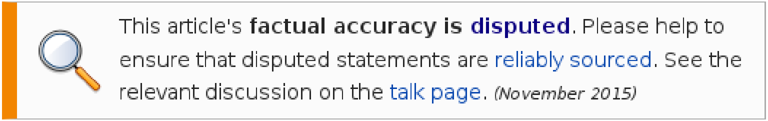
  > This article's **factual accuracy is disputed**. Please help to ensure that disputed statements are reliably sourced. See the relevant discussion on the talk page. *(November 2015)*

- Entity disambiguation:
  "The place and function of Venus in Ovid..."
  "Computed backscattering function of Venus and the moon..."

- Anaphora resolution:
  "Obama told Hollande that he was not a spying target"

Many sources of uncertainty

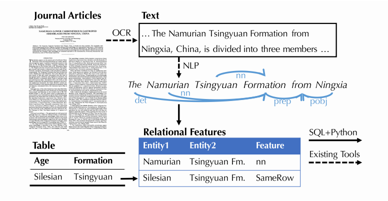
- Errors in sources:

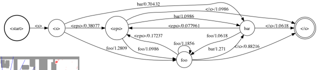

This article's **factual accuracy is disputed**. Please help to ensure that disputed statements are reliably sourced. See the relevant discussion on the talk page. *(November 2015)*

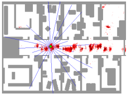
- Entity disambiguation:
  "The place and function of Venus in Ovid..."
  "Computed backscattering function of Venus and the moon..."
- Anaphora resolution:
  "Obama told Hollande that he was not a spying target"
- Incompleteness

Many uncertain data applications

- Information extraction
- Machine learning
- Speech recognition
- Data integration
- Crowdsourcing
- …







Possible Mapping		Prob
$m_1 =$	{(pname, name), (email-addr, email), (current-addr, mailing-addr), (permanent-addr, home-addr)}	0.5
$m_2 =$	{(pname, name), (email-addr, email), (permanent-addr, mailing-addr), (current-addr, home-addr)}	0.4
$m_3 =$	{(pname, name), (email-addr, mailing-addr), (current-addr, home-addr)}	0.1

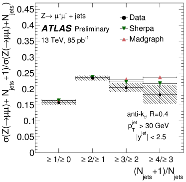
(a)

pname	email-addr	current-addr	permanent-addr
Alice	alice@	Mountain View	Sunnyvale
Bob	bob@	Sunnyvale	Sunnyvale

(b)

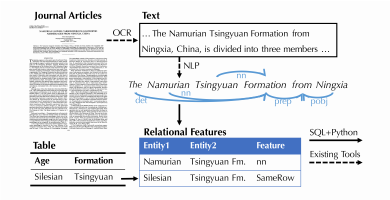
Tuple (mailing-addr)	Prob
('Sunnyvale')	0.9
('Mountain View')	0.5
('alice@')	0.1
('bob@')	0.1

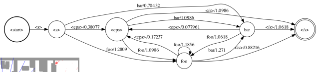
(c)

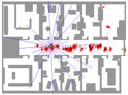


Many uncertain data applications

- Information extraction
- Machine learning
- Speech recognition
- Data integration
- Crowdsourcing
- ...
- PhD defense scheduling







Possible Mapping		Prob
$m_1 =$	{(pname, name), (email-addr, email), (current-addr, mailing-addr), (permanent-addr, home-addr)}	0.5
$m_2 =$	{(pname, name), (email-addr, email), (permanent-addr, mailing-addr), (current-addr, home-addr)}	0.4
$m_3 =$	{(pname, name), (email-addr, mailing-addr), (current-addr, home-addr)}	0.1

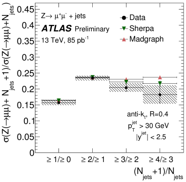
(a)

pname	email-addr	current-addr	permanent-addr
Alice	alice@	Mountain View	Sunnyvale
Bob	bob@	Sunnyvale	Sunnyvale

(b)

Tuple (mailing-addr)	Prob
('Sunnyvale')	0.9
('Mountain View')	0.5
('alice@')	0.1
('bob@')	0.1

(c)



Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics
Number of people invited

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics	
Number of people invited	87

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics	
Number of people invited	87
Number of definite yes answers	

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics	
Number of people invited	87
Number of definite yes answers	46

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics	
Number of people invited	87
Number of definite yes answers	46
Number of definite no answers	

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics	
Number of people invited	87
Number of definite yes answers	46
Number of definite no answers	14

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics	
Number of people invited	87
Number of definite yes answers	46
Number of definite no answers	14
Number of uncertain answers	

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics	
Number of people invited	87
Number of definite yes answers	46
Number of definite no answers	14
Number of uncertain answers	27

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics	
Number of people invited	87
Number of definite yes answers	46
Number of definite no answers	14
Number of uncertain answers	27
Number of additional people showing up	

Uncertainty applied to PhD defenses

Who will attend this PhD defense?

Statistics	
Number of people invited	87
Number of definite yes answers	46
Number of definite no answers	14
Number of uncertain answers	27
Number of additional people showing up	??

Why is uncertainty challenging?

- Data is uncertain if we don't know its exact state
- A possible world is an actual outcome

Why is uncertainty challenging?

- Data is uncertain if we don't know its exact state
- A possible world is an actual outcome
- Simplest method: write out all possible worlds

Why is uncertainty challenging?

- Data is uncertain if we don't know its exact state
- A possible world is an actual outcome
- Simplest method: write out all possible worlds

List of the people
who may show up:

- Dave
- Guy
- Tat
- ...
- more?

Why is uncertainty challenging?

- Data is uncertain if we don't know its exact state
- A possible world is an actual outcome
- Simplest method: write out all possible worlds

List of the people
who may show up:

- Dave
- Guy
- Tat
- ...
- more?

→ 27 uncertain people

Why is uncertainty challenging?

- Data is uncertain if we don't know its exact state
- A possible world is an actual outcome
- Simplest method: write out all possible worlds

List of the people
who may show up:

- Dave
- Guy
- Tat
- ...
- more?

→ 27 uncertain people

→ 134 217 728 possibilities

Why is uncertainty challenging?

- Data is uncertain if we don't know its exact state
- A possible world is an actual outcome
- Simplest method: write out all possible worlds

List of the people
who may show up:

- Dave
- Guy
- Tat
- ...
- more?

→ 27 uncertain people

→ 134 217 728 possibilities

→ If the list of people is incomplete,
infinitely many possible completions

Uncertainty representation and semantics

Uncertain databases represent implicitly the possible worlds

Uncertainty representation and semantics

Uncertain databases represent implicitly the possible worlds

→ Probabilities

Dave	0.4
Guy	0.3
Tat	0.2
⋮	

Uncertainty representation and semantics

Uncertain databases represent implicitly the possible worlds

→ Probabilities

Dave	0.4
Guy	0.3
Tat	0.2
⋮	

→ Correlations

- Only one of Isa and Pal can come
- Mat and Val either come together or not
- Nell will probably come if Mike does

Uncertainty representation and semantics

Uncertain databases represent implicitly the possible worlds

→ Probabilities

Dave	0.4
Guy	0.3
Tat	0.2
⋮	

→ Correlations

- Only one of Isa and Pal can come
- Mat and Val either come together or not
- Nell will probably come if Mike does

→ Logical constraints

- If someone comes to the defense then they will also come to the drinks

Summary of uncertainty goals

→ End goal: A database system with first-class uncertainty

- Feed uncertain data to the system
- Get uncertain query results

Summary of uncertainty goals

→ End goal: A database system with first-class uncertainty

- Feed uncertain data to the system
- Get uncertain query results

Summary of uncertainty goals

→ End goal: A database system with first-class uncertainty

- Feed uncertain data to the system
- Get uncertain query results

Uncertain data

→ Probas

Flo	0.4
Guy	0.3
Tat	0.2
⋮	

→ Correlations

- Only one of Isa and Pal can come
- Mat and Val either come together or not
- Nell will probably come if Mike does

→ Logical rules If defense then drinks

Summary of uncertainty goals

→ End goal: A database system with first-class uncertainty

- Feed uncertain data to the system
- Get uncertain query results

Uncertain data

→ Probas

Flo	0.4
Guy	0.3
Tat	0.2
⋮	

→ Correlations

- Only one of Isa and Pal can come
- Mat and Val either come together or not
- Nell will probably come if Mike does

→ Logical rules If defense then drinks

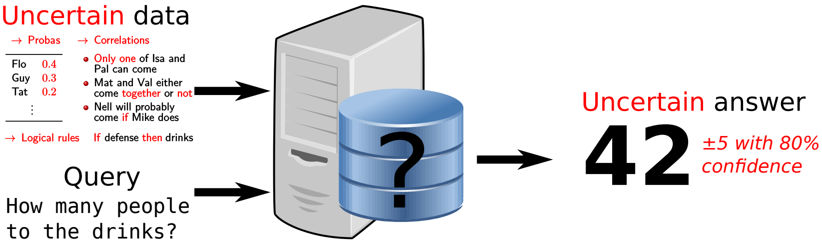
Query

```
How many people
to the drinks?
```

Summary of uncertainty goals

→ End goal: A database system with first-class uncertainty

- Feed uncertain data to the system
- Get uncertain query results



Summary of uncertainty goals

→ End goal: A database system with first-class uncertainty

- Feed uncertain data to the system
- Get uncertain query results

Uncertain data

→ Probas

Flo	0.4
Guy	0.3
Tat	0.2
⋮	

→ Correlations

- Only one of Isa and Pal can come
- Mat and Val either come together or not
- Nell will probably come if Mike does

→ Logical rules If defense then drinks

Query

`How many people to the drinks?`

Uncertain answer

42 ±5 with 80% confidence

- Representing our knowledge about the data
- Computing numerical probabilities
- Reasoning with logical constraints

Summary of uncertainty goals

→ End goal: A database system with first-class uncertainty

- Feed uncertain data to the system
- Get uncertain query results

Uncertain data

→ Probas

Flo	0.4
Guy	0.3
Tat	0.2
⋮	

→ Correlations

- Only one of Isa and Pal can come
- Mat and Val either come together or not
- Nell will probably come if Mike does

→ Logical rules: If defense then drinks

Query

```
How many people
to the drinks?
```

Uncertain answer

42 ±5 with 80% confidence

- Representing our knowledge about the data

⇒ Computing numerical probabilities

- Reasoning with logical constraints

Why are uncertainty and probabilities challenging?

Uncertain attendees

Dave	0.4
Guy	0.3
Tat	0.2
Ell	0.1
$\vdots$	

Why are uncertainty and probabilities challenging?

Uncertain attendees

Dave	0.4
Guy	0.3
Tat	0.2
Ell	0.1
⋮	

People who should meet

Dave	Guy
Ell	Tat
Ell	Guy

Why are uncertainty and probabilities challenging?

Uncertain attendees

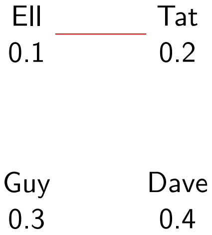
Dave	0.4
Guy	0.3
Tat	0.2
Ell	0.1
⋮	

People who should meet

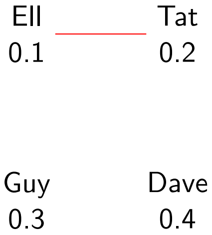
Dave	Guy
Ell	Tat
Ell	Guy

What is the probability that one of the pairs can meet?

Computing probabilities



Computing probabilities



$$0.1 \times 0.2$$

Computing probabilities

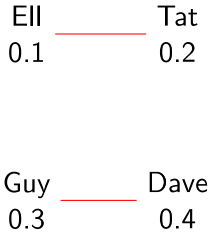
Ell 0.1 — Tat 0.2

Guy 0.3

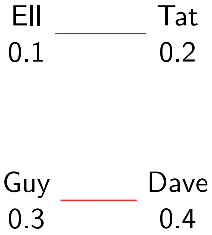
Dave 0.4

$$0.1 \times 0.2 = 0.02$$

Computing probabilities

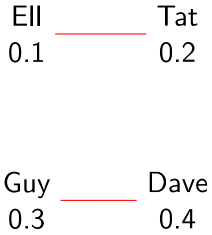


Computing probabilities



$$0.1 \times 0.2$$

Computing probabilities



$$0.1 \times 0.2 = 0.02$$

Computing probabilities

Ell ——— Tat
0.1 0.2

Guy ——— Dave
0.3 0.4

$$0.1 \times 0.2 = 0.02$$
$$0.3 \times 0.4$$

Computing probabilities

Ell — Tat
0.1 0.2

$$0.1 \times 0.2 = 0.02$$
$$0.3 \times 0.4 = 0.12$$

Guy — Dave
0.3 0.4

Computing probabilities

Ell —— Tat
0.1 0.2

Guy —— Dave
0.3 0.4

$$0.1 \times 0.2 = 0.02$$
$$0.3 \times 0.4 = 0.12$$
$$1 - (1 - 0.02) \times (1 - 0.12)$$

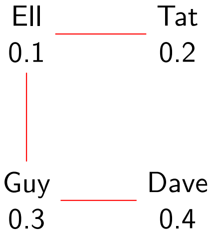
Computing probabilities

Ell ______ Tat
0.1 0.2

Guy ______ Dave
0.3 0.4

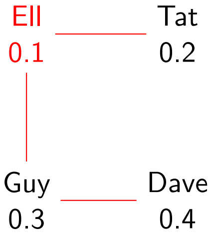
$$0.1 \times 0.2 = 0.02$$
$$0.3 \times 0.4 = 0.12$$
$$1 - (1 - 0.02) \times (1 - 0.12) = 0.1376$$

Computing probabilities

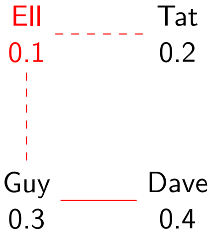


Computing probabilities

If Ell is missing:

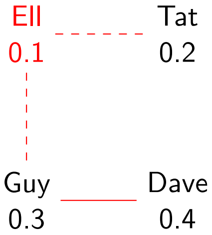


Computing probabilities



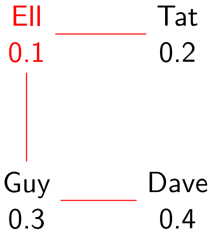
If Ell is missing:
0.3×0.4

Computing probabilities



If Ell is missing:
$0.3 \times 0.4 = 0.12$

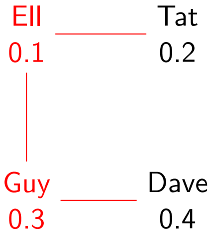
Computing probabilities



If Ell is missing:
$0.3 \times 0.4 = 0.12$

If Ell is here:

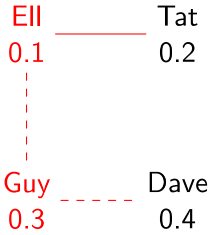
Computing probabilities



If Ell is missing:
$0.3 \times 0.4 = 0.12$

If Ell is here:
If Guy is missing:

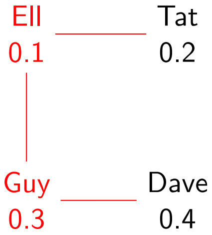
Computing probabilities



If Ell is missing:
$0.3 \times 0.4 = 0.12$

If Ell is here:
- If Guy is missing:
  - We need Tat: 0.2

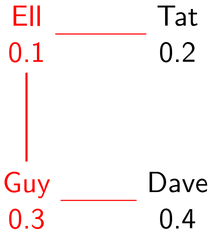
Computing probabilities



If Ell is missing:
$0.3 \times 0.4 = 0.12$

If Ell is here:
- If Guy is missing:
  - We need Tat: 0.2
- If Guy is here:

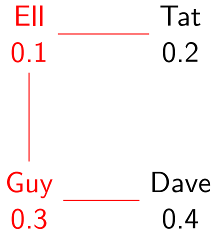
Computing probabilities



If Ell is missing:
$0.3 \times 0.4 = 0.12$

If Ell is here:
- If Guy is missing:
  - We need Tat: 0.2
- If Guy is here: success!

Computing probabilities

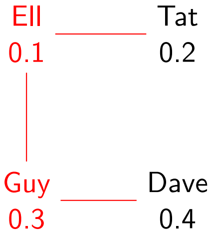


If Ell is missing:
$0.3 \times 0.4 = 0.12$

If Ell is here:
- If Guy is missing:
  - We need Tat: 0.2
- If Guy is here: success!

Total:

Computing probabilities

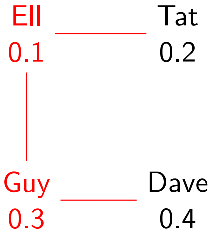


If Ell is missing:
$0.3 \times 0.4 = 0.12$

If Ell is here:
- If Guy is missing:
  - We need Tat: 0.2
- If Guy is here: success!

Total: $(1 - 0.1) \times 0.12$

Computing probabilities



If Ell is missing:

$0.3 \times 0.4 = 0.12$

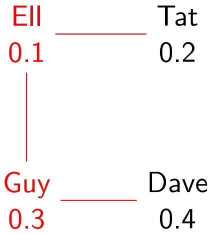
If Ell is here:

- If Guy is missing:
  - We need Tat: 0.2
- If Guy is here: success!

Total: $(1 - 0.1) \times 0.12$

$+ 0.1 \times$

Computing probabilities

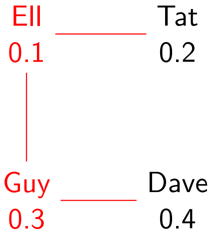


If Ell is missing:
$0.3 \times 0.4 = 0.12$

If Ell is here:
- If Guy is missing:
  - We need Tat: 0.2
- If Guy is here: success!

Total: $(1 - 0.1) \times 0.12$
$+ 0.1 \times \big(0.3 + (1 - 0.3) \times 0.2\big)$

Computing probabilities

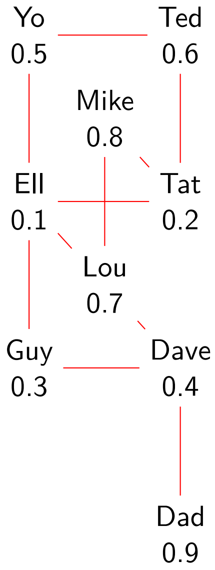


If Ell is missing:
$0.3 \times 0.4 = 0.12$

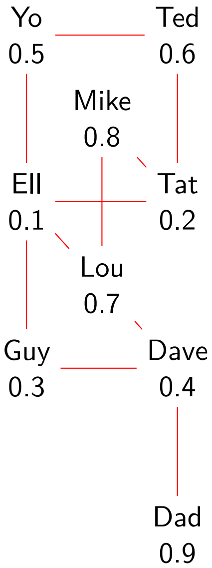
If Ell is here:
- If Guy is missing:
  - We need Tat: 0.2
- If Guy is here: success!

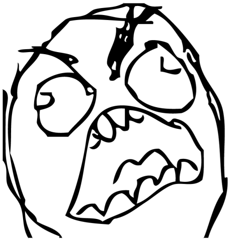
Total: $(1 - 0.1) \times 0.12$
$+ 0.1 \times \big(0.3 + (1 - 0.3) \times 0.2\big)$
$= 0.152$

Computing probabilities

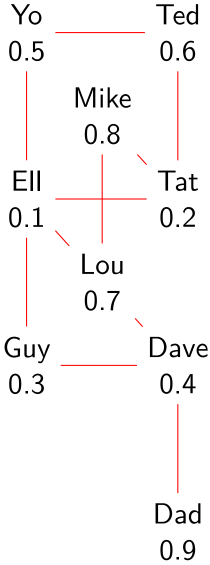


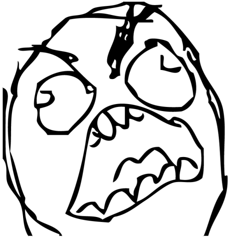
Computing probabilities





Computing probabilities





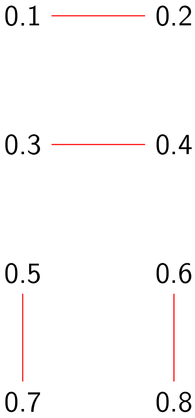
- This task is intractable (#P-hard)
- Many other tasks on uncertain data are intractable or even undecidable

My PhD topic

→ Make it easier to use uncertain data
by making assumptions on the structure of data

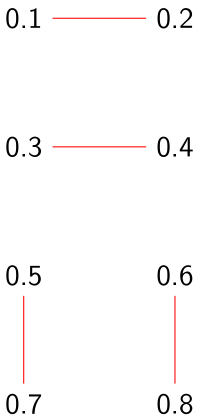
My PhD topic

→ Make it easier to use uncertain data
by making assumptions on the structure of data



My PhD topic

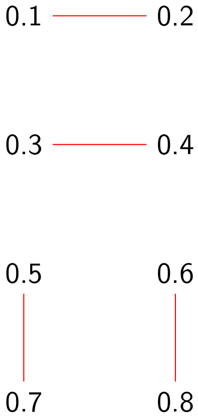
→ Make it easier to use uncertain data
by making assumptions on the structure of data



- $0.1 \times 0.2 = 0.02$

My PhD topic

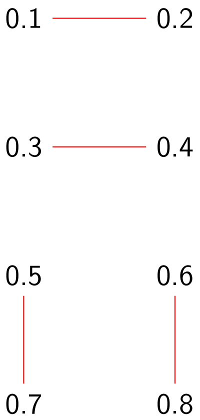
→ Make it easier to use uncertain data
by making assumptions on the structure of data



- $0.1 \times 0.2 = 0.02$
- $0.3 \times 0.4 = 0.12$

My PhD topic

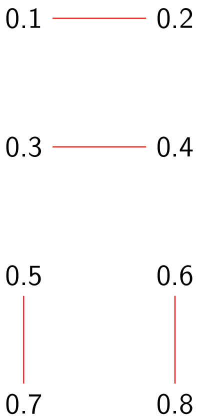
→ Make it easier to use uncertain data
by making assumptions on the structure of data



- $0.1 \times 0.2 = 0.02$
- $0.3 \times 0.4 = 0.12$
- $0.5 \times 0.7 = 0.35$

My PhD topic

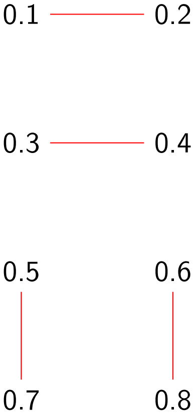
→ Make it easier to use uncertain data
by making assumptions on the structure of data



- $0.1 \times 0.2 = 0.02$
- $0.3 \times 0.4 = 0.12$
- $0.5 \times 0.7 = 0.35$
- $0.6 \times 0.8 = 0.48$

My PhD topic

→ Make it easier to use uncertain data
by making assumptions on the structure of data

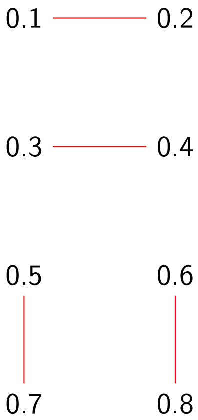


- $0.1 \times 0.2 = 0.02$
- $0.3 \times 0.4 = 0.12$
- $0.5 \times 0.7 = 0.35$
- $0.6 \times 0.8 = 0.48$

→ $1 - (1 - 0.02) \times \cdots \times (1 - 0.48)$

My PhD topic

→ Make it easier to use uncertain data
by making assumptions on the structure of data

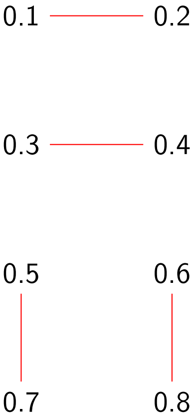


- $0.1 \times 0.2 = 0.02$
- $0.3 \times 0.4 = 0.12$
- $0.5 \times 0.7 = 0.35$
- $0.6 \times 0.8 = 0.48$

→ $1 - (1 - 0.02) \times \cdots \times (1 - 0.48)$
$= 0.7085088$

My PhD topic

→ Make it easier to use uncertain data
by making assumptions on the structure of data



- $0.1 \times 0.2 = 0.02$
- $0.3 \times 0.4 = 0.12$
- $0.5 \times 0.7 = 0.35$
- $0.6 \times 0.8 = 0.48$

→ $1 - (1 - 0.02) \times \cdots \times (1 - 0.48)$
$= 0.7085088$

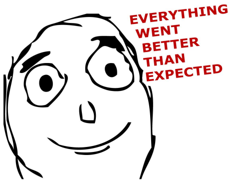


Table of contents

Roadmap

I investigated various kinds of uncertain data:

Roadmap

I investigated various kinds of uncertain data:

Partially ordered data. Representation and querying

- Possibility and certainty on ordered relations
  Preprint: A., Ba, Deutch, Senellart 2016
- Completing uncertain ordered numerical values
  Preprint: A., Amsterdamer, Milo, Senellart 2016

Roadmap

I investigated various kinds of uncertain data:

Partially ordered data. Representation and querying

- Possibility and certainty on ordered relations
  Preprint: A., Ba, Deutch, Senellart 2016
- Completing uncertain ordered numerical values
  Preprint: A., Amsterdamer, Milo, Senellart 2016

Incomplete data. Open-world reasoning under constraints

- Combining several decidable constraint languages
  A., Benedikt 2015a, IJCAI'15
- Addressing the finiteness hypothesis
  A., Benedikt 2015b, LICS'15; Thesis Part II

Roadmap

I investigated various kinds of uncertain data:

Partially ordered data. Representation and querying

- Possibility and certainty on ordered relations
  Preprint: A., Ba, Deutch, Senellart 2016
- Completing uncertain ordered numerical values
  Preprint: A., Amsterdamer, Milo, Senellart 2016

Incomplete data. Open-world reasoning under constraints

- Combining several decidable constraint languages
  A., Benedikt 2015a, IJCAI'15
- Addressing the finiteness hypothesis
  A., Benedikt 2015b, LICS'15; Thesis Part II

Probabilistic data. Query evaluation assuming treelikeness
A., Bourhis, Senellart 2015, 2016, ICALP'15, PODS'16; Thesis Part I

Roadmap

I investigated various kinds of uncertain data:

Partially ordered data. Representation and querying

- Possibility and certainty on ordered relations
  Preprint: A., Ba, Deutch, Senellart 2016
- Completing uncertain ordered numerical values
  Preprint: A., Amsterdamer, Milo, Senellart 2016

Incomplete data. Open-world reasoning under constraints

- Combining several decidable constraint languages
  A., Benedikt 2015a, IJCAI'15
- Addressing the finiteness hypothesis
  A., Benedikt 2015b, LICS'15; Thesis Part II

Probabilistic data. Query evaluation assuming treelikeness
A., Bourhis, Senellart 2015, 2016, ICALP'15, PODS'16; Thesis Part I

Other work: (A. 2014, 2015a,b; A., Allauzen, Mohri 2015; A., Amsterdamer, Milo 2014a,b; A., Maniu, Senellart 2015; A., Galárraga, Preda, Suchanek 2014; Talaika, Biega, A., Suchanek 2015; Tang, A., Senellart, Bressan 2014a,b)

Uncertain ordered relations

Food

tiramisu kougelhopf

bretzel

munster

Drinks

champagne

riesling

Uncertain ordered relations

Food

tiramisu kougelhopf

bretzel

munster

Drinks

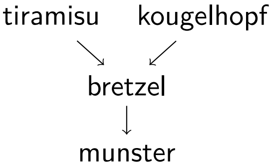
champagne

riesling

- I partially know guest preferences

Uncertain ordered relations

Food



Drinks

- I partially know guest preferences

Uncertain ordered relations

Food

tiramisu → bretzel ← kougelhopf

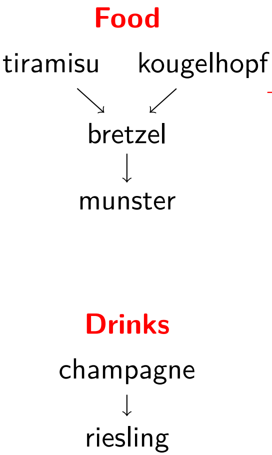
bretzel → munster

Drinks

champagne → riesling

- I partially know guest preferences
- What should my parents bring?

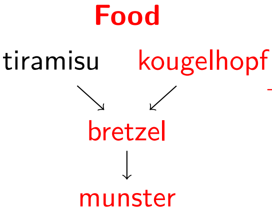
Uncertain ordered relations



- I partially know guest preferences
- What should my parents bring?

→ What are the top two Alsatian products?

Uncertain ordered relations



- I partially know guest preferences
- What should my parents bring?

→ What are the top two Alsatian products?

Uncertain ordered relations

Food

tiramisu → bretzel ← kougelhopf

bretzel → munster

Drinks

champagne → riesling

- I partially know guest preferences
- What should my parents bring?

→ What are the top two Alsatian products?

Possible:

riesling → kougelhopf

Uncertain ordered relations

Food

tiramisu → bretzel ← kougelhopf

bretzel

↓

munster

Drinks

champagne

↓

riesling

- I partially know guest preferences
- What should my parents bring?

→ What are the top two Alsatian products?

Possible:

riesling

↓

kougelhopf

Not possible:

kougelhopf

↓

munster

Uncertain ordered relations

Food

tiramisu ↘ bretzel ↙ kougelhopf

bretzel ↓ munster

- I partially know guest preferences
- What should my parents bring?

→ What are the top two Alsatian products?

Possible: riesling ↓ kougelhopf

Not possible: kougelhopf ↓ munster

Drinks

champagne ↓ riesling

→ I extended relational algebra (bag semantics, including aggregation) to uncertain ordered data

→ I showed complexity results for possible and certain answers depending on the query and data

Uncertain numerical values

- How much food do people eat?

Uncertain numerical values

- How much food do people eat?
- Let's ask friends who defended recently

Uncertain numerical values

small sweet	tiny both	small salty
medium sweet	small both	medium salty
large sweet	medium both	large salty
	large both	

- How much food do people eat?
- Let's ask friends who defended recently

Uncertain numerical values

small
sweet

tiny
both

small
salty

medium
sweet

small
both

medium
salty

large
sweet

medium
both

large
salty

large
both

- How much food do people eat?
- Let's ask friends who defended recently

Uncertain numerical values

small sweet

tiny both

small salty

medium sweet

small both

medium salty

large sweet

medium both

large salty

large both

- How much food do people eat?
- Let's ask friends who defended recently
- How to estimate for my own defense?

Uncertain numerical values

small sweet

tiny both

small salty

medium sweet

small both

medium salty

large sweet

medium both

large salty

large both

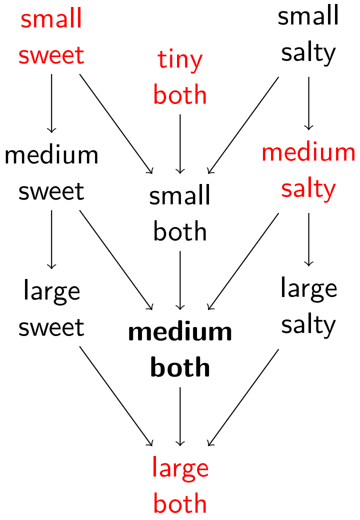
- How much food do people eat?
- Let's ask friends who defended recently
- How to estimate for my own defense?

Uncertain numerical values

small sweet	tiny both	small salty
medium sweet	small both	medium salty
large sweet	**medium both**	large salty
	large both	

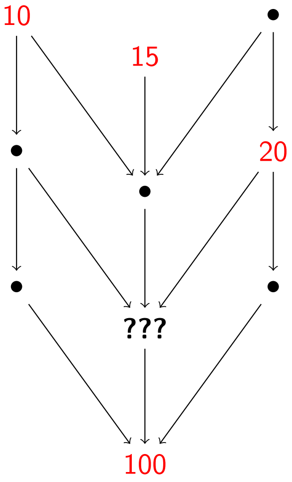
- How much food do people eat?
- Let's ask friends who defended recently
- How to estimate for my own defense?
- Some order relations are implied

Uncertain numerical values



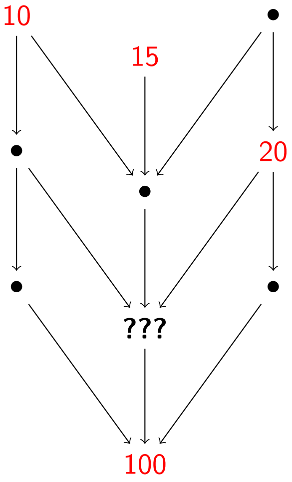
- How much food do people eat?
- Let's ask friends who defended recently
- How to estimate for my own defense?
- Some order relations are implied

Uncertain numerical values



- How much food do people eat?
- Let's ask friends who defended recently
- How to estimate for my own defense?
- Some order relations are implied

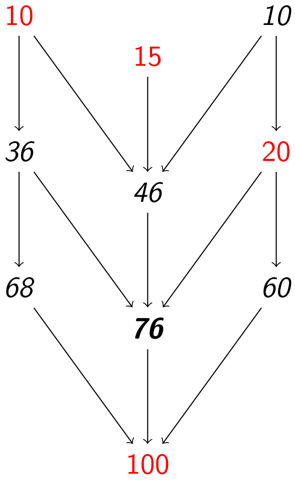
Uncertain numerical values



- How much food do people eat?
- Let's ask friends who defended recently
- How to estimate for my own defense?
- Some order relations are implied

→ I extended interpolation to posets based on integration on polytopes

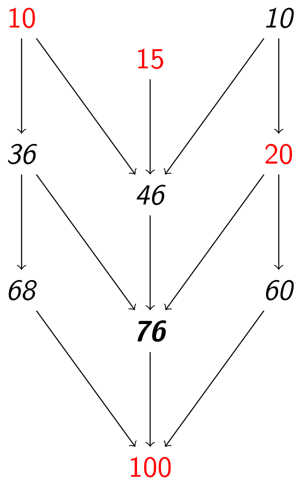
Uncertain numerical values



- How much food do people eat?
- Let's ask friends who defended recently
- How to estimate for my own defense?
- Some order relations are implied

→ I extended interpolation to posets based on integration on polytopes

Uncertain numerical values



- How much food do people eat?
- Let's ask friends who defended recently
- How to estimate for my own defense?
- Some order relations are implied

→ I extended interpolation to posets based on integration on polytopes

→ I showed hardness of the problem and identified tractable cases

Open-world query answering

Incomplete data:

- Fabian advises Luis
- Fabian is at the defense
- Fabian is in DBWeb

Open-world query answering

Incomplete data:

- Fabian advises Luis
- Fabian is at the defense
- Fabian is in DBWeb

Logical constraints:

- People at the defense will have drinks
- All DBWeb students will have drinks
- If your advisor is in DBWeb
  then you are a DBWeb student

Open-world query answering

Incomplete data:

- Fabian advises Luis
- Fabian is at the defense
- Fabian is in DBWeb

Logical constraints:

- People at the defense will have drinks
- All DBWeb students will have drinks
- If your advisor is in DBWeb then you are a DBWeb student

Is the following query certain?

→ Will a DBWeb student meet their advisor at the drinks?

Open-world query answering

Incomplete data:

- Fabian advises Luis
- Fabian is at the defense
- Fabian is in DBWeb
- Fabian comes to the drinks

Logical constraints:

- People at the defense will have drinks
- All DBWeb students will have drinks
- If your advisor is in DBWeb then you are a DBWeb student

Is the following query certain?

→ Will a DBWeb student meet their advisor at the drinks?

Open-world query answering

Incomplete data:

- Fabian advises Luis
- Fabian is at the defense
- Fabian is in DBWeb
- Fabian comes to the drinks
- Luis is a DBWeb student

Logical constraints:

- People at the defense will have drinks
- All DBWeb students will have drinks
- If your advisor is in DBWeb then you are a DBWeb student

Is the following query certain?

→ Will a DBWeb student meet their advisor at the drinks?

Open-world query answering

Incomplete data:

- Fabian advises Luis
- Fabian is at the defense
- Fabian is in DBWeb
- Fabian comes to the drinks
- Luis is a DBWeb student
- Luis comes to the drinks

Logical constraints:

- People at the defense will have drinks
- All DBWeb students will have drinks
- If your advisor is in DBWeb then you are a DBWeb student

Is the following query certain?

→ Will a DBWeb student meet their advisor at the drinks?

Open-world query answering

Incomplete data:

- Fabian advises Luis
- Fabian is at the defense
- Fabian is in DBWeb
- Fabian comes to the drinks
- Luis is a DBWeb student
- Luis comes to the drinks

Logical constraints:

- People at the defense will have drinks
- All DBWeb students will have drinks
- If your advisor is in DBWeb then you are a DBWeb student

Is the following query certain?

→ Will a DBWeb student meet their advisor at the drinks?

→ Yes!

Open-world query answering

Incomplete data:

- Fabian advises Luis
- Fabian is at the defense
- Fabian is in DBWeb
- Fabian comes to the drinks
- Luis is a DBWeb student
- Luis comes to the drinks

Logical constraints:

- People at the defense will have drinks
- All DBWeb students will have drinks
- If your advisor is in DBWeb then you are a DBWeb student

Is the following query certain?

→ Will a DBWeb student meet their advisor at the drinks?

→ Yes!

→ For which constraint languages is this task decidable?

Expressive open-world query answering

Different communities use different kinds of constraints:

- Constraints with facts of arity > 2

Expressive open-world query answering

Different communities use different kinds of constraints:

- Constraints with facts of arity > 2
  - Fabian supervises Luis: arity 2
  - Antoine's defense is in B312 on Monday: arity 3

Expressive open-world query answering

Different communities use different kinds of constraints:

- Constraints with facts of arity > 2
  - Fabian supervises Luis: arity 2
  - Antoine's defense is in B312 on Monday: arity 3
- Constraints with number restrictions

Expressive open-world query answering

Different communities use different kinds of constraints:

- Constraints with facts of arity > 2
  - Fabian supervises Luis: arity 2
  - Antoine's defense is in B312 on Monday: arity 3
- Constraints with number restrictions
  - Everyone can invite at most one person
  - Students have at most two advisors

Expressive open-world query answering

Different communities use different kinds of constraints:

- Constraints with facts of arity > 2
  - Fabian supervises Luis: arity 2
  - Antoine's defense is in B312 on Monday: arity 3
- Constraints with number restrictions
  - Everyone can invite at most one person
  - Students have at most two advisors

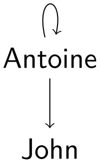
→ I proposed a language that combines these features (with some restrictions on the higher-arity rules)

→ I showed that query answering for the language is decidable

Query answering assuming finiteness

Consider the guests to the defense, $\longrightarrow$ shows who invites whom

Data:

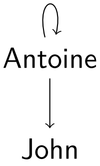


Rules:

Query answering assuming finiteness

Consider the guests to the defense, $\longrightarrow$ shows who invites whom

Data:



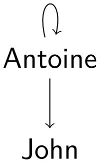
Rules:

- Each guest invites someone

Query answering assuming finiteness

Consider the guests to the defense, $\longrightarrow$ shows who invites whom

Data:



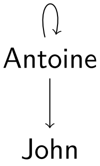
Rules:

- Each guest invites someone
- Nobody is invited by two people

Query answering assuming finiteness

Consider the guests to the defense, $\longrightarrow$ shows who invites whom

Data:



Rules:

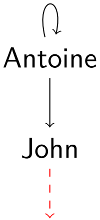
- Each guest invites someone
- Nobody is invited by two people

$\rightarrow$ Is this sensible?

Query answering assuming finiteness

Consider the guests to the defense, $\longrightarrow$ shows who invites whom

Data:



Rules:

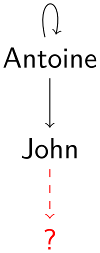
- Each guest invites someone
- Nobody is invited by two people

$\rightarrow$ Is this sensible?

Query answering assuming finiteness

Consider the guests to the defense, ⟶ shows who invites whom

Data:



Rules:

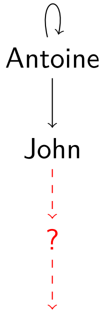
- Each guest invites someone
- Nobody is invited by two people

→ Is this sensible?

Query answering assuming finiteness

Consider the guests to the defense, ⟶ shows who invites whom

Data:



Rules:

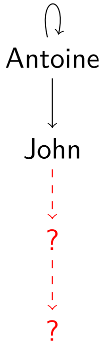
- Each guest invites someone
- Nobody is invited by two people

→ Is this sensible?

Query answering assuming finiteness

Consider the guests to the defense, ⟶ shows who invites whom

Data:



Rules:

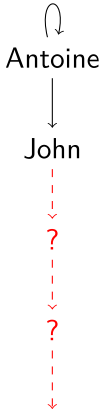
- Each guest invites someone
- Nobody is invited by two people

→ Is this sensible?

Query answering assuming finiteness

Consider the guests to the defense, ⟶ shows who invites whom

Data:



Rules:

- Each guest invites someone
- Nobody is invited by two people

→ Is this sensible?

Query answering assuming finiteness

Consider the guests to the defense, ⟶ shows who invites whom

Data:

Rules:

- Each guest invites someone
- Nobody is invited by two people

→ Is this sensible?

Query answering assuming finiteness

Consider the guests to the defense, ⟶ shows who invites whom

Data:

Rules:

- Each guest invites someone
- Nobody is invited by two people

→ Is this sensible? **No!**

Query answering assuming finiteness

Consider the guests to the defense, ⟶ shows who invites whom

Data:

Rules:

- Each guest invites someone
- Nobody is invited by two people
- There are finitely many guests!

Query answering assuming finiteness

Consider the guests to the defense, ⟶ shows who invites whom

Data:

Rules:

- Each guest invites someone
- Nobody is invited by two people
- There are finitely many guests!

→ Can we do reasoning assuming finiteness?

Query answering assuming finiteness

Consider the guests to the defense, ⟶ shows who invites whom

Data:

Rules:

- Each guest invites someone
- Nobody is invited by two people
- There are finitely many guests!

→ Can we do reasoning assuming finiteness?

→ What difference does it make?

Finite open-world query answering

- I study the following constraints on arbitrary arity:
  - Inclusion dependencies with one exported element

Finite open-world query answering

- I study the following constraints on arbitrary arity:
  - Inclusion dependencies with one exported element

    $\rightarrow$ If x invites y then y invites some z

Finite open-world query answering

- I study the following constraints on arbitrary arity:
  - Inclusion dependencies with one exported element
    $\rightarrow$ If x invites y then y invites some z
  - Functional dependencies

Finite open-world query answering

- I study the following constraints on arbitrary arity:
  - Inclusion dependencies with one exported element

    $\rightarrow$ If x invites y then y invites some z
  - Functional dependencies

    $\rightarrow$ If x and y invite z then $x = y$

Finite open-world query answering

- I study the following constraints on arbitrary arity:
  - Inclusion dependencies with one exported element
    - $\rightarrow$ If x invites y then y invites some z
  - Functional dependencies
    - $\rightarrow$ If x and y invite z then $x = y$
- I showed the following results (difficult proof):
  - We can compute new constraints implied by finiteness using (Cosmadakis, Kanellakis, Vardi 1990)

Finite open-world query answering

- I study the following constraints on arbitrary arity:
  - Inclusion dependencies with one exported element
    $\rightarrow$ If x invites y then y invites some z
  - Functional dependencies
    $\rightarrow$ If x and y invite z then $x = y$
- I showed the following results (difficult proof):
  - We can compute new constraints implied by finiteness using (Cosmadakis, Kanellakis, Vardi 1990)
  - With the new constraints, we can forget finiteness

Finite open-world query answering

- I study the following constraints on arbitrary arity:
  - Inclusion dependencies with one exported element
    → If x invites y then y invites some z
  - Functional dependencies
    → If x and y invite z then $x = y$
- I showed the following results (difficult proof):
  - We can compute new constraints implied by finiteness using (Cosmadakis, Kanellakis, Vardi 1990)
  - With the new constraints, we can forget finiteness

→ First techniques for open-world query answering with arbitrary arity signatures and functional dependencies where assuming finiteness makes a difference

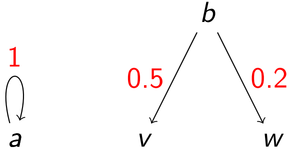
Table of contents

Tuple-independent databases (TID)

S		
a	*a*	1
b	*v*	0.5
b	*w*	0.2

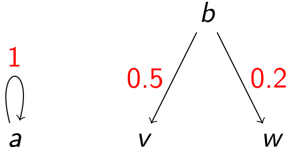
Tuple-independent databases (TID)

S		
a	a	1
b	v	0.5
b	w	0.2



Tuple-independent databases (TID)

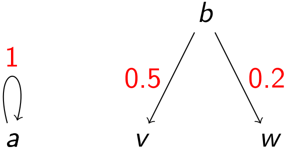
S		
a	*a*	1
b	*v*	0.5
b	*w*	0.2



This TID instance represents the following probability distribution:

Tuple-independent databases (TID)

S		
a	a	1
b	v	0.5
b	w	0.2



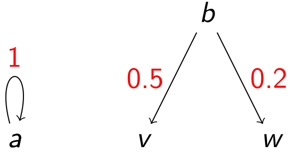
This TID instance represents the following probability distribution:

0.5×0.2

S	
a	a
b	v
b	w

Tuple-independent databases (TID)

S		
a	a	1
b	v	0.5
b	w	0.2



This TID instance represents the following probability distribution:

0.5×0.2

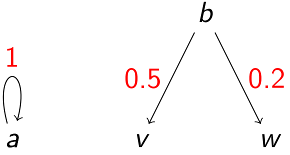
S	
a	a
b	v
b	w

$0.5 \times (1 - 0.2)$

S	
a	a
b	v

Tuple-independent databases (TID)

S		
a	a	1
b	v	0.5
b	w	0.2



This TID instance represents the following probability distribution:

0.5×0.2

S	
a	a
b	v
b	w

$0.5 \times (1 - 0.2)$

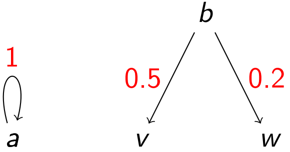
S	
a	a
b	v

$(1 - 0.5) \times 0.2$

S	
a	a
b	w

Tuple-independent databases (TID)

S		
a	*a*	1
b	*v*	0.5
b	*w*	0.2



This TID instance represents the following probability distribution:

0.5×0.2

S	
a	*a*
b	*v*
b	*w*

$0.5 \times (1 - 0.2)$

S	
a	*a*
b	*v*

$(1 - 0.5) \times 0.2$

S	
a	*a*
b	*w*

$(1 - 0.5) \times (1 - 0.2)$

S	
a	*a*

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$$

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\ R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \land S(x, y) \land T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \land S(x, y) \land T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

- The query is true iff $R(b)$ is here and one of:

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

- The query is true iff $R(b)$ is here and one of:
  - $S(b, v)$ and $T(v)$ are here

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

- The query is true iff $R(b)$ is here and one of:
  - $S(b, v)$ and $T(v)$ are here
  - $S(b, w)$ and $T(w)$ are here

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	*a*	1
b	*v*	0.5
b	*w*	0.2

T	
v	0.3
w	0.7
b	1

- The query is true iff $R(b)$ is here and one of:
  - $S(b, v)$ and $T(v)$ are here
  - $S(b, w)$ and $T(w)$ are here

→ Probability:

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\ R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

- The query is true iff $R(b)$ is here and one of:
  - $S(b, v)$ and $T(v)$ are here
  - $S(b, w)$ and $T(w)$ are here

→ Probability:

$0.4 \times$

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	*a*	1
b	*v*	0.5
b	*w*	0.2

T	
v	0.3
w	0.7
b	1

- The query is true iff $R(b)$ is here and one of:
  - $S(b, v)$ and $T(v)$ are here
  - $S(b, w)$ and $T(w)$ are here

→ Probability:

$0.4 \times \big(1 -$

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\ R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

- The query is true iff $R(b)$ is here and one of:
  - $S(b, v)$ and $T(v)$ are here
  - $S(b, w)$ and $T(w)$ are here

→ Probability:
$0.4 \times \big(1 - (1 - 0.5 \times 0.3)$

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \land S(x, y) \land T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

- The query is true iff $R(b)$ is here and one of:
  - $S(b, v)$ and $T(v)$ are here
  - $S(b, w)$ and $T(w)$ are here

→ Probability:
$0.4 \times \left(1 - (1 - 0.5 \times 0.3) \times (1 - 0.2 \times 0.7)\right)$

Query evaluation on probabilistic instances

We want to evaluate the probability of a query on a TID instance

$$q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$$

R	
a	1
b	0.4
c	0.6

S		
a	a	1
b	v	0.5
b	w	0.2

T	
v	0.3
w	0.7
b	1

- The query is true iff $R(b)$ is here and one of:
  - $S(b, v)$ and $T(v)$ are here
  - $S(b, w)$ and $T(w)$ are here

→ Probability:
$$0.4 \times \big(1 - (1 - 0.5 \times 0.3) \times (1 - 0.2 \times 0.7)\big) = 0.1076$$

Complexity of probabilistic query evaluation (PQE)

What is the data complexity of probabilistic query evaluation on TID depending on the class $\mathcal{Q}$ of queries and class $\mathcal{I}$ of instances?

Complexity of probabilistic query evaluation (PQE)

What is the data complexity of probabilistic query evaluation on TID depending on the class $\mathcal{Q}$ of queries and class $\mathcal{I}$ of instances?

- Existing dichotomy result: (Dalvi, Suciu 2012)
  - $\mathcal{Q}$ are (unions of) conjunctive queries, $\mathcal{I}$ is all TID instances
  - There is a class $\mathcal{S} \subseteq \mathcal{Q}$ of safe queries

Complexity of probabilistic query evaluation (PQE)

What is the data complexity of probabilistic query evaluation on TID depending on the class $\mathcal{Q}$ of queries and class $\mathcal{I}$ of instances?

- Existing dichotomy result: (Dalvi, Suciu 2012)
  - $\mathcal{Q}$ are (unions of) conjunctive queries, $\mathcal{I}$ is all TID instances
  - There is a class $\mathcal{S} \subseteq \mathcal{Q}$ of safe queries
  - PQE is PTIME for any $q \in \mathcal{S}$ on all instances

Complexity of probabilistic query evaluation (PQE)

What is the data complexity of probabilistic query evaluation on TID depending on the class $\mathcal{Q}$ of queries and class $\mathcal{I}$ of instances?

- Existing dichotomy result: (Dalvi, Suciu 2012)
  - $\mathcal{Q}$ are (unions of) conjunctive queries, $\mathcal{I}$ is all TID instances
  - There is a class $\mathcal{S} \subseteq \mathcal{Q}$ of safe queries
  - PQE is PTIME for any $q \in \mathcal{S}$ on all instances
  - PQE is #P-hard for any $q \in \mathcal{Q} \backslash \mathcal{S}$ on all instances

Complexity of probabilistic query evaluation (PQE)

What is the data complexity of probabilistic query evaluation on TID depending on the class $\mathcal{Q}$ of queries and class $\mathcal{I}$ of instances?

- Existing dichotomy result: (Dalvi, Suciu 2012)
  - $\mathcal{Q}$ are (unions of) conjunctive queries, $\mathcal{I}$ is all TID instances
  - There is a class $\mathcal{S} \subseteq \mathcal{Q}$ of safe queries
  - PQE is PTIME for any $q \in \mathcal{S}$ on all instances
  - PQE is #P-hard for any $q \in \mathcal{Q} \backslash \mathcal{S}$ on all instances
  - $q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$ is unsafe!

Complexity of probabilistic query evaluation (PQE)

What is the data complexity of probabilistic query evaluation on TID depending on the class $\mathcal{Q}$ of queries and class $\mathcal{I}$ of instances?

- Existing dichotomy result: (Dalvi, Suciu 2012)
  - $\mathcal{Q}$ are (unions of) conjunctive queries, $\mathcal{I}$ is all TID instances
  - There is a class $\mathcal{S} \subseteq \mathcal{Q}$ of safe queries
  - PQE is PTIME for any $q \in \mathcal{S}$ on all instances
  - PQE is #P-hard for any $q \in \mathcal{Q} \setminus \mathcal{S}$ on all instances
  - $q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$ is unsafe!

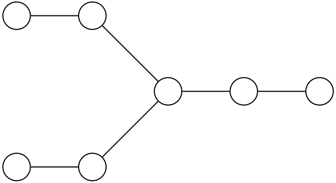
Is there a smaller class $\mathcal{I}$ such that PQE is tractable for a larger $\mathcal{Q}$?

Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)

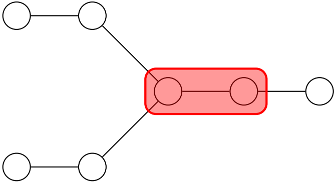
Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



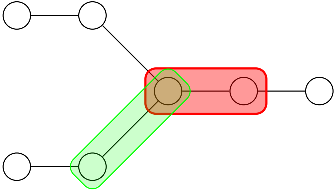
Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



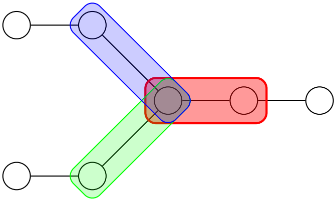
Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



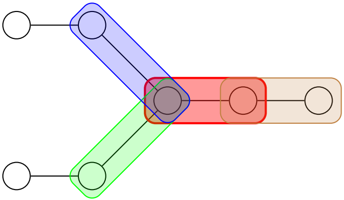
Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



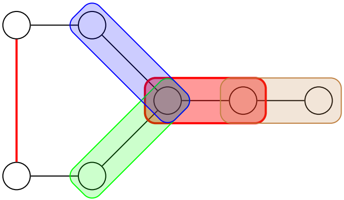
Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



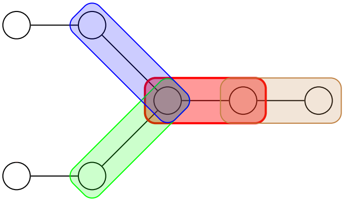
Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



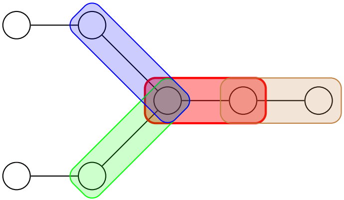
Trees and treelike instances

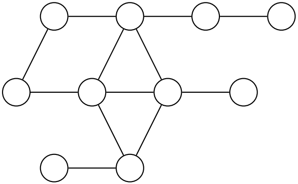
- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



Trees and treelike instances

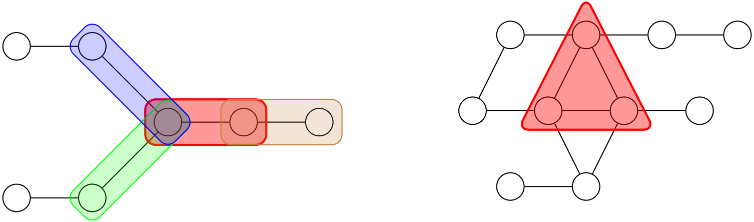
- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)





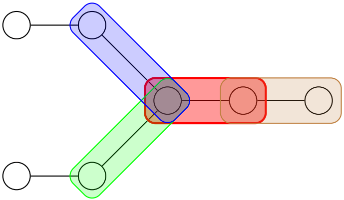
Trees and treelike instances

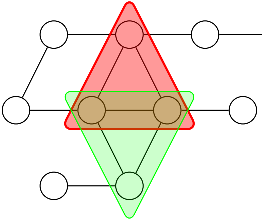
- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



Trees and treelike instances

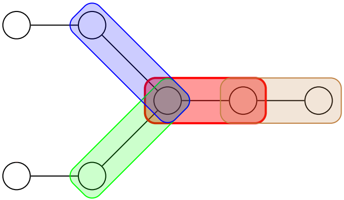
- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)

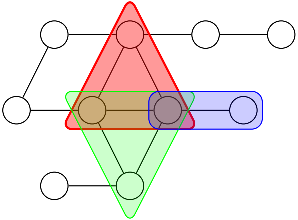




Trees and treelike instances

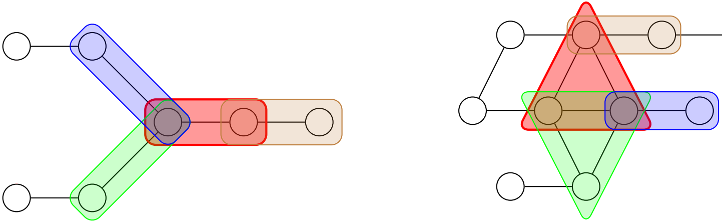
- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)





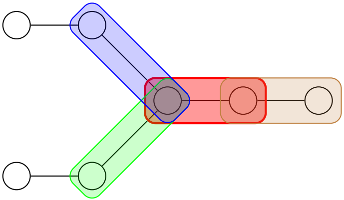
Trees and treelike instances

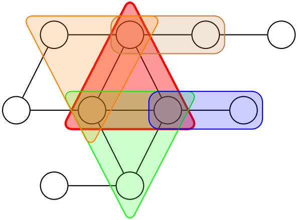
- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



Trees and treelike instances

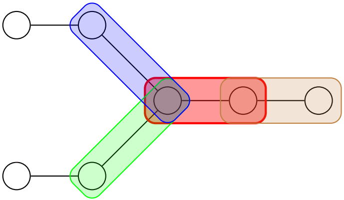
- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)

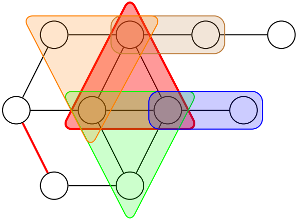




Trees and treelike instances

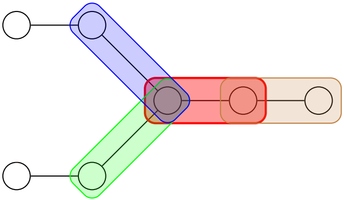
- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)

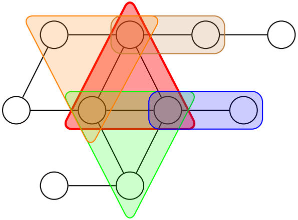




Trees and treelike instances

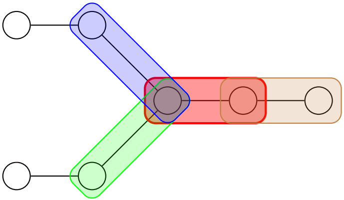
- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)

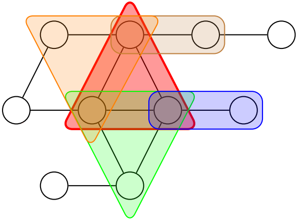




Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)

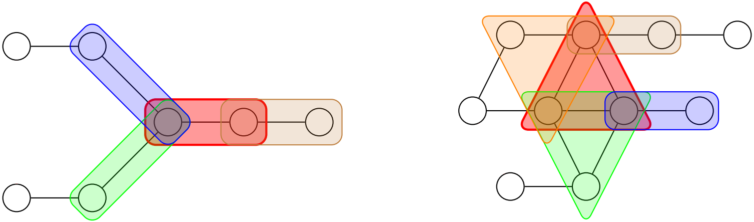




  - Trees have treewidth 1
  - Cycles have treewidth 2
  - k-cliques and $(k-1)$-grids have treewidth $k-1$

Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)



  - Trees have treewidth 1
  - Cycles have treewidth 2
  - k-cliques and $(k-1)$-grids have treewidth $k-1$

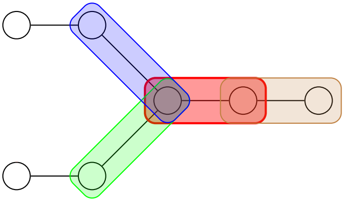
$\rightarrow$ Known results (Courcelle 1990):

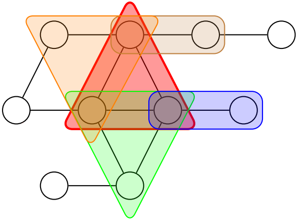
  - $\mathcal{I}$: treelike instances; $\mathcal{Q}$: monadic second-order queries

  $\rightarrow$ non-probabilistic QE is in linear time

Trees and treelike instances

- Idea: let $\mathcal{I}$ be treelike instances (constant bound on treewidth)





  - Trees have treewidth 1
  - Cycles have treewidth 2
  - k-cliques and $(k-1)$-grids have treewidth $k-1$

$\rightarrow$ Known results (Courcelle 1990):

  - $\mathcal{I}$: treelike instances; $\mathcal{Q}$: monadic second-order queries

  $\rightarrow$ non-probabilistic QE is in linear time

$\rightarrow$ Does this extend to probabilistic QE?

Our main result

An instance-based dichotomy result:

Upper bound.

For $\mathcal{I}$ the treelike instances and $\mathcal{Q}$ the MSO queries

$\rightarrow$ PQE is in linear time modulo arithmetic costs

Our main result

An instance-based dichotomy result:

Upper bound.

For $\mathcal{I}$ the treelike instances and $\mathcal{Q}$ the MSO queries

$\rightarrow$ PQE is in linear time modulo arithmetic costs

- Also for expressive provenance representations
- Also with bounded-treewidth correlations

Our main result

An instance-based dichotomy result:

Upper bound.

For $\mathcal{I}$ the treelike instances and $\mathcal{Q}$ the MSO queries

$\rightarrow$ PQE is in linear time modulo arithmetic costs

- Also for expressive provenance representations
- Also with bounded-treewidth correlations

Lower bound.

For any unbounded-tw family $\mathcal{I}$ and $\mathcal{Q}$ the FO queries

$\rightarrow$ PQE is #P-hard under RP reductions assuming:

- Signature arity is 2 (graphs)
- High-tw instances in $\mathcal{I}$ are easily constructible

Technical tool: lineages

The lineage of a query q on an instance I:

- Boolean function ϕ whose variables are the facts of I
- A subinstance of I satisfies q iff ϕ is true for that valuation

Technical tool: lineages

The lineage of a query q on an instance I:

- Boolean function ϕ whose variables are the facts of I
- A subinstance of I satisfies q iff ϕ is true for that valuation

Example: $q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$

Technical tool: lineages

The lineage of a query q on an instance I:

- Boolean function ϕ whose variables are the facts of I
- A subinstance of I satisfies q iff ϕ is true for that valuation

Example: $q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$

R	
a	f_1
b	f_2
c	f_3

S		
a	a	g_1
b	v	g_2
b	w	g_3

T	
v	h_1
w	h_2
b	h_3

Technical tool: lineages

The lineage of a query q on an instance I:

- Boolean function ϕ whose variables are the facts of I
- A subinstance of I satisfies q iff ϕ is true for that valuation

Example: $q : \exists x\, y\ R(x) \wedge S(x, y) \wedge T(y)$

R	
a	f_1
b	f_2
c	f_3

S		
a	a	g_1
b	v	g_2
b	w	g_3

T	
v	h_1
w	h_2
b	h_3

$\rightarrow$ Lineage:

Technical tool: lineages

The lineage of a query q on an instance I:

- Boolean function ϕ whose variables are the facts of I
- A subinstance of I satisfies q iff ϕ is true for that valuation

Example: $q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$

R	
a	f_1
b	f_2
c	f_3

S		
a	a	g_1
b	v	g_2
b	w	g_3

T	
v	h_1
w	h_2
b	h_3

$\rightarrow$ Lineage: $f_2 \wedge$

Technical tool: lineages

The lineage of a query q on an instance I:

- Boolean function ϕ whose variables are the facts of I
- A subinstance of I satisfies q iff ϕ is true for that valuation

Example: $q : \exists x\, y\; R(x) \land S(x, y) \land T(y)$

R	
a	f_1
b	f_2
c	f_3

S		
a	a	g_1
b	v	g_2
b	w	g_3

T	
v	h_1
w	h_2
b	h_3

$\rightarrow$ Lineage: $f_2 \land \big($

Technical tool: lineages

The lineage of a query q on an instance I:

- Boolean function ϕ whose variables are the facts of I
- A subinstance of I satisfies q iff ϕ is true for that valuation

Example: $q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$

R	
a	f_1
b	f_2
c	f_3

S		
a	a	g_1
b	v	g_2
b	w	g_3

T	
v	h_1
w	h_2
b	h_3

$\rightarrow$ Lineage: $f_2 \wedge \big((g_2 \wedge h_1)$

Technical tool: lineages

The lineage of a query q on an instance I:

- Boolean function ϕ whose variables are the facts of I
- A subinstance of I satisfies q iff ϕ is true for that valuation

Example: $q : \exists x\, y\; R(x) \land S(x, y) \land T(y)$

R	
a	f_1
b	f_2
c	f_3

S		
a	a	g_1
b	v	g_2
b	w	g_3

T	
v	h_1
w	h_2
b	h_3

$\rightarrow$ Lineage: $f_2 \land \big((g_2 \land h_1) \lor (g_3 \land h_2)\big)$

Technical tool: lineages

The lineage of a query q on an instance I:

- Boolean function ϕ whose variables are the facts of I
- A subinstance of I satisfies q iff ϕ is true for that valuation

Example: $q : \exists x\, y\; R(x) \wedge S(x, y) \wedge T(y)$

R	
a	f_1
b	f_2
c	f_3

S		
a	a	g_1
b	v	g_2
b	w	g_3

T	
v	h_1
w	h_2
b	h_3

$\rightarrow$ Lineage: $f_2 \wedge \big((g_2 \wedge h_1) \vee (g_3 \wedge h_2)\big)$

$\rightarrow$ For all $\nu : I \rightarrow \{0, 1\}$ we have $\nu(\phi) = 1$ iff $\{F \in I \mid \nu(F) = 1\} \models q$

Using lineages

- Use lineage for PQE:

Using lineages

- Use lineage for PQE:
  - Compute a lineage representation efficiently

Using lineages

- Use lineage for PQE:
  - Compute a lineage representation efficiently

  $\rightarrow$ Probability of the lineage $=$ probability of the query

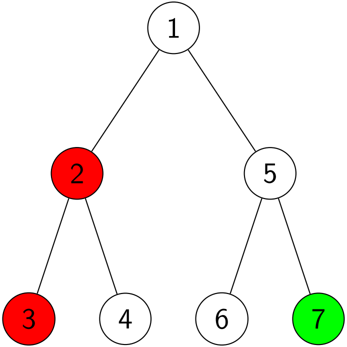
Using lineages

- Use lineage for PQE:
  - Compute a lineage representation efficiently
  - → Probability of the lineage = probability of the query
  - Compute the lineage probability efficiently
    (show it is not #P-hard as in the general case)

Uncertain trees

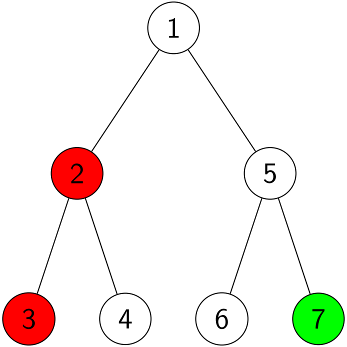
- First compute lineages on uncertain trees then use (Courcelle 1990)

Uncertain trees



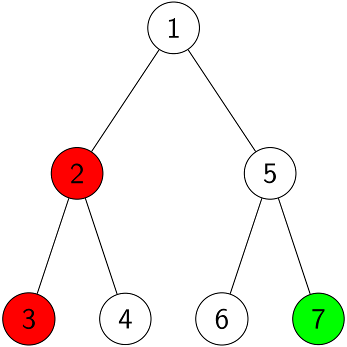
- First compute lineages on uncertain trees then use (Courcelle 1990)
- Uncertain trees: node labels may be discarded

Uncertain trees



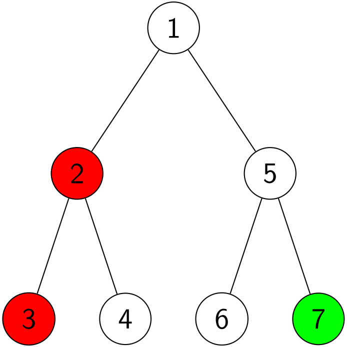
- First compute lineages on uncertain trees then use (Courcelle 1990)
- Uncertain trees: node labels may be discarded
- A valuation indicates which labels are kept

Uncertain trees



- First compute lineages on uncertain trees then use (Courcelle 1990)
- Uncertain trees: node labels may be discarded
- A valuation indicates which labels are kept
- Example query:
  "Is there both a red and a green node?"

Uncertain trees

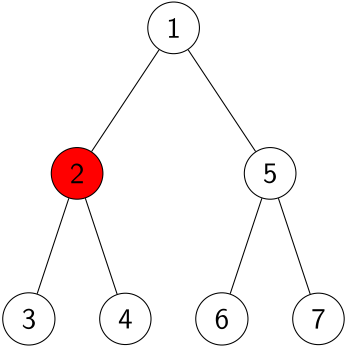


- First compute lineages on uncertain trees then use (Courcelle 1990)
- Uncertain trees: node labels may be discarded
- A valuation indicates which labels are kept
- Example query:
  "Is there both a red and a green node?"

Valuation: $\{2, 3, 7\}$

The query is true

Uncertain trees

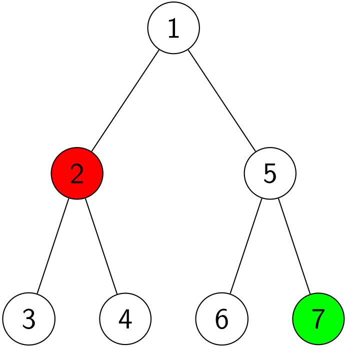


- First compute lineages on uncertain trees then use (Courcelle 1990)
- Uncertain trees: node labels may be discarded
- A valuation indicates which labels are kept
- Example query:
  "Is there both a red and a green node?"

Valuation: $\{2\}$

The query is false

Uncertain trees

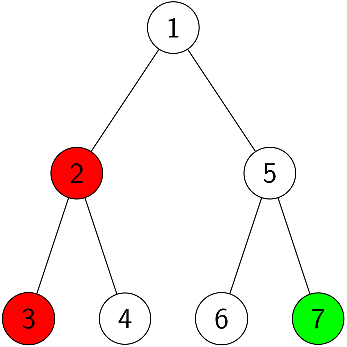


- First compute lineages on uncertain trees then use (Courcelle 1990)
- Uncertain trees: node labels may be discarded
- A valuation indicates which labels are kept
- Example query:
  "Is there both a red and a green node?"

Valuation: $\{2, 7\}$

The query is true

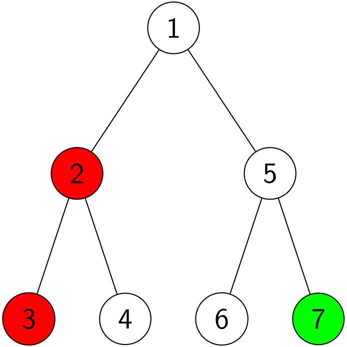
Lineage circuits on trees



q: Is there both a red and a green node?

- Which valuations satisfy q? ($\Leftrightarrow$ lineage)

Lineage circuits on trees

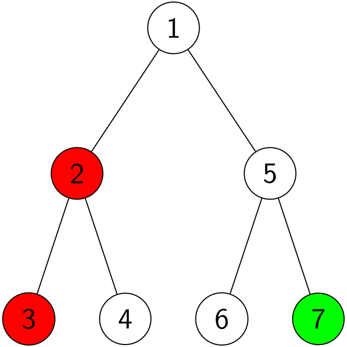


q: Is there both a red and a green node?

- Which valuations satisfy q? ($\Leftrightarrow$ lineage)
- Lineage circuit of a query q on an uncertain tree T
  - Boolean circuit C
  - with input gates g_2, g_3, g_7

  $\rightarrow$ $\nu(T)$ satisfies q iff $\nu(C)$ is true

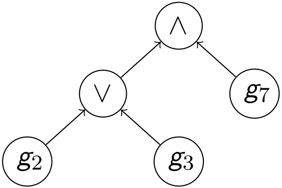
Lineage circuits on trees



q: Is there both a red and a green node?

- Which valuations satisfy q? ($\Leftrightarrow$ lineage)
- Lineage circuit of a query q on an uncertain tree T
  - Boolean circuit C
  - with input gates g_2, g_3, g_7

  $\rightarrow$ $\nu(T)$ satisfies q iff $\nu(C)$ is true



Our main results

Theorem

For any query q given as a bottom-up tree automaton A, for any input tree T, we can build a lineage circuit of A on T in linear time in $|A| \cdot |T|$.

Our main results

Theorem

For any query q given as a bottom-up tree automaton A, for any input tree T, we can build a lineage circuit of A on T in linear time in $|A| \cdot |T|$.

MSO on treelike instances ⇒ MSO on trees (Courcelle 1990).

Our main results

Theorem

For any query q given as a bottom-up tree automaton A, for any input tree T, we can build a lineage circuit of A on T in linear time in $|A| \cdot |T|$.

MSO on treelike instances $\Rightarrow$ MSO on trees (Courcelle 1990).

Theorem

For any fixed MSO query q and $k \in \mathbb{N}$, for any input instance I of treewidth $\leq k$, we can build in linear time in I a lineage circuit of q on I.

Our main results

Theorem

For any query q given as a bottom-up tree automaton A, for any input tree T, we can build a lineage circuit of A on T in linear time in $|A| \cdot |T|$.

MSO on treelike instances ⇒ MSO on trees (Courcelle 1990).

Theorem

For any fixed MSO query q and $k \in \mathbb{N}$, for any input instance I of treewidth $\leq k$, we can build in linear time in I a lineage circuit of q on I.

The lineage circuits are themselves treelike, hence:

Our main results

Theorem

For any query q given as a bottom-up tree automaton A, for any input tree T, we can build a lineage circuit of A on T in linear time in $|A| \cdot |T|$.

MSO on treelike instances ⇒ MSO on trees (Courcelle 1990).

Theorem

For any fixed MSO query q and $k \in \mathbb{N}$, for any input instance I of treewidth $\leq k$, we can build in linear time in I a lineage circuit of q on I.

The lineage circuits are themselves treelike, hence:

Corollary

Probabilistic query evaluation of MSO queries on treelike instances is in linear time up to arithmetic costs.

Extension 1: general semirings

- Positive Boolean functions are a semiring $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$

Extension 1: general semirings

- Positive Boolean functions are a semiring $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$
- Provenance semirings: (Green, Karvounarakis, Tannen 2007)
  - Provenance for arbitrary (commutative) semirings
  - For queries in the positive relational algebra and Datalog

Extension 1: general semirings

- Positive Boolean functions are a semiring $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$
- Provenance semirings: (Green, Karvounarakis, Tannen 2007)
  - Provenance for arbitrary (commutative) semirings
  - For queries in the positive relational algebra and Datalog
- Our construction can be extended to $\mathbb{N}[X]$-provenance for conjunctive queries and unions of conjunctive queries (UCQ):

Extension 1: general semirings

- Positive Boolean functions are a semiring $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$
- Provenance semirings: (Green, Karvounarakis, Tannen 2007)
  - Provenance for arbitrary (commutative) semirings
  - For queries in the positive relational algebra and Datalog
- Our construction can be extended to $\mathbb{N}[X]$-provenance for conjunctive queries and unions of conjunctive queries (UCQ):

Theorem

For any fixed UCQ q and $k \in \mathbb{N}$, for any input instance I of treewidth $\leq k$, we can build in linear time a $\mathbb{N}[X]$-provenance circuit of q on I.

Extension 1: general semirings

- Positive Boolean functions are a semiring $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$
- Provenance semirings: (Green, Karvounarakis, Tannen 2007)
  - Provenance for arbitrary (commutative) semirings
  - For queries in the positive relational algebra and Datalog
- Our construction can be extended to $\mathbb{N}[X]$-provenance for conjunctive queries and unions of conjunctive queries (UCQ):

Theorem

For any fixed UCQ q and $k \in \mathbb{N}$,
for any input instance I of treewidth $\leq k$,
we can build in linear time a $\mathbb{N}[X]$-provenance circuit of q on I.

→ We have a linear-size (and treelike) arithmetic circuit instead of a polynomial-size $\mathbb{N}[X]$-formula

Extension 2: correlations

- Our probabilistic instances assume independence on all facts

Extension 2: correlations

- Our probabilistic instances assume independence on all facts
- More expressive: Block-Independent Disjoint instances:

Extension 2: correlations

- Our probabilistic instances assume independence on all facts
- More expressive: Block-Independent Disjoint instances:

<u>**name**</u>	**favorite**	*p*
john	kougelhopf	0.8
john	bretzel	0.2
jane	kougelhopf	0.1
jane	bretzel	0.9

Extension 2: correlations

- Our probabilistic instances assume independence on all facts
- More expressive: Block-Independent Disjoint instances:

name	favorite	p
john	kougelhopf	0.8
john	bretzel	0.2
jane	kougelhopf	0.1
jane	bretzel	0.9

Theorem

Probabilistic query evaluation of MSO queries on treelike BID is in linear time up to arithmetic operations.

Extension 2: correlations

- Our probabilistic instances assume independence on all facts
- More expressive: Block-Independent Disjoint instances:

name	favorite	p
john	kougelhopf	0.8
john	bretzel	0.2
jane	kougelhopf	0.1
jane	bretzel	0.9

Theorem

Probabilistic query evaluation of MSO queries on treelike BID is in linear time up to arithmetic operations.

Generalises to pc-tables with treelike correlations

Lower bound

- Class $\mathcal{I}$ of unbounded-treewidth instances, query q in class $\mathcal{Q}$
- Show that probabilistic query evaluation of q on $\mathcal{I}$ is hard

Lower bound

- Class $\mathcal{I}$ of unbounded-treewidth instances, query q in class $\mathcal{Q}$
- Show that probabilistic query evaluation of q on $\mathcal{I}$ is hard

$\rightarrow$ Restrict to arity-2 ($=$ labeled graphs) for technical reasons

Lower bound

- Class $\mathcal{I}$ of unbounded-treewidth instances, query q in class $\mathcal{Q}$
- Show that probabilistic query evaluation of q on $\mathcal{I}$ is hard

→ Restrict to arity-2 (= labeled graphs) for technical reasons

→ Impose that $\mathcal{I}$ is tw-constructible:

Lower bound

- Class $\mathcal{I}$ of unbounded-treewidth instances, query q in class $\mathcal{Q}$
- Show that probabilistic query evaluation of q on $\mathcal{I}$ is hard

→ Restrict to arity-2 (= labeled graphs) for technical reasons

→ Impose that $\mathcal{I}$ is tw-constructible:

  - Given $k \in \mathbb{N}$, we can construct in time $\mathrm{Poly}(k)$ an instance of $\mathcal{I}$ of treewidth $\geq k$

Lower bound

- Class $\mathcal{I}$ of unbounded-treewidth instances, query q in class $\mathcal{Q}$
- Show that probabilistic query evaluation of q on $\mathcal{I}$ is hard

→ Restrict to arity-2 (= labeled graphs) for technical reasons

→ Impose that $\mathcal{I}$ is tw-constructible:
  - Given $k \in \mathbb{N}$, we can construct in time $\mathrm{Poly}(k)$ an instance of $\mathcal{I}$ of treewidth $\geq k$

Theorem

There is a first-order query q such that for any unbounded-tw, tw-constructible, arity-2 instance family $\mathcal{I}$, probabilistic query eval for q on $\mathcal{I}$ is #P-hard under RP reductions.

Lower bound

- Class $\mathcal{I}$ of unbounded-treewidth instances, query q in class $\mathcal{Q}$
- Show that probabilistic query evaluation of q on $\mathcal{I}$ is hard

→ Restrict to arity-2 ($=$ labeled graphs) for technical reasons

→ Impose that $\mathcal{I}$ is tw-constructible:

  - Given $k \in \mathbb{N}$, we can construct in time $\mathrm{Poly}(k)$ an instance of $\mathcal{I}$ of treewidth $\geq k$

Theorem

There is a first-order query q such that for any unbounded-tw, tw-constructible, arity-2 instance family $\mathcal{I}$, probabilistic query eval for q on $\mathcal{I}$ is #P-hard under RP reductions.

Proven by extracting arbitrary graphs as minors of high-treewidth families using (Chekuri, Chuzhoy 2014)

Table of contents

Conclusion

Main contributions to the study of uncertain data management:

Conclusion

Main contributions to the study of uncertain data management:

- I proved that reasoning is decidable for new constraint languages on incomplete data, in particular assuming finiteness

Conclusion

Main contributions to the study of uncertain data management:

- I proved that reasoning is decidable for new constraint languages on incomplete data, in particular assuming finiteness
- I proposed new representations of uncertain ordered data and proved complexity results including tractable cases

Conclusion

Main contributions to the study of uncertain data management:

- I proved that reasoning is decidable for new constraint languages on incomplete data, in particular assuming finiteness
- I proposed new representations of uncertain ordered data and proved complexity results including tractable cases
- I showed an instance-based dichotomy for probabilistic data including extensions to semiring provenance and correlations

Ongoing and future work

- Probabilistic query answering
  - Tractability in combined complexity for some queries
  - Hybrid tractability criteria based on instance and query
  - Practical implementation with partial decompositions

Ongoing and future work

- Probabilistic query answering
  - Tractability in combined complexity for some queries
  - Hybrid tractability criteria based on instance and query
  - Practical implementation with partial decompositions
- Open-world query answering
  - Find a uniform decidable language capturing our results
  - Managing order relations and transitive relations
  - Simplify and generalize our results on finiteness

Ongoing and future work

- Probabilistic query answering
  - Tractability in combined complexity for some queries
  - Hybrid tractability criteria based on instance and query
  - Practical implementation with partial decompositions
- Open-world query answering
  - Find a uniform decidable language capturing our results
  - Managing order relations and transitive relations
  - Simplify and generalize our results on finiteness
- Longer term: Extend provenance to open-world reasoning

Ongoing and future work

- Probabilistic query answering
  - Tractability in combined complexity for some queries
  - Hybrid tractability criteria based on instance and query
  - Practical implementation with partial decompositions
- Open-world query answering
  - Find a uniform decidable language capturing our results
  - Managing order relations and transitive relations
  - Simplify and generalize our results on finiteness
- Longer term: Extend provenance to open-world reasoning

Thanks for your attention!

Main publications:

(A., Amsterdamer, Milo 2014a) ICDT'14
(A., Benedikt 2015a) IJCAI'15
(A., Benedikt 2015b) LICS'15
(A. 2014) AMW'14
(A., Bourhis, Senellart 2015) ICALP'15
(A., Bourhis, Senellart 2016) PODS'16

Image sources

- Slides 2 and 14: https://openclipart.org/download/163711/database-server.svg
- Slide 3: SMSSecure https://smssecure.org/ and AOSP https://source.android.com/
- Slide 7: https://openclipart.org/download/36529/interrogation.svg
- Slide 8: http://rtw.ml.cmu.edu/, https://openclipart.org/download/25537/HMTL.svg, and https://twitter.com/cmunell
- Slide 9: https://en.wikipedia.org/wiki/Template:Disputed
- Slide 10: Zhang 2015, p. 9, Dong, Halevy, Yu 2009, p. 4, https://atlas.web.cern.ch/Atlas/GROUPS/PHYSICS/CONFNOTES/ATLAS-CONF-2015-041/fig_06b.png, https://code.google.com/p/transducersaurus/wiki/CascadeTutorial, https://www.cs.washington.edu/robotics/mcl/
- Slide 16: https://diaryofawhinyguy.files.wordpress.com/2013/01/rage-guy.png
- Slide 17: http://mylolface.com/assets/faces/happy-everything-went-better-than-expected.jpg

References I

- Amarilli, Antoine (2014). "The Possibility Problem for Probabilistic XML". In: *Proc. AMW*. URL: http://ceur-ws.org/Vol-1189/paper_2.pdf.
- Amarilli, Antoine (2015a). "Possibility for Probabilistic XML". In: *Ingénierie des Systèmes d'Information*. URL: http://arxiv.org/abs/1404.3131.
- Amarilli, Antoine (2015b). "Structurally Tractable Uncertain Data". In: *Proc. PhD Symposium of SIGMOD/PODS*. URL: http://arxiv.org/abs/1507.04955.
- Amarilli, Antoine, Cyril Allauzen, Mehryar Mohri (2015). *Minimum Bayesian Risk Methods for Automatic Speech Recognition*. United States Patent 9123333. URL: https://a3nm.net/publications/amarilli2014minimum.pdf.

References II

- Amarilli, Antoine, Yael Amsterdamer, Tova Milo (2014a). "On the Complexity of Mining Itemsets from the Crowd Using Taxonomies". In: *Proc. ICDT*. URL: http://arxiv.org/abs/1312.3248.
- Amarilli, Antoine, Yael Amsterdamer, Tova Milo (2014b). "Uncertainty in Crowd Data Sourcing Under Structural Constraints". In: *Proc. UnCrowd*. URL: http://arxiv.org/abs/1403.0783.
- Amarilli, Antoine, Michael Benedikt (2015a). "Combining Existential Rules and Description Logics". In: *Proc. IJCAI*. URL: http://arxiv.org/abs/1505.00326.
- Amarilli, Antoine, Michael Benedikt (2015b). "Finite Open-World Query Answering with Number Restrictions". In: *Proc. LICS*. URL: http://arxiv.org/abs/1505.04216.

References III

- Amarilli, Antoine, Pierre Bourhis, Pierre Senellart (2015). "Provenance Circuits for Trees and Treelike Instances". In: *Proc. ICALP*. URL: http://arxiv.org/abs/1511.08723.
- Amarilli, Antoine, Pierre Bourhis, Pierre Senellart (2016). "Tractable Lineages on Treelike Instances: Limits and Extensions". In: *Proc. PODS*. To appear. URL: https://a3nm.net/publications/amarilli2016tractable.pdf.
- Amarilli, Antoine, Silviu Maniu, Pierre Senellart (2015). "Intensional Data on the Web". In: *SIGWEB Newsletter*. URL: https://a3nm.net/publications/amarilli2015intensional.pdf.
- Amarilli, Antoine et al. (2014). "Recent Topics of Research around the YAGO Knowledge Base". In: *Proc. APWEB*. URL: https://zenodo.org/record/34912.

References IV

- Amarilli, Antoine et al. (2016). "Possible and Certain Answers for Queries over Order-Incomplete Data". Preprint: https://a3nm.net/publications/amarilli2016possible.pdf.
- Amarilli, Antoine et al. (2016). "Top-*k* Queries on Unknown Values under Order Constraints". Preprint: https://a3nm.net/publications/amarilli2016top.pdf.
- Chaudhuri, Surajit, Moshe Y. Vardi (1992). "On the Equivalence of Recursive and Nonrecursive Datalog Programs". In: *Proc. PODS*.
- Chekuri, Chandra, Julia Chuzhoy (2014). "Polynomial Bounds for the Grid-Minor Theorem". In: *Proc. STOC*.
- Cosmadakis, Stavros S., Paris C. Kanellakis, Moshe Y. Vardi (1990). "Polynomial-Time Implication Problems for Unary Inclusion Dependencies". In: *J. ACM*.

References V

- Courcelle, Bruno (1990). "The Monadic Second-Order Logic of Graphs. I. Recognizable Sets of Finite Graphs". In: *Inf. Comput.*
- Dalvi, Nilesh, Dan Suciu (2012). "The Dichotomy of Probabilistic Inference for Unions of Conjunctive Queries". In: *J. ACM.*
- Darwiche, Adnan (2001). "On the Tractable Counting of Theory Models and its Application to Truth Maintenance and Belief Revision". In: *J. Applied Non-Classical Logics.*
- Dong, Xin Luna, Alon Halevy, Cong Yu (2009). "Data integration with uncertainty". In: *The VLDB Journal—The International Journal on Very Large Data Bases.*
- Frick, Markus, Martin Grohe (2001). "Deciding first-order properties of locally tree-decomposable structures". In: *J. ACM.*
- Ganian, Robert et al. (2014). "Lower Bounds on the Complexity of MSO_1 Model-Checking". In: *JCSS.*

References VI

- Green, Todd J., Grigoris Karvounarakis, Val Tannen (2007). "Provenance Semirings". In: *Proc. PODS.*
- Lauritzen, Steffen L., David J. Spiegelhalter (1988). "Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems". In: *J. Royal Statistical Society. Series B.*
- Robertson, Neil, Paul D. Seymour (1986). "Graph minors. V. Excluding a Planar Graph". In: *J. Comb. Theory, Ser. B.*
- Talaika, Aliaksandr et al. (2015). "IBEX: Harvesting Entities from the Web Using Unique Identifiers". In: *Proc. WebDB.* URL: http://arxiv.org/abs/1505.00841.
- Tang, Ruiming et al. (2014a). "A Framework for Sampling-Based XML Data Pricing". In: *Transactions on Large-Scale Data and Knowledge-Centered Systems.* URL: https://a3nm.net/publications/tang2014framework.pdf.

References VII

- Tang, Ruiming et al. (2014b). "Get a Sample for a Discount". In: *Proc. DEXA*. URL: https://a3nm.net/publications/tang2014get.pdf.
- Thatcher, James W., Jesse B. Wright (1968). "Generalized Finite Automata Theory with an Application to a Decision Problem of Second-Order Logic". In: *Math. Systems Theory*.
- Zhang, Ce (2015). "DeepDive: A Data Management System for Automatic Knowledge Base Construction". https://cs.stanford.edu/people/czhang/zhang.thesis.pdf. PhD thesis. University of Winconsin–Madison.

Our main result on trees

Theorem

For any bottom-up (nondet) tree automaton A and input tree T, we can build a provenance circuit of A on T in linear time in A and T.

Our main result on trees

Theorem

For any bottom-up (nondet) tree automaton A and input tree T, we can build a provenance circuit of A on T in linear time in A and T.

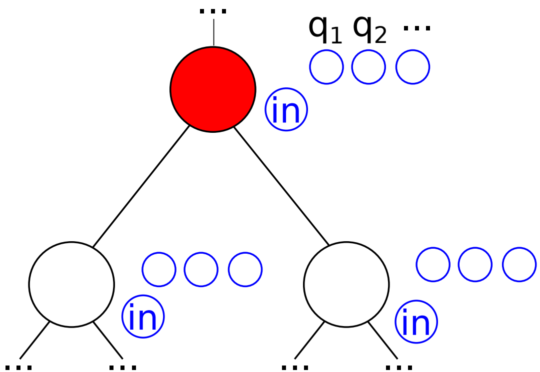
Construct the Boolean provenance circuit bottom-up

Our main result on trees

Theorem

For any bottom-up (nondet) tree automaton A and input tree T, we can build a provenance circuit of A on T in linear time in A and T.

Construct the Boolean provenance circuit bottom-up

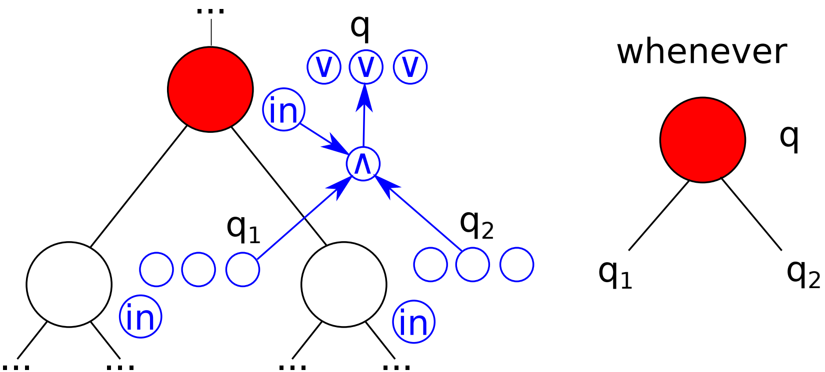


Our main result on trees

Theorem

For any bottom-up (nondet) tree automaton A and input tree T, we can build a provenance circuit of A on T in linear time in A and T.

Construct the Boolean provenance circuit bottom-up

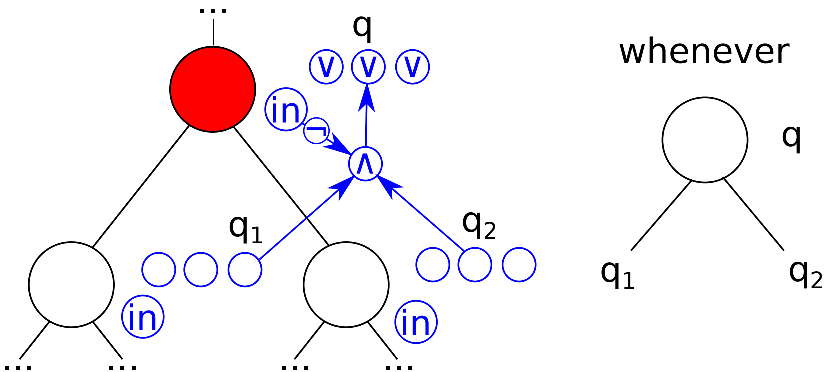


Our main result on trees

Theorem

For any bottom-up (nondet) tree automaton A and input tree T, we can build a provenance circuit of A on T in linear time in A and T.

Construct the Boolean provenance circuit bottom-up



Treelike instances

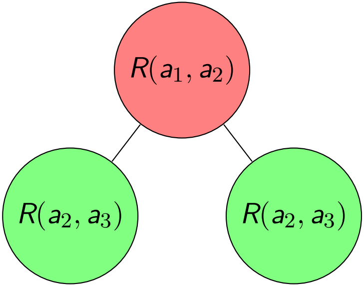
- Treelike instance I
- Tree encoding: tree E on fixed alphabet, represents I
- MSO query on I translates to
  $\rightarrow$ MSO query on E by Courcelle 1990
  $\rightarrow$ tree automaton on E by Thatcher, Wright 1968

Treelike instances

- Treelike instance I
- Tree encoding: tree E on fixed alphabet, represents I
- MSO query on I translates to
  → MSO query on E by Courcelle 1990
  → tree automaton on E by Thatcher, Wright 1968
- Uncertain instance: each fact can be present or absent

→ Possible subinstances are possible valuations of the encoding

R	
a	b
b	c
b	d

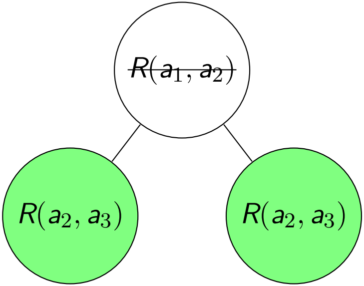


Treelike instances

- Treelike instance I
- Tree encoding: tree E on fixed alphabet, represents I
- MSO query on I translates to
  $\rightarrow$ MSO query on E by Courcelle 1990
  $\rightarrow$ tree automaton on E by Thatcher, Wright 1968
- Uncertain instance: each fact can be present or absent

$\rightarrow$ Possible subinstances are possible valuations of the encoding

R	
~~a~~	~~b~~
b	c
b	d

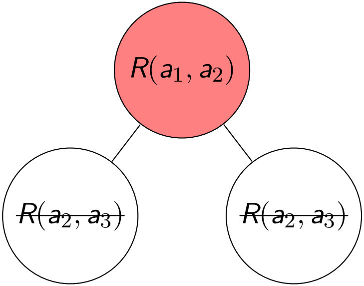


Treelike instances

- Treelike instance I
- Tree encoding: tree E on fixed alphabet, represents I
- MSO query on I translates to
  $\rightarrow$ MSO query on E by Courcelle 1990
  $\rightarrow$ tree automaton on E by Thatcher, Wright 1968
- Uncertain instance: each fact can be present or absent

$\rightarrow$ Possible subinstances are possible valuations of the encoding

R	
a	b
~~b~~	~~c~~
~~b~~	~~d~~

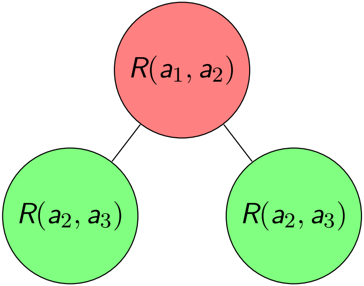


Treelike instances

- Treelike instance I
- Tree encoding: tree E on fixed alphabet, represents I
- MSO query on I translates to
  $\rightarrow$ MSO query on E by Courcelle 1990
  $\rightarrow$ tree automaton on E by Thatcher, Wright 1968
- Uncertain instance: each fact can be present or absent

$\rightarrow$ Possible subinstances are possible valuations of the encoding

R	
a	b
b	c
b	d



Our main result on treelike instances

Theorem

For any fixed MSO query q and $k \in \mathbb{N}$,
for any input instance I of treewidth $\leq k$,
we can build in linear time in I a provenance circuit of q on I.

Probability evaluation

Two alternate ways to see why probability evaluation is tractable on our provenance circuits:

Probability evaluation

Two alternate ways to see why probability evaluation is tractable on our provenance circuits:

- They have bounded treewidth themselves
  - Follows the structure of the tree encoding
  - Width only depends on number of automaton states

  → Apply message passing (Lauritzen, Spiegelhalter 1988)

Probability evaluation

Two alternate ways to see why probability evaluation is tractable on our provenance circuits:

- They have bounded treewidth themselves
  - Follows the structure of the tree encoding
  - Width only depends on number of automaton states
  - → Apply message passing (Lauritzen, Spiegelhalter 1988)
- If the tree automaton is deterministic
  - All conjunctions depend on disjoint sets of input gates
  - All disjunctions are on mutually exclusive outcomes
  - → Circuit is a d-DNNF (Darwiche 2001)

Probability evaluation

Two alternate ways to see why probability evaluation is tractable on our provenance circuits:

- They have bounded treewidth themselves
  - Follows the structure of the tree encoding
  - Width only depends on number of automaton states
  - → Apply message passing (Lauritzen, Spiegelhalter 1988)
- If the tree automaton is deterministic
  - All conjunctions depend on disjoint sets of input gates
  - All disjunctions are on mutually exclusive outcomes
  - → Circuit is a d-DNNF (Darwiche 2001)

Corollary

Probabilistic query evaluation of MSO queries on treelike instances is in linear time up to arithmetic operations.

Encoding treelike instances (Chaudhuri, Vardi 1992)

Instance:

N	
a	b
b	c
c	d
d	e
e	f

S	
a	c
b	e

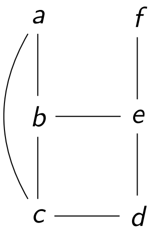
Encoding treelike instances (Chaudhuri, Vardi 1992)

Instance:

N	
a	*b*
b	*c*
c	*d*
d	*e*
e	*f*

S	
a	*c*
b	*e*

Gaifman graph:



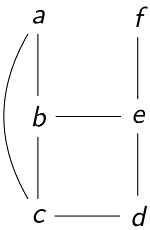
Encoding treelike instances (Chaudhuri, Vardi 1992)

Instance:

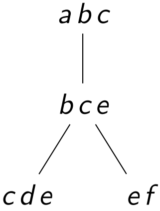
N	
a	*b*
b	*c*
c	*d*
d	*e*
e	*f*

S	
a	*c*
b	*e*

Gaifman graph:



Tree decomp.:



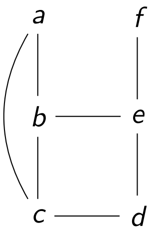
Encoding treelike instances (Chaudhuri, Vardi 1992)

Instance:

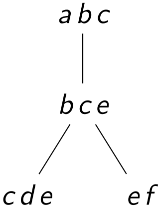
N	
a	b
b	c
c	d
d	e
e	f

S	
a	c
b	e

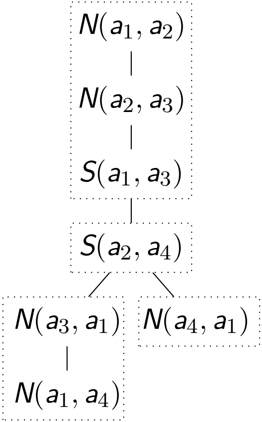
Gaifman graph:



Tree decomp.:



Tree encoding:



Provenance semirings

- Semiring of positive Boolean functions $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$

Provenance semirings

- Semiring of positive Boolean functions $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$
- Provenance semirings: (Green, Karvounarakis, Tannen 2007)
  - Provenance generalized to arbitrary (commutative) semirings
  - For queries in the positive relational algebra and Datalog

Provenance semirings

- Semiring of positive Boolean functions $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$
- Provenance semirings: (Green, Karvounarakis, Tannen 2007)
  - Provenance generalized to arbitrary (commutative) semirings
  - For queries in the positive relational algebra and Datalog

$\rightarrow$ Our circuits capture $\mathrm{PosBool}[X]$-provenance in this sense

Provenance semirings

- Semiring of positive Boolean functions $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$
- Provenance semirings: (Green, Karvounarakis, Tannen 2007)
  - Provenance generalized to arbitrary (commutative) semirings
  - For queries in the positive relational algebra and Datalog

→ Our circuits capture $\mathrm{PosBool}[X]$-provenance in this sense
  - The definitions match: all subinstances that satisfy the query

Provenance semirings

- Semiring of positive Boolean functions $(\mathrm{PosBool}[X], \vee, \wedge, \mathfrak{f}, \mathfrak{t})$
- Provenance semirings: (Green, Karvounarakis, Tannen 2007)
  - Provenance generalized to arbitrary (commutative) semirings
  - For queries in the positive relational algebra and Datalog

$\rightarrow$ Our circuits capture $\mathrm{PosBool}[X]$-provenance in this sense

  - The definitions match: all subinstances that satisfy the query
  - For monotone queries, we can construct positive circuits

Universal provenance

- Universal semiring of polynomials $(\mathbb{N}[X], +, \times, 0, 1)$

  $\rightarrow$ The provenance for $\mathbb{N}[X]$ can be specialized to any $K[X]$

Universal provenance

- Universal semiring of polynomials $(\mathbb{N}[X], +, \times, 0, 1)$
  $\rightarrow$ The provenance for $\mathbb{N}[X]$ can be specialized to any $K[X]$
- Captures many useful semirings:
  - counting the number of matches of a query
  - computing the security level of a query result
  - computing the cost of a query result

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\; R(x, y) \wedge R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$\rightarrow$ $\mathbb{N}[X]$-provenance:

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\; R(x, y) \wedge R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$\rightarrow$ $\mathbb{N}[X]$-provenance:

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\ R(x, y) \wedge R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$$(x_1 \wedge x_2)$$

$\rightarrow$ $\mathbb{N}[X]$-provenance:

$$(x_1 \times x_2)$$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\; R(x, y) \wedge R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$(x_1 \wedge x_2)$

$\rightarrow$ $\mathbb{N}[X]$-provenance:

$(x_1 \times x_2)$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\; R(x, y) \wedge R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$$(x_1 \wedge x_2) \vee (x_3 \wedge x_4)$$

$\rightarrow$ $\mathbb{N}[X]$-provenance:

$$(x_1 \times x_2) + (x_3 \times x_4)$$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\ R(x, y) \wedge R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$$(x_1 \wedge x_2) \vee (x_3 \wedge x_4)$$

$\rightarrow$ $\mathbb{N}[X]$-provenance:

$$(x_1 \times x_2) + (x_3 \times x_4)$$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\; R(x, y) \land R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$$(x_1 \land x_2) \lor (x_3 \land x_4)$$

$\rightarrow$ $\mathbb{N}[X]$-provenance:

$$(x_1 \times x_2) + (x_3 \times x_4) + (x_4 \times x_3)$$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\; R(x, y) \wedge R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$$(x_1 \wedge x_2) \vee (x_3 \wedge x_4)$$

$\rightarrow$ $\mathbb{N}[X]$-provenance:

$$(x_1 \times x_2) + (x_3 \times x_4) + (x_4 \times x_3)$$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\; R(x, y) \wedge R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$$(x_1 \wedge x_2) \vee (x_3 \wedge x_4) \quad \vee \quad x_5$$

$\rightarrow$ $\mathbb{N}[X]$-provenance:

$$(x_1 \times x_2) + (x_3 \times x_4) + (x_4 \times x_3) + (x_5 \times x_5)$$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\; R(x, y) \wedge R(y, z)$$

$\rightarrow$ $\mathrm{PosBool}[X]$-provenance:

$$(x_1 \wedge x_2) \vee (x_3 \wedge x_4) \qquad \vee \quad x_5$$

$\rightarrow$ $\mathbb{N}[X]$-provenance:

$$(x_1 \times x_2) + (x_3 \times x_4) + (x_4 \times x_3) + (x_5 \times x_5)$$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

R		
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\,y\,z\; R(x,y) \wedge R(y,z)$$

→ $\mathrm{PosBool}[X]$-provenance:

$$(x_1 \wedge x_2) \vee (x_3 \wedge x_4) \qquad \vee \quad x_5$$

→ $\mathbb{N}[X]$-provenance:

$$(x_1 \times x_2) + (x_3 \times x_4) + (x_4 \times x_3) + (x_5 \times x_5)$$
$$= x_1 x_2 + 2 x_3 x_4 + x_5^2$$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

$\mathbb{N}[X]$-provenance example

	R	
a	b	x_1
b	c	x_2
d	e	x_3
e	d	x_4
f	f	x_5

$$\exists x\, y\, z\ R(x, y) \land R(y, z)$$

→ $\text{PosBool}[X]$-provenance:

$$(x_1 \land x_2) \lor (x_3 \land x_4) \qquad \lor \quad x_5$$

→ $\mathbb{N}[X]$-provenance:

$$(x_1 \times x_2) + (x_3 \times x_4) + (x_4 \times x_3) + (x_5 \times x_5)$$
$$= x_1 x_2 + 2 x_3 x_4 + x_5^2$$

- Definition of provenance for conjunctive queries:
  - Sum over query matches
  - Multiply over matched facts

How is $\mathbb{N}[X]$ more expressive than $\text{PosBool}[X]$?

→ Coefficients: counting multiple matches

→ Exponents: using facts multiple times

Capturing $\mathbb{N}[X]$-provenance

Our construction can be extended to $\mathbb{N}[X]$-provenance
for conjunctive queries and unions of conjunctive queries (UCQ):

Capturing $\mathbb{N}[X]$-provenance

Our construction can be extended to $\mathbb{N}[X]$-provenance for conjunctive queries and unions of conjunctive queries (UCQ):

Theorem

For any fixed UCQ q and $k \in \mathbb{N}$, for any input instance I of treewidth $\leq k$, we can build in linear time a $\mathbb{N}[X]$-provenance circuit of q on I.

Capturing $\mathbb{N}[X]$-provenance

Our construction can be extended to $\mathbb{N}[X]$-provenance
for conjunctive queries and unions of conjunctive queries (UCQ):

Theorem

For any fixed UCQ q and $k \in \mathbb{N}$,
for any input instance I of treewidth $\leq k$,
we can build in linear time a $\mathbb{N}[X]$-provenance circuit of q on I.

→ What fails for MSO and Datalog?

- Unbounded maximal multiplicity of fact uses

Correlations

- Our probabilistic instances assume independence on all facts
  $\rightarrow$ Not very expressive!

Correlations

- Our probabilistic instances assume independence on all facts
  → Not very expressive!

More expressive formalism: Block-Independent Disjoint instances:

name	city	iso	p
pods	san francisco	us	0.8
pods	los angeles	us	0.2
icalp	rome	it	0.1
icalp	florence	it	0.9

pc-tables

More generally, pc-tables to represent arbitrary correlations

pc-tables

More generally, pc-tables to represent arbitrary correlations

date	teacher	room	
04	John	C42	$\neg x_1$
04	Jane	C42	x_1
11	John	C017	$x_2 \wedge \neg x_1$
11	Jane	C017	$x_2 \wedge x_1$
11	John	C47	$\neg x_2 \wedge \neg x_1$
11	Jane	C47	$\neg x_2 \wedge x_1$

pc-tables

More generally, pc-tables to represent arbitrary correlations

date	teacher	room	
04	John	C42	$\neg x_1$
04	Jane	C42	x_1
11	John	C017	$x_2 \wedge \neg x_1$
11	Jane	C017	$x_2 \wedge x_1$
11	John	C47	$\neg x_2 \wedge \neg x_1$
11	Jane	C47	$\neg x_2 \wedge x_1$

x_1 John gets sick

$\rightarrow$ Probability 0.1

x_2 Room C017 is available

$\rightarrow$ Probability 0.2

Our results

Probabilistic query evaluation on instances with correlations is tractable if the instance and correlations are bounded-tw:

Our results

Probabilistic query evaluation on instances with correlations is tractable if the instance and correlations are bounded-tw:

Theorem

Probabilistic query evaluation of MSO queries on treelike BID is in linear time up to arithmetic operations.

Our results

Probabilistic query evaluation on instances with correlations
is tractable if the instance and correlations are bounded-tw:

Theorem

Probabilistic query evaluation of MSO queries on treelike BID is in linear time up to arithmetic operations.

“Tree-like” just means the underlying instance (easy correlations)

Our results

Probabilistic query evaluation on instances with correlations is tractable if the instance and correlations are bounded-tw:

Theorem

Probabilistic query evaluation of MSO queries on treelike BID is in linear time up to arithmetic operations.

"Tree-like" just means the underlying instance (easy correlations)

Theorem

Probabilistic query evaluation of MSO queries on treelike pc-tables is in linear time up to arithmetic operations.

Our results

Probabilistic query evaluation on instances with correlations
is tractable if the instance and correlations are bounded-tw:

Theorem

Probabilistic query evaluation of MSO queries on treelike BID is in linear time up to arithmetic operations.

"Tree-like" just means the underlying instance (easy correlations)

Theorem

Probabilistic query evaluation of MSO queries on treelike pc-tables is in linear time up to arithmetic operations.

"Tree-like" refers to the underlying instance, adding facts
to represent variable occurrences and co-occurrences

Idea: extracting topological minors

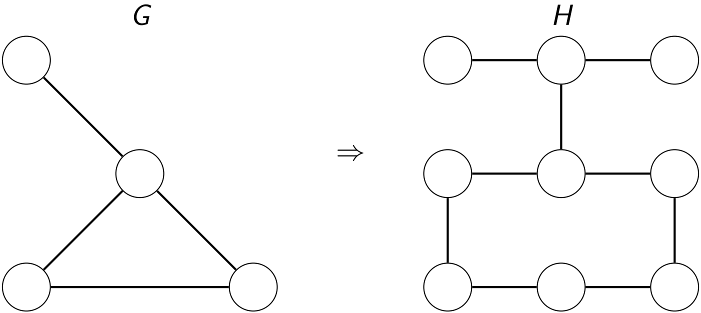
- Let G be a planar graph of degree ≤ 3

Idea: extracting topological minors

- Let G be a planar graph of degree ≤ 3
- G is a topological minor of H if:

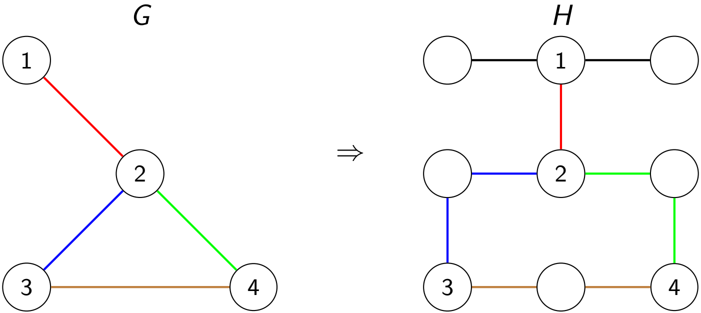
Idea: extracting topological minors

- Let G be a planar graph of degree ≤ 3
- G is a topological minor of H if:



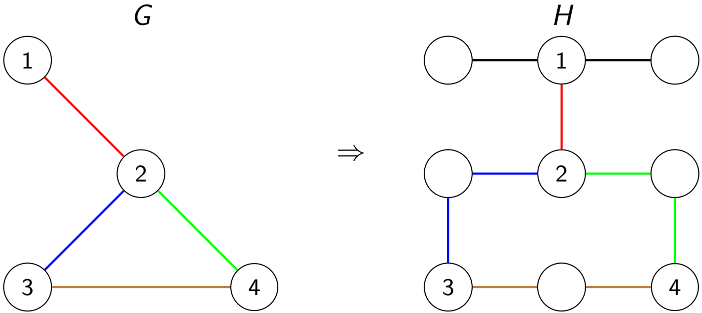
Idea: extracting topological minors

- Let G be a planar graph of degree ≤ 3
- G is a topological minor of H if:



Idea: extracting topological minors

- Let G be a planar graph of degree ≤ 3
- G is a topological minor of H if:



- Map vertices to vertices
- Map edges to vertex-disjoint paths

Topological minor extraction results

Theorem ((Robertson, Seymour 1986))

For any planar graph G of degree ≤ 3,
for any graph H of sufficiently high treewidth,
G is a topological minor of H.

Topological minor extraction results

Theorem ((Robertson, Seymour 1986))

For any planar graph G of degree ≤ 3,
for any graph H of sufficiently high treewidth,
G is a topological minor of H.

More recently:

Theorem ((Chekuri, Chuzhoy 2014))

There is a certain constant $c \in \mathbb{N}$ such that
for any planar graph G of degree ≤ 3,
for any graph H of treewidth $\geq |G|^c$,
G is a topological minor of H and
we can embed G in H (with high probability) in PTIME in $|H|$.

Intuition for our result: reduction

- Choose a problem from which to reduce:
  - Must be #P-hard on planar degree-3 graphs
  - Must be encodable to an FO query q (more later)
- $\rightarrow$ We use the problem of counting matchings

Intuition for our result: reduction

- Choose a problem from which to reduce:
  - Must be #P-hard on planar degree-3 graphs
  - Must be encodable to an FO query q (more later)

  → We use the problem of counting matchings
- Given an input graph G, compute $k := |G|^c$

Intuition for our result: reduction

- Choose a problem from which to reduce:
  - Must be #P-hard on planar degree-3 graphs
  - Must be encodable to an FO query q (more later)

  $\rightarrow$ We use the problem of counting matchings
- Given an input graph G, compute $k := |G|^c$
- Compute in PTIME an instance I of $\mathcal{I}$ of treewidth $\geq k$

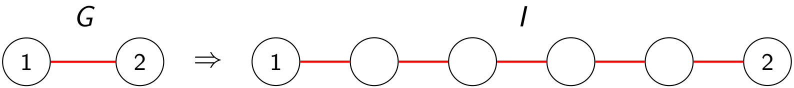
Intuition for our result: reduction

- Choose a problem from which to reduce:
  - Must be #P-hard on planar degree-3 graphs
  - Must be encodable to an FO query q (more later)
  - $\rightarrow$ We use the problem of counting matchings
- Given an input graph G, compute $k := |G|^c$
- Compute in PTIME an instance I of $\mathcal{I}$ of treewidth $\geq k$
- Compute in randomized PTIME an embedding of G in I

Intuition for our result: reduction

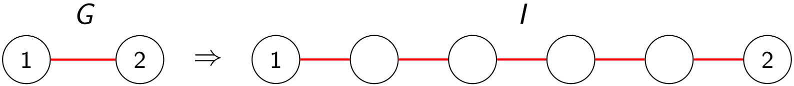
- Choose a problem from which to reduce:
  - Must be #P-hard on planar degree-3 graphs
  - Must be encodable to an FO query q (more later)
  - $\rightarrow$ We use the problem of counting matchings
- Given an input graph G, compute $k := |G|^c$
- Compute in PTIME an instance I of $\mathcal{I}$ of treewidth $\geq k$
- Compute in randomized PTIME an embedding of G in I
- Construct a probability valuation π of I such that:
  - Unneccessary edges of I are removed
  - Probability eval for q gives the answer to the hard problem

Technical issue



- In the embedding, edges of G can become long paths in I
- q must answer the hard problem on G despite subdivisions

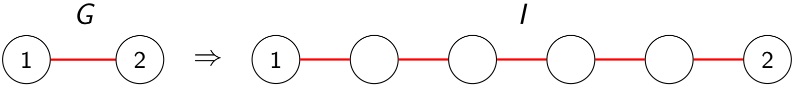
Technical issue



- In the embedding, edges of G can become long paths in I
- q must answer the hard problem on G despite subdivisions

→ Our q restricts to a subset of the worlds of known weight and gives the right answer up to renormalization

Technical issue



- In the embedding, edges of G can become long paths in I
- q must answer the hard problem on G despite subdivisions

→ Our q restricts to a subset of the worlds of known weight and gives the right answer up to renormalization

→ For non-probabilistic evaluation, using FO does not work (Frick, Grohe 2001)

→ Lower bounds for non-probabilistic evaluation are for MSO (Ganian et al. 2014)

Can we do better?

- We can use a non-monotone FO or a monotone MSO query
- Can we use a weaker query language? (e.g., monotone FO)

Can we do better?

- We can use a non-monotone FO or a monotone MSO query
- Can we use a weaker query language? (e.g., monotone FO)

$\rightarrow$ We cannot use a connected CQ even with inequalities

$\rightarrow$ We cannot use a query closed under homomorphisms

Can we do better?

- We can use a non-monotone FO or a monotone MSO query
- Can we use a weaker query language? (e.g., monotone FO)

→ We cannot use a connected CQ even with inequalities

→ We cannot use a query closed under homomorphisms

- A good candidate query:

$$q : (E(x,y) \vee E(y,x)) \wedge (E(y,z) \wedge E(z,y)) \wedge x \neq z$$

Can we do better?

- We can use a non-monotone FO or a monotone MSO query
- Can we use a weaker query language? (e.g., monotone FO)

→ We cannot use a connected CQ even with inequalities

→ We cannot use a query closed under homomorphisms

- A good candidate query:

$$q : (E(x, y) \vee E(y, x)) \wedge (E(y, z) \wedge E(z, y)) \wedge x \neq z$$

→ This UCQ with inequalities is hard in a weaker sense
(no polynomial-size OBDD representations of provenance)

→ We don't know whether it's #P-hard (because of subdivisions)