Thomas Feller, Tim Lyon, Piotr Ostropolski-Nalewaja, Sebastian Rudolph
Faculty of Computer Science, Institute of Artificial Intelligence, Computational Logic Group

Finite-Cliquewidth Sets of Existential Rules

Toward a General Criterion for Decidable yet Highly Expressive Querying // ICDT 2023, Ioannina, 29.03.2023

Introduction: Existential Rules

Existential Rules are sentences of first-order logic of a particular shape:

Introduction: Existential Rules

Existential Rules are sentences of first-order logic of a particular shape:

$$\forall x.\mathtt{Person}(x) \to \exists y.\mathtt{Person}(y) \land \mathtt{MotherOf}(y, x)$$

Introduction: Existential Rules

Existential Rules are sentences of first-order logic of a particular shape:

$$\forall x.\mathtt{Person}(x) \rightarrow \exists y.\mathtt{Person}(y) \land \mathtt{MotherOf}(y,x)$$

body

Introduction: Existential Rules

Existential Rules are sentences of first-order logic of a particular shape:

$$\forall x.\mathtt{Person}(x) \rightarrow \exists y.\mathtt{Person}(y) \land \mathtt{MotherOf}(y, x)$$

body **head**

Introduction: Existential Rules

Existential Rules are sentences of first-order logic of a particular shape:

$$\forall x.\mathtt{Person}(x) \rightarrow \exists y.\mathtt{Person}(y) \wedge \mathtt{MotherOf}(y,x)$$

body **head**

Introduction: Existential Rules

Existential Rules are sentences of first-order logic of a particular shape:

$$\forall x.\texttt{Person}(x) \rightarrow \exists y.\texttt{Person}(y) \wedge \texttt{MotherOf}(y, x)$$

body **head**

Introduction: Existential Rules

Existential Rules are sentences of first-order logic of a particular shape:

$$\forall x.\mathtt{Person}(x) \rightarrow \exists y.\mathtt{Person}(y) \wedge \mathtt{MotherOf}(y,x)$$

body **single-head**

Introduction: Existential Rules

Existential Rules are sentences of first-order logic of a particular shape:

$$\forall x.\mathtt{Person}(x) \rightarrow \exists y.\mathtt{Person}(y) \wedge \mathtt{MotherOf}(y, x)$$

body **single-head**

A ***ruleset*** $\mathcal{R}$ is a finite set of existential rules.

Introduction: Existential Rules

Existential Rules are sentences of first-order logic of a particular shape:

$$\forall x.\mathtt{Person}(x) \rightarrow \exists y.\mathtt{Person}(y) \wedge \mathtt{MotherOf}(y,x)$$

body **single-head**

A ***ruleset*** $\mathcal{R}$ is a finite set of existential rules.

An ***ontology*** $(\mathcal{D}, \mathcal{R})$ consists of a database $\mathcal{D}$ and a set of rules $\mathcal{R}$.

Introduction: Universal Models and Ontology Mediated Query Entailment

Introduction: Universal Models and Ontology Mediated Query Entailment

$\mathfrak{U}$ is a ***universal model*** if it maps into any model of the ontology.

Introduction: Universal Models and Ontology Mediated Query Entailment

$\mathcal{U}$ is a ***universal model*** if it maps into any model of the ontology.

Problem: Ontology Mediated Query Entailment

Input: Database $\mathcal{D}$, ruleset $\mathcal{R}$ and a boolean query q.

Question: Does q hold in every model of $(\mathcal{D}, \mathcal{R})$?

Introduction: Universal Models and Ontology Mediated Query Entailment

$\mathcal{U}$ is a ***universal model*** if it maps into any model of the ontology.

Problem: Ontology Mediated Query Entailment
Input: Database $\mathcal{D}$, ruleset $\mathcal{R}$ and a boolean query q.
Question: Does q hold in every model of $(\mathcal{D}, \mathcal{R})$?

Problem: Undecidable!

Introduction: Universal Models and Ontology Mediated Query Entailment

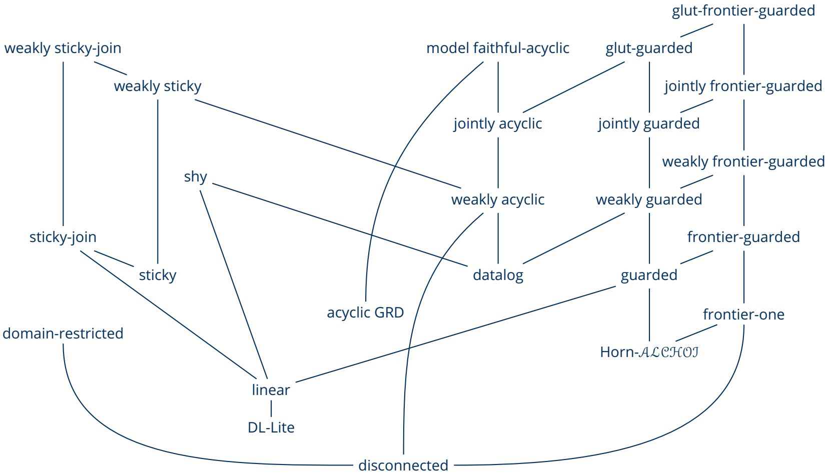
$\mathcal{U}$ is a ***universal model*** if it maps into any model of the ontology.

Problem: Ontology Mediated Query Entailment
Input: Database $\mathcal{D}$, ruleset $\mathcal{R}$ and a boolean query q.
Question: Does q hold in every model of $(\mathcal{D}, \mathcal{R})$?

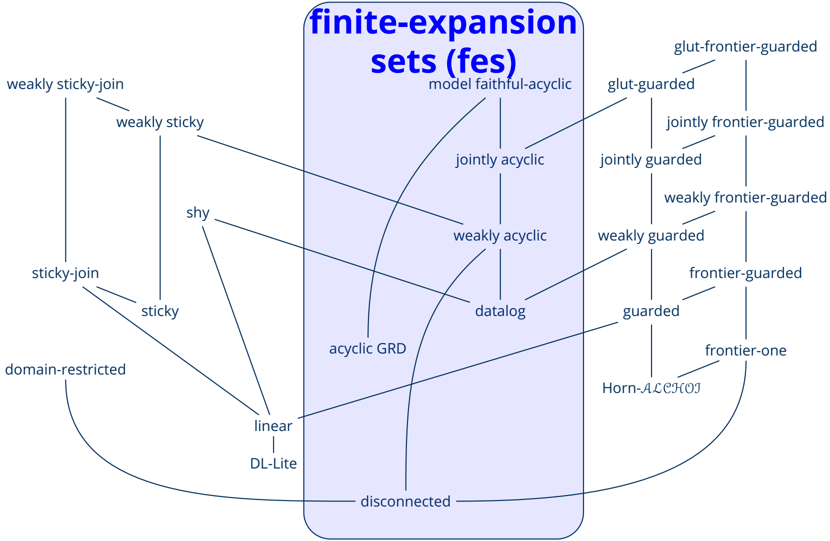
Problem: Undecidable!

Solution: Restricting rulesets.

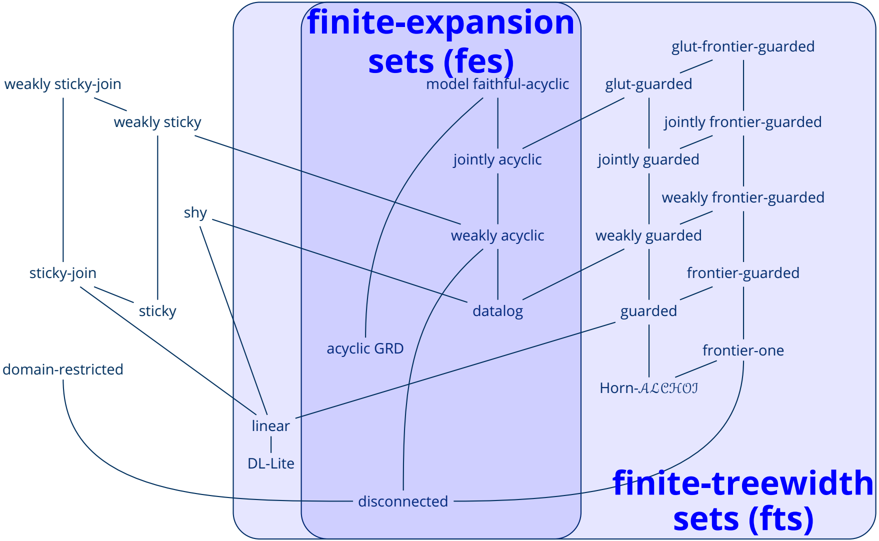
Zoo of Decidable Existential Rules



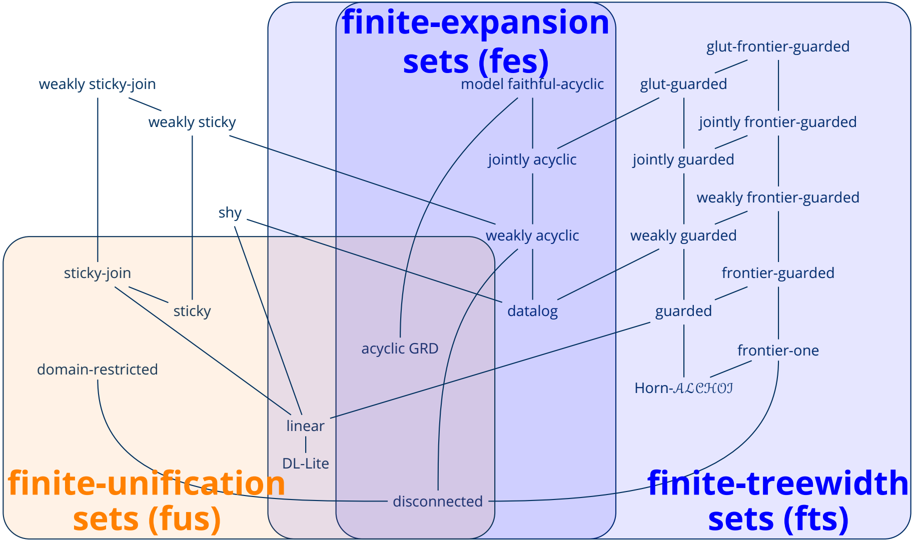
Zoo of Decidable Existential Rules



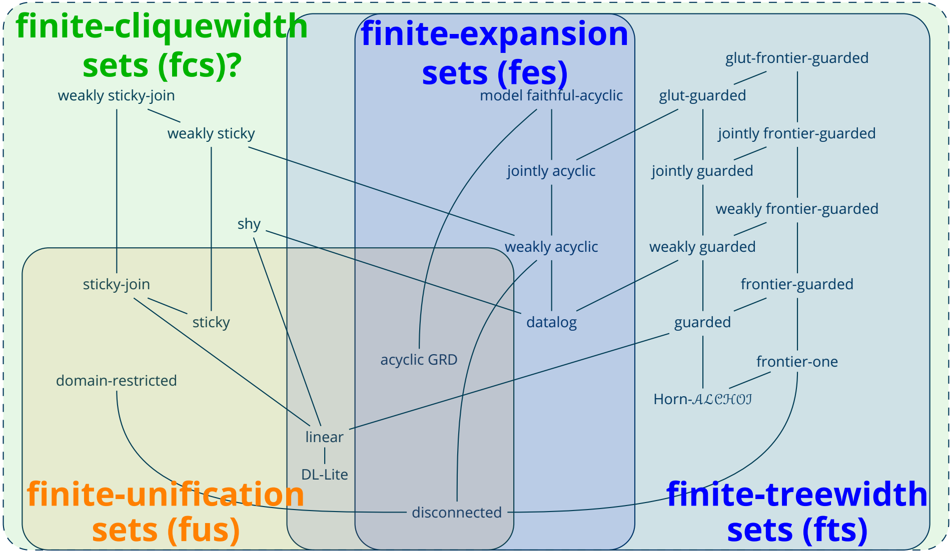
Zoo of Decidable Existential Rules



Zoo of Decidable Existential Rules



Zoo of Decidable Existential Rules



Example: fts, neither fes nor fus

$\forall x, y. A(x) \wedge E(x, y) \rightarrow A(y)$ $\forall x. A(x) \rightarrow \exists y. E(x, y)$

Example: fts, neither fes nor fus

$\forall x, y. A(x) \wedge E(x, y) \rightarrow A(y)$ $\qquad$ $\forall x. A(x) \rightarrow \exists y. E(x, y)$

Example: fts, neither fes nor fus

$\forall x, y. A(x) \land E(x, y) \rightarrow A(y)$

$\forall x. A(x) \rightarrow \exists y. E(x, y)$

Example: fts, neither fes nor fus

$\forall x, y. A(x) \land E(x, y) \rightarrow A(y)$

$\forall x. A(x) \rightarrow \exists y. E(x, y)$

Example: fts, neither fes nor fus

$\forall x, y. A(x) \land E(x, y) \rightarrow A(y)$

$\forall x. A(x) \rightarrow \exists y. E(x, y)$

Example: fts, neither fes nor fus

$\forall x, y. A(x) \land E(x, y) \rightarrow A(y)$

$\forall x. A(x) \rightarrow \exists y. E(x, y)$

Example: fts, neither fes nor fus

$\forall x, y . A(x) \wedge E(x, y) \rightarrow A(y)$

$\forall x . A(x) \rightarrow \exists y . E(x, y)$

Example: fts, neither fes nor fus

$\forall x, y. A(x) \land E(x, y) \to A(y)$

$\forall x. A(x) \to \exists y. E(x, y)$

Example: fts, neither fes nor fus

$\forall x, y.A(x) \land E(x, y) \to A(y)$

$\forall x.A(x) \to \exists y.E(x, y)$

Example: fts, neither fes nor fus

$$\forall x, y. A(x) \land E(x, y) \rightarrow A(y)$$

$$\forall x. A(x) \rightarrow \exists y. E(x, y)$$

Example: fts, neither fes nor fus

$\forall x, y. A(x) \land E(x, y) \rightarrow A(y)$

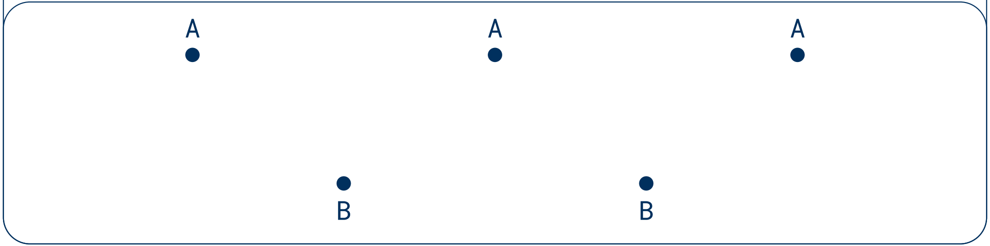
$\forall x. A(x) \rightarrow \exists y. E(x, y)$

Example: Concept Products, fus but not fts

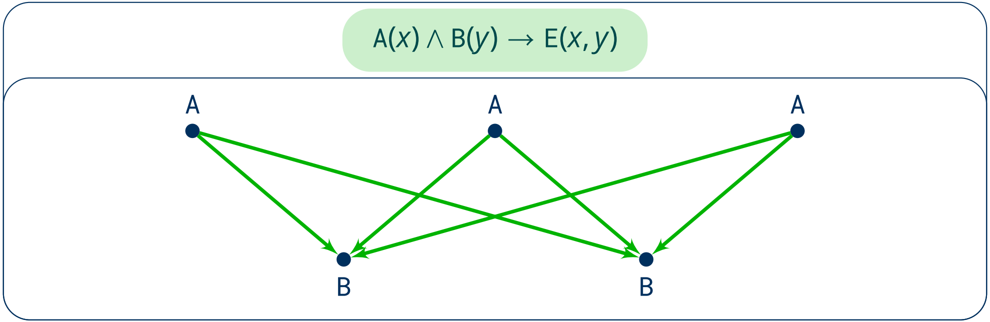
$$A(x) \wedge B(y) \rightarrow E(x, y)$$

Example: Concept Products, fus but not fts

$A(x) \wedge B(y) \rightarrow E(x, y)$

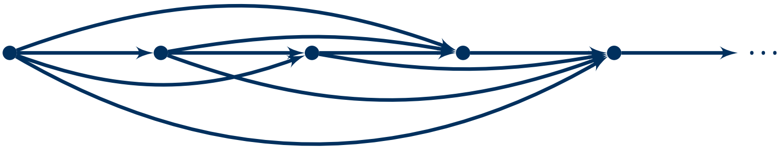


Example: Concept Products, fus but not fts



Example: Transitivity, neither fus nor fts!

$\forall x, y. E(x, y) \rightarrow \exists z. E(y, z)$ $\forall x, y. E(x, y) \wedge E(y, z) \rightarrow E(x, z)$



Cliquewidth

Cliquewidth is a width notion based on tree encodings.

Cliquewidth

Cliquewidth is a width notion based on tree encodings.

Those tree encodings are strongly connected to Monadic Second-Order Logic (MSO).

Cliquewidth

Cliquewidth is a width notion based on tree encodings.

Those tree encodings are strongly connected to Monadic Second-Order Logic (MSO).

Established by Courcelle for finite/infinite graphs and finite hypergraphs.

Cliquewidth

Cliquewidth is a width notion based on tree encodings.

Those tree encodings are strongly connected to Monadic Second-Order Logic (MSO).

Established by Courcelle for finite/infinite graphs and finite hypergraphs.

In this talk: Will be introduced by example for finite and infinite instances in the binary and in the higher arity setting.

Our Contributions

We adapt cliquewidth for **infinite** instances in the **higher arity** setting.

Our Contributions

We adapt cliquewidth for **infinite** instances in the **higher arity** setting.

↓

Based on cliquewidth, we define **fcs** as a class of **decidable** existential rules.

Our Contributions

We adapt cliquewidth for **infinite** instances in the **higher arity** setting.

↓

Based on cliquewidth, we define **fcs** as a class of **decidable** existential rules.

↓

We show that fcs **generalises** fts (with a detour: **Reification**)

Our Contributions

We adapt cliquewidth for **infinite** instances in the **higher arity** setting.

↓

Based on cliquewidth, we define **fcs** as a class of **decidable** existential rules.

↙ ↘

We show that fcs **generalises** fts (with a detour: **Reification**)

We prove that fcs **subsumes** binary single-head fus

Our Contributions

This talk!

We adapt cliquewidth for **infinite** instances in the **higher arity** setting.

↓

Based on cliquewidth, we define **fcs** as a class of **decidable** existential rules.

Take a peek into the paper!

We show that fcs **generalises** fts (with a detour: **Reification**)

We prove that fcs **subsumes** binary single-head fus

Cliquewidth: A finite and binary example

expression

instance

syntax tree

# of colors used

cliquewidth

Cliquewidth: A finite and binary example

expression

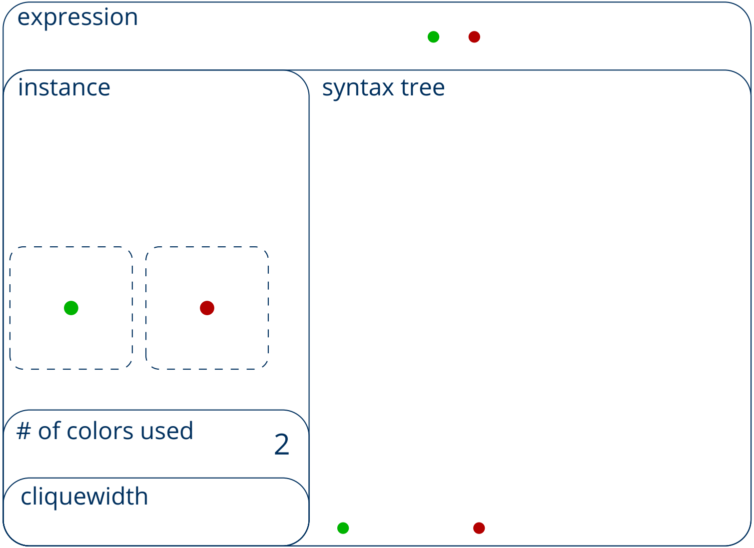
instance

syntax tree

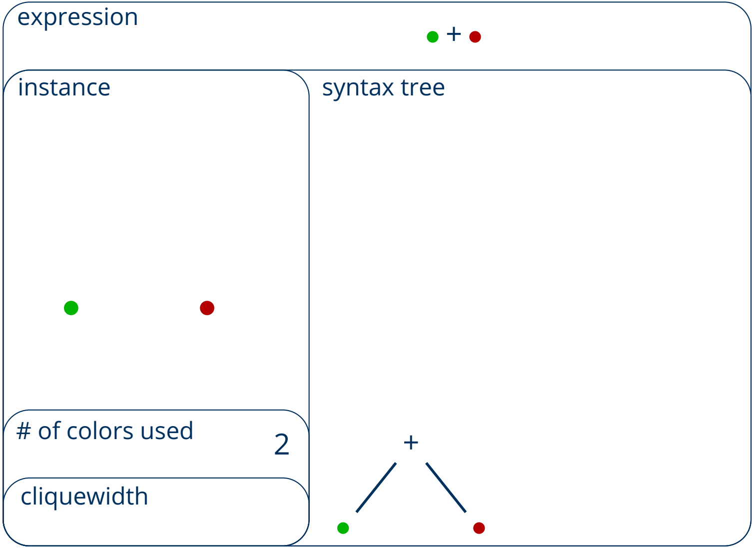
# of colors used 1

cliquewidth

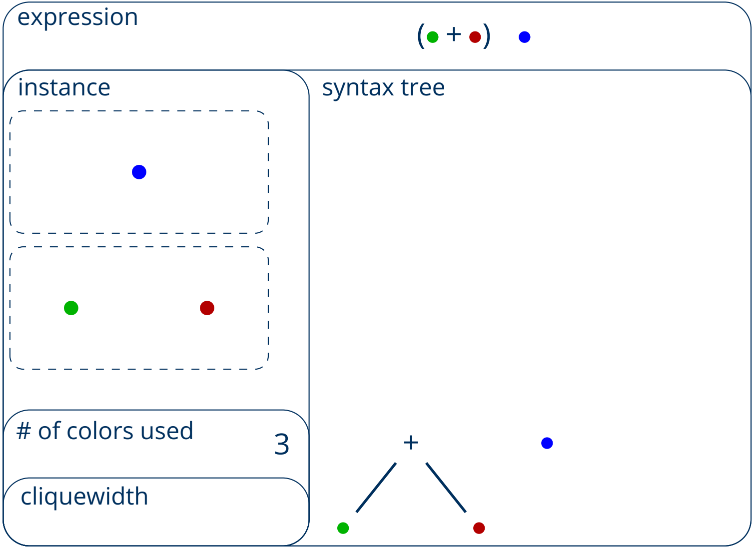
Cliquewidth: A finite and binary example



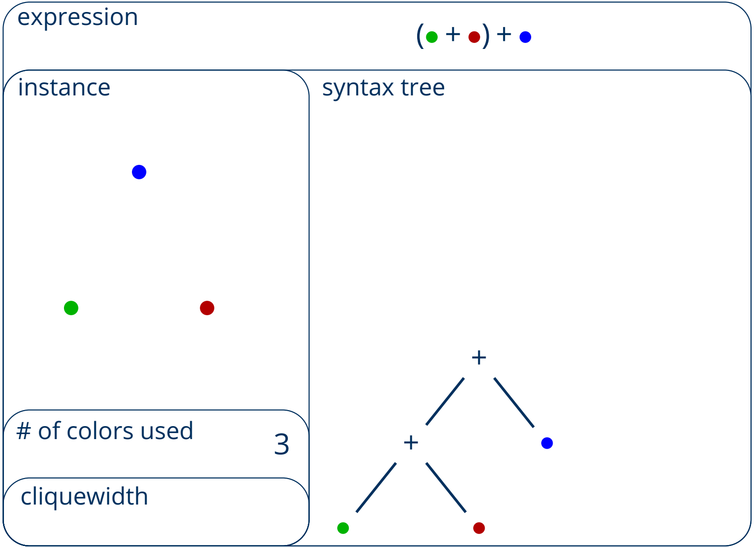
Cliquewidth: A finite and binary example



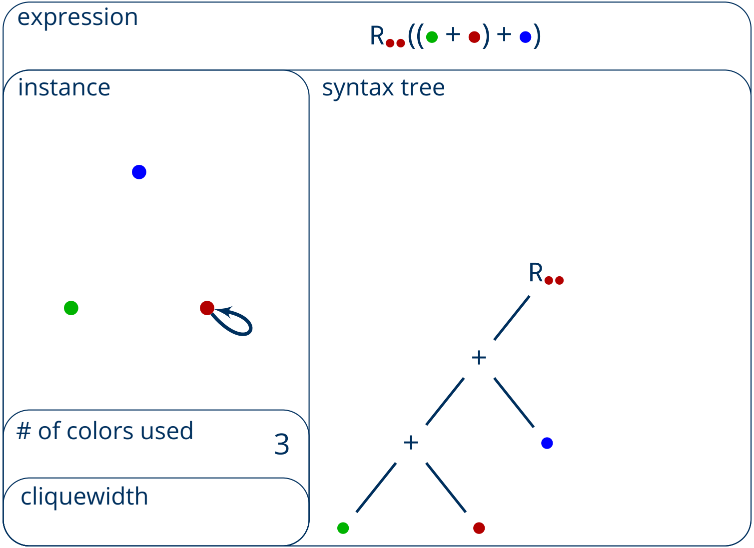
Cliquewidth: A finite and binary example



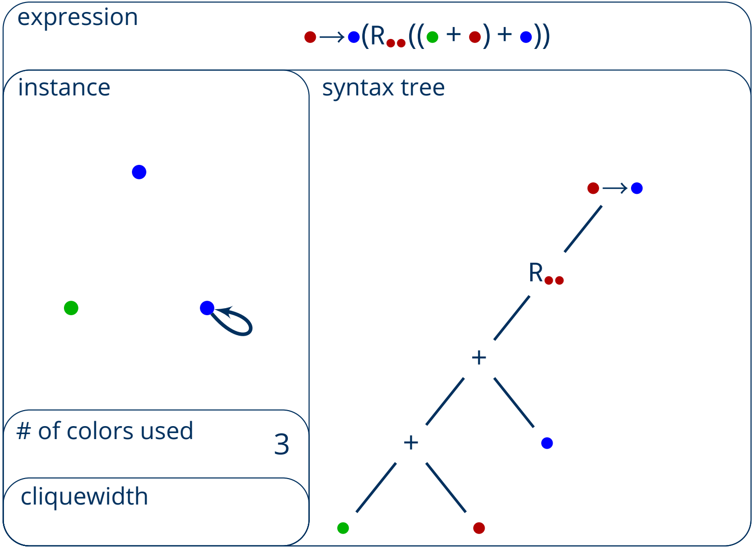
Cliquewidth: A finite and binary example



Cliquewidth: A finite and binary example



Cliquewidth: A finite and binary example

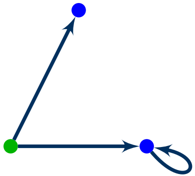


Cliquewidth: A finite and binary example

expression

R●●(●→●(R●●((● + ●) + ●)))

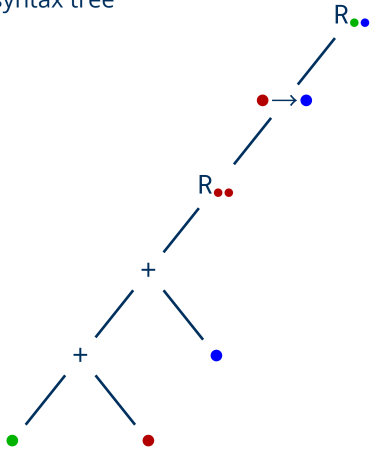
instance



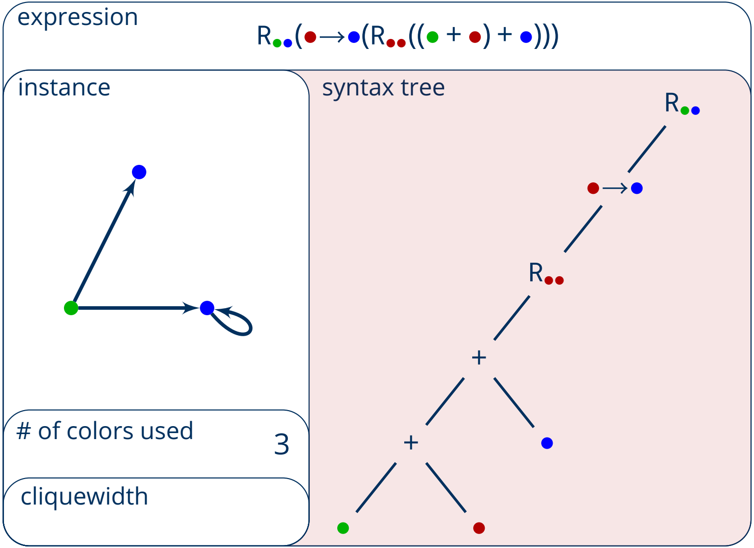
# of colors used 3

cliquewidth

syntax tree



Cliquewidth: A finite and binary example

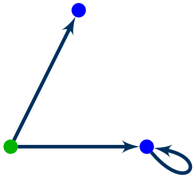


A finite, binary, and non-optimal example

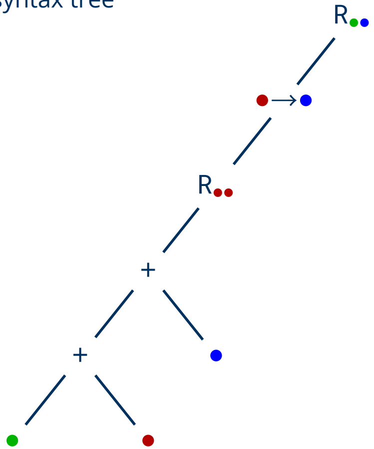
expression

R●●(●→●(R●●((● + ●) + ●)))

instance



syntax tree



# of colors used 3

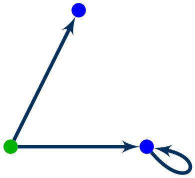
cliquewidth ≤ 3

A finite, binary, and optimal example

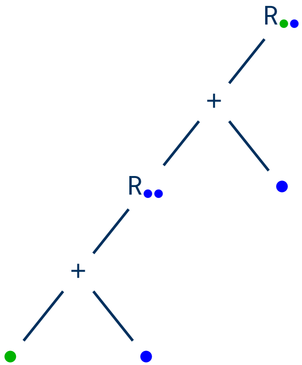
expression

R••((R••(• + •)) + •)

instance



syntax tree



# of colors used 2

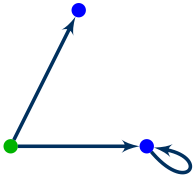
cliquewidth ≤ 2

A finite, binary, and optimal example

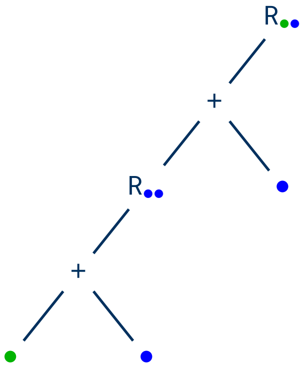
expression

R••((R••(• + •)) + •)

instance



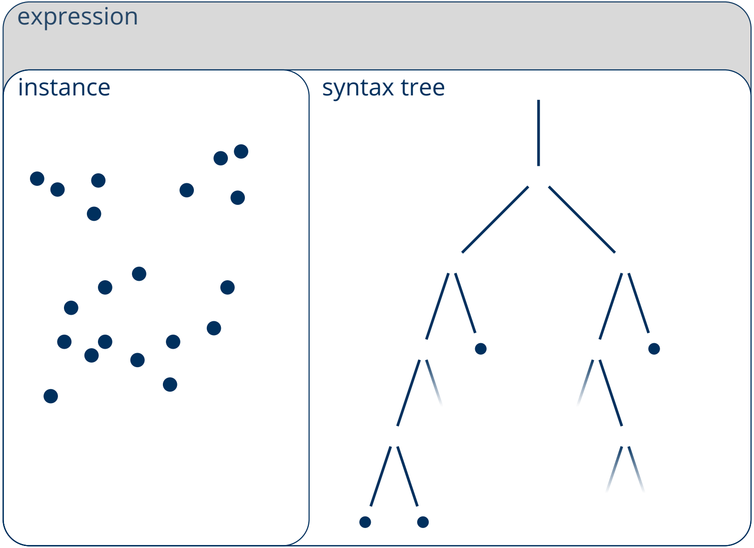
syntax tree



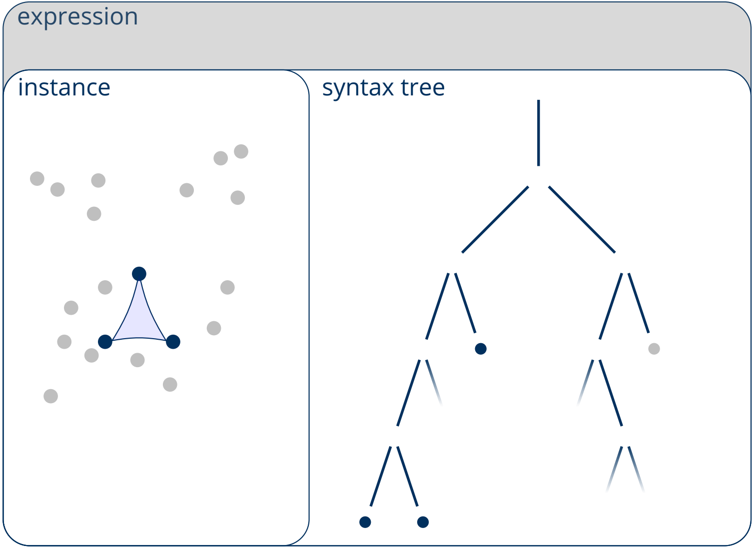
# of colors used 2

cliquewidth = 2

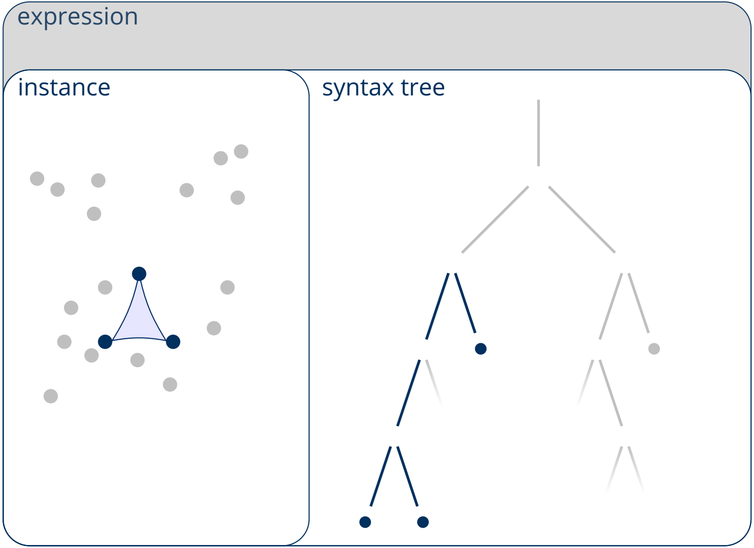
What about the infinite case?



What about the infinite case?

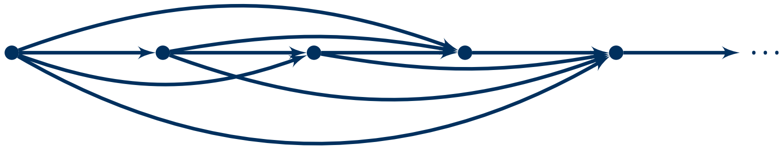


What about the infinite case?



Reminder: Transitivity, neither fus nor fts!

$\forall x, y.E(x, y) \rightarrow \exists z.E(y, z)$ $\quad$ $\forall x, y.E(x, y) \wedge E(y, z) \rightarrow E(x, z)$



An (infinite) instance of finite Cliquewidth

Transitive chain

$\mathbb{N}$ with $\mathrm{R}(n, m)$ iff $n < m$ has a cliquewidth of 2.

expression

instance

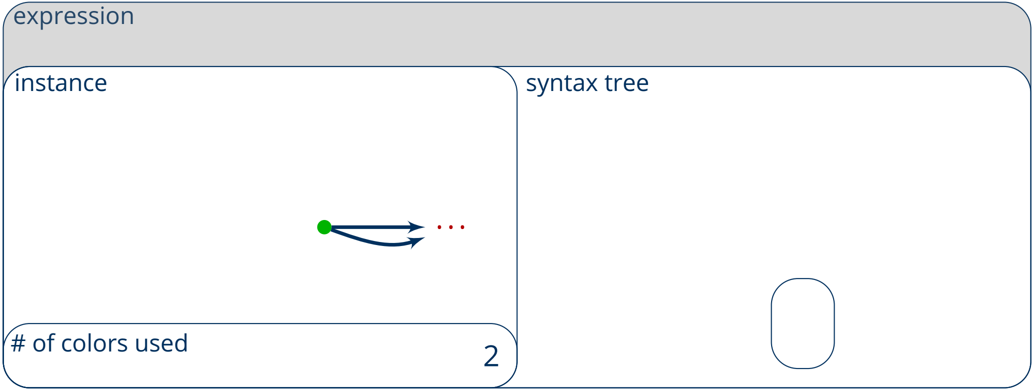
syntax tree

# of colors used

An (infinite) instance of finite Cliquewidth

Transitive chain

$\mathbb{N}$ with $\mathrm{R}(n, m)$ iff $n < m$ has a cliquewidth of 2.



An (infinite) instance of finite Cliquewidth

Transitive chain

$\mathbb{N}$ with $\mathrm{R}(n, m)$ iff $n < m$ has a cliquewidth of 2.

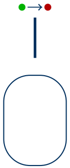
expression

instance

# of colors used

2

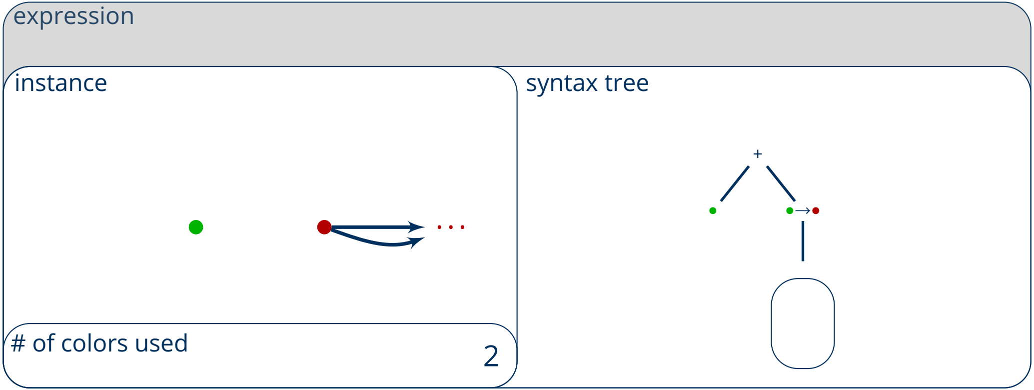
syntax tree



An (infinite) instance of finite Cliquewidth

Transitive chain

$\mathbb{N}$ with R(n, m) iff $n < m$ has a cliquewidth of 2.



An (infinite) instance of finite Cliquewidth

Transitive chain

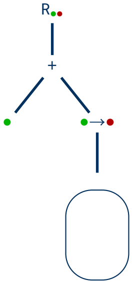
$\mathbb{N}$ with $R(n, m)$ iff $n < m$ has a cliquewidth of 2.

expression

instance

# of colors used 2

syntax tree



An (infinite) instance of finite Cliquewidth

Transitive chain

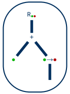
$\mathbb{N}$ with R(n, m) iff $n < m$ has a cliquewidth of 2.

expression

instance

# of colors used 2

syntax tree



An (infinite) instance of finite Cliquewidth

Transitive chain

$\mathbb{N}$ with R(n, m) iff $n < m$ has a cliquewidth of 2.

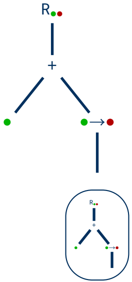
expression

instance

# of colors used

2

syntax tree



Reminder: Ontology Mediated Query Entailment

Problem: Ontology Mediated Query Entailment

Input: Database $\mathcal{D}$, ruleset $\mathcal{R}$ and a boolean query q.

Question: Does q hold in every model of $(\mathcal{D}, \mathcal{R})$?

Reminder: Ontology Mediated Query Entailment

Problem: Ontology Mediated Query Entailment
Input: Database $\mathcal{D}$, ruleset $\mathcal{R}$ and a boolean query q.
Question: Does q hold in every model of $(\mathcal{D}, \mathcal{R})$?

Generally an undecidable problem.

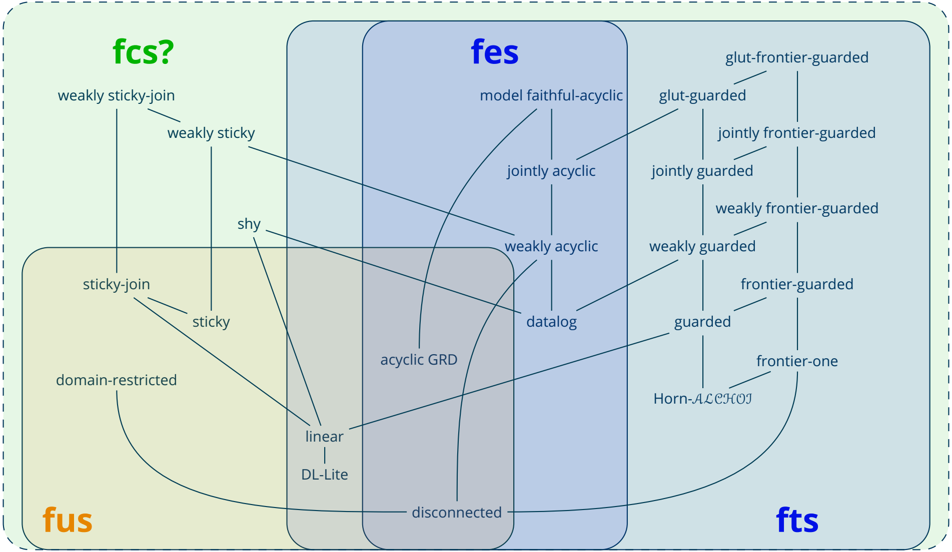
Reminder: Ontology Mediated Query Entailment

Problem: Ontology Mediated Query Entailment
Input: Database $\mathcal{D}$, ruleset $\mathcal{R}$ and a boolean query q.
Question: Does q hold in every model of $(\mathcal{D}, \mathcal{R})$?

Generally an undecidable problem.

Idea: Constrain universal models by demanding finite cliquewidth.

Reminder: Zoo of Decidable Existential Rules



Finite cliquewidth and Monadic Second Order Logic

Instances of finite cliquewidth have a treelike encoding.

Finite cliquewidth and Monadic Second Order Logic

Instances of finite cliquewidth have a treelike encoding.

So through clever use of Rabin's Tree Theorem we get:

Finite cliquewidth and Monadic Second Order Logic

Instances of finite cliquewidth have a treelike encoding.

So through clever use of Rabin's Tree Theorem we get:

Theorem

Determining if a given MSO-formula Φ has a model $\mathcal{I}$ with cliquewidth $\leq n$ is decidable for a fixed $n \in \mathbb{N}$.

Deciding query entailment

Definition (fcs)

A ruleset $\mathcal{R}$ is a ***finite-cliquewidth set*** (or *is fcs*), if for any database $\mathcal{D}$ there exists a universal model $\mathcal{U}$ for $(\mathcal{D}, \mathcal{R})$ of finite cliquewidth.

Deciding query entailment

Definition (fcs)

A ruleset $\mathcal{R}$ is a ***finite-cliquewidth set*** (or *is fcs*), if for any database $\mathcal{D}$ there exists a universal model $\mathcal{U}$ for $(\mathcal{D}, \mathcal{R})$ of finite cliquewidth.

Corollary

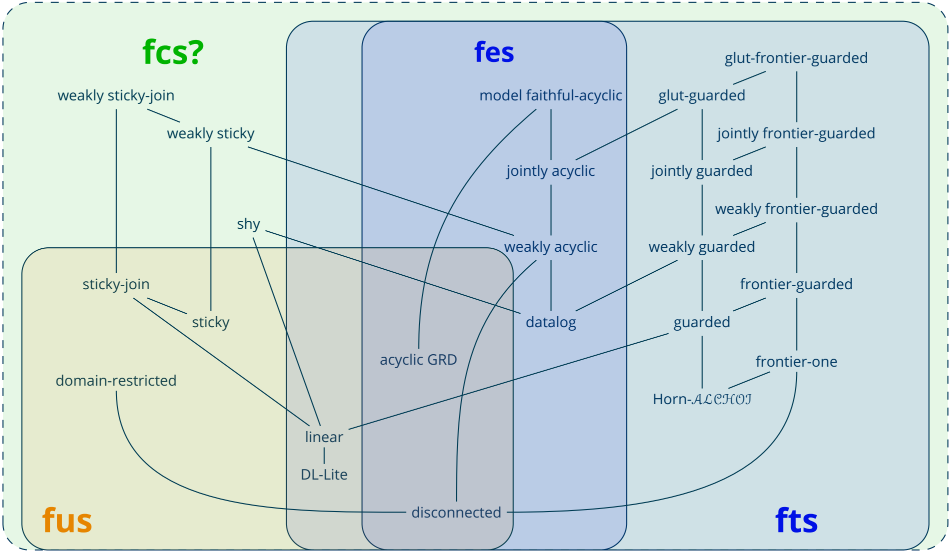
For arbitrary databases $\mathcal{D}$, if $\mathcal{R}$ is fcs and q a query expressible in MSO and Datalog, then query entailment is decidable.

Results: Binary single-head fus is fcs

Theorem

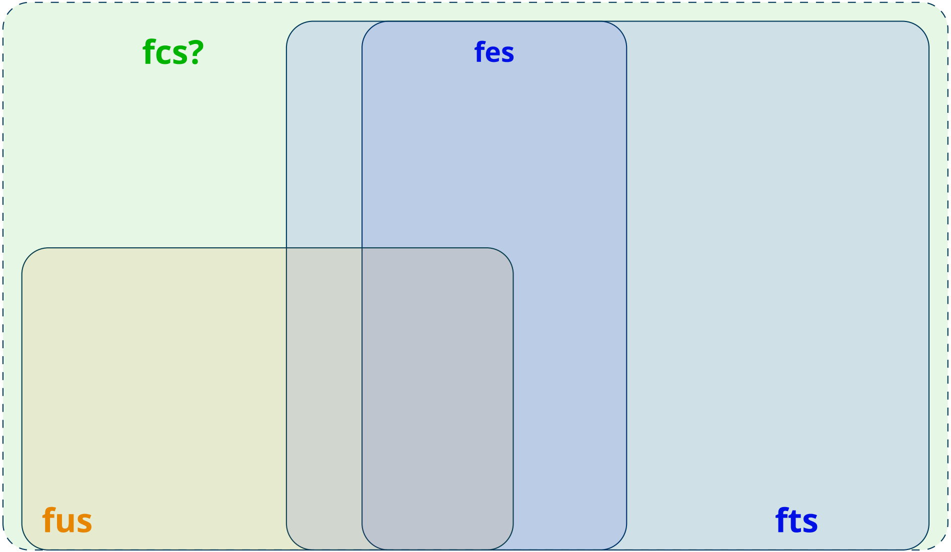
Any fus ruleset of single-headed rules over a binary signature is fcs.

Summary

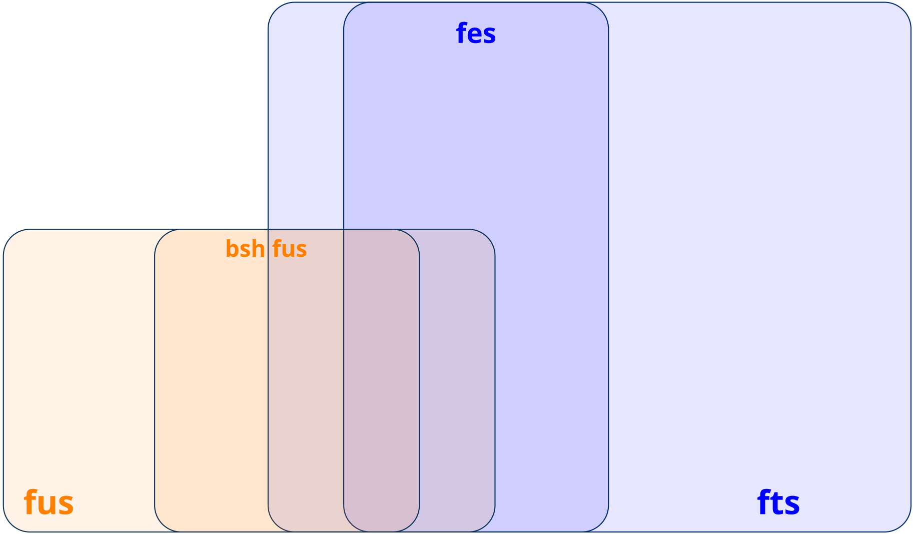


fcs?
fes
fus
fts
weakly sticky-join
weakly sticky
shy
sticky-join
sticky
domain-restricted
linear
DL-Lite
acyclic GRD
disconnected
model faithful-acyclic
jointly acyclic
weakly acyclic
datalog
glut-frontier-guarded
glut-guarded
jointly frontier-guarded
jointly guarded
weakly frontier-guarded
weakly guarded
frontier-guarded
guarded
frontier-one
Horn-$\mathcal{ALCHOI}$

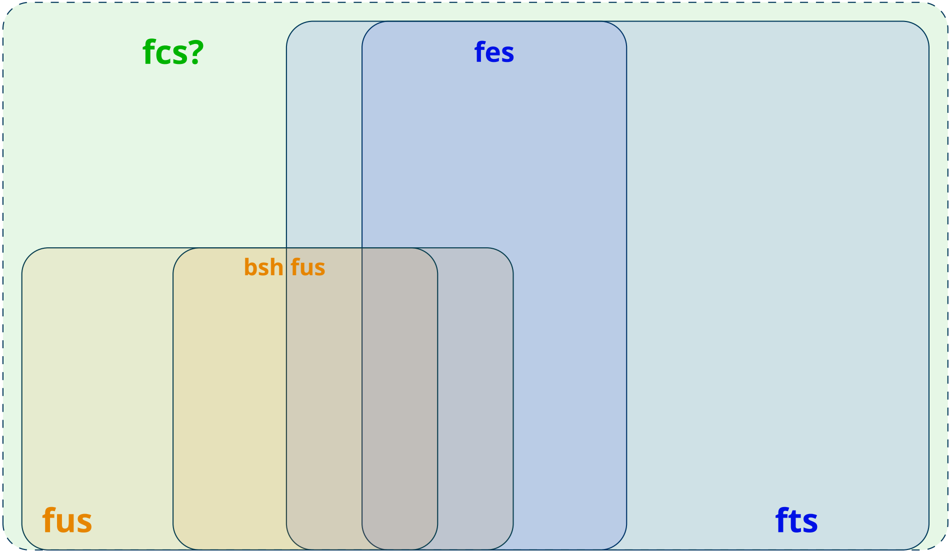
Summary: The binary picture



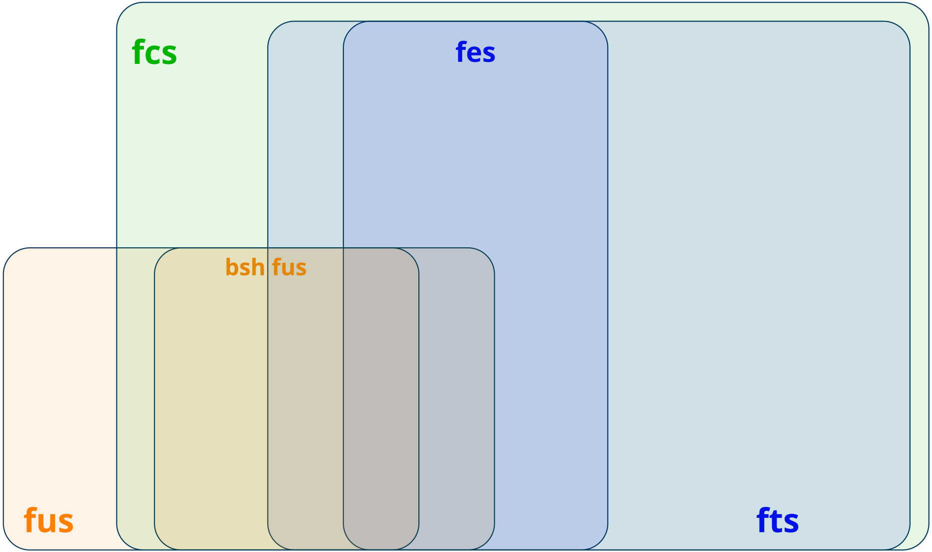
Summary: The binary picture



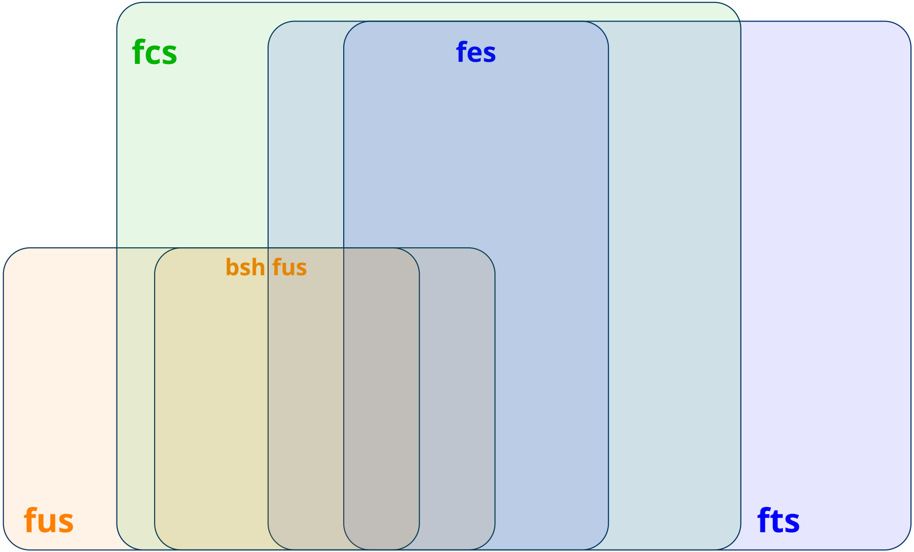
Summary: The binary picture



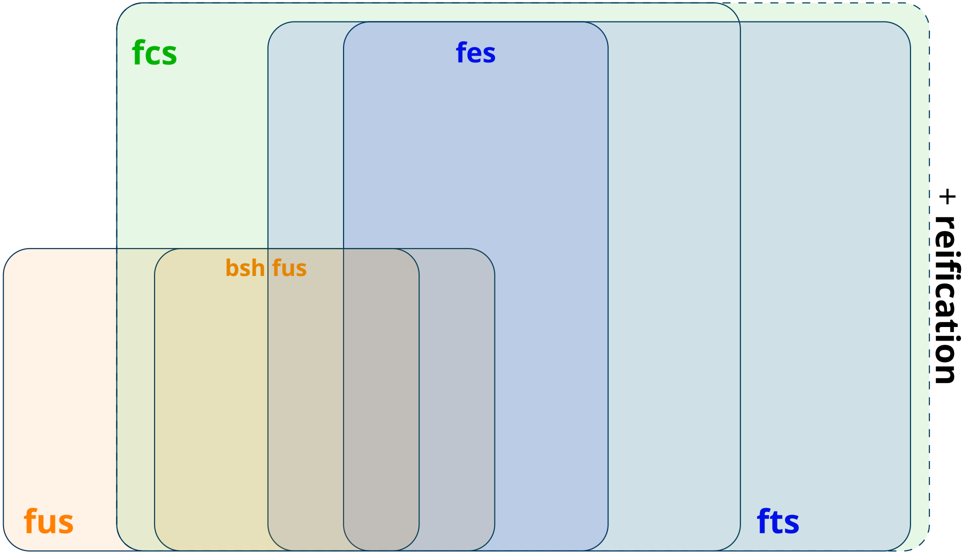
Summary: The binary picture



Summary: The higher arity case



Summary: The higher arity case



We adapt cliquewidth for **infinite** instances in the **higher arity** setting.

↓

Based on cliquewidth, we define **fcs** as a class of **decidable** existential rules.

This talk!

Take a peek into the paper!

We show that fcs **generalises** fts (with a detour: **Reification**)

We prove that fcs **subsumes** binary single-head fus

We adapt cliquewidth for **infinite** instances in the **higher arity** setting.

This talk!

Based on cliquewidth, we define **fcs** as a class of **decidable** existential rules.

Take a peek into the paper!

We show that fcs **generalises** fts (with a detour: **Reification**)

We prove that fcs **subsumes** binary single-head fus

Thank you for your attention!