TECHNISCHE UNIVERSITÄT DRESDEN

International Center for Computational Logic

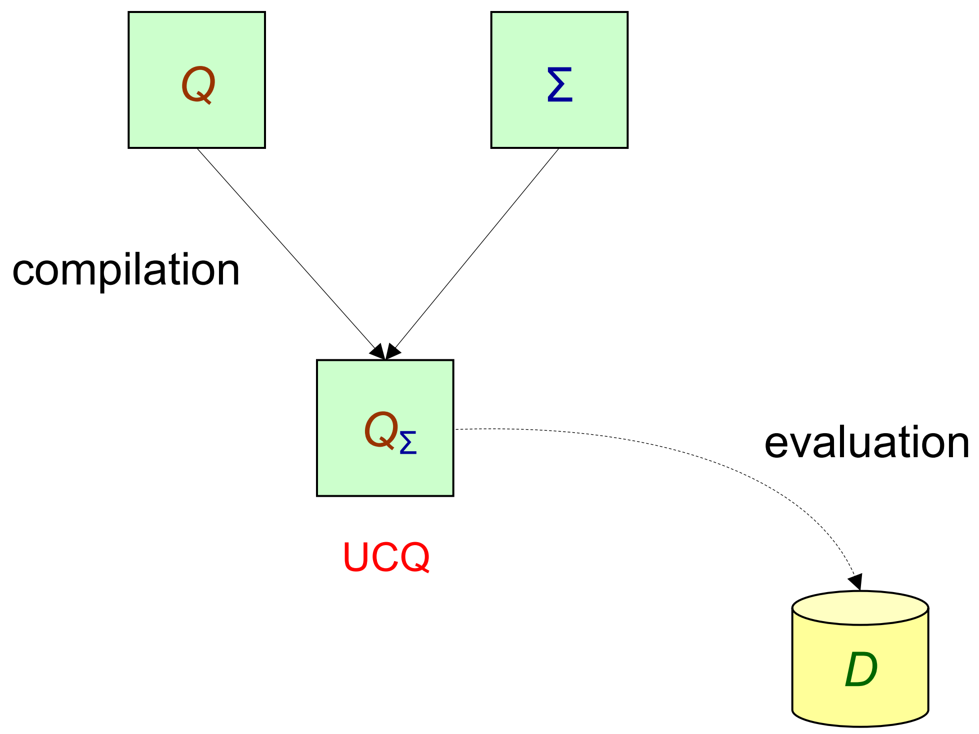
Computational Logic ∴ Group

Sebastian Rudolph
International Center for Computational Logic
TU Dresden

Existential Rules – Lecture 9

Adapted from slides by Andreas Pieris and Michaël Thomazo
Summer Term 2023

UCQ-Rewritability



$\forall D : D \wedge \Sigma \models Q \quad \Leftrightarrow \quad D \models Q_\Sigma$

evaluated and optimized by exploiting existing technology

Limitations of UCQ-Rewritability

$\forall D : D \wedge \Sigma \models Q \Leftrightarrow D \models Q_\Sigma$

evaluated and optimized by exploiting existing technology

- What about the size of Q_Σ? - very large, no rewritings of polynomial size

Exponentially Sized UCQ-Rewritings

$\Sigma = \{\forall X\,(R_k(X) \rightarrow P_k(X))\}_{k \in \{1,\ldots,n\}}$ $\qquad Q = \exists X\,(P_1(X) \wedge \ldots \wedge P_n(X))$

$$\exists X\,(P_1(X) \wedge \ldots \wedge P_n(X))$$

$P_1(X) \vee R_1(X)$ $\qquad$ $P_n(X) \vee R_n(X)$

thus, we need to consider 2^n disjuncts

Limitations of UCQ-Rewritability

$\forall D : D \wedge \Sigma \models Q \Leftrightarrow D \models Q_\Sigma$

evaluated and optimized by exploiting existing technology

- What about the size of Q_Σ? - very large, no rewritings of polynomial size
- What kind of ontology languages can be used for Σ? - below PTIME

PTIME-hard Languages

BCQ-Answering under PTIME-hard languages is not UCQ-rewritable

- Assume that BCQ-Answering is UCQ-rewritable
- Thus, BCQ-Answering is in AC_0 w.r.t. to the data complexity
- Therefore, AC_0 = PTIME which is a contradiction

Limitations of UCQ-Rewritability

$\forall D : D \land \Sigma \models Q \Leftrightarrow D \models Q_\Sigma$

evaluated and optimized by exploiting existing technology

- What about the size of Q_Σ? - very large, no rewritings of polynomial size
- What kind of ontology languages can be used for Σ? - below PTIME

…what about FO-rewritability?

Size of FO-Rewritings

$\Sigma = \{\forall X\, (R_k(X) \rightarrow P_k(X))\}_{k \in \{1,\ldots,n\}}$ $\qquad Q = \exists X\, (P_1(X) \wedge \ldots \wedge P_n(X))$

$\exists X\, ((P_1(X) \vee R_1(X)) \wedge \ldots \wedge (P_n(X) \vee R_n(X)))$

…however, it is known that there are no FO-rewritings of polynomial size,

unless the polynomial hierarchy collapses

Limitations of UCQ/FO-Rewritability

$\forall D : D \land \Sigma \models Q \quad \Leftrightarrow \quad D \models Q_\Sigma$

evaluated and optimized by exploiting existing technology

- What about the size of Q_Σ? - very large, no rewritings of polynomial size
- What kind of ontology languages can be used for Σ? - below PTIME

$\Rightarrow$ a more refined approach is needed

Modify the Database

- An approach proposed in the context of description logics
- Several promising results - applied on (extensions of) EL, and members of the DL-Lite family

D = $\{P(a), S_1(a), P(b), S_2(b)\}$

Σ = $\{\forall X\, (P(X) \rightarrow \exists Y\, (R(X,Y) \wedge P(Y)))\}$

Q = $\exists X \exists Y \exists Z\, (R(X,Y) \wedge R(Z,Y) \wedge S_1(X) \wedge S_2(Z))$

auxiliary constant for satisfying ∃-variables

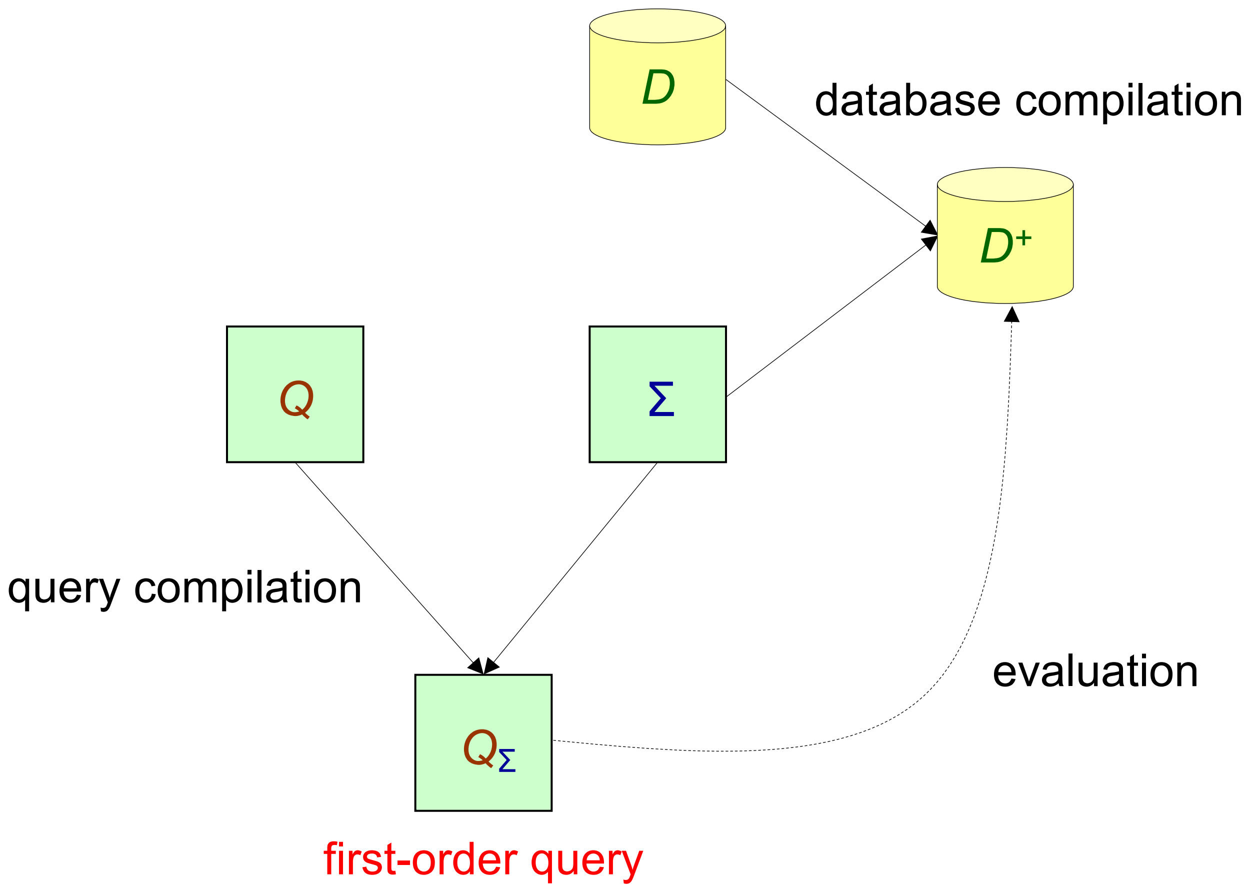
Step 1: Saturate the database, without inventing new nulls

$$D^+ = \{P(a), S_1(a), P(b), S_2(b)\} \cup \{Ex(c)\} \cup \{R(a,c), R(b,c), P(c), R(c,c)\}$$

Step 2: Eliminate unsound answers by rewrting the query into a FO-query

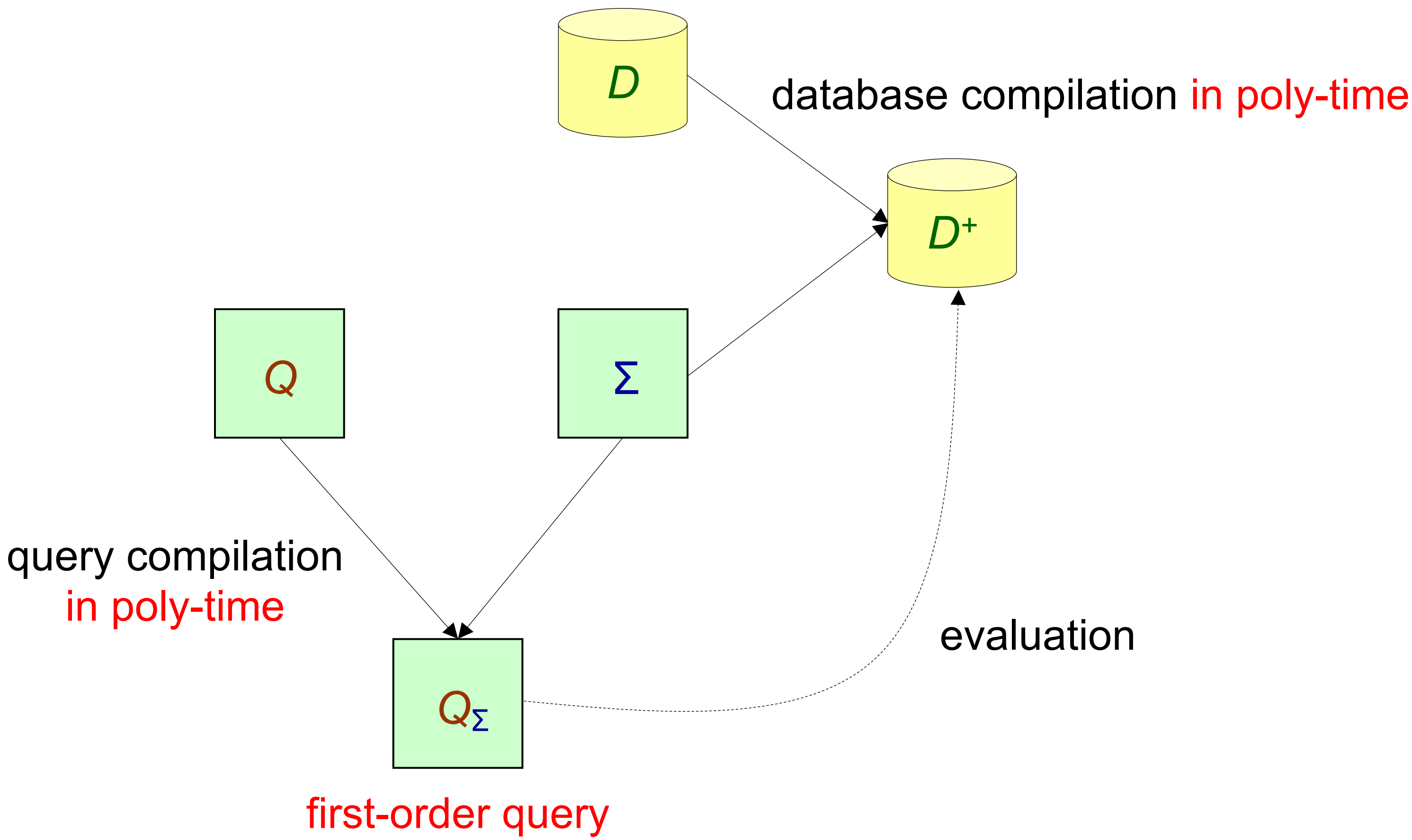
$$Q_{FO} = \exists X \exists Y \exists Z\, ((R(X,Y) \wedge R(Z,Y) \wedge S_1(X) \wedge S_2(Z)) \wedge (Ex(Y) \rightarrow X = Z))$$

Combined FO-Rewritability



$\forall D : D \wedge \Sigma \models Q \quad \Leftrightarrow \quad D^{+} \models Q_{\Sigma}$

Polynomial Combined FO-Rewritability



$$\forall D : D \wedge \Sigma \models Q \quad \Leftrightarrow \quad D^{+} \models Q_{\Sigma}$$

First-Order (FO) Queries

A first-order query Q is a first-order logic formula

$$\varphi(X)$$

with X be the free variables of φ

$$Q(J) = \{t \in adom(J)^{|X|} \mid J \models \varphi(t)\}$$

Polynomial Combined FO-Rewritability: Definition

Consider a class of existential rules $\mathcal{L}$.

BCQ-Answering under $\mathcal{L}$ is polynomially combined FO-rewritable if,

for every database D, $\Sigma \in \mathcal{L}$ and BCQ Q, we can construct in poly-time

a FO-query Q_Σ independently of D, and a database D_Σ independently of Q

such that $D \wedge \Sigma \models Q$ iff $D_\Sigma \models Q_\Sigma$

NOTE: The procedure is not database-independent – the combined approach to query rewriting

Polynomial Combined FO-Rewritability

assumptions on the underlying schema

Size	Arity	FULL	ACYCLIC	LINEAR
∞	∞	[×]	[[×]]	✓
∞	bounded	?	[[×]]	✓
bounded	∞	?	✓	✓
bounded	bounded	?	✓	✓

[×] - assuming that PSPACE $\neq$ EXPTIME

[[×]] - assuming that PSPACE $\neq$ NEXPTIME

Negative Cases

Evaluating a first-order query is in PSPACE

+

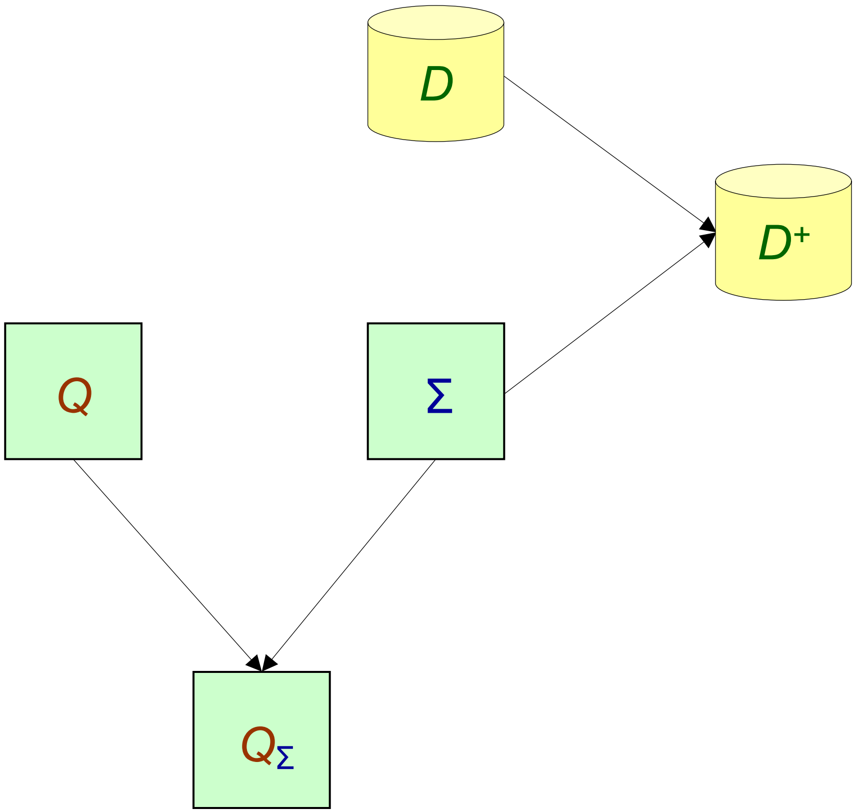
FULL is EXPTIME-hard | ACYCLIC is NEXPTIME-hard

⇓

the polynomial combined approach cannot be applied

unless PSPACE = EXPTIME | unless PSPACE = NEXPTIME

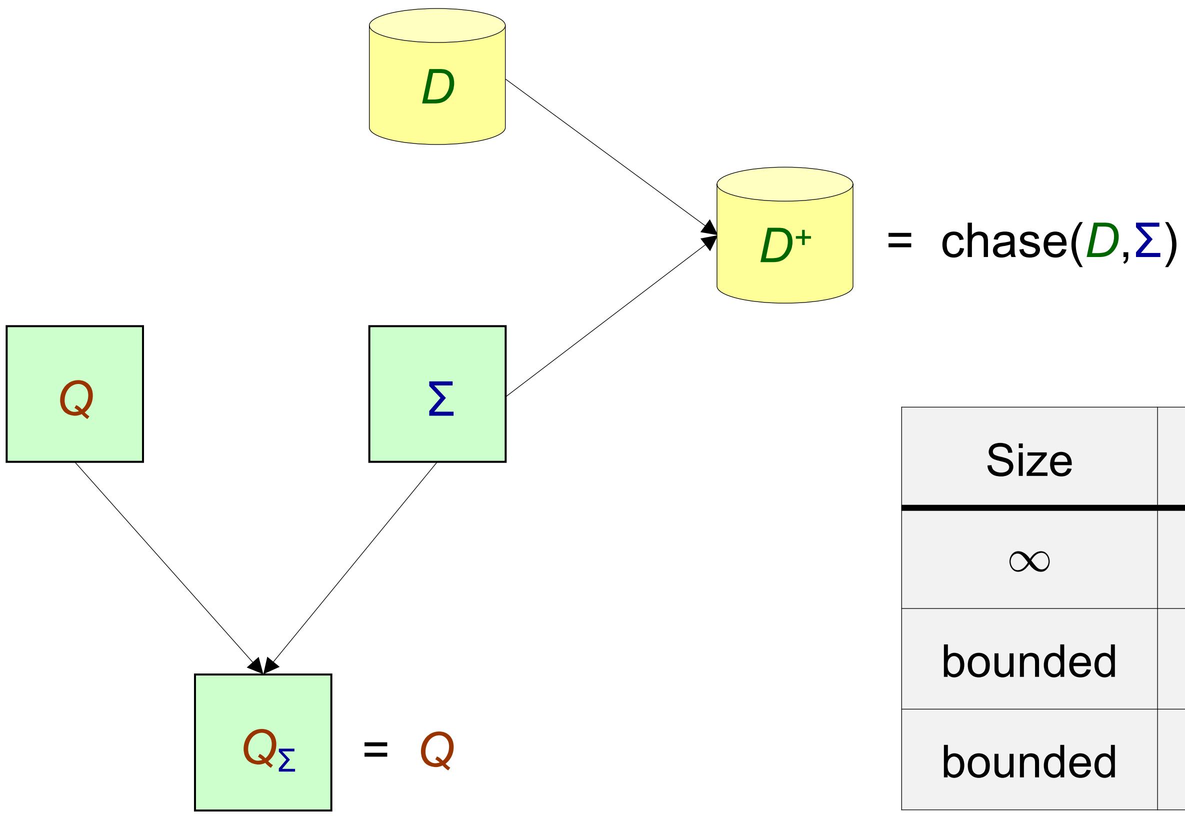
Unknown Cases



Size	Arity	FULL
∞	bounded	?
bounded	∞	?
bounded	bounded	?

Any ideas?

Unknown Cases



Size	Arity	FULL
∞	bounded	?
bounded	∞	?
bounded	bounded	?

$$(|sch(\Sigma)| \cdot (|adom(D)|)^{maxarity})^2 \cdot |\Sigma| \cdot (|adom(D)|)^{maxvariables(\Sigma)} \cdot maxbody(\Sigma)$$

the database compilation phase is costly

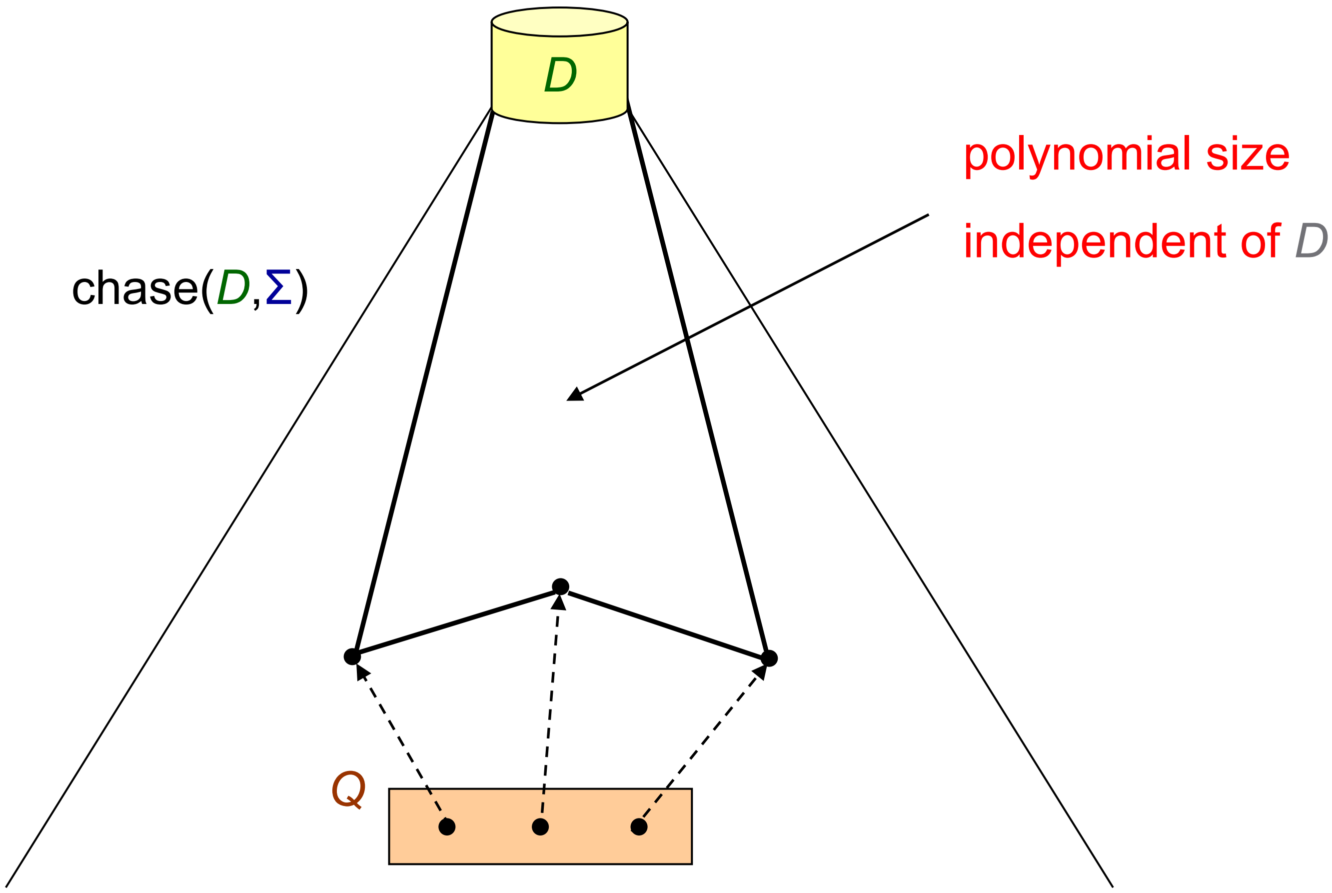
Polynomial Combined FO-Rewritability

assumptions on the underlying schema

Size	Arity	FULL	ACYCLIC	LINEAR
∞	∞	[×]	[[×]]	✓
∞	bounded	?	[[×]]	✓
bounded	∞	?	✓	✓
bounded	bounded	?	✓	✓

by exploiting the polynomial witness property

Polynomial Witness Property (PWP)



chase(*D*,Σ) ⊨ *Q* ⇒ the query admits a small witness

Polynomial Witness Property (PWP)

Theorem: The PWP implies that BCQ-Answering is polynomially combined FO-rewritable

Proof (hint):

- We simulate the polynomially sized witness via a polynomially sized first-order query (query compilation)
- Notice that the number of nulls that appear in the witness depends on the query, and thus can not be explicitly added in the database
- We simulate these nulls via tuples of 0s and 1s - the constants 0 and 1 are explicitly added in the database (database compilation)

Polynomial Combined FO-Rewritability

assumptions on the underlying schema

Size	Arity	FULL	ACYCLIC	LINEAR
∞	∞	[×]	[[×]]	✓
∞	bounded	?	[[×]]	✓
bounded	∞	?	✓	✓
bounded	bounded	?	✓	✓

no witness of polynomial size

Witnesses and Linear Rules

SUCC = $\{\Sigma_n\}_{n>0}$, where

$$\Sigma_n = \{\forall Z \forall O \forall B_1 \ldots \forall B_n\, (num(Z,O,B_1,\ldots,B_{n-i},Z,\overbrace{O,\ldots,O}^{i\,-1}) \rightarrow$$

$$num(Z,O,B_1,\ldots,B_{n-i},O,\underbrace{Z,\ldots,Z}_{i\,-1}))\}_{i \in \{1,\ldots,n\}}$$

- Σ_n simulates the successor operator on binary numbers
- The binary number $b_1b_2\ldots b_m$ is encoded as $num(0,1,b_1,b_2,\ldots,b_m)$
- $D = \{num(0,1,0,\ldots,0)\}$ & $Q = num(0,1,1,\ldots,1)$ - witness of exponential size

$\Rightarrow$ Linear rules (even with one predicate) do not enjoy the PWP

Polynomial Combined FO-Rewritability

assumptions on the underlying schema

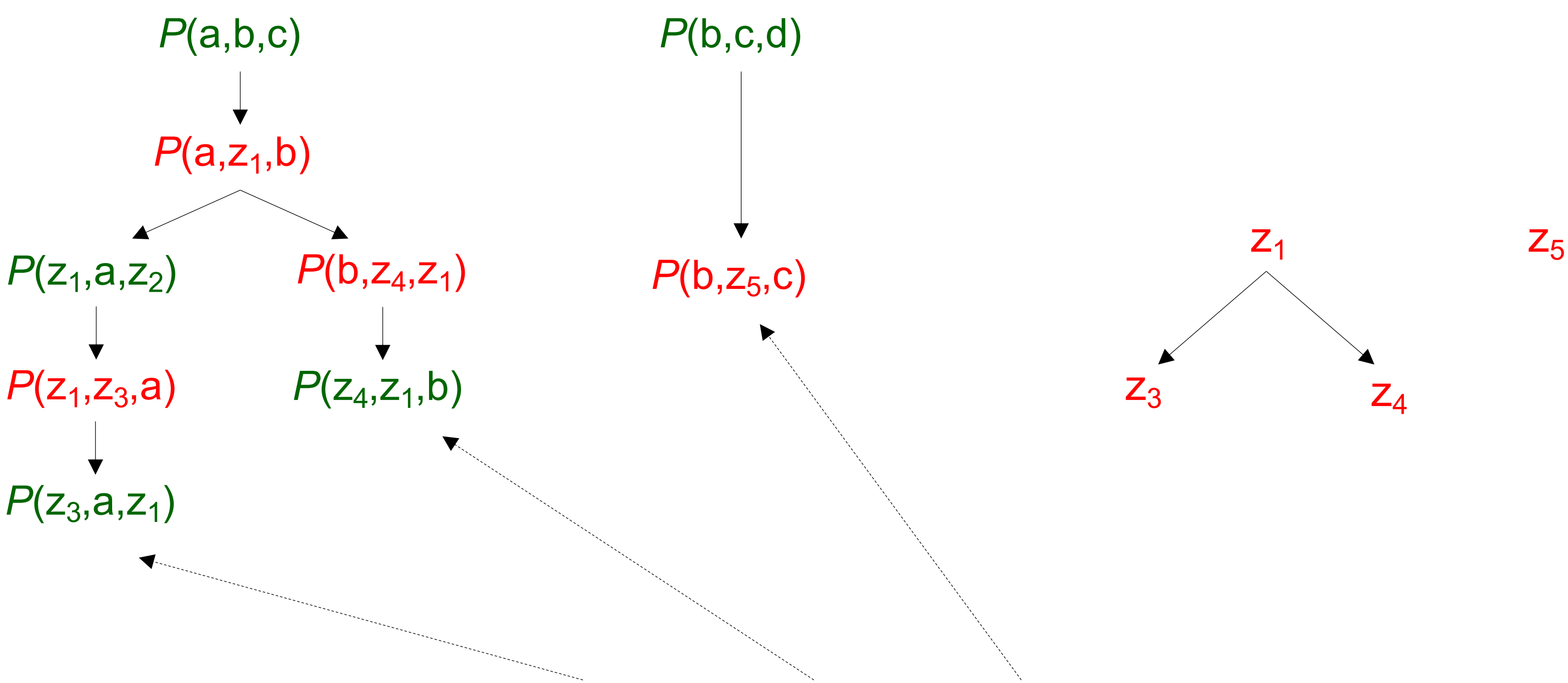
Size	Arity	FULL	ACYCLIC	LINEAR
∞	∞	[×]	[[×]]	✓
∞	bounded	?	[[×]]	✓
bounded	∞	?	✓	✓
bounded	bounded	?	✓	✓

Challenge: Simulate witnesses of exponential size via FO-queries of polynomial size

Witness Generator

D = $\{P(a,b,c), P(b,c,d)\}$

Σ = $\{\forall X \forall Y \forall Z\, (P(X,Y,Z) \rightarrow \exists W\, P(X,W,Y)),\quad \forall X \forall Y \forall Z\, (P(X,Y,Z) \rightarrow \exists W\, P(Z,W,Y))$
$\forall X \forall Y \forall Z\, (P(X,Y,Z) \rightarrow \exists W\, P(Y,X,W)),\quad \forall X \forall Y \forall Z\, (P(X,Y,Z) \rightarrow P(Y,Z,X))\}$



Q = $\exists A \exists B \exists C \exists D\, (P(A,a,B) \wedge P(C,B,b) \wedge P(D,c,b))$

Witness Generator

Witness generator for Q w.r.t. D and Σ

$$(\,h,\ v,\ F\,)$$

h: var(Q) $\rightarrow$ (dom(D) $\cup$ NULLS)

the homomorphism that maps Q to chase(D,Σ)

v: null(h(Q)) $\rightarrow$ chase(D,Σ)

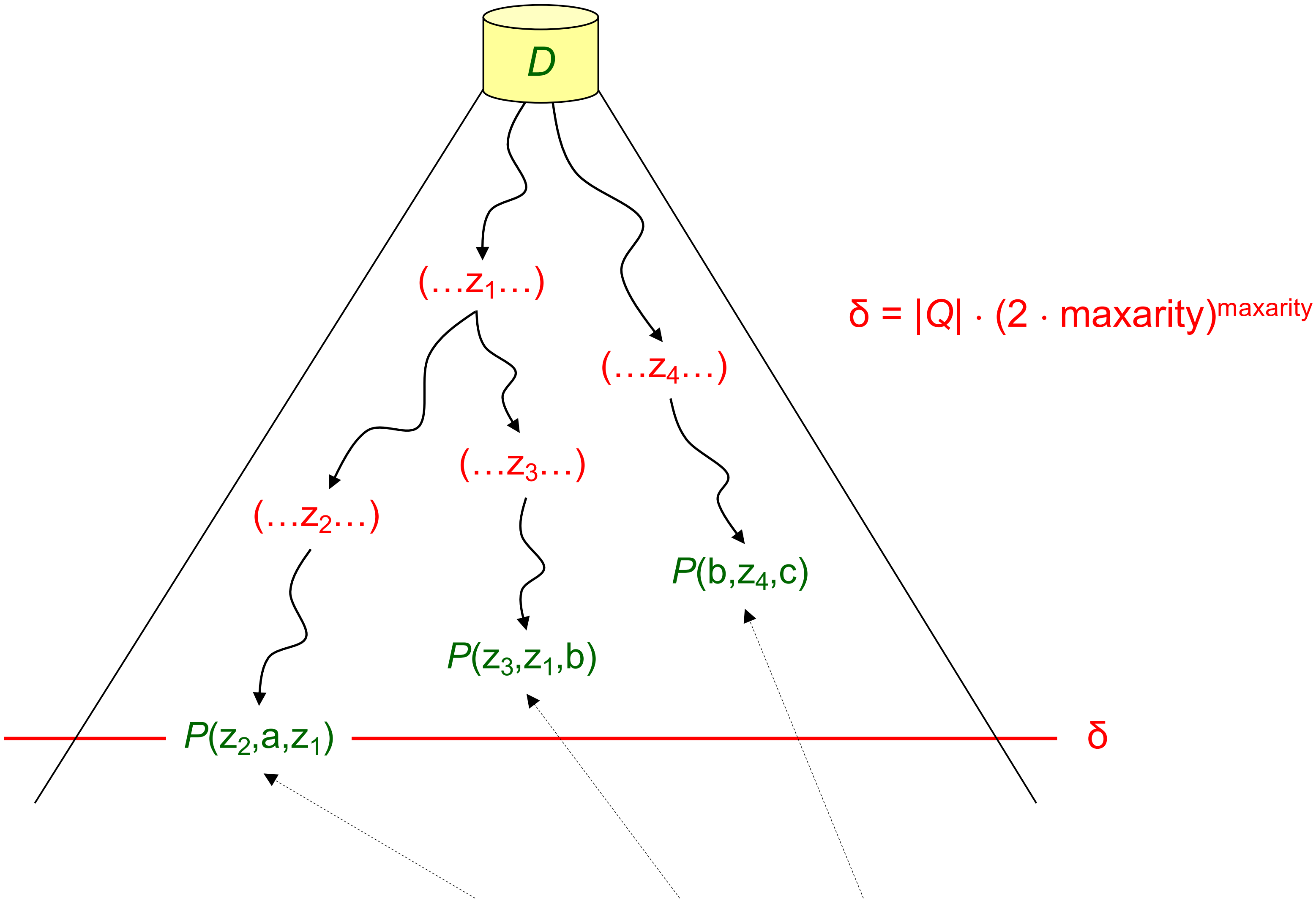
the function that gives the atoms of chase(D,Σ) in which the nulls of h(Q) are invented

F: rooted forest on null(h(Q))

ensures that the reachability checks succeed

Lemma: $D \wedge \Sigma \models Q \quad \Leftrightarrow \quad$ there exists a witness generator for Q w.r.t. D and Σ

Reachability on the Chase Graph



$Q = \exists A \exists B \exists C \exists D\ (P(A,a,B) \wedge P(C,B,b) \wedge P(D,c,b))$

Reachability Checks

$\Pi_k(X,Y)$:= $P(Y)$ is reachable from $P(X)$ via a path of length at most 2^k

reach(X,Y) := $\Pi_{\lceil \log \delta \rceil}(X,Y)$

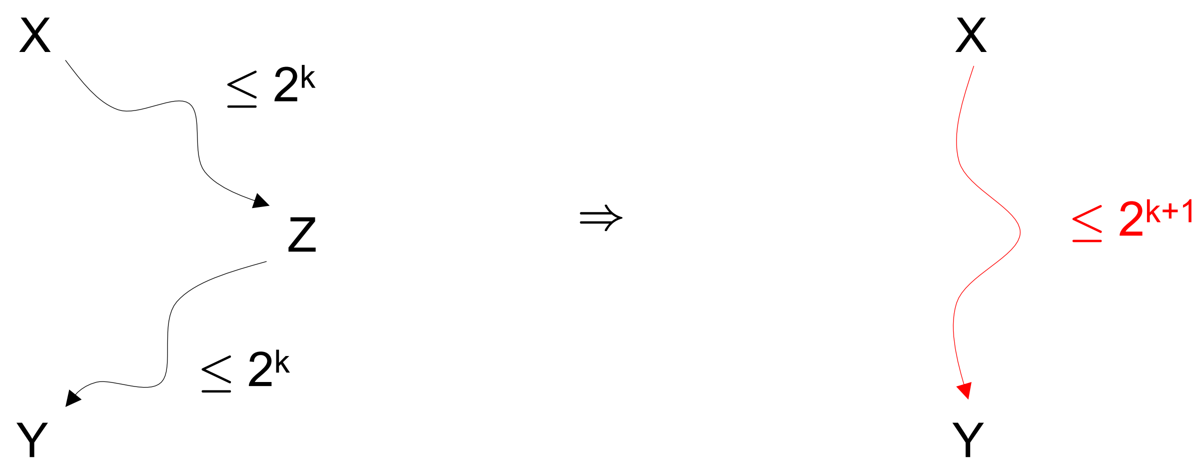
$\delta = |Q| \cdot (2 \cdot \text{maxarity})^{\text{maxarity}}$, and thus $\lceil \log \delta \rceil$ is polynomial, independent of D

Reachability Checks

$\Pi_k(X,Y)$ is defined inductively as follows:

$\Pi_0(X,Y)$:= $P(Y)$ can be obtained from $P(X)$ by applying a rule of Σ

$$\Pi_{k+1}(X,Y) := \exists Z\, (\forall U \forall V\, ((((U = X) \wedge (V = Z)) \vee ((U = Z) \wedge (V = Y))) \rightarrow \Pi_k(U,V)))$$



Reachability Checks

depth of the witness is at most

$$\delta = |Q| \cdot (2 \cdot \text{ maxarity})^{\text{maxarity}}$$

$$\Downarrow$$

maximum number of nulls in the proof is

$$(|Q| \cdot \delta \cdot \text{ maxarity})$$

explicitly added in *D*

$$\Downarrow$$

the nulls in the witness can be represented via tuples of $\{0,1\}^{\alpha}$, where

$\alpha = \lceil \log (|Q| \cdot \delta \cdot \text{ maxarity}) \rceil$ - polynomial, and independent of *D*

Polynomial Combined FO-Rewritability

assumptions on the underlying schema

Size	Arity	FULL	ACYCLIC	LINEAR
∞	∞	[×]	[[×]]	✓
∞	bounded	?	[[×]]	✓
bounded	∞	?	✓	✓
bounded	bounded	?	✓	✓

[×] - assuming that PSPACE $\neq$ EXPTIME

[[×]] - assuming that PSPACE $\neq$ NEXPTIME

Research Directions & Open Problems

Query Rewriting

- Construct (pure) rewritings efficiently - field of intense research
- Existing results on the combined approach are of theoretical nature - far from being practical
- Full existential rules and polynomial combined FO-rewritability - currently under investigation

Ultimate Goal: An efficient reasoner for rule-based languages

Additional Modelling Features

- Counting quantifiers - very little is known

$$\forall X\ (\mathit{professor}(X) \rightarrow \exists_{\leq 4} Y\ (\mathit{supervisorOf}(X,Y) \wedge \mathit{student}(Y))$$

- Default negation (or negation as failure) - lot of recent results, but not completely understood

$$\forall X\ (\mathit{person}(X) \rightarrow \exists Y\ (\mathit{hasParent}(X,Y) \wedge \mathit{person}(Y))$$

$$\forall X\ (\mathit{person}(X) \wedge \text{not } \mathit{even}(X) \rightarrow \mathit{odd}(X))$$

$$\forall X\ (\mathit{person}(X) \wedge \text{not } \mathit{odd}(X) \rightarrow \mathit{even}(X))$$

Last Words: The Bigger Picture