A Collective, Probabilistic Approach to Schema Mapping

Angelika Kimmig, Alex Memory, Renée Miller, Lise Getoor

ILP 2017 (published at ICDE 2017)

Context: Data Exchange & Data Integration

source

emp

id	name	company
1	Alice	SAP
2	Bob	IBM
3	Pat	MS
4	Carl	X
5	Tom	Y7

proj

topic	mgr	lead
BigData	1	2
ML	1	1
eGov	4	5
DM	5	5

target

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111
MT	Justin	444
Deep	Ann	333

leader

name
Alice
Bob
Jim
Ann
Igor

org

oid	name
111	SAP
222	MS
333	Z
444	HC

Context: Data Exchange & Data Integration

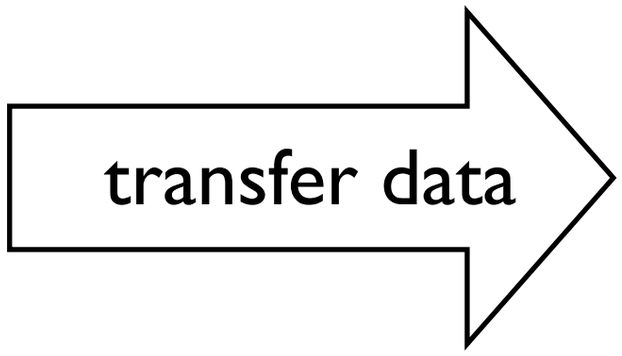
source

emp

id	name	company
1	Alice	SAP
2	Bob	IBM
3	Pat	MS
4	Carl	X
5	Tom	Y7

proj

topic	mgr	lead
BigData	1	2
ML	1	1
eGov	4	5
DM	5	5



target

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111
MT	Justin	444
Deep	Ann	333

leader

name
Alice
Bob
Jim
Ann
Igor

org

oid	name
111	SAP
222	MS
333	Z
444	HC

Context: Data Exchange & Data Integration

source

emp

id	name	company
1	Alice	SAP
2	Bob	IBM
3	Pat	MS
4	Carl	X
5	Tom	Y7

proj

topic	mgr	lead
BigData	1	2
ML	1	1
eGov	4	5
DM	5	5

transfer data

rewrite queries

target

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111
MT	Justin	444
Deep	Ann	333

leader

name
Alice
Bob
Jim
Ann
Igor

org

oid	name
111	SAP
222	MS
333	Z
444	HC

Context: Data Exchange & Data Integration

source

emp

id	name	company
1	Alice	SAP
2	Bob	IBM
3	Pat	MS
4	Carl	X
5	Tom	Y7

proj

topic	mgr	lead
BigData	1	2
ML	1	1
eGov	4	5
DM	5	5

transfer data

need **schema mapping**

rewrite queries

target

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111
MT	Justin	444
Deep	Ann	333

leader

name
Alice
Bob
Jim
Ann
Igor

org

oid	name
111	SAP
222	MS
333	Z
444	HC

Context: Data Exchange & Data Integration

source

target

emp

id	name	company
1	Alice	SAP
2		
3		
4		
5		

transfer data

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111
		4
		3

$$emp(I,N,C) \rightarrow leader(N)$$

$$proj(T,M,L) \rightarrow \exists\, S.\, \exists\, O.\, task(T,S,O)$$

$$proj(T,M,L)\ \&\ emp(L,N,C) \rightarrow \exists\, O.\, task(T,N,O)\ \&\ org(O,C)$$

pro

topic	mgr	lead
BigData	1	2
ML	1	1
eGov	4	5
DM	5	5

rewrite queries

Bob
Jim
Ann
Igor

	ame
	SAP
222	MS
333	Z
444	HC

Schema Mapping

- **st tgd** = source-target tuple generating dependency

[Fagin et al, 05; ten Cate & Kolaitis, 10]

- first order rule

$$\forall \mathbf{x}\ \phi(\mathbf{x}) \rightarrow \exists \mathbf{y}\ \psi(\mathbf{x}, \mathbf{y})$$

conjunctive query on source schema

conjunctive query on target schema

- a **schema mapping** is a set of st tgds

Goal: learn schema mapping from data

proj

topic	mgr	lead
BigData	1	2
ML	1	1

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111

emp

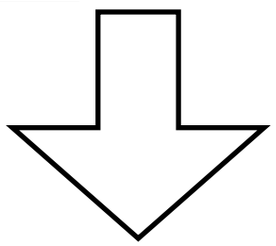
id	name	company
1	Alice	SAP
2	Bob	IBM
3	Pat	MS

leader

name
Alice
Bob

org

oid	name
111	SAP
222	MS



$proj(T,M,L) \;\&\; emp(L,N,C) \rightarrow leader(N)$

$emp(I,N,C) \rightarrow \exists\, O.\; org(O,C)$

$proj(T,M,L) \;\&\; emp(M,N,C) \rightarrow \exists\, O.\; task(T,N,O) \;\&\; org(O,C)$

Goal: learn schema mapping from data

proj

topic	mgr	lead
BigData	1	2
ML	1	1

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111

emp

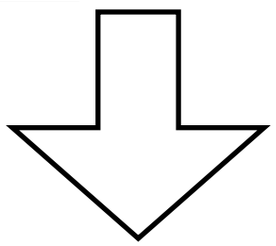
id	name	company
1	Alice	SAP
2	Bob	IBM
3	Pat	MS

leader

name
Alice
Bob

org

oid	name
111	SAP
222	MS



proj(T,M,L) & emp(L,N,C) → leader(N)
emp(I,N,C) → ∃ O. org(O,C)
proj(T,M,L) & emp(M,N,C) → ∃ O. task(T,N,O) & org(O,C)

Challenges:

- ambiguous metadata
- imperfect data
- existentials / nulls

Here:

- **Given** data example (I,J), candidate set C
- **Find** optimal M⊆C for (I,J)

I

proj

topic	mgr	lead
BigData	1	2
ML	1	1

emp

id	name	company
1	Alice	SAP
2	Bob	IBM
3	Pat	MS

J

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111

leader

name
Alice
Bob

org

oid	name
111	SAP
222	MS

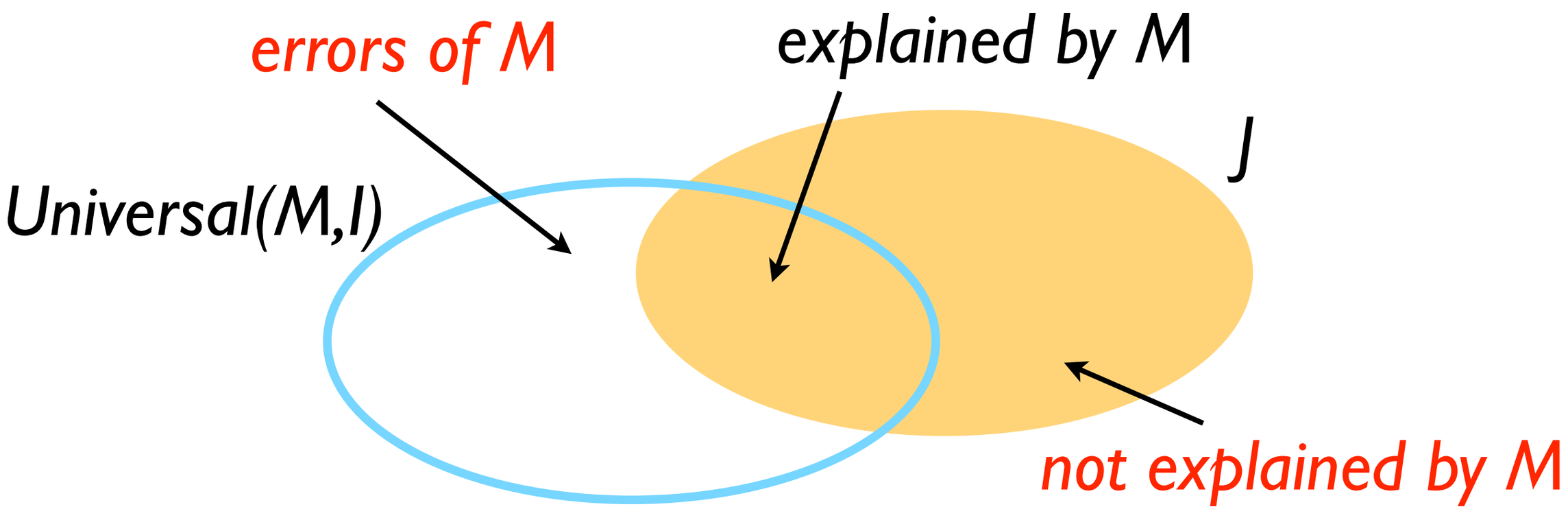
C

emp(I,N,C) → leader(N)
proj(T,M,L) → ∃ S. ∃ O. task(T,S,O)
proj(T,M,L) & emp(L,N,C) → leader(N)
proj(T,M,L) & emp(M,N,C)
→ ∃ O. task(T,N,O) & org(O,C)
proj(T,M,L) & emp(L,N,C)
→ ∃ O. task(T,N,O) & org(O,C)
emp(I,N,C) → ∃ O. org(O,C)
emp(I,N,C) → ∃ O. org(O,N)
…

M

proj(T,M,L) & emp(L,N,C) → leader(N)
emp(I,N,C) → ∃ O. org(O,C)
proj(T,M,L) & emp(M,N,C) → ∃ O. task(T,N,O) & org(O,C)

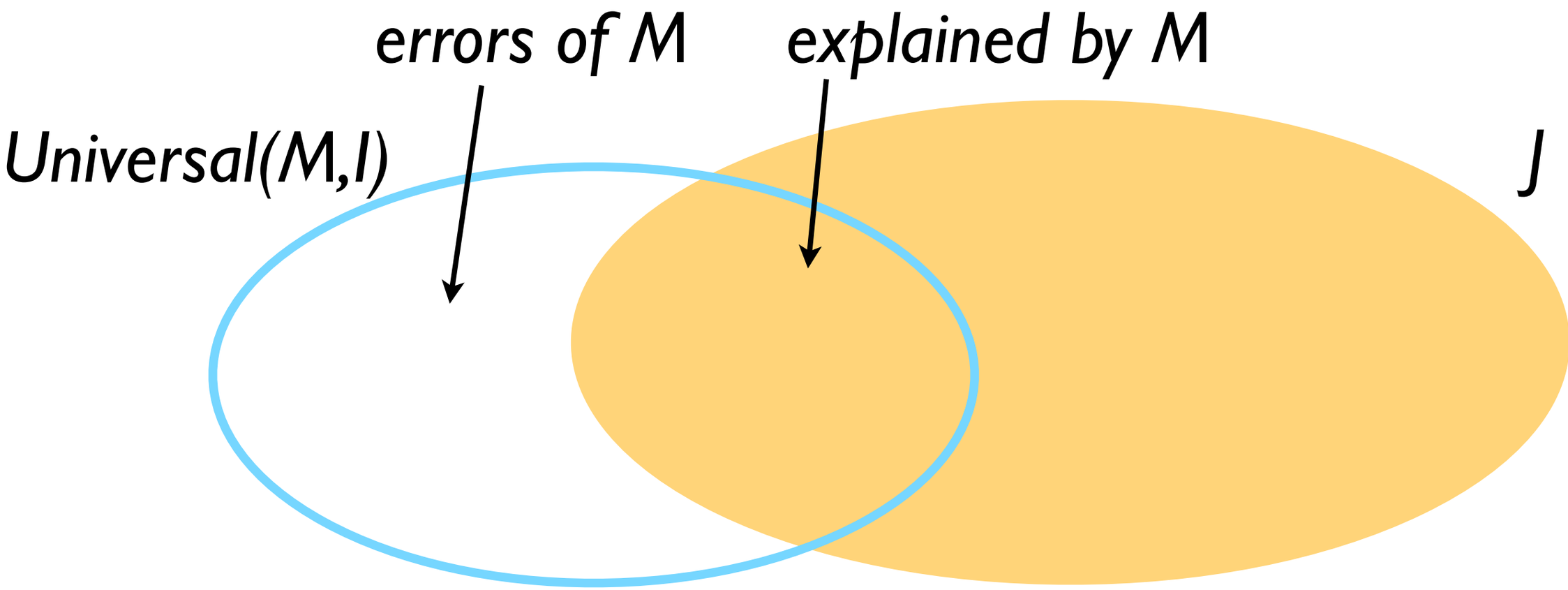
Full st tgds (no ∃)



goal: find small M that maximizes intersection

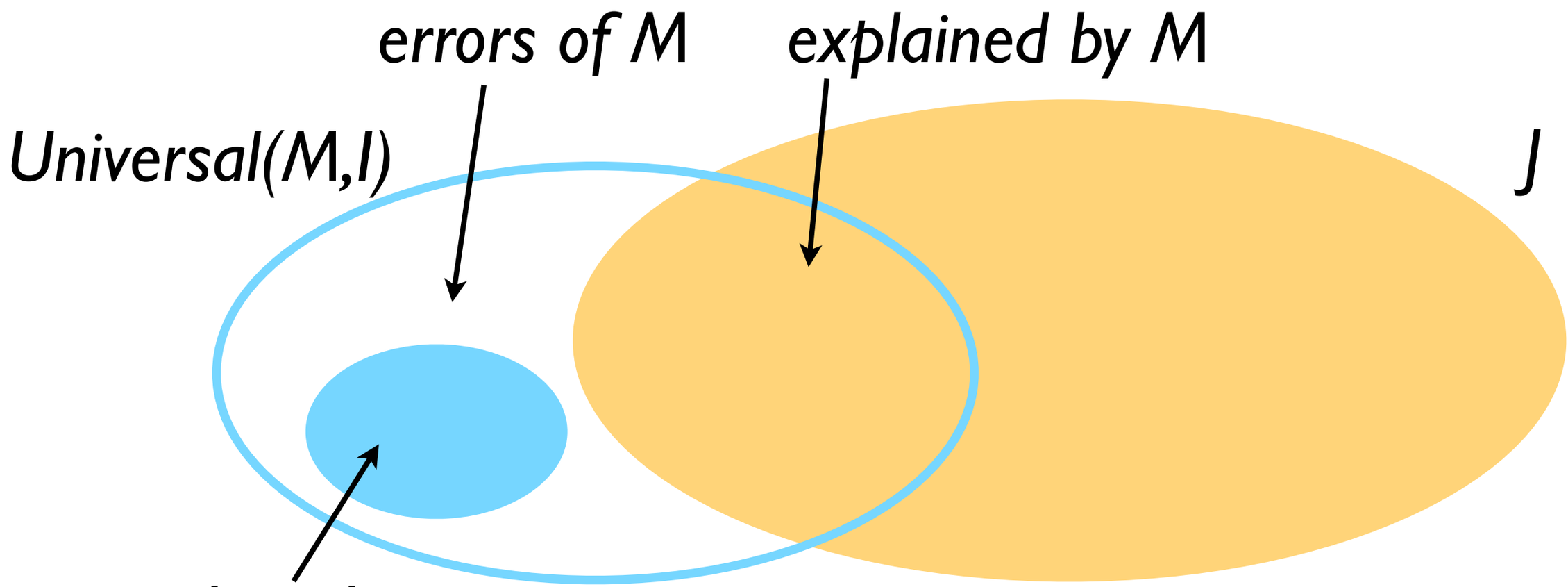
Arbitrary st tgds:

replace containment with homomorphism checks



Arbitrary st tgds:

replace containment with homomorphism checks

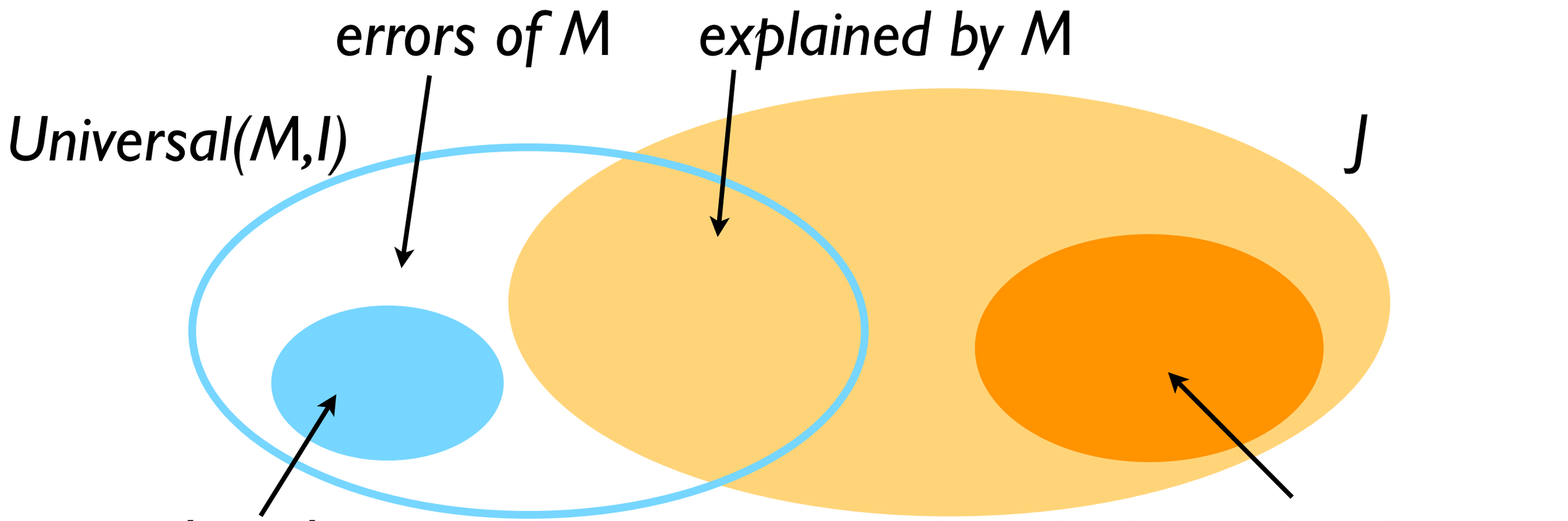


tuples where we can replace nulls by constants such that we get tuples in J

= *no errors*

Arbitrary st tgds:

replace containment with homomorphism checks



tuples where we can replace nulls by constants such that we get tuples in J

= *no errors*

tuples which can be obtained from tuples in Universal(M,I) replacing nulls by constants

= *(partially) explained*

Example

task

title	supervisor	oid
BigData	Bob	null1
ML	Alice	null2

leader

name

org

oid	name
null1	IBM
null2	SAP

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111

leader

name
Alice
Bob

org

oid	name
111	SAP
222	MS

Example

task

title	supervisor	oid
BigData	Bob	null1
ML	Alice	null2

leader

name

org

oid	name
null1	IBM
null2	SAP

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111

leader

name
Alice
Bob

org

oid	name
111	SAP
222	MS

no errors: replace null2 by 111...

Example

task

title	supervisor	oid
BigData	Bob	null1
ML	Alice	null2

leader

name

org

oid	name
null1	IBM
null2	SAP

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111

leader

name
Alice
Bob

org

oid	name
111	SAP
222	MS

no errors: replace null2 by 111... ... to explain these

Example

errors: can't replace null1 to get tuples in J

task

title	supervisor	oid
BigData	Bob	null1
ML	Alice	null2

leader

name

org

oid	name
null1	IBM
null2	SAP

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111

leader

name
Alice
Bob

org

oid	name
111	SAP
222	MS

no errors: replace null2 by 111... ... to explain these

Example

errors: can't replace null1 to get tuples in J

not explained by any tuple in *Universal(M,I)*

task

title	supervisor	oid
BigData	Bob	null1
ML	Alice	null2

leader

name

org

oid	name
null1	IBM
null2	SAP

task

title	supervisor	oid
BigData	Alice	111
ML	Alice	111

leader

name
Alice
Bob

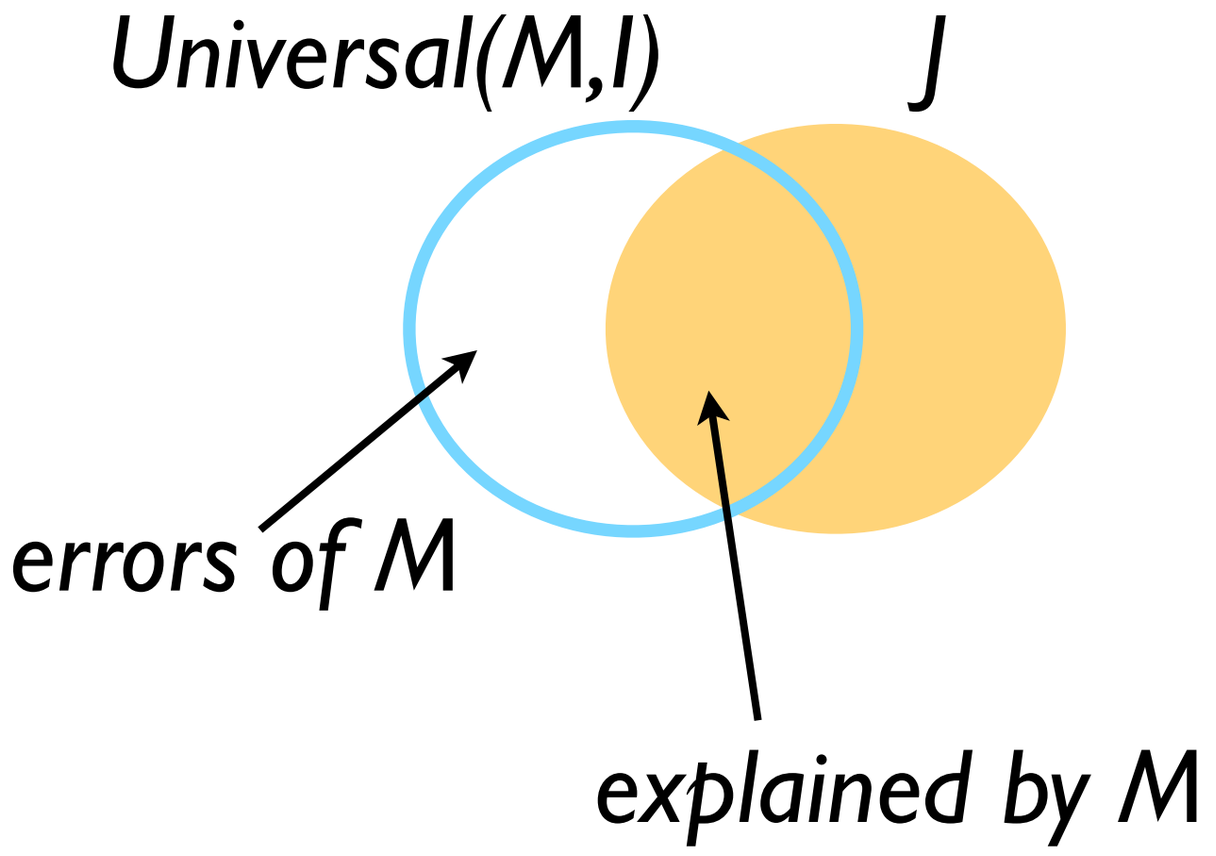
org

oid	name
111	SAP
222	MS

no errors: replace null2 by 111... ... to explain these

Task

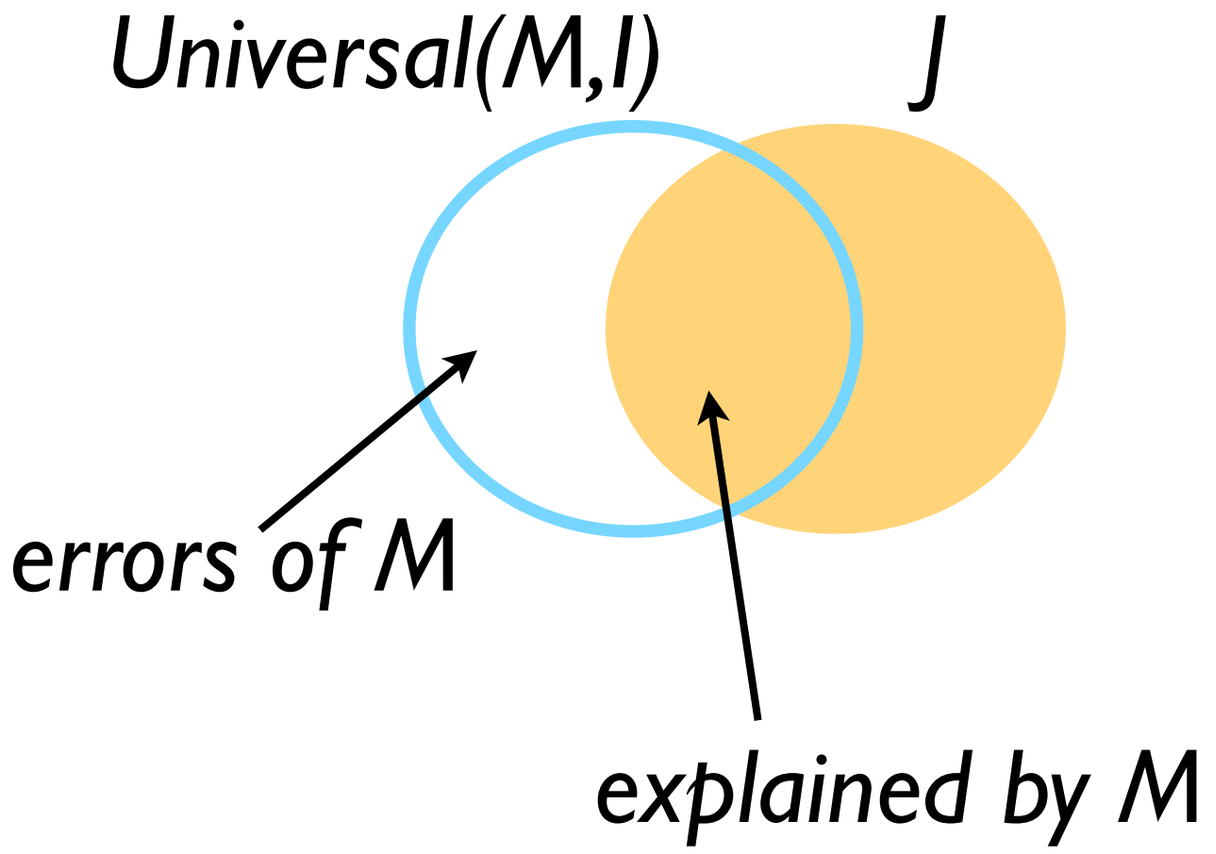
- **Given**
  - source schema S, target schema T
  - data example (I,J)
    - set C of candidate st tgds
- **Find** an optimal mapping M, i.e.,



$$\arg\min_{M \subseteq C} \left[\text{size}(M) + \sum_{t \in J} (1 - \text{explains}(M, t)) + \sum_{t \in \text{Universal}(\mathbf{C},\mathbf{I}) - J} \text{error}(M, t) \right]$$

Task

- **Given**
  - source schema S, target schema T
  - data example (I,J)
  - set C of candidate st tgds
- **Find** an optimal mapping M, i.e.,



NP-hard even for full st tgds

$$\arg\min_{M \subseteq C} \left[\text{size}(M) + \sum_{t \in J} (1 - \text{explains}(M, t)) + \sum_{t \in \text{Universal}(\mathbf{C},\mathbf{I}) - J} \text{error}(M, t) \right]$$

Probabilistic Soft Logic (PSL)

- declarative **language** to specify probabilistic models over logical atoms / relational domains
- PSL program = set of **weighted first order rules**

$$w: \quad b_1(\vec{X}) \wedge \ldots \wedge b_n(\vec{X}) \rightarrow h_1(\vec{X}) \vee \ldots \vee h_m(\vec{X})$$

- MPE **inference** = finding most likely model
- **efficient** approximate solver with **guarantee** on solution quality

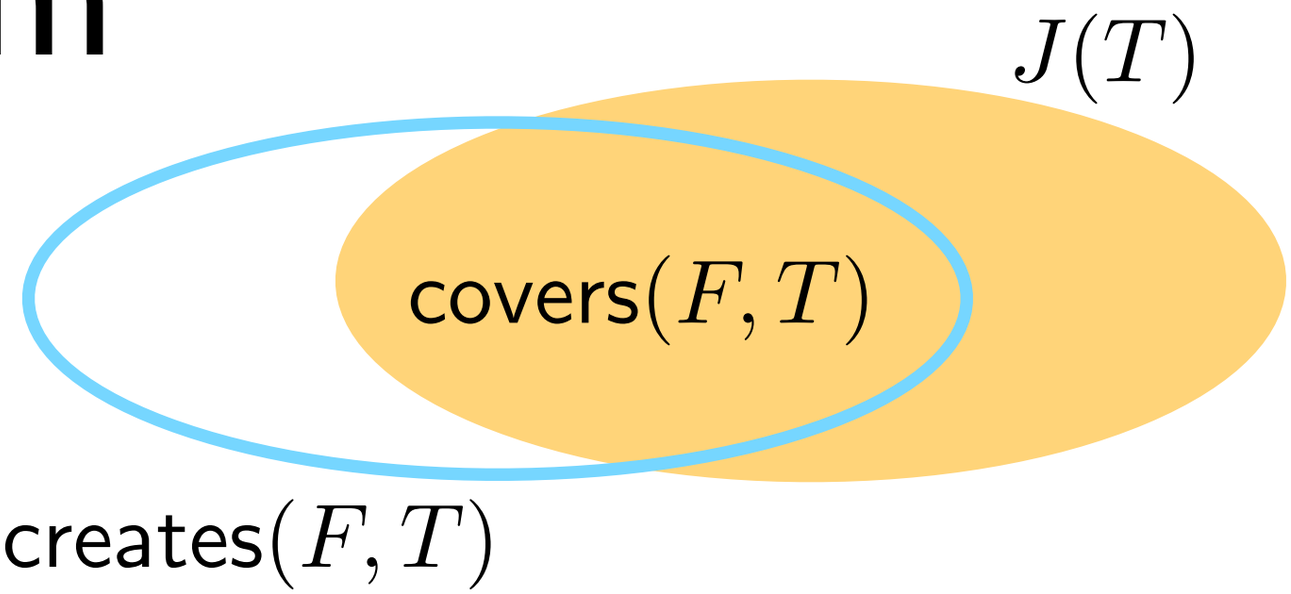
http://psl.linqs.org

Our PSL program

given $J(T)$

covers(F, T)

creates(F, T)

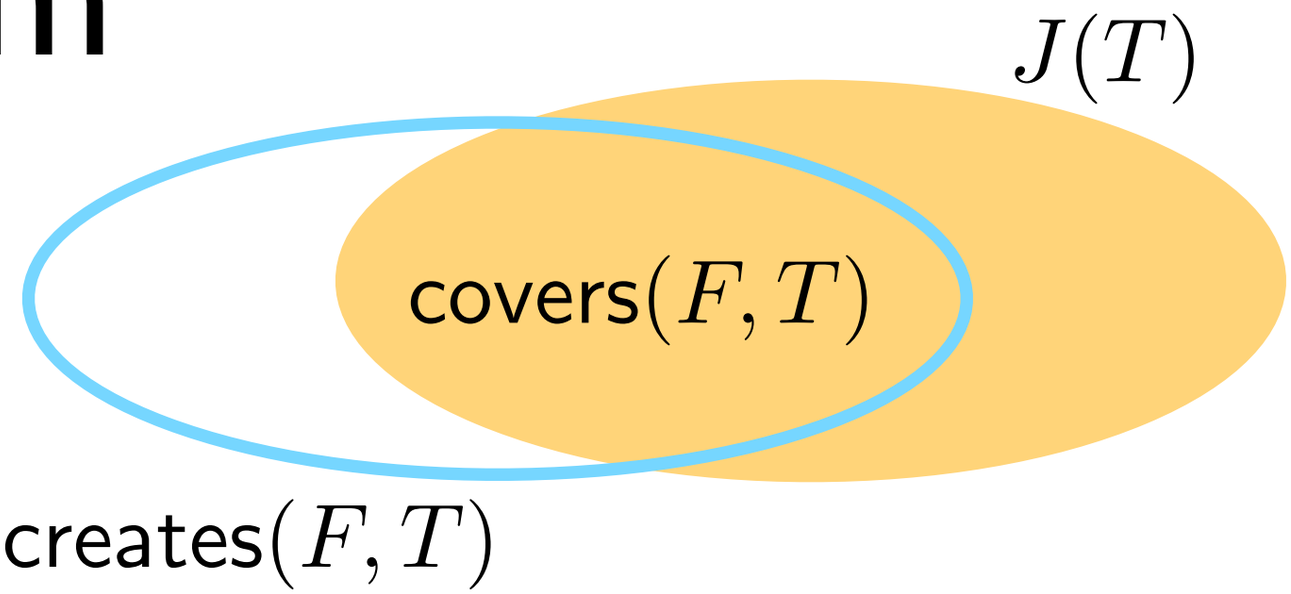


Our PSL program

given $J(T)$

covers(F, T)

creates(F, T)



find optimal M:

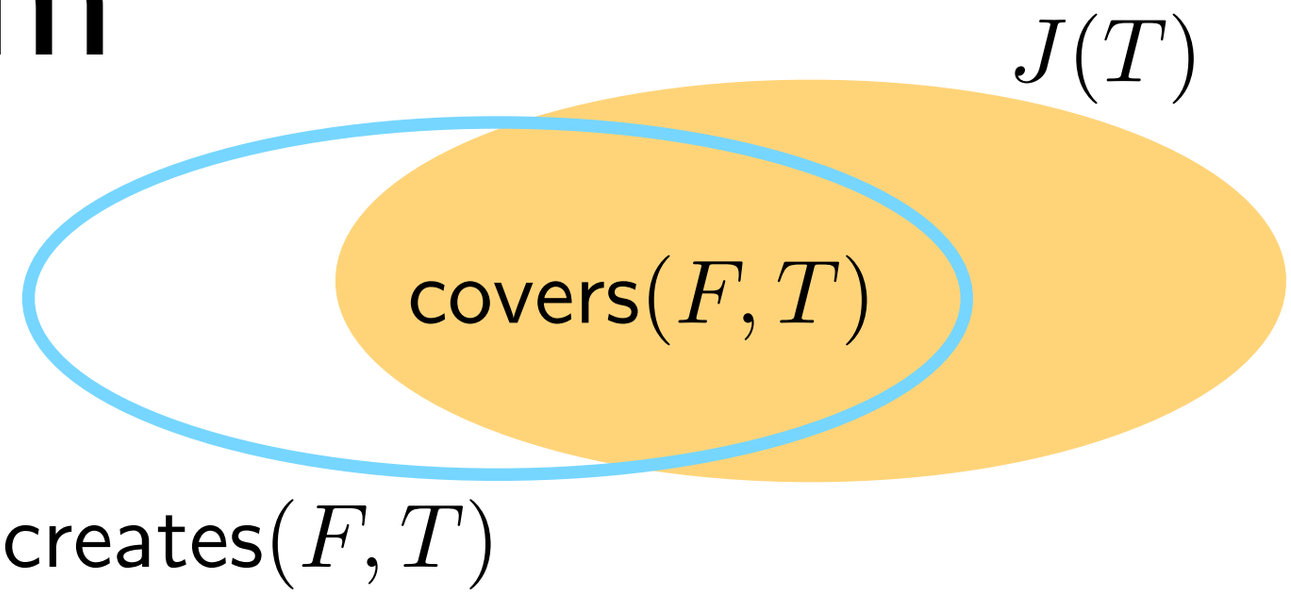
$$in(F) \Leftrightarrow F \in M$$

Our PSL program

given $J(T)$

covers(F, T)

creates(F, T)



find optimal M:

$$in(F) \Leftrightarrow F \in M$$

minimize #errors:

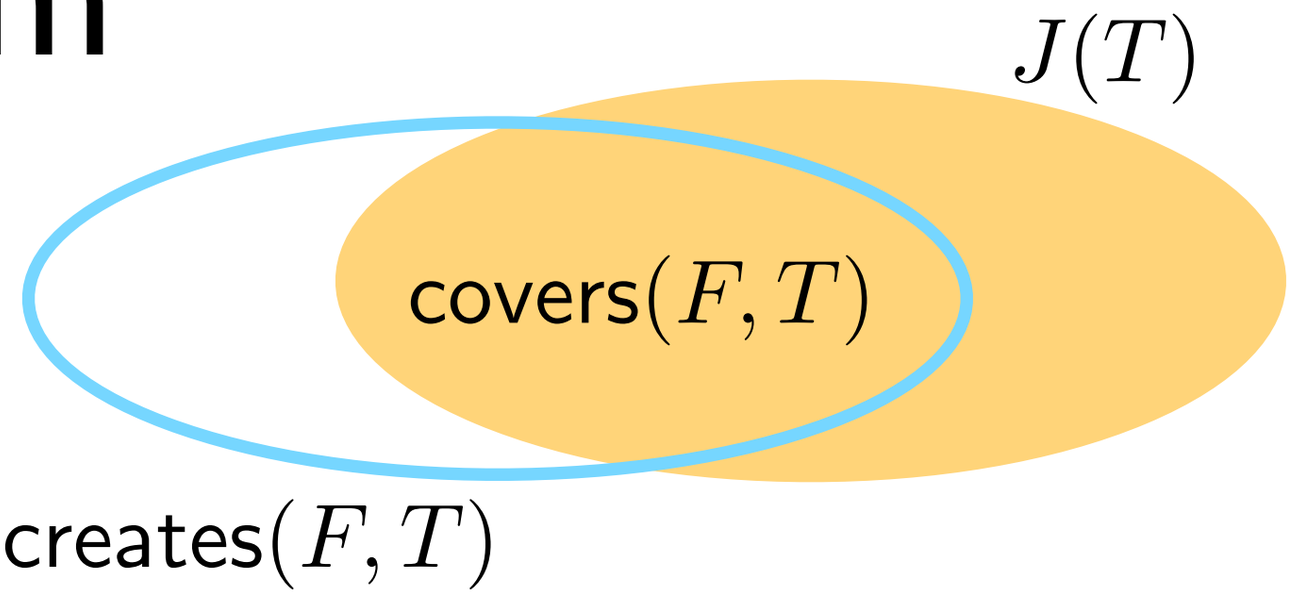
$$1 : in(F) \wedge \text{creates}(F, T) \rightarrow J(T)$$

Our PSL program

given $J(T)$

covers(F, T)

creates(F, T)



find optimal M:

$$in(F) \Leftrightarrow F \in M$$

minimize #errors: $1 : in(F) \wedge \text{creates}(F, T) \rightarrow J(T)$

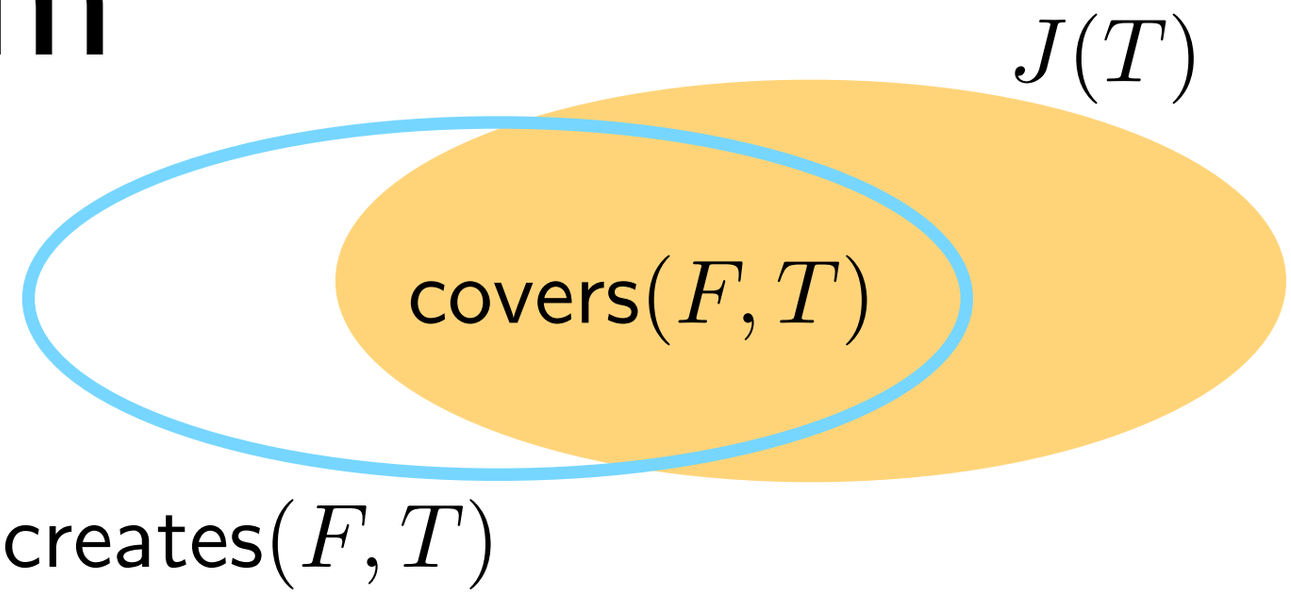
minimize #unexplained: $1 : J(T) \rightarrow \exists F.\text{covers}(F, T) \wedge in(F)$

Our PSL program

given $J(T)$

covers(F, T)

creates(F, T)



find optimal M:

$$in(F) \Leftrightarrow F \in M$$

minimize #errors: $1 : in(F) \wedge \text{creates}(F, T) \rightarrow J(T)$

minimize #unexplained: $1 : J(T) \rightarrow \exists F.\text{covers}(F, T) \wedge in(F)$

minimize size of M: $\text{size}(F) : in(F) \rightarrow \bot$

New PSL construct: prioritized disjunction rules

$$1 : J(T) \rightarrow \exists F.\text{covers}(F, T) \wedge in(F)$$

to be inferred

observed **priority** between
0 (low) and 1 (high)

automatically transformed into set of standard PSL rules expressing a preference for inferred atoms with higher priority

Experimental evaluation

- scenarios generated using iBench [Arocena et al, 15]
- candidate st tgds generated using Clio [Fagin et al, 09]
- E1: increasingly noisy candidates, perfect data
  - metadata-only baseline suffers,
    we get perfect mappings
- E2: ambiguous set of candidates, increasingly noisy data
  - high quality mappings found for up to 25%
    unexpected and 10% missing target tuples in J

Our Contributions

Given
- metadata
- data example
- candidate st tgds

Find
- small set of st tgds
- minimally invalid
- maximally explaining

Our Contributions

Given
- metadata
- data example
- candidate st tgds

Collective Mapping Discovery (CMD)

declarative probabilistic model
+
MPE inference

using Probabilistic Soft Logic (PSL)

Find
- small set of st tgds
- minimally invalid
- maximally explaining

Our Contributions

Given
- metadata
- data example
- candidate st tgds

Collective Mapping Discovery (CMD)

declarative probabilistic model
+
MPE inference

using Probabilistic Soft Logic (PSL)

Find
- small set of st tgds
- minimally invalid
- maximally explaining

- supports arbitrary st tgds
- jointly reasons about metadata and data
- handles noisy input
- efficient solver with quality guarantee
- declarative, extensible definition of optimization task

Our Contributions

Given
- metadata
- data example
- candidate st tgds

Collective Mapping Discovery (CMD)

declarative probabilistic model
+
MPE inference

using Probabilistic Soft Logic (PSL)

Find
- small set of st tgds
- minimally invalid
- maximally explaining

- supports arbitrary st tgds
- jointly reasons about metadata and data
- handles noisy input
- efficient solver with quality guarantee
- declarative, extensible definition of optimization task

Thanks!