Sebastian Rudolph
International Center for Computational Logic
TU Dresden

Existential Rules – Lecture 2

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

Some Notation

- Our basic vocabulary:
  - A countable set **C** of constants - domain of a database
  - A countable set **N** of (labeled) nulls - globally ∃-quantified variables
  - A countable set **V** of (regular) variables - used in rule and queries

- A term is a constant, null or variable

- An atom has the form $P(t_1, \dots, t_n)$ where P is an n-ary predicate and each t_i is a term

- Sets of atoms are typically understood as the conjunction over their elements

Syntax of Existential Rules

An existential rule is an expression

$$\underbrace{\forall \mathbf{X} \forall \mathbf{Y} \, (\varphi(\mathbf{X},\mathbf{Y})}_{\text{body}} \rightarrow \underbrace{\exists \mathbf{Z} \, \psi(\mathbf{X},\mathbf{Z}))}_{\text{head}}$$

- $\mathbf{X}$,$\mathbf{Y}$ and $\mathbf{Z}$ are tuples of variables of $\mathbf{V}$
- $\varphi(\mathbf{X},\mathbf{Y})$ and $\psi(\mathbf{X},\mathbf{Z})$ are (constant-free) conjunctions of atoms

…a.k.a. tuple-generating dependencies, and Datalog$^{\pm}$ rules

Homomorphism

- Semantics of existential rules via the key notion of homomorphism

- A substitution from a set of symbols **S** to a set of symbols **T** is a function $h : \mathbf{S} \rightarrow \mathbf{T}$, i.e., a set of assignments of the form $s \mapsto t$, with $s \in \mathbf{S}$ and $t \in \mathbf{T}$

- A homomorphism from a set of atoms **A** to a set of atoms **B** is a substitution $h : \mathbf{C} \cup \mathbf{N} \cup \mathbf{V} \rightarrow \mathbf{C} \cup \mathbf{N} \cup \mathbf{V}$ such that:

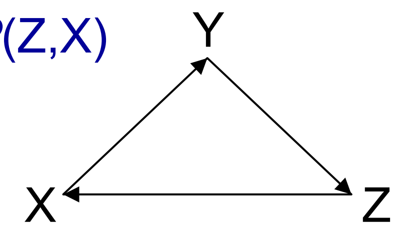
  (i) $t \in \mathbf{C} \;\Rightarrow\; h(t) = t$ (cf. unique name assumption)

  (ii) $P(t_1, \ldots, t_n) \in \mathbf{A} \;\Rightarrow\; h(P(t_1, \ldots, t_n)) := P(h(t_1), \ldots, h(t_n)) \in \mathbf{B}$

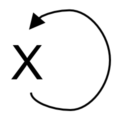
- Can be naturally extended to sets (and thus conjunctions) of atoms

Exercise: Find the Homomorphisms

$\varphi_1 = P(X,Y) \wedge P(Y,Z) \wedge P(Z,X)$



$\varphi_2 = P(X,X)$



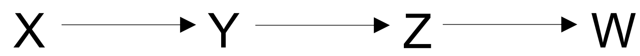
$\varphi_3 = P(X,Y) \wedge P(Y,X) \wedge P(Y,Y)$

$\varphi_4 = P(X,Y) \wedge P(Y,X)$

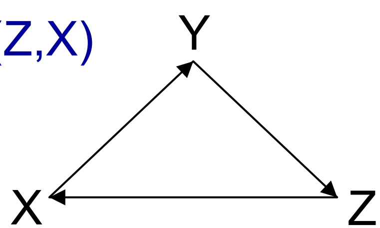
$\varphi_5 = P(X,Y) \wedge P(Y,Z) \wedge P(Z,W)$

Exercise: Find the Homomorphisms

$\varphi_5 \;=\; P(X,Y) \wedge P(Y,Z) \wedge P(Z,W)$

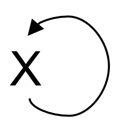


$\varphi_1 \;=\; P(X,Y) \wedge P(Y,Z) \wedge P(Z,X)$



$\varphi_4 \;=\; P(X,Y) \wedge P(Y,X)$

$\varphi_2 \;=\; P(X,X)$



$\varphi_3 \;=\; P(X,Y) \wedge P(Y,X) \wedge P(Y,Y)$

Semantics of Existential Rules

- An instance J is a model of the rule

  $$\sigma \;=\; \forall \mathbf{X} \forall \mathbf{Y}\, (\varphi(\mathbf{X},\mathbf{Y}) \rightarrow \exists \mathbf{Z}\, \psi(\mathbf{X},\mathbf{Z}))$$

  written as $J \models \sigma$, if the following holds:

  whenever there exists a homomorphism h such that $h(\varphi(\mathbf{X},\mathbf{Y})) \subseteq J$,

  then there exists $g \supseteq h_{|\mathbf{X}}$ such that $g(\psi(\mathbf{X},\mathbf{Z})) \subseteq J$

  $\{t \mapsto h(t) \mid t \in \mathbf{X}\}$ – the restriction of h to **X**

- Given a set Σ of existential rules, J is a model of Σ, written as $J \models \Sigma$, if the following holds: for each $\sigma \in \Sigma$, $J \models \sigma$

- It can be shown that $J \models \Sigma$ iff J is a model of the first-order theory $\bigwedge_{\sigma \in \Sigma} \sigma$

Existential Rules vs. DLs

Existential rules and DLs rely on first-order semantics - comparable formalisms

DL-Lite Axioms	Existential Rules
$A \sqsubseteq B$	$\forall X\ (A(X) \rightarrow B(X))$
$A \sqsubseteq \exists R$	$\forall X\ (A(X) \rightarrow \exists Y\ R(X,Y))$
$\exists R \sqsubseteq A$	$\forall X \forall Y\ (R(X,Y) \rightarrow A(X))$
$\exists R \sqsubseteq \exists P$	$\forall X \forall Y\ (R(X,Y) \rightarrow \exists Z\ P(X,Z))$
$A \sqsubseteq \exists R.B$	$\forall X\ (A(X) \rightarrow \exists Y\ (R(X,Y) \wedge B(Y)))$
$R \sqsubseteq P$	$\forall X \forall Y\ (R(X,Y) \rightarrow P(X,Y))$
$A \sqsubseteq \neg B$	$\forall X\ (A(X) \wedge B(X) \rightarrow \bot)$

Existential Rules vs. DLs

Existential rules and DLs rely on first-order semantics - comparable formalisms

EL Axioms	Existential Rules
$A \sqsubseteq B$	$\forall X\ (A(X) \rightarrow B(X))$
$A \sqcap B \sqsubseteq C$	$\forall X\ (A(X) \wedge B(X) \rightarrow C(X))$
$A \sqsubseteq \exists R.B$	$\forall X\ (A(X) \rightarrow \exists Y\ (R(X,Y) \wedge B(Y)))$
$\exists R.B \sqsubseteq A$	$\forall X \forall Y\ (R(X,Y) \wedge B(Y) \rightarrow A(X))$

Existential Rules vs. DLs

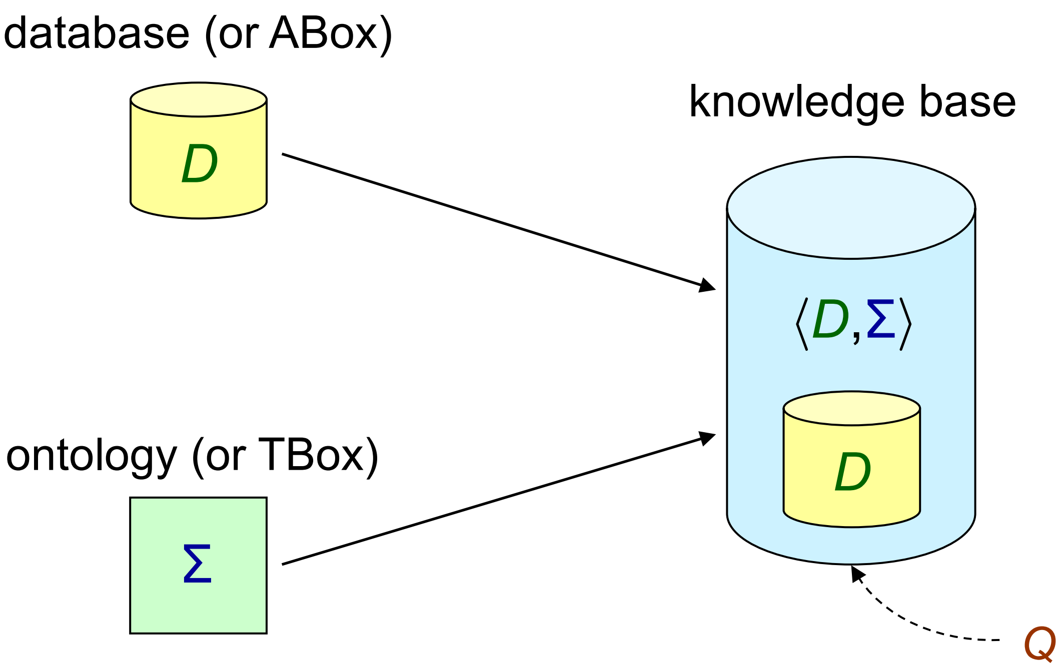
- Several Horn DLs (without disjunction) can be expressed via existential rules
- But, existential rules can express more

$$\forall X\ (\mathit{Boss}(X) \rightarrow \mathit{supervisorOf}(X,X))$$

$$\forall X \forall Y\ (\mathit{siblingOf}(X,Y) \rightarrow \exists Z\ (\mathit{parentOf}(Z,X) \wedge \mathit{parentOf}(Z,Y)))$$

- Higher arity predicates allow for more flexibility
  - Direct translation of database relations
  - Adding contextual information is easy (provenance, trust, etc.)

Ontology-Based Query Answering (OBQA)

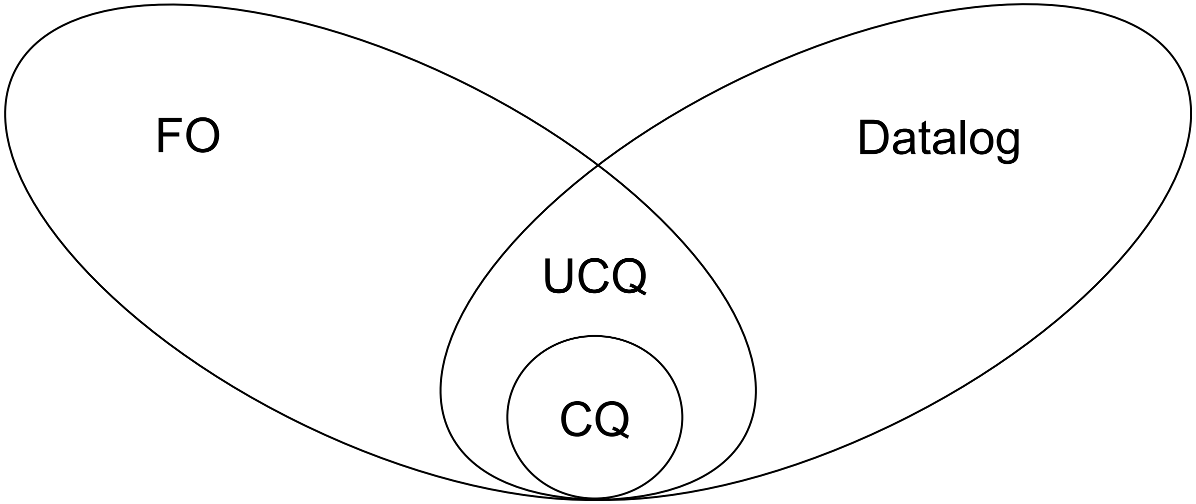


existential rules

$\forall \mathbf{X} \forall \mathbf{Y} \, (\varphi(\mathbf{X},\mathbf{Y}) \rightarrow \exists \mathbf{Z} \, \psi(\mathbf{X},\mathbf{Z}))$

Query Languages

- The four most important query languages
  - Conjunctive Queries (CQ)
  - Unions of Conjunctive Queries (UCQ)
  - First-order Queries (FO)
  - Datalog



Syntax of Conjunctive Queries

A conjunctive query (CQ) is an expression

$$\exists \mathbf{Y} \, (\varphi(\mathbf{X},\mathbf{Y}))$$

- **X** and **Y** are tuples of variables of **V**
- $\varphi(\mathbf{X},\mathbf{Y})$ is a conjunction of atoms (possibly with constants)

The most important query language used in practice

Forms the SELECT-FROM-WHERE fragment of SQL

Semantics of Conjunctive Queries

- A match of a CQ $\exists \mathbf{Y}\ (\varphi(\mathbf{X},\mathbf{Y}))$ in an instance J is a homomorphism h such that $h(\varphi(\mathbf{X},\mathbf{Y})) \subseteq J$ i.e., all the atoms of the query are satisfied

- The answer to $Q = \exists \mathbf{Y}\ (\varphi(\mathbf{X},\mathbf{Y}))$ over J is the set of tuples

$$Q(J) = \{h(\mathbf{X}) \mid h \text{ is a match of } Q \text{ in } J\}$$

- The answer consists of the witnesses for the free variables of the query

Conjunctive Queries: Example

Find the researchers who work for the project DIADEM

Researcher(id), *Project*(id), *worksFor*(rid, pid), *PrName*(pid, name)

$\exists Y\ (\mathit{Researcher}(X) \wedge \mathit{worksFor}(X,Y) \wedge \mathit{Project}(Y) \wedge \mathit{PrName}(Y, \text{“DIADEM”}))$

SELECT R.id

FROM *Researcher* R, *worksFor* W, *Project* P, *PrName* N

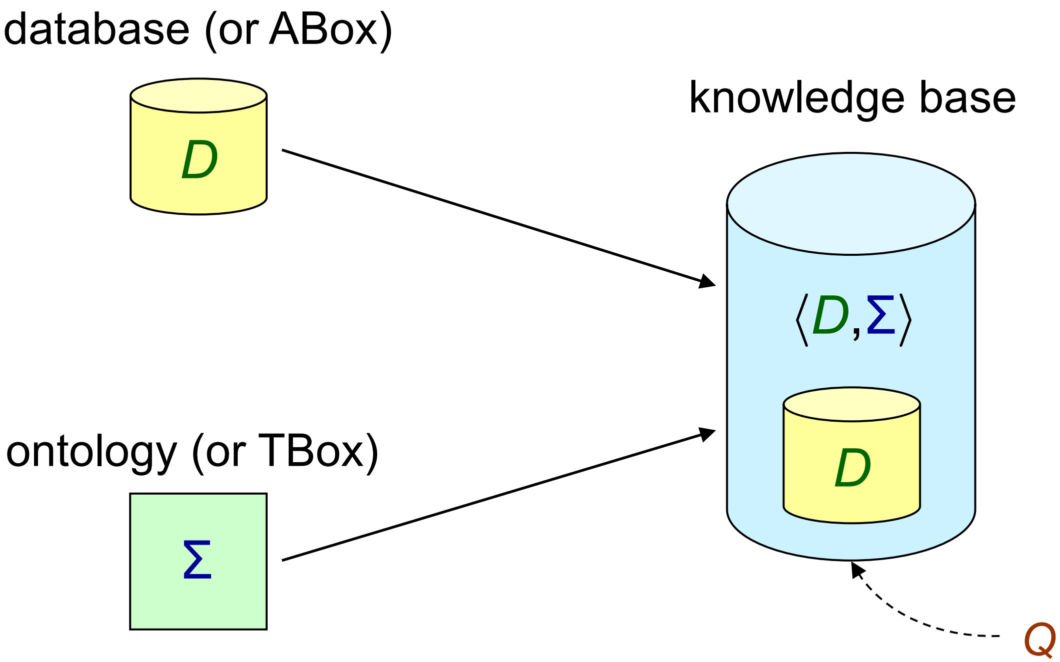
WHERE R.id = W.rid AND

W.pid = P.id AND

P.id = N.pid AND

N.name = “DIADEM”

Ontology-Based Query Answering (OBQA)



existential rules

$\forall \mathbf{X} \forall \mathbf{Y} \; (\varphi(\mathbf{X},\mathbf{Y}) \rightarrow \exists \mathbf{Z} \; \psi(\mathbf{X},\mathbf{Z}))$

conjunctive queries

$\exists \mathbf{Y} \; (\varphi(\mathbf{X},\mathbf{Y}))$

OBQA: Formal Definition

active domain – constants occurring in D

CQ-Answering:

Input: database D, existential rules Σ, CQ $Q = \exists \mathbf{Y} (\varphi(\mathbf{X},\mathbf{Y}))$, tuple t $\in \mathrm{adom}(D)^{|\mathbf{X}|}$

Question: decide whether $\mathbf{t} \in \mathrm{certain}(Q,\langle D,\Sigma\rangle) = \bigcap_{J \in \mathrm{models}(D \wedge \Sigma)} Q(J)_{\downarrow}$

$$\mathrm{models}(D \wedge \Sigma) = \{J \mid J \supseteq D \text{ and } J \models \Sigma\}$$

$\downarrow$ - we are interested only on ground answers that contain values from D

OBQA: Formal Definition

active domain – constants occurring in D

CQ-Answering:

Input: database D, existential rules Σ, CQ $Q = \exists \mathbf{Y}\,(\varphi(\mathbf{X},\mathbf{Y}))$, tuple $t \in \mathrm{adom}(D)^{|\mathbf{X}|}$

Question: decide whether $\mathbf{t} \in \mathrm{certain}(Q,\langle D,\Sigma\rangle) = \bigcap_{J \in \mathrm{models}(D \wedge \Sigma)} Q(J)_{\downarrow}$

$$\mathbf{t} \in \mathrm{certain}(Q,\langle D\ \Sigma\rangle) \quad \text{iff} \quad \mathbf{t} \in \bigcap_{J \in \mathrm{models}(D \wedge \Sigma)} Q(J)_{\downarrow}$$

$$\text{iff} \quad \forall J \in \mathrm{models}(D \wedge \Sigma),\ J \vDash \exists \mathbf{Y}\,(\varphi(\mathbf{t},\mathbf{Y}))$$

$$\text{iff} \quad D \wedge \Sigma \vDash \exists \mathbf{Y}\,(\varphi(\mathbf{t},\mathbf{Y}))$$

Boolean CQ (BCQ) – no free variables

OBQA: Formal Definition

BCQ-Answering:

Input: database D, existential rules Σ, BCQ $Q = \exists Y\ (\varphi(Y))$

Question: decide whether $D \wedge \Sigma \models Q$ – the answer is yes or no

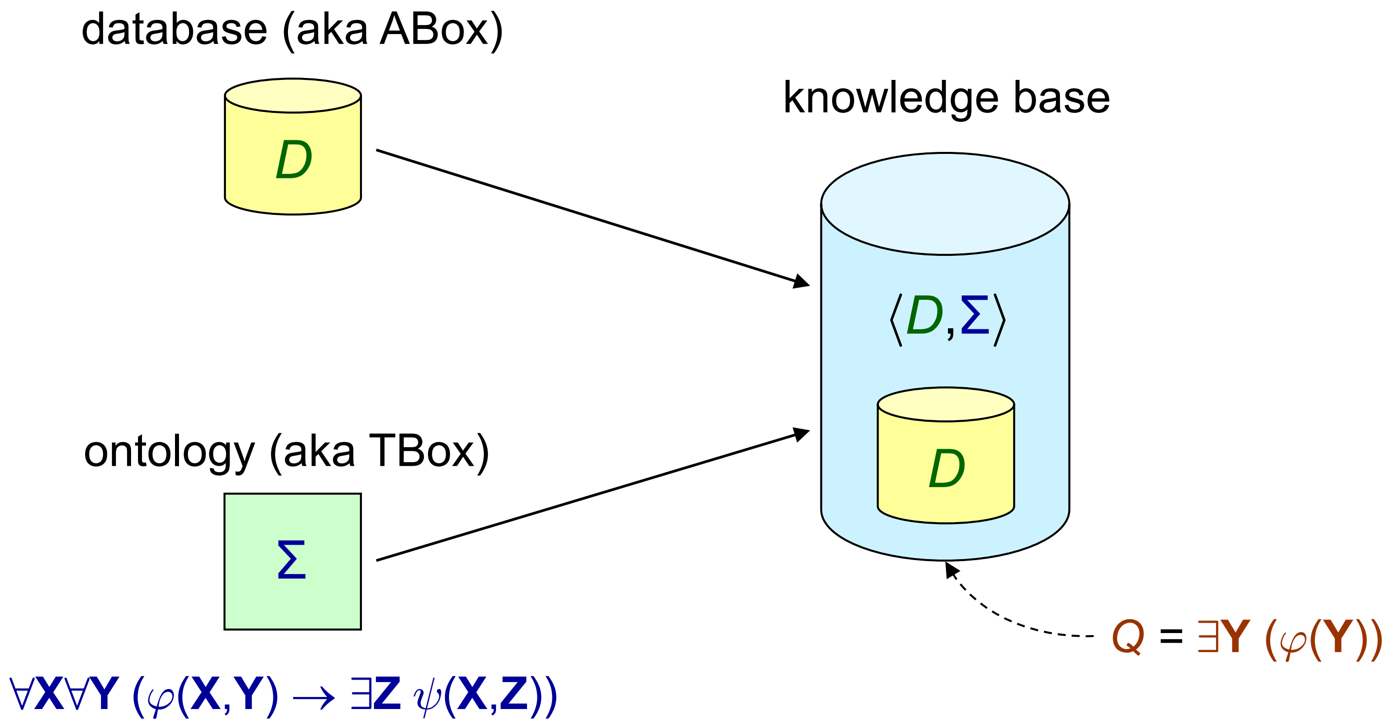
Lemma: CQ-Answering $\equiv_{\text{LOGSPACE}}$ BCQ-Answering

Proof: ($\leq_{\text{LOGSPACE}}$) By simply instantiating the free variables of Q

($\geq_{\text{LOGSPACE}}$) By definition, a Boolean CQ is a CQ

…for brevity, we focus on BCQ-Answering

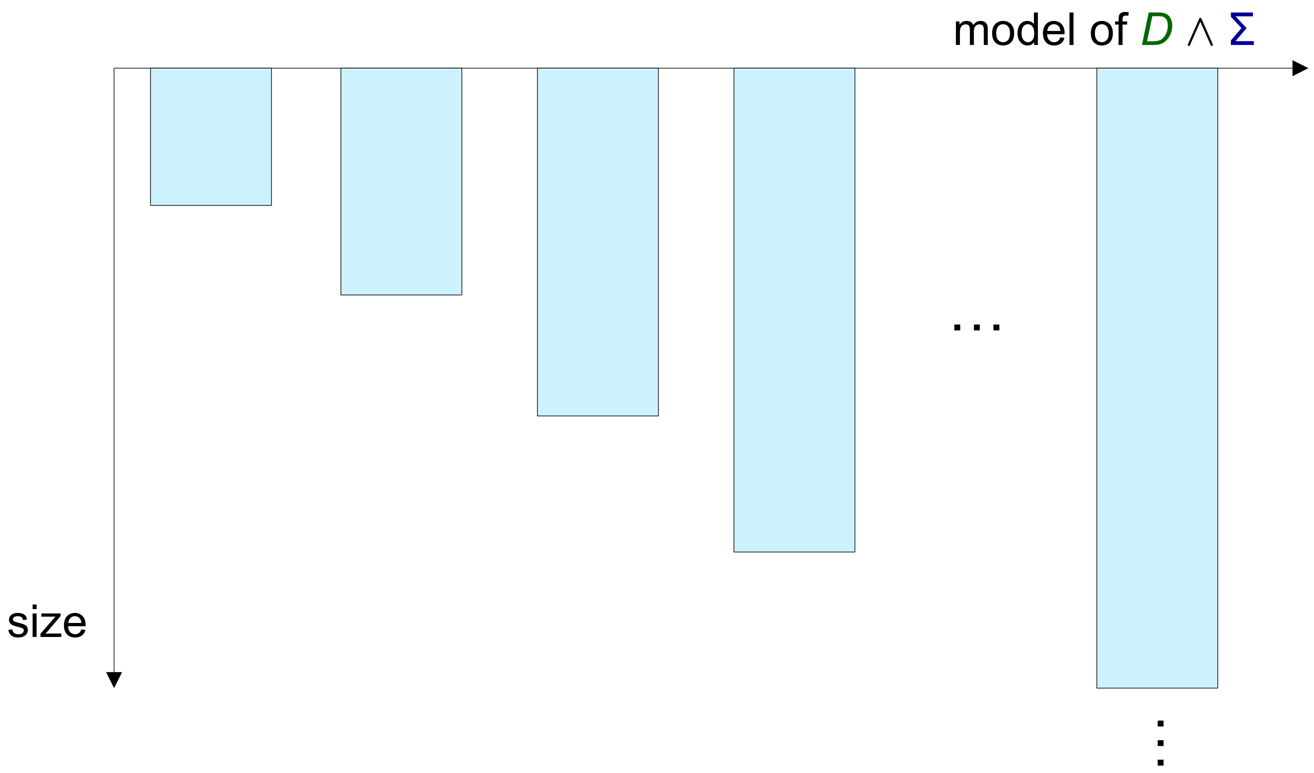
BCQ-Answering: Our Main Decision Problem



decide whether $D \wedge \Sigma \vDash Q$

The Two Dimensions of Infinity

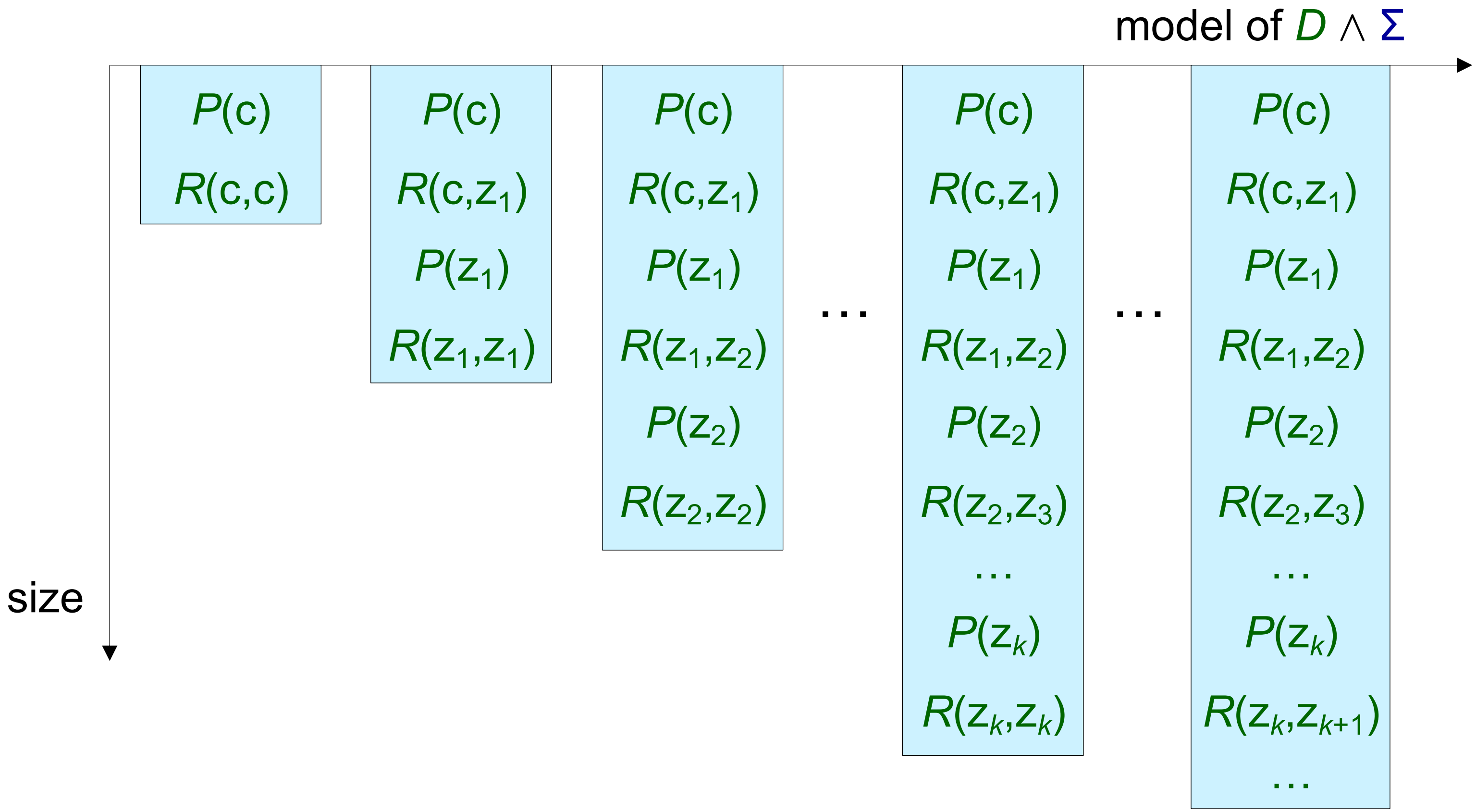
Consider the database *D*, and the set of existential rules Σ



D ∧ Σ admits infinitely many models, and each one may be of infinite size

The Two Dimensions of Infinity

$D = \{P(c)\}$ $\Sigma = \{\forall X\ (P(X) \to \exists Y\ (R(X,Y) \land P(Y)))\}$



$z_1, z_2, z_3, \ldots$ are nulls of N

Taming the First Dimension of Infinity

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

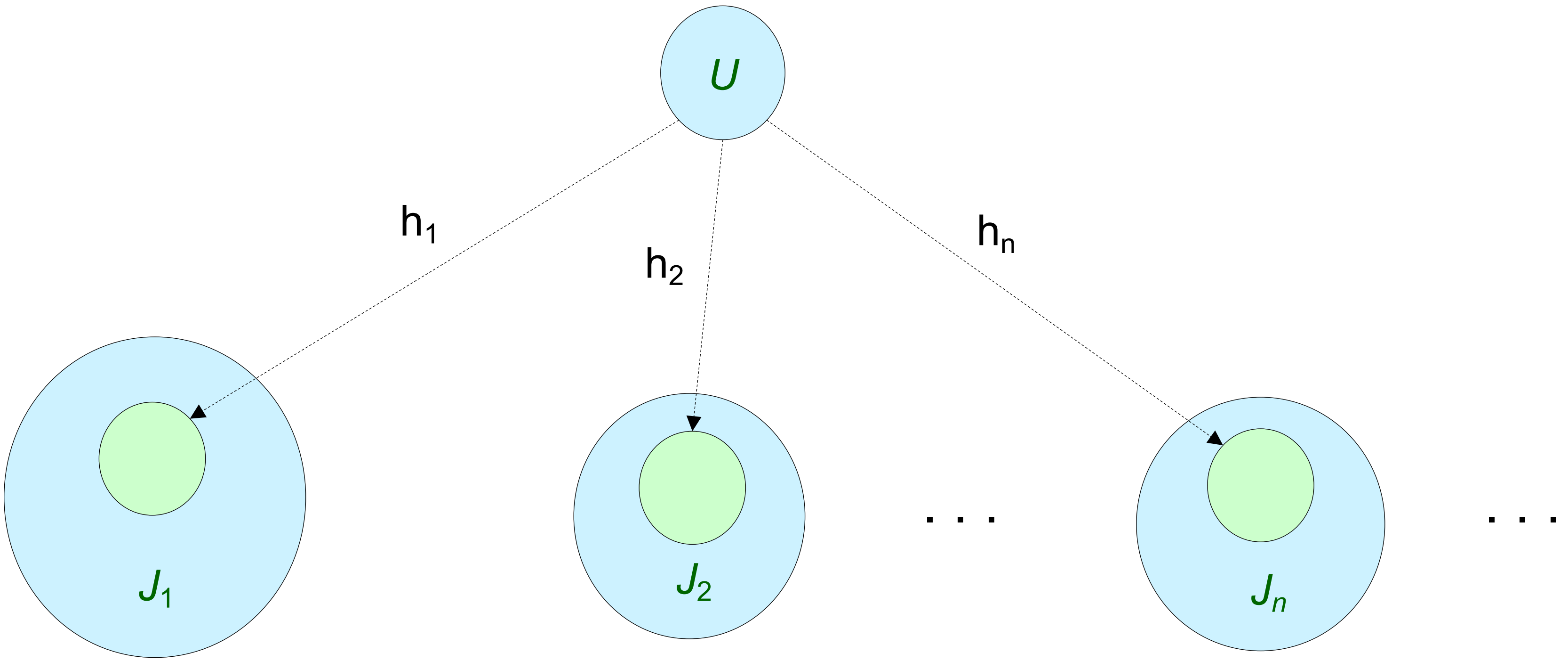
model of $D \wedge \Sigma$ →

size ↓

$P(c)$ $R(c,c)$	$P(c)$ $R(c,z_1)$ $P(z_1)$ $R(z_1,z_1)$	$P(c)$ $R(c,z_1)$ $P(z_1)$ $R(z_1,z_2)$ $P(z_2)$ $R(z_2,z_2)$	…	$P(c)$ $R(c,z_1)$ $P(z_1)$ $R(z_1,z_2)$ $P(z_2)$ $R(z_2,z_3)$ … $P(z_k)$ $R(z_k,z_k)$	…	$P(c)$ $R(c,z_1)$ $P(z_1)$ $R(z_1,z_2)$ $P(z_2)$ $R(z_2,z_3)$ … $P(z_k)$ $R(z_k,z_{k+1})$ …
✗	✗	✗		✗		✓

Key Idea: Focus on a representative,
a model that is as general as possible

Universal Models (a.k.a. Canonical Models)



An instance U is a universal model of $D \wedge \Sigma$ if the following holds:

1. U is a model of $D \wedge \Sigma$

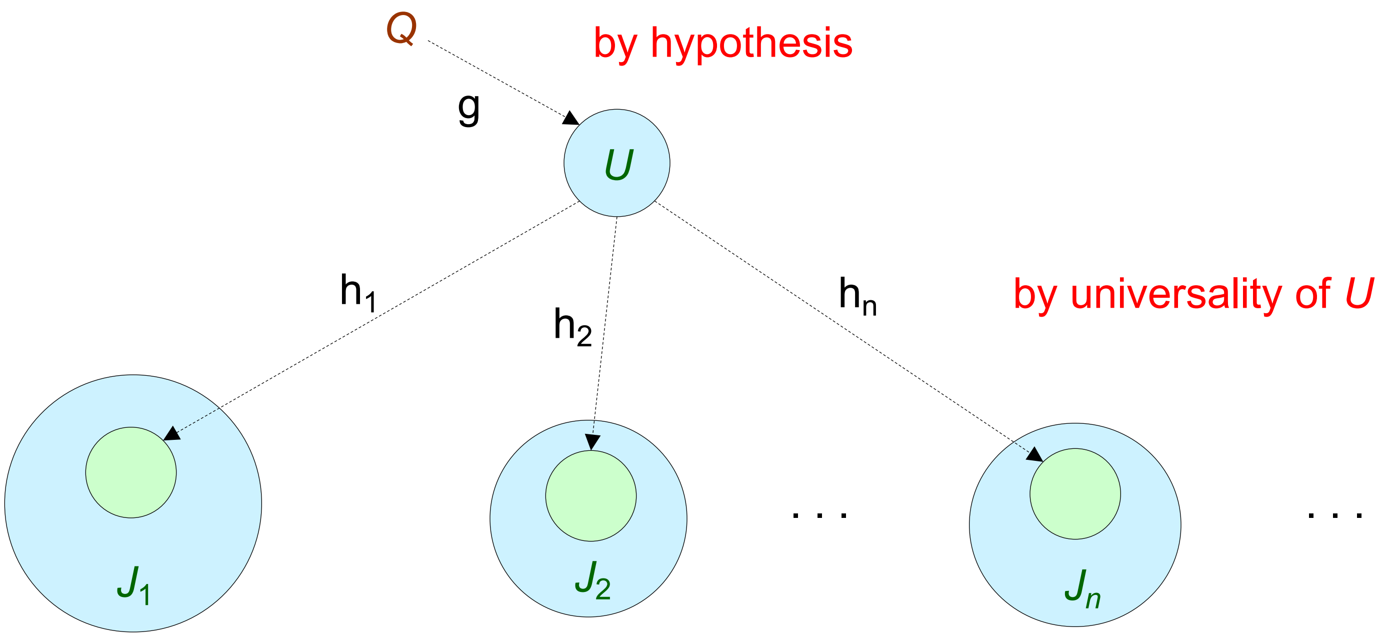
2. $\forall J \in \text{models}(D \wedge \Sigma)$, there exists a homomorphism h_J such that $h_J(U) \subseteq J$

Query Answering via Universal Models

Theorem: $D \wedge \Sigma \models Q$ iff $U \models Q$, where U is a universal model of $D \wedge \Sigma$

Proof: (⇒) Trivial since, for every $J \in$ models($D \wedge \Sigma$), $J \models Q$

(⇐) By exploiting the universality of U



$\forall J \in$ models($D \wedge \Sigma$), $\exists h_J$ such that $h_J(g(Q)) \subseteq J$ $\Rightarrow$ $\forall J \in$ models($D \wedge \Sigma$), $J \models Q$

$\Rightarrow$ $D \wedge \Sigma \models Q$

The Chase Procedure

- Fundamental algorithmic tool used in databases

- It has been applied to a wide range of problems:
    - Checking containment of queries under constraints
    - Computing data exchange solutions
    - Computing certain answers in data integration settings
    - …

… what's the reason for the ubiquity of the chase in databases?

it constructs universal models

The Chase Procedure

D

person(Alice)

Σ

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

chase(D,Σ) = D ∪

The Chase Procedure

D

person(Alice)

Σ

$\forall X\ (\mathit{Person}(X) \rightarrow \exists Y\ (\mathit{hasParent}(X,Y) \land \mathit{Person}(Y)))$

chase(D,Σ) = $D \cup \{\mathit{hasParent}(\text{Alice}, z_1), \mathit{Person}(z_1)$

The Chase Procedure

D

person(Alice)

Σ

$\forall X\, (Person(X) \rightarrow \exists Y\, (hasParent(X,Y) \wedge Person(Y)))$

$chase(D,\Sigma) = D \cup \{hasParent(Alice, z_1), Person(z_1),$

$hasParent(z_1, z_2), Person(z_2)$

The Chase Procedure

D

person(Alice)

Σ

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

$\mathrm{chase}(D,\Sigma) = D \cup \{\mathit{hasParent}(\mathrm{Alice}, z_1), \mathit{Person}(z_1),$

$\mathit{hasParent}(z_1, z_2), \mathit{Person}(z_2),$

$\mathit{hasParent}(z_2, z_3), \mathit{Person}(z_3)$

The Chase Procedure

D

person(Alice)

Σ

∀X (*Person*(X) → ∃Y (*hasParent*(X,Y) ∧ *Person*(Y)))

chase(*D*,Σ) = *D* ∪ {*hasParent*(Alice, z_1), *Person*(z_1),

hasParent(z_1, z_2), *Person*(z_2),

hasParent(z_2, z_3), *Person*(z_3), …

infinite instance

The Chase Procedure: Formal Definition

- Chase rule - the building block of the chase procedure

- A rule $\sigma = \forall \mathbf{X} \forall \mathbf{Y}\, (\varphi(\mathbf{X},\mathbf{Y}) \rightarrow \exists \mathbf{Z}\, \psi(\mathbf{X},\mathbf{Z}))$ is applicable to instance J if:

  1. There exists a homomorphism h such that $h(\varphi(\mathbf{X},\mathbf{Y})) \subseteq J$
  2. There is no $g \supseteq h_{|\mathbf{X}}$ such that $g(\psi(\mathbf{X},\mathbf{Z})) \subseteq J$

$J = \{R(a), P(a,b)\}$

$h = \{X \rightarrow a\}$ $g = \{X \rightarrow a, Y \rightarrow b\}$

$\forall X\, (R(X) \rightarrow \exists Y\, P(X,Y))$

✗

$J = \{R(a), P(b,a)\}$

$h = \{X \rightarrow a\}$ ✗

$\forall X\, (R(X) \rightarrow \exists Y\, P(X,Y))$

✓

The Chase Procedure: Formal Definition

- Chase rule - the building block of the chase procedure

- A rule $\sigma = \forall \mathbf{X} \forall \mathbf{Y} (\varphi(\mathbf{X},\mathbf{Y}) \rightarrow \exists \mathbf{Z}\ \psi(\mathbf{X},\mathbf{Z}))$ is applicable to instance J if:
  1. There exists a homomorphism h such that $h(\varphi(\mathbf{X},\mathbf{Y})) \subseteq J$
  2. There is no $g \supseteq h_{|\mathbf{X}}$ such that $g(\psi(\mathbf{X},\mathbf{Z})) \subseteq J$

- Let $J_+ = J \cup \{g(\psi(\mathbf{X},\mathbf{Z}))\}$, where $g \supseteq h_{|\mathbf{X}}$ and $g(\mathbf{Z})$ are “fresh” nulls not in J

- The result of applying σ to J is J_+, denoted $J\langle\sigma,h\rangle J_+$ - single chase step

The Chase Procedure: Formal Definition

- A finite chase of D w.r.t. Σ is a finite sequence

$$D\langle\sigma_1,h_1\rangle J_1\langle\sigma_2,h_2\rangle J_2\langle\sigma_3,h_3\rangle J_3 \ \dots\ \langle\sigma_n,h_n\rangle J_n$$

and chase(D,Σ) is defined as the instance J_n

all applicable rules will eventually be applied

- An infinite chase of D w.r.t. Σ is a fair finite sequence

$$D\langle\sigma_1,h_1\rangle J_1\langle\sigma_2,h_2\rangle J_2\langle\sigma_3,h_3\rangle J_3 \ \dots\ \langle\sigma_n,h_n\rangle J_n \dots$$

and chase(D,Σ) is defined as the instance $\cup_{k \geq 0} J_k$ (with $J_0 = D$)

least fixpoint of a monotonic operator - chase step

Chase: A Universal Model

Theorem: chase(D,Σ) is a universal model of $D \wedge \Sigma$

the result of the chase after k applications of the chase step

Proof:

- By construction, chase(D,Σ) $\in$ models($D \wedge \Sigma$)
- It remains to show that chase(D, Σ) can be homomorphically embedded into every other model of $D \wedge \Sigma$
- Fix an arbitrary instance $J \in$ models($D \wedge \Sigma$). We need to show that there exists h such that h(chase(D,Σ)) $\subseteq J$
- By induction on the number of applications of the chase step, we show that for every $k \geq 0$, there exists h_k such that $h_k(\text{chase}^{[k]}(D,\Sigma)) \subseteq J$, and h_k is compatible with h_{k-1}
- Clearly, $\cup_{k \geq 0}\, h_k$ is a well-defined homomorphism that maps chase(D,Σ) to J
- The claim follows with $h = \cup_{k \geq 0}\, h_k$

Chase: Uniqueness Property

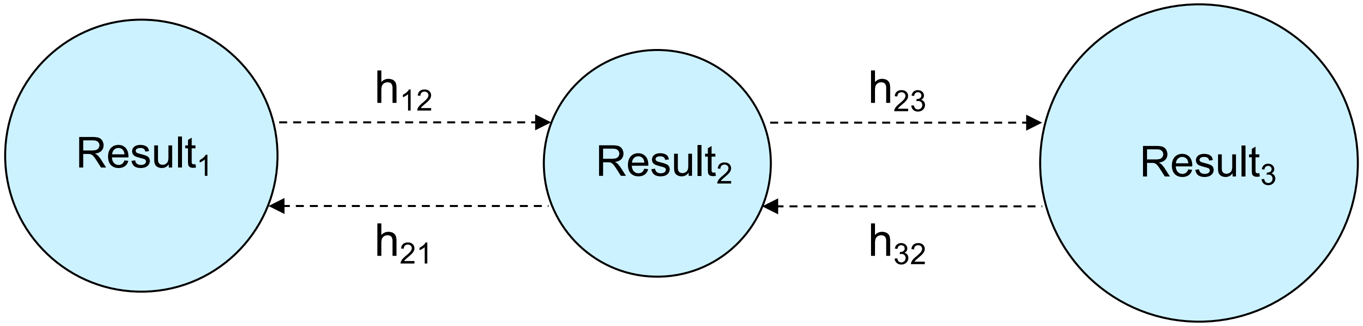
- The result of the chase is not unique - depends on the order of rule application

$D = \{P(a)\}$ $\sigma_1 = \forall X\, (P(X) \rightarrow \exists Y\, R(Y))$ $\sigma_2 = \forall X\, (P(X) \rightarrow R(X))$

$\text{Result}_1 = \{P(a), R(z), R(a)\}$ σ_1 then σ_2

$\text{Result}_2 = \{P(a), R(a)\}$ σ_2 then σ_1

- But, it is unique up to homomorphic equivalence



- Thus, it is unique for query answering purposes

Query Answering via the Chase

Theorem: $D \wedge \Sigma \models Q$ iff $U \models Q$, where U is a universal model of $D \wedge \Sigma$

+

Theorem: chase(D, Σ) is a universal model of $D \wedge \Sigma$

=

Corollary: $D \wedge \Sigma \models Q$ iff chase(D,Σ) $\models Q$

- We can tame the first dimension of infinity by exploiting the chase procedure
- But, what about the second dimension of infinity? - the chase may be infinite

Rest of the Lectrure

- Undecidability of BCQ-Answering
- Gaining decidability - terminating chase
- Full Existential Rules
- Acyclic Existential Rules

Undecidability of BCQ-Answering

Theorem: BCQ-Answering is undecidable

Proof : By simulating a deterministic Turing machine with an empty tape

Deterministic Turing Machine (DTM)

$S\setminus\{s_{acc}\} \times \Lambda \rightarrow S \times \Lambda \times \{-1,0,+1\}$

accepting state

$M \;=\; (S, \Lambda, \sqcup, \delta, s_0, s_{acc})$

states

tape symbols

blank symbol

initial state

$\delta(s_1, \alpha) = (s_2, \beta, +1)$

IF at some time instant τ the machine is in sate s_1, the cursor points to cell κ, and this cell contains α

THEN at instant $\tau+1$ the machine is in state s_2, cell κ contains β, and the cursor points to cell $\kappa+1$

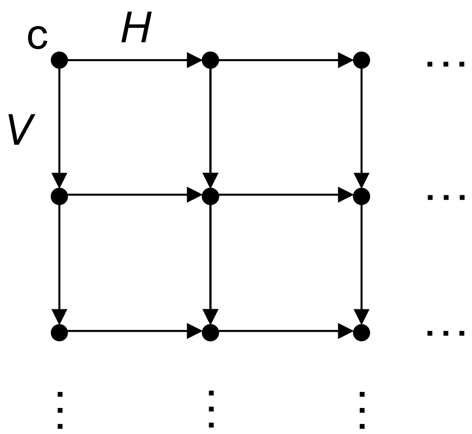
Undecidability of BCQ-Answering

Our Goal: Encode the computation of a DTM M with an empty tape

using a database D, a set Σ of existential rules, and a BCQ Q such that

$D \wedge \Sigma \models Q$ iff M accepts

Build an Infinite Grid



k-th horizontal line represents the

k-th configuration of the machine

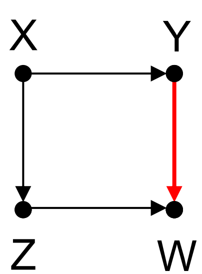
$\forall X\ (Start(X) \rightarrow Node(X) \land Initial(X))$

D = {*Start*(c)}

fixes the origin of the grid

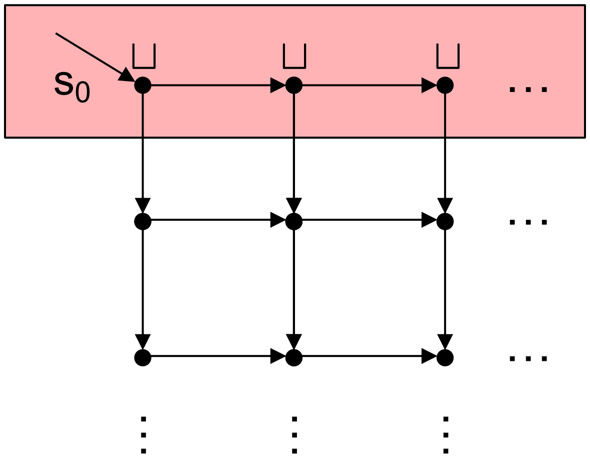
$\forall X\ (Node(X) \rightarrow \exists Y\ (H(X,Y) \land Node(Y)))$

$\forall X\ (Node(X) \rightarrow \exists Y\ (V(X,Y) \land Node(Y)))$



$\forall X \forall Y \forall Z \forall W\ (H(X,Y)\ H(Z,W)\ V(X,Z) \rightarrow V(Y,W))$

Initialization Rules

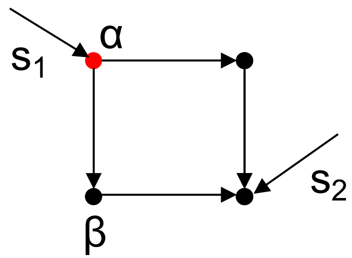


$$\forall X \forall Y\ (\mathit{Initial}(X) \land H(X,Y) \rightarrow \mathit{Initial}(Y))$$

$$\forall X\ (\mathit{Start}(X) \rightarrow \mathit{Cursor}[s_0](X))$$

$$\forall X\ (\mathit{Initial}(X) \rightarrow \mathit{Symbol}[\sqcup](X))$$

Transition Rules

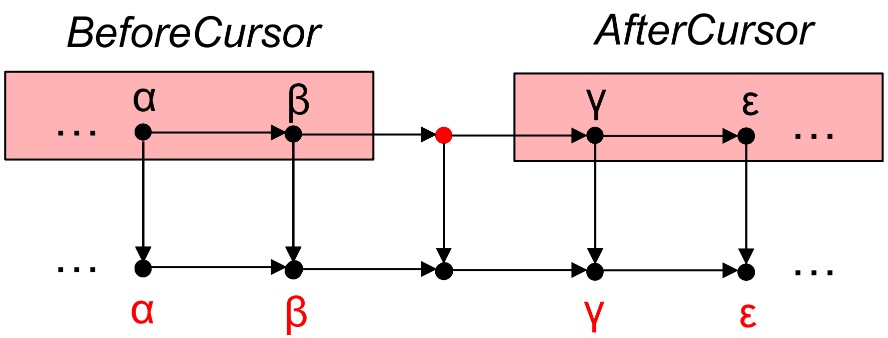


$$\delta(s_1,\alpha) \;=\; (s_2,\beta,+1)$$

$$\forall X \forall Y \forall Z\, (\mathit{Cursor}[s_1](X) \wedge \mathit{Symbol}[\alpha](X) \wedge V(X,Y) \wedge H(Y,Z) \rightarrow$$

$$\mathit{Cursor}[s_2](Z) \wedge \mathit{Symbol}[\beta](Y) \wedge \mathit{Mark}(X))$$

Inertia Rules



$\forall X \forall Y\ (\mathit{Mark}(X) \wedge H(X,Y) \rightarrow \mathit{AfterCursor}(Y))$

$\forall X \forall Y\ (\mathit{AfterCursor}(X) \wedge H(X,Y) \rightarrow \mathit{AfterCursor}(Y))$

$\forall X \forall Y\ (\mathit{AfterCursor}(X) \wedge \mathit{Symbol}[\alpha](X) \wedge V(X,Y) \rightarrow \mathit{Symbol}[\alpha](Y))$

…we have similar rules for the cells before the cursor

Accepting Rule

Once we reach the accepting state we accept

$$\forall X\ (\mathit{Cursor}[s_{acc}](X) \rightarrow \mathit{Accept}(X))$$

$D \land \Sigma \vDash \exists X\ \mathit{Accept}(X)$ iff the DTM *M* accepts

Undecidability of BCQ-Answering

Theorem: BCQ-Answering is undecidable

Proof : By simulating a deterministic Turing machine with an empty tape

…syntactic restrictions are needed!!!