Computing Cores for Existential Rules
with the Standard Chase and ASP

Markus Krötzsch
Knowledge-Based Systems, TU Dresden

TECHNISCHE UNIVERSITÄT DRESDEN

Rules are simple, but what do they mean?

$$R1: \quad \text{father}(x, y) \rightarrow \text{male}(y)$$

$$R2: \quad \text{person}(x) \rightarrow \exists v.\text{father}(x, v) \wedge \text{male}(v)$$

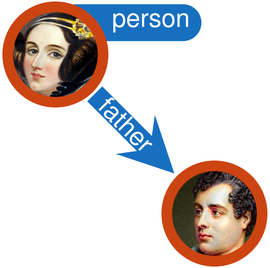
$$\text{person}(\text{ada}) \qquad \text{father}(\text{ada}, \text{george})$$

Rules are simple, but what do they mean?

$$R1: \quad \text{father}(x, y) \to \text{male}(y)$$

$$R2: \quad \text{person}(x) \to \exists v.\text{father}(x, v) \land \text{male}(v)$$

person(ada) father(ada, george)

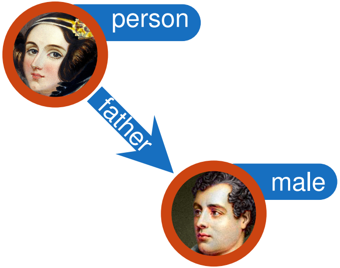


Rules are simple, but what do they mean?

$$R1: \quad \text{father}(x, y) \rightarrow \text{male}(y)$$

$$R2: \quad \text{person}(x) \rightarrow \exists v.\text{father}(x, v) \wedge \text{male}(v)$$

person(ada) father(ada, george)

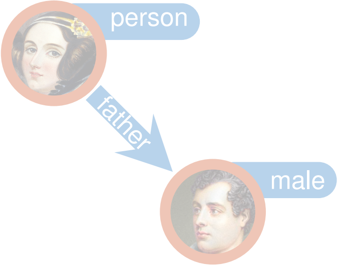


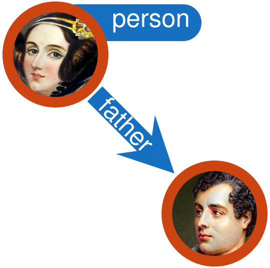
Rules are simple, but what do they mean?

$$R1: \quad \text{father}(x, y) \to \text{male}(y)$$

$$R2: \quad \text{person}(x) \to \exists v.\text{father}(x, v) \land \text{male}(v)$$

person(ada) father(ada, george)





Rules are simple, but what do they mean?

$R1$: $\text{father}(x, y) \rightarrow \text{male}(y)$

$R2$: $\text{person}(x) \rightarrow \exists v.\text{father}(x, v) \wedge \text{male}(v)$

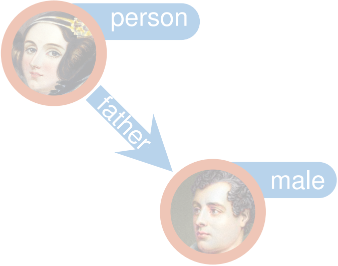
person(ada) father(ada, george)

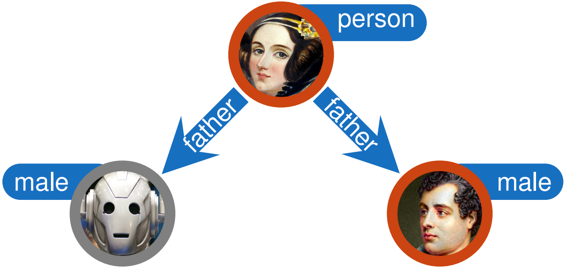
Rules are simple, but what do they mean?

$$R1: \quad \text{father}(x, y) \to \text{male}(y)$$

$$R2: \quad \text{person}(x) \to \exists v.\text{father}(x, v) \wedge \text{male}(v)$$

person(ada) father(ada, george)



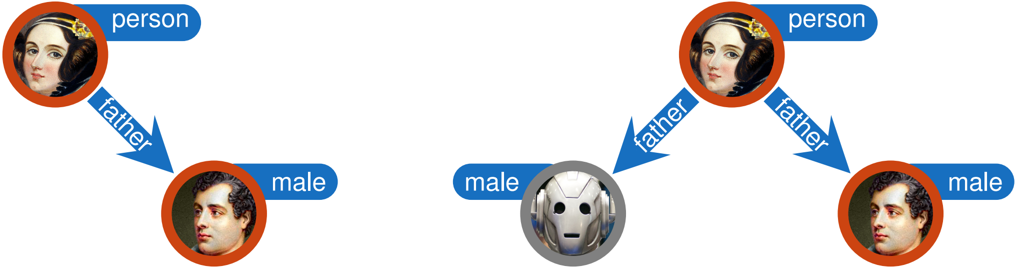


Rules are simple, but what do they mean?

$$R1: \quad \text{father}(x, y) \rightarrow \text{male}(y)$$

$$R2: \quad \text{person}(x) \rightarrow \exists v.\text{father}(x, v) \wedge \text{male}(v)$$

person(ada) father(ada, george)



The Core

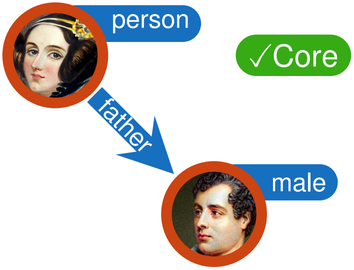
Simplification: We will only talk about finite chases here.

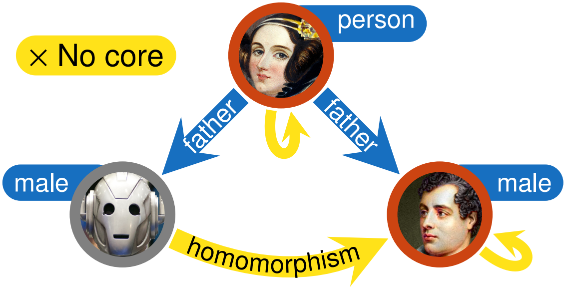
A **core** is a finite structure C where every homomorphism $C \to C$ is an isomorphism.

The Core

Simplification: We will only talk about finite chases here.

A **core** is a finite structure C where every homomorphism $C \to C$ is an isomorphism.





Cores in Practice

The core is the "best among all universal solutions"
– Fagin, Kolaitis, and Popa 2005

- Can be computed effectively
- Possible during the chase: "core chase"

Cores in Practice

The core is the "best among all universal solutions"
– Fagin, Kolaitis, and Popa 2005

- Can be computed effectively
- Possible during the chase: "core chase"

And yet: No current system implements the core chase!

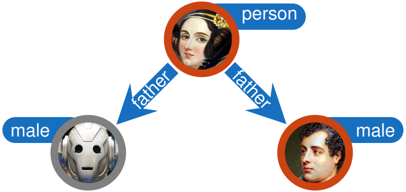
Problem: Computing the core takes exponential time in the size of the chase.

Cores from the Standard Chase

Idea: Couldn't we get cores with the standard chase?

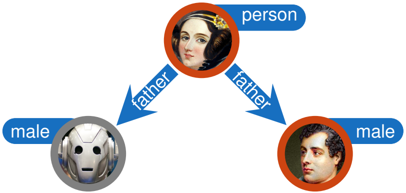
Idea: Couldn't we get cores with the standard chase?

Analysis: What went wrong here?



Idea: Couldn't we get cores with the standard chase?

Analysis: What went wrong here?



- We applied rule $R2$ to a **match**:

  person(ada) → father(ada, $null$) ∧ male($null$)

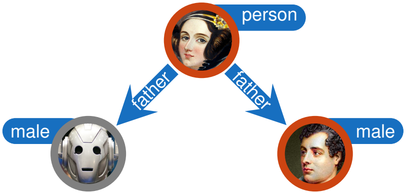
- In the final chase, this instance is satisfied by an **alternative match**:

  person(ada) → father(ada, george) ∧ male(george)

Cores from the Standard Chase

Idea: Couldn't we get cores with the standard chase?

Analysis: What went wrong here?



- We applied rule $R2$ to a **match**:

  person(ada) $\to$ father(ada, $null$) $\wedge$ male($null$)

- In the final chase, this instance is satisfied by an **alternative match**:

  person(ada) $\to$ father(ada, george) $\wedge$ male(george)

Theorem:
Every chase without alternative matches yields a core.

A Characterisation in ASP

Idea: Characterise alternative-match-free standard chases in ASP.

A Characterisation in ASP

Idea: Characterise alternative-match-free standard chases in ASP.

Encoding:

- Use terms with (skolem) function symbols instead of named nulls
- Augment rules with precondition that they are “not blocked”
- Add rules that derive that a rule is “blocked” when an alternative match is found

A Characterisation in ASP

Idea: Characterise alternative-match-free standard chases in ASP.

Encoding:

- Use terms with (skolem) function symbols instead of named nulls
- Augment rules with precondition that they are “not blocked”
- Add rules that derive that a rule is “blocked” when an alternative match is found

Theorem: Cores from a chase without alternative matches correspond to the stable models of suitable normal logic programs.

Can we guide the standard chase to produce a core?

Core Stratification:

- Define $R1 \prec^{\Box} R2$ to mean "$R1$ could produce structures that enable alternative matches for $R2$"
- Stratify the order of rule applications w.r.t. $\prec^{\Box}$ (together with a more usual positive "dependency" $\prec^{+}$)

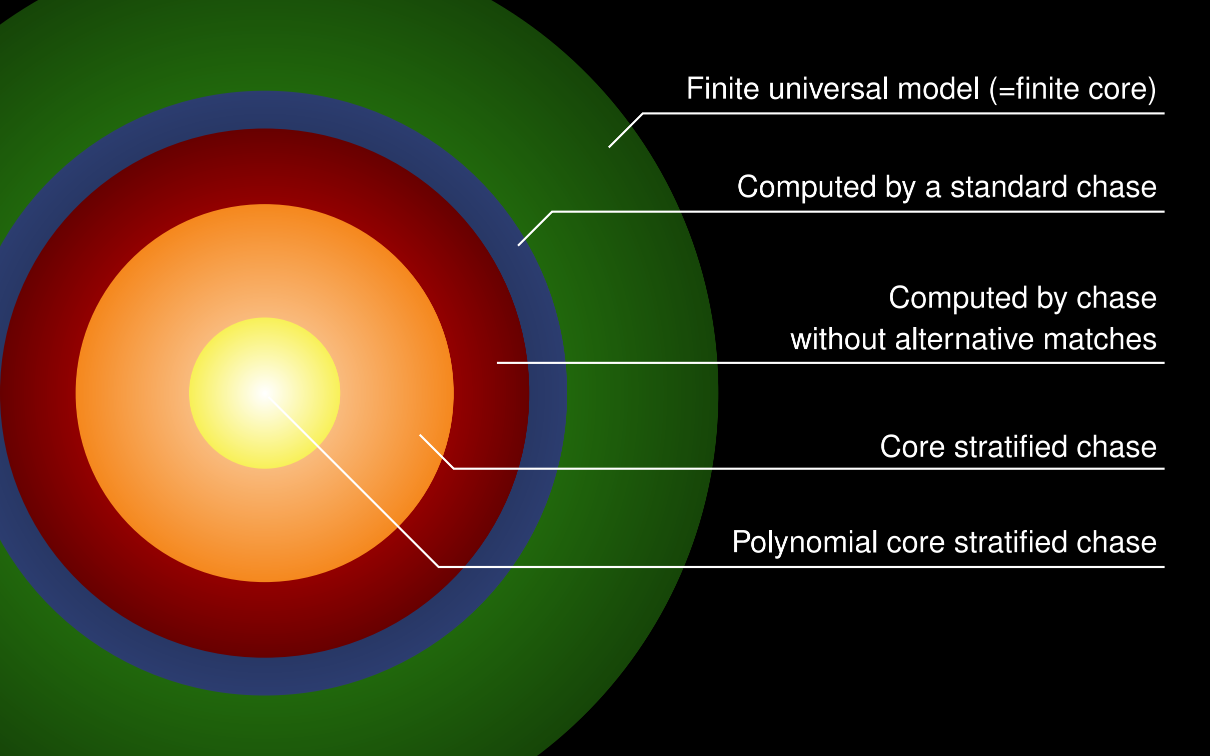
Can we guide the standard chase to produce a core?

Core Stratification:

- Define $R1 \prec^{\square} R2$ to mean "$R1$ could produce structures that enable alternative matches for $R2$"
- Stratify the order of rule applications w.r.t. $\prec^{\square}$ (together with a more usual positive "dependency" $\prec^{+}$)

Results:

- Core stratification of a rule set can be decided in Σ_2^P.
- If a chase is core stratified, then it has no alternative matches (and therefore yields a core).


Finite universal model (=finite core)
Computed by a standard chase
Computed by chase
without alternative matches
Core stratified chase
Polynomial core stratified chase

Existentials and Negation

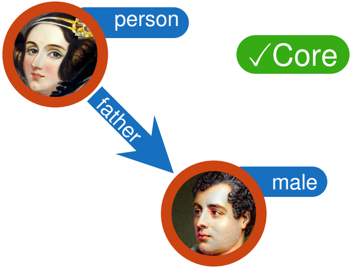
A (classically) stratified logic program:

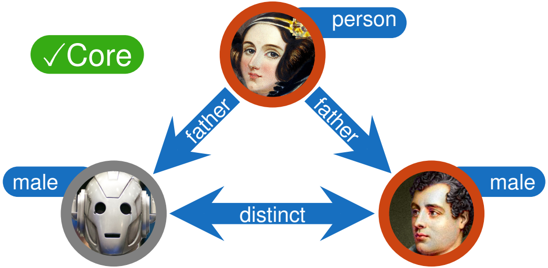
$$
\begin{aligned}
R1: &\quad \mathsf{father}(x,y) \to \mathsf{male}(y) \\
R2: &\quad \mathsf{person}(x) \to \exists v.\mathsf{father}(x,v) \wedge \mathsf{male}(v) \\
R3: &\quad \mathsf{father}(x,y) \to \mathsf{equals}(y,y) \\
R4: &\quad \mathsf{father}(x,y_1) \wedge \mathsf{father}(x,y_2) \wedge \\
&\quad \textbf{not}\ \mathsf{equals}(y_1,y_2) \to \mathsf{distinct}(y_1,y_2)
\end{aligned}
$$

Existentials and Negation

A (classically) stratified logic program:

$$R1: \quad \text{father}(x, y) \to \text{male}(y)$$
$$R2: \quad \text{person}(x) \to \exists v.\text{father}(x, v) \wedge \text{male}(v)$$
$$R3: \quad \text{father}(x, y) \to \text{equals}(y, y)$$
$$R4: \quad \text{father}(x, y_1) \wedge \text{father}(x, y_2) \wedge \textbf{not}\ \text{equals}(y_1, y_2) \to \text{distinct}(y_1, y_2)$$





Idea: Combine core stratification & classical stratification.

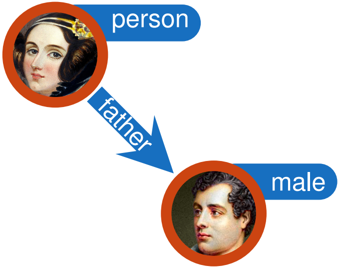
$\leadsto$ "Full stratification"

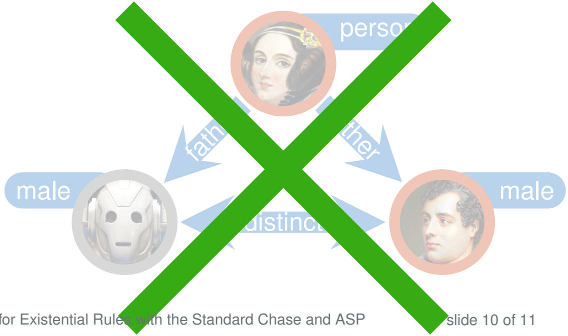
Perfect Core Models

Idea: Combine core stratification & classical stratification.

$\leadsto$ "Full stratification"

Theorem: A finite, fully stratified chase yield a unique stable model that is a core, the perfect core model.





Main insight: Cores are in reach for practical uses

- Existing ASP engines can compute them
- Existing chase implementations can compute them
- Cores could be key to mix existentials and non-monotonic negation

Next questions:

- How do practical implementations perform?
- Is core stratification common in practice?
- Can we generalise perfect core models?

TECHNISCHE UNIVERSITÄT DRESDEN