Local First Order Logic with Data: Toward Specification of Distributed Algorithms

Olivier Stietel

Under the supervision of:
Benedikt Bollig & Arnaud Sangnier

Jeudi 14 décembre 2023

Institut de Recherche en Informatique Fondamentale

Université Paris Cité

Outline :

Motivations - Making decisions as a group...

...of people before the computer era

Motivations - Making decisions as a group...

...of people before the computer era

...of computer nowadays

- multicore programming
- servers
- cloud
- ...

Motivations - Making decisions as a group...

...of people before the computer era

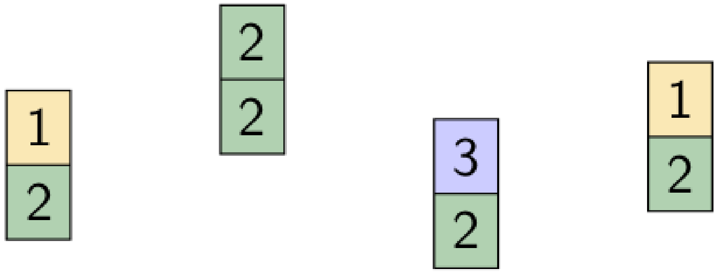
...of computer nowadays

- multicore programming
- servers
- cloud
- ...

Verification coming in

Motivations - Consensus problem

- Famous problem in distributed computing
- The goal is to design an algorithm such that:
  - all entities in a network have an input value
  - they should all agree on the same value
  - the chosen value should be one of the input values



Outline :

Data Logic - Introduction

Context

- Data-aware systems are omnipresent
  - Database
  - Sets of data for learning
  - Distributed/ Concurrent Systems
- Need for specification languages to describe systems with data

Data Logic - Introduction

Context

- Data-aware systems are omnipresent
  - Database
  - Sets of data for learning
  - Distributed/ Concurrent Systems
- Need for specification languages to describe systems with data

Requirements

- Logic to specify input-output behavior of distributed algorithms
- Structures with two data values
- The input values can be compared with the output values

Data Logic - Structures

A data value $\approx$ Element of a countable set (here $\mathbb{N}$)

Data Logic - Structures

A data value $\approx$ Element of a countable set (here $\mathbb{N}$)

Parameters:

- Σ finite set of unary relation symbols
- $\kappa \geq 0$ an integer (the number of data values per element)

Definition

A κ-structure is a tuple $\mathfrak{A} = (A, (P_\sigma)_{\sigma \in \Sigma}, f_1, \ldots, f_\kappa)$ where:

- A is the nonempty finite universe
- $P_\sigma \subseteq A$ for all $\sigma \in \Sigma$
- $f_1, \ldots, f_\kappa : A \to \mathbb{N}$ map each element to κ data values

Data Logic - Structures

A data value $\approx$ Element of a countable set (here $\mathbb{N}$)

Parameters:

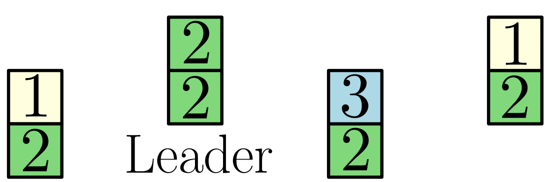
- Σ finite set of unary relation symbols
- $\kappa \geq 0$ an integer (the number of data values per element)

Definition

A κ-structure is a tuple $\mathfrak{A} = (A, (P_\sigma)_{\sigma \in \Sigma}, f_1, \dots, f_\kappa)$ where:

- A is the nonempty finite universe
- $P_\sigma \subseteq A$ for all $\sigma \in \Sigma$
- $f_1, \dots, f_\kappa : A \to \mathbb{N}$ map each element to κ data values

Example (2-structure):



Data Logic - Logic

Parameters:

- Σ finite set of unary relation symbols
- $\kappa \geq 0$ an integer

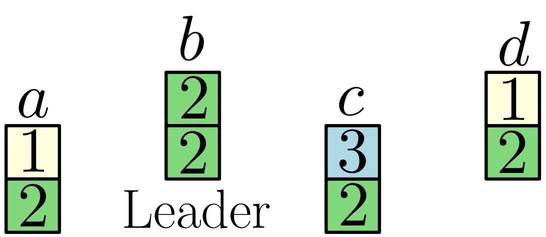
Definition

The logic $\mathsf{FO}_\kappa[\Sigma]$ is given as follows:

$$\varphi ::= \sigma(x) \mid x\,{}_i{\sim}_j\,y \mid x = y \mid \varphi \vee \varphi \mid \neg\varphi \mid \exists x.\varphi$$

where $\sigma \in \Sigma$ and $i, j \in \{1, \ldots, \kappa\}$

Example:



$\models a\,{}_1{\sim}_1\,d$

$\models a\,{}_2{\sim}_1\,b$

$\not\models c\,{}_1{\sim}_2\,c$

Parameters:

- Σ finite set of unary relation symbols
- $\kappa \geq 0$ an integer

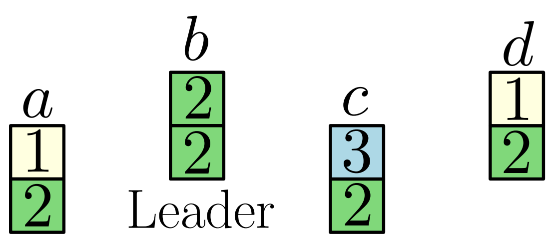
Definition

The logic $\mathsf{FO}_\kappa[\Sigma]$ is given as follows:

$$\varphi ::= \sigma(x) \mid x\,{}_i{\sim}_j\,y \mid x = y \mid \varphi \vee \varphi \mid \neg\varphi \mid \exists x.\varphi$$

where $\sigma \in \Sigma$ and $i, j \in \{1, \ldots, \kappa\}$

Example:



$\models a\,{}_1{\sim}_1\,d$

$\models a\,{}_2{\sim}_1\,b$

$\not\models c\,{}_1{\sim}_2\,c$

$$\models \begin{array}{l} \exists^{=1} x.\mathsf{leader}(x) \\ \wedge \forall y.\exists x.(\mathsf{leader}(x) \wedge x\,{}_1{\sim}_2\,y) \end{array}$$

Data Logic - Examples

Everybody takes a new name...

$$\forall x.\forall y.\neg x\,{}_2{\sim}_1\,y$$

...different from everyone else

$$\forall x.\forall y.\neg x\,{}_2{\sim}_2\,y$$

Data Logic - Examples

Everybody takes a new name...

$$\forall x. \forall y. \neg x\,{}_2{\sim}_1\,y$$

...different from everyone else

$$\forall x. \forall y. \neg x\,{}_2{\sim}_2\,y$$

We can force infinite models :

$$\begin{aligned} &\exists x. \forall y. \neg\, x\,{}_1{\sim}_2\,y \\ \wedge &\forall x. \exists y. x\,{}_2{\sim}_1\,y \\ \wedge &\forall x. \forall y. \neg x\,{}_2{\sim}_2\,y \end{aligned}$$

Data Logic - Examples

Everybody takes a new name...

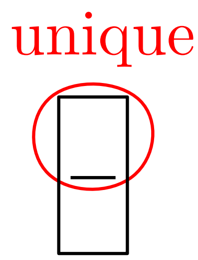
$$\forall x.\forall y.\neg x\,{}_2{\sim}_1\,y$$

...different from everyone else

$$\forall x.\forall y.\neg x\,{}_2{\sim}_2\,y$$

We can force infinite models :

$$\begin{aligned} &\exists x.\forall y.\neg\, x\,{}_1{\sim}_2\,y \\ \wedge&\forall x.\exists y.x\,{}_2{\sim}_1\,y \\ \wedge&\forall x.\forall y.\neg x\,{}_2{\sim}_2\,y \end{aligned}$$



Data Logic - Examples

Everybody takes a new name...

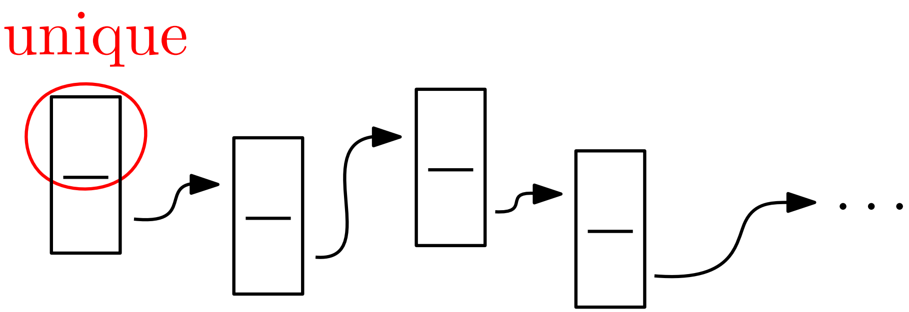
$$\forall x.\forall y.\neg x \,{}_2\!\sim_1 y$$

...different from everyone else

$$\forall x.\forall y.\neg x \,{}_2\!\sim_2 y$$

We can force infinite models :

$$\begin{aligned} &\exists x.\forall y.\neg\, x \,{}_1\!\sim_2 y \\ \wedge &\forall x.\exists y. x \,{}_2\!\sim_1 y \\ \wedge &\forall x.\forall y.\neg x \,{}_2\!\sim_2 y \end{aligned}$$



Data Logic - Examples

Everybody takes a new name...

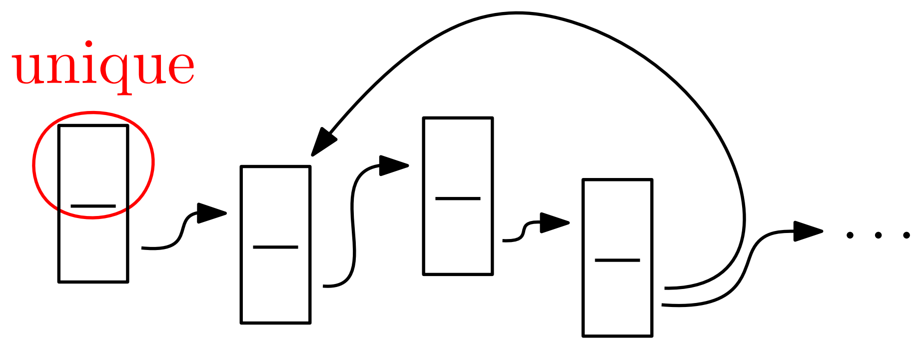
$$\forall x.\forall y.\neg x\,{}_2{\sim}_1\,y$$

...different from everyone else

$$\forall x.\forall y.\neg x\,{}_2{\sim}_2\,y$$

We can force infinite models :

$$\begin{aligned} &\exists x.\forall y.\neg\, x\,{}_1{\sim}_2\,y \\ \wedge &\forall x.\exists y. x\,{}_2{\sim}_1\,y \\ \wedge &\forall x.\forall y.\neg x\,{}_2{\sim}_2\,y \end{aligned}$$



Data Logic - Examples

Everybody takes a new name...

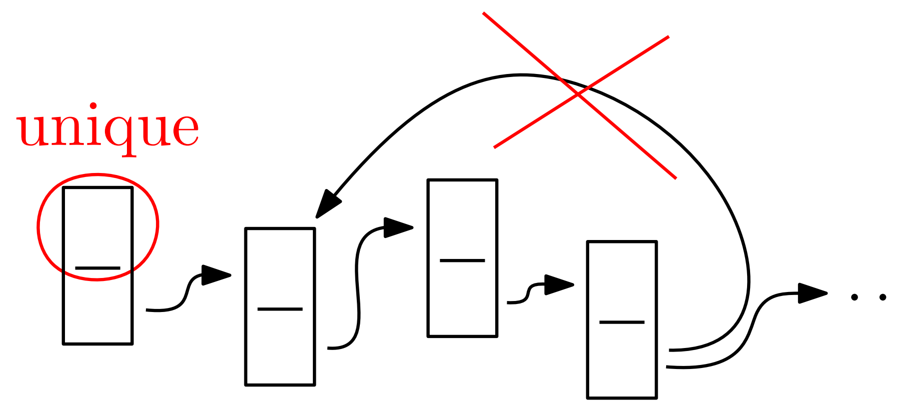
$$\forall x. \forall y. \neg x\,{}_2{\sim}_1\,y$$

...different from everyone else

$$\forall x. \forall y. \neg x\,{}_2{\sim}_2\,y$$

We can force infinite models :

$$\begin{aligned} &\exists x. \forall y. \neg\, x\,{}_1{\sim}_2\,y \\ \wedge &\forall x. \exists y. x\,{}_2{\sim}_1\,y \\ \wedge &\forall x. \forall y. \neg x\,{}_2{\sim}_2\,y \end{aligned}$$



Data Logic - Examples

Everybody takes a new name...

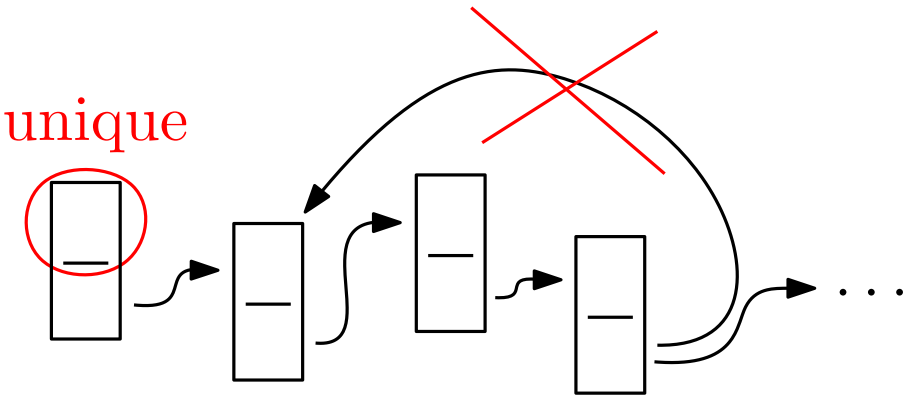
$$\forall x.\forall y.\neg x\,{}_2{\sim}_1\,y$$

...different from everyone else

$$\forall x.\forall y.\neg x\,{}_2{\sim}_2\,y$$

We can force infinite models :

$$\begin{aligned} &\exists x.\forall y.\neg\, x\,{}_1{\sim}_2\,y \\ \wedge&\forall x.\exists y.x\,{}_2{\sim}_1\,y \\ \wedge&\forall x.\forall y.\neg x\,{}_2{\sim}_2\,y \end{aligned}$$



With only one data value per element,

$$\mathsf{FO}_1[\Sigma] \quad \approx \quad \mathsf{FO} \text{ over one equivalence relation}$$

Data Logic - Satisfiability Problem

Question: How to know that a given specification is consistent ?

Data Logic - Satisfiability Problem

Question: How to know that a given specification is consistent ?

Definition

The problem $\textsc{Sat}(\mathcal{F})$ is defined as follows:
Input: Finite set Σ; sentence $\varphi \in \mathcal{F}[\Sigma]$.
Question: Is there a data structure $\mathfrak{A}$ such that $\mathfrak{A} \models \varphi$?

Data Logic - Satisfiability Problem

Question: How to know that a given specification is consistent ?

Definition

The problem $\textsc{Sat}(\mathcal{F})$ is defined as follows:

Input: Finite set Σ; sentence $\varphi \in \mathcal{F}[\Sigma]$.

Question: Is there a data structure $\mathfrak{A}$ such that $\mathfrak{A} \models \varphi$?

Theorem

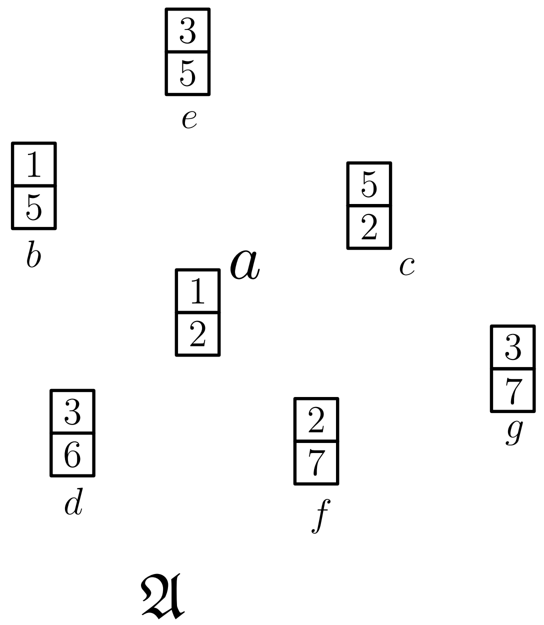
$\textsc{Sat}(\mathsf{FO}_2)$ is undecidable, even when $\Sigma = \emptyset$ and without using ${}_1{\sim}_2$ and ${}_2{\sim}_1$.

Other related works:

- Works where the number of variables is bounded [Kieronski and Tendera, 2009]
- Two-variable logic on data words [Bojanczyk, David, Muscholl, Schwentick, and Segoufin, 2011]

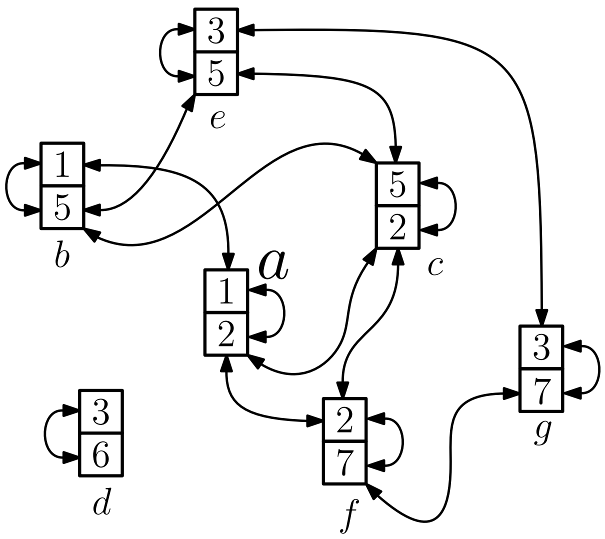
Outline :

Locality Explained - From Structures to Views



- $\mathfrak{A}$ — 2-data structure

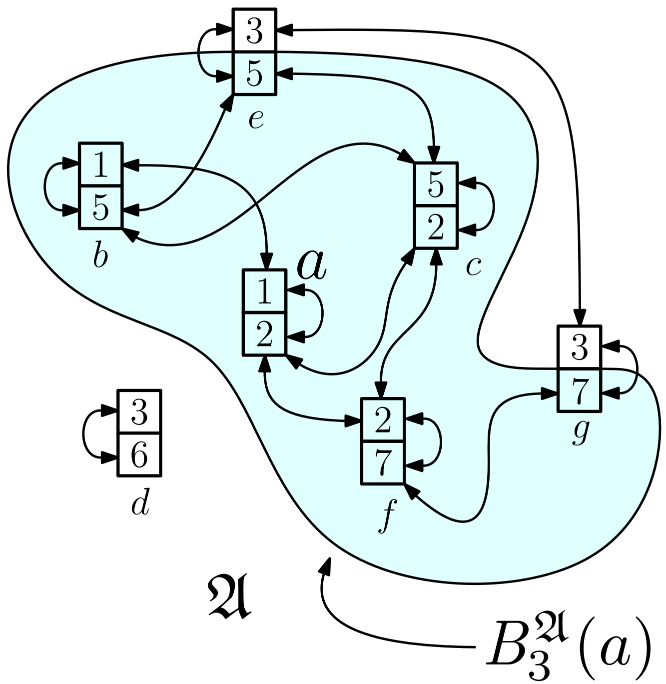
Locality Explained - From Structures to Views



$\mathfrak{A}$

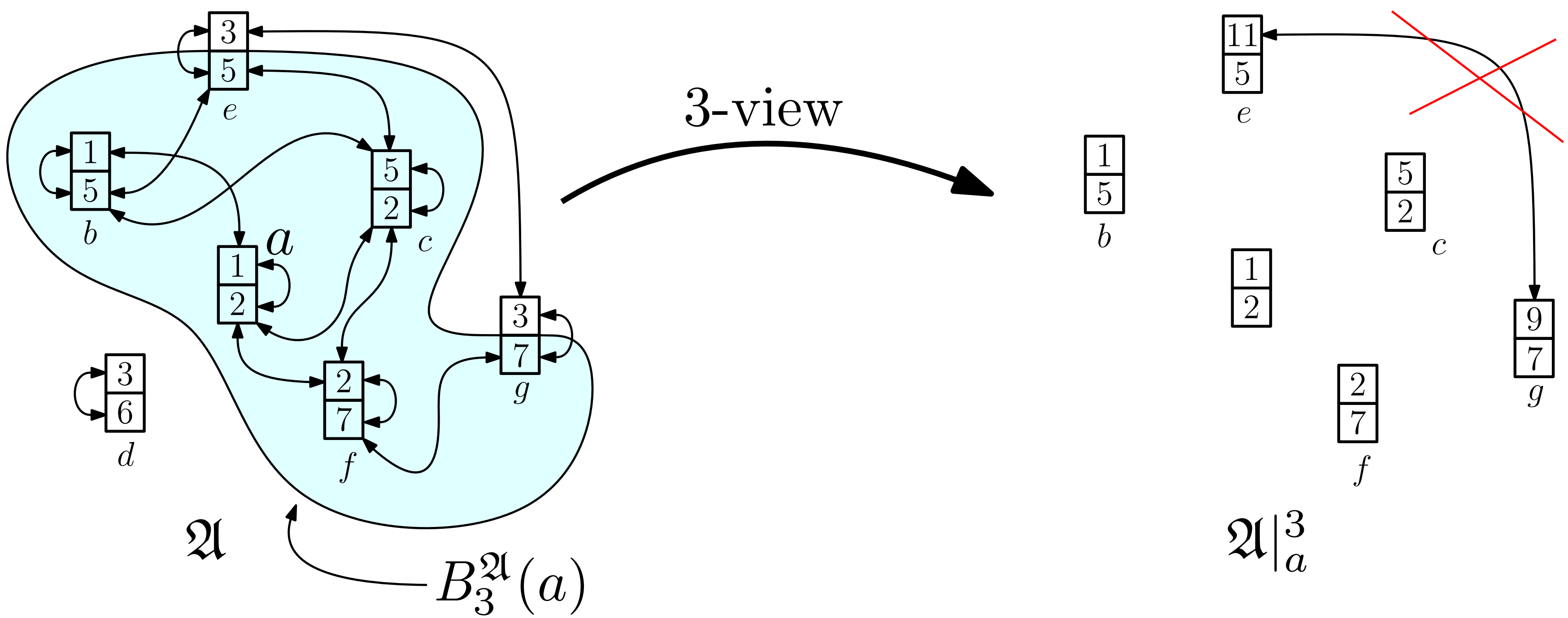
- $\mathfrak{A}$ — 2-data structure
- $\mathcal{G}(\mathfrak{A})$ — data graph

Locality Explained - From Structures to Views



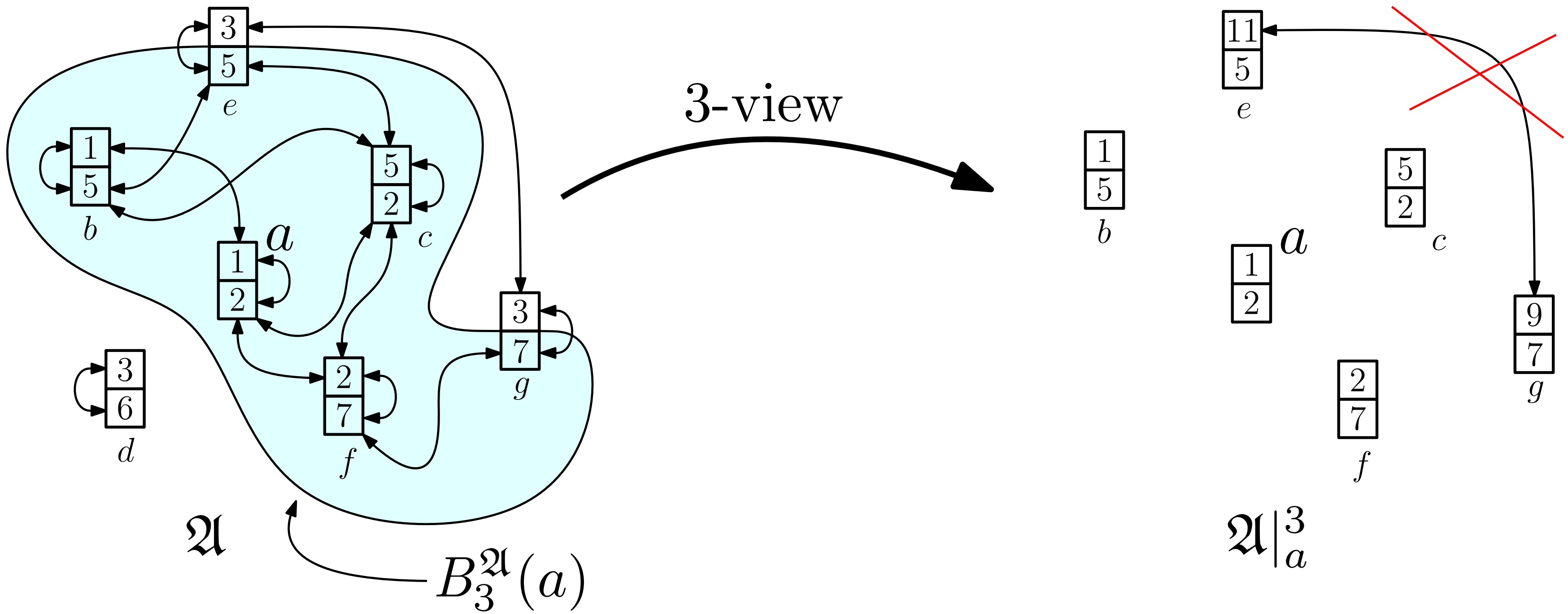
- $\mathfrak{A}$ — 2-data structure
- $\mathcal{G}(\mathfrak{A})$ — data graph
- $B_3^{\mathfrak{A}}(a)$ — 3-ball

Locality Explained - From Structures to Views



- $\mathfrak{A}$ — 2-data structure
- $\mathcal{G}(\mathfrak{A})$ — data graph
- $B_3^{\mathfrak{A}}(a)$ — 3-ball
- $\mathfrak{A}|_a^3$ — 3-view of a

Locality Explained - From Structures to Views



- $\mathfrak{A}$ — 2-data structure
- $\mathcal{G}(\mathfrak{A})$ — data graph
- $B_3^{\mathfrak{A}}(a)$ — 3-ball
- $\mathfrak{A}|_a^3$ — 3-view of a
- Local modality $\langle\!\langle \psi \rangle\!\rangle_x^3$ with $\psi \in \mathsf{FO}_\kappa[\Sigma]$

  ψ has only access to $\mathfrak{A}|_a^3$

Locality Explained - Local Fragment

Parameters :

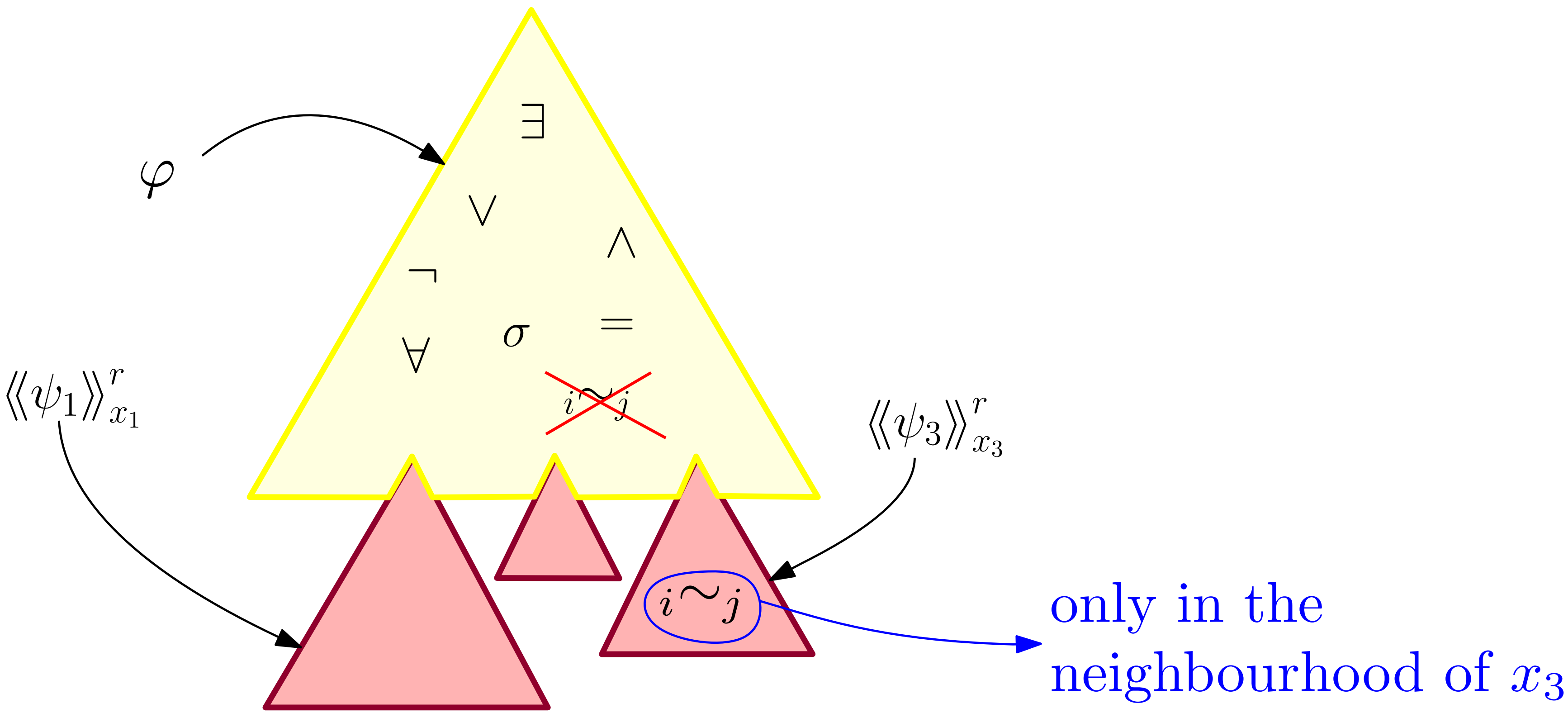
- Σ finite set of unary relation symbols
- $\kappa > 0$ an integer
- $r \geq 0$ an integer

Definition

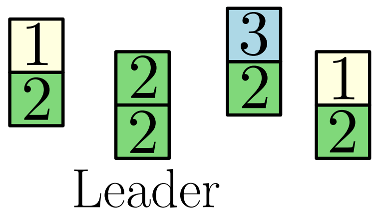
The logic r-$\mathsf{LF}_\kappa[\Sigma]$ is given as follows:

$$\varphi \ ::= \ \langle\!\langle \psi \rangle\!\rangle_x^r \ | \ x = y \ | \ \exists x.\varphi \ | \ \varphi \vee \varphi \ | \ \neg\varphi$$

where $\psi \in \mathsf{FO}_\kappa[\Sigma]$.



Locality Explained - Consensus & Inclusions



$$\models \quad \begin{array}{l} \exists^{=1} x. \langle\!\langle \mathsf{leader}(x) \rangle\!\rangle^{1}_{x} \quad \wedge \\ \forall y. \langle\!\langle \exists x. \mathsf{leader}(x) \wedge y \,{}_{2}{\sim}_{1}\, x \rangle\!\rangle^{1}_{y} \\ \qquad \in \mathsf{1\text{-}LF}_{2}[\{\mathsf{leader}\}] \end{array}$$

Inclusion analysis:

$$\mathsf{1\text{-}LF}_{\kappa}[\Sigma] \subseteq \mathsf{2\text{-}LF}_{\kappa}[\Sigma] \subseteq \mathsf{3\text{-}LF}_{\kappa}[\Sigma] \subseteq \cdots \quad \subseteq \mathsf{FO}_{\kappa}[\Sigma]$$

Outline :

The General Fragment - Positive Results

Theorem

$\textsc{Sat}(1\text{-}\mathsf{LF}_2)$ is decidable.
(with relations in ${}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2$)

The General Fragment - Positive Results

Theorem

$\textsc{Sat}(1\text{-}\mathsf{LF}_2)$ is decidable.
(with relations in ${}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2$)

Wait, ${}_1{\sim}_2$ but not ${}_2{\sim}_1$?

The General Fragment - Positive Results

Theorem

$\textsc{Sat}(1\text{-}\mathsf{LF}_2)$ is decidable.
(with relations in ${}_1\!\sim_1, {}_2\!\sim_2, {}_1\!\sim_2$)

Wait, ${}_1\!\sim_2$ but not ${}_2\!\sim_1$?

Features of the proof:

- reduction to two-variable FO over two equivalence relations

The General Fragment - Positive Results

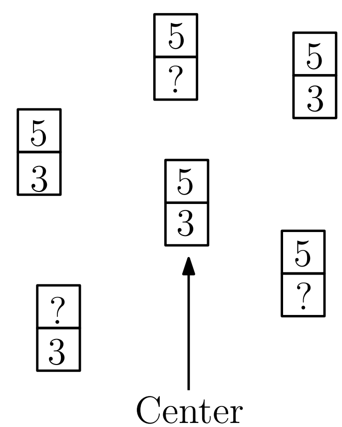
Theorem

$\textsc{Sat}(1\text{-}\mathsf{LF}_2)$ is decidable.
(with relations in ${}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2$)

Wait, ${}_1{\sim}_2$ but not ${}_2{\sim}_1$?

Features of the proof:

- reduction to two-variable FO over two equivalence relations
- a view $\approx$ counting



Type	Number
${}_1{\sim}_1$	2
${}_2{\sim}_2$	1
${}_1{\sim}_1$ and ${}_2{\sim}_2$	2

The General Fragment - Positive Results

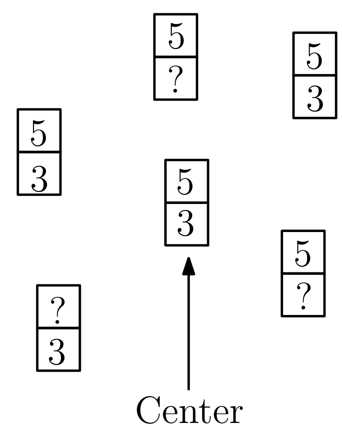
Theorem

$\textsc{Sat}(1\text{-}\mathsf{LF}_2)$ is decidable.
(with relations in ${}_1{\sim}_1, {}_2{\sim}_2, {}_1{\sim}_2$)

Wait, ${}_1{\sim}_2$ but not ${}_2{\sim}_1$?

Features of the proof:

- reduction to two-variable FO over two equivalence relations
- a view $\approx$ counting



Type	Number
${}_1{\sim}_1$	2
${}_2{\sim}_2$	1
${}_1{\sim}_1$ and ${}_2{\sim}_2$	2

- reduce two-variable FO with counting to two-variable FO without it
  - two-variable FO with counting is decidable
  - but it involves to duplicates binary relations and this does not work with equivalence

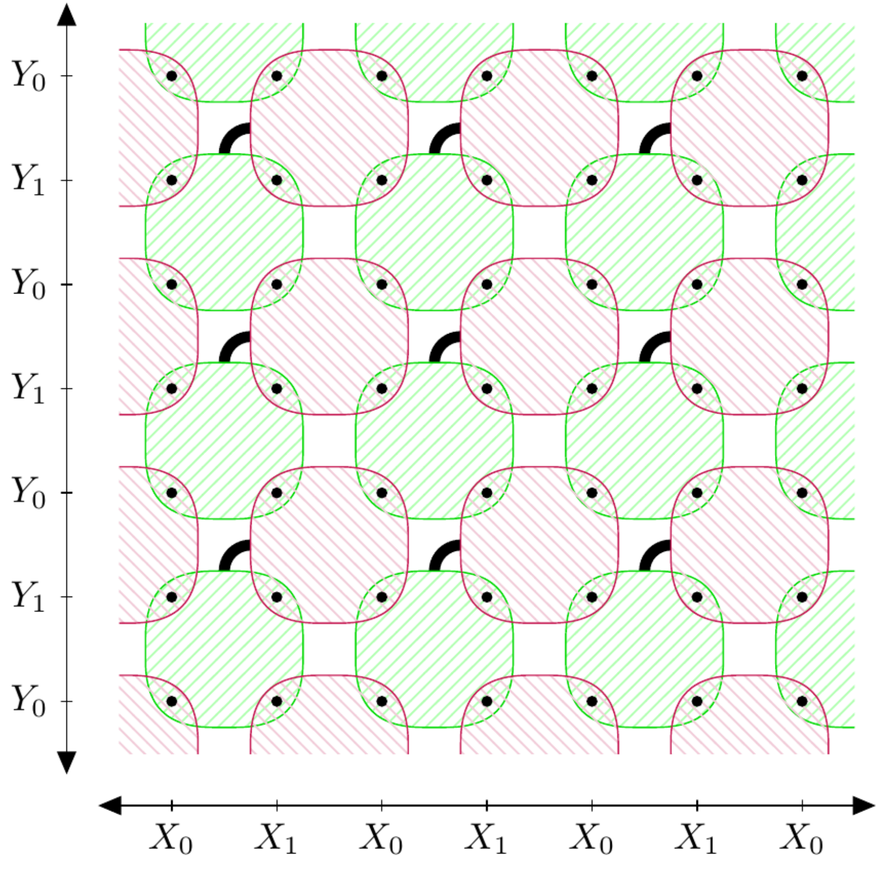
The General Fragment - Negative Results

Theorem

$\textsc{Sat}(2\text{-}\mathsf{LF}_2)$ is undecidable

Proof: - Reduction from the tiling problem

- Uses a technique developped by M.Otto in [Otto01] "Two variable first-order logic over ordered domains."



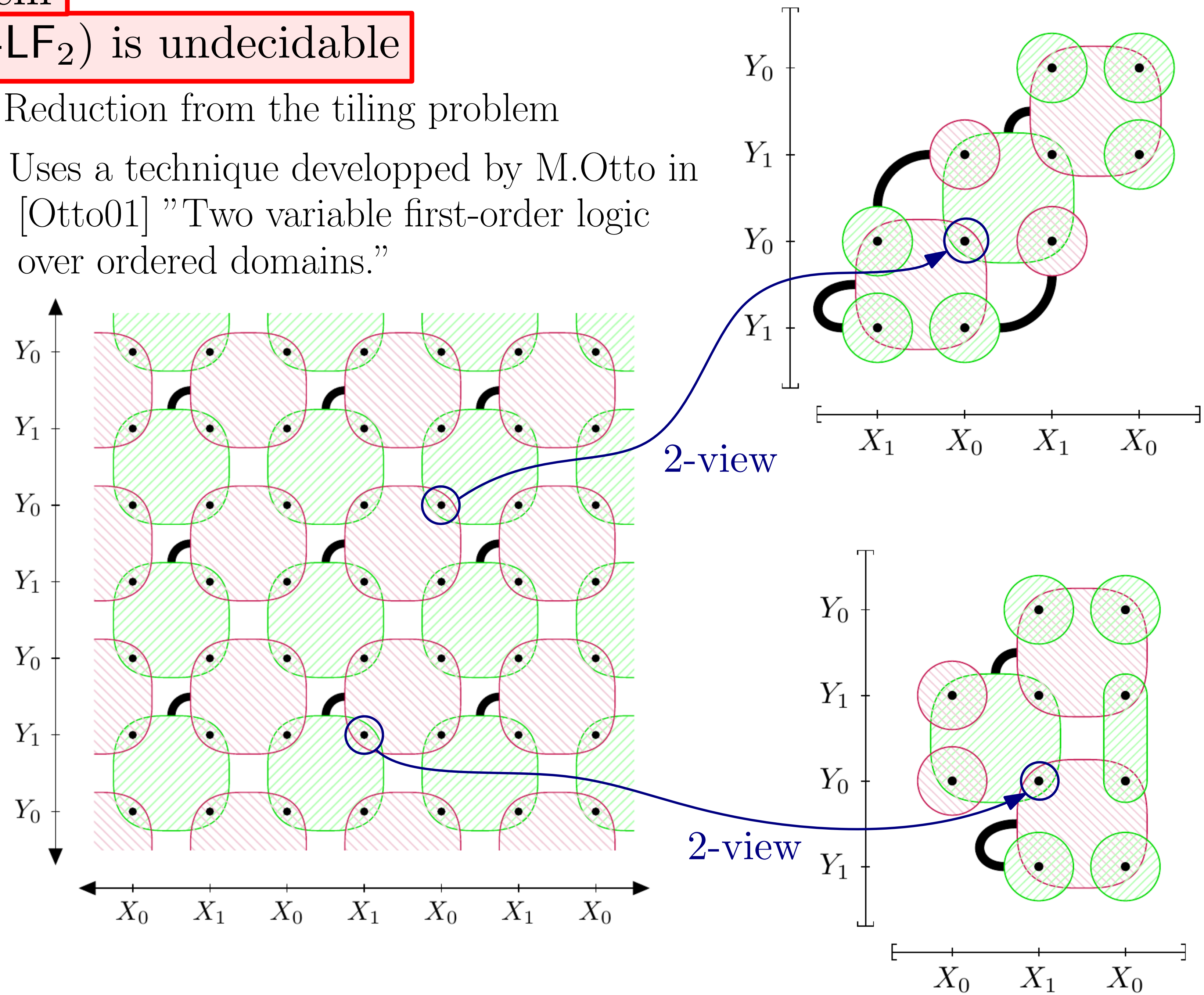
The General Fragment - Negative Results

Theorem

SAT(2-LF_2) is undecidable

Proof: - Reduction from the tiling problem

- Uses a technique developped by M.Otto in [Otto01] "Two variable first-order logic over ordered domains."



The General Fragment - Conclusions

Summary:

- $\textsc{Sat}(1\text{-}\mathsf{LF}_2)$ is decidable (with restriction)
- $\textsc{Sat}(2\text{-}\mathsf{LF}_2)$ is undecidable

Decidability of full $\textsc{Sat}(1\text{-}\mathsf{LF}_2)$ is an open problem.

From this, what are other decidable fragments?

Outline :

The Existential Fragment - Definition

Parameters:

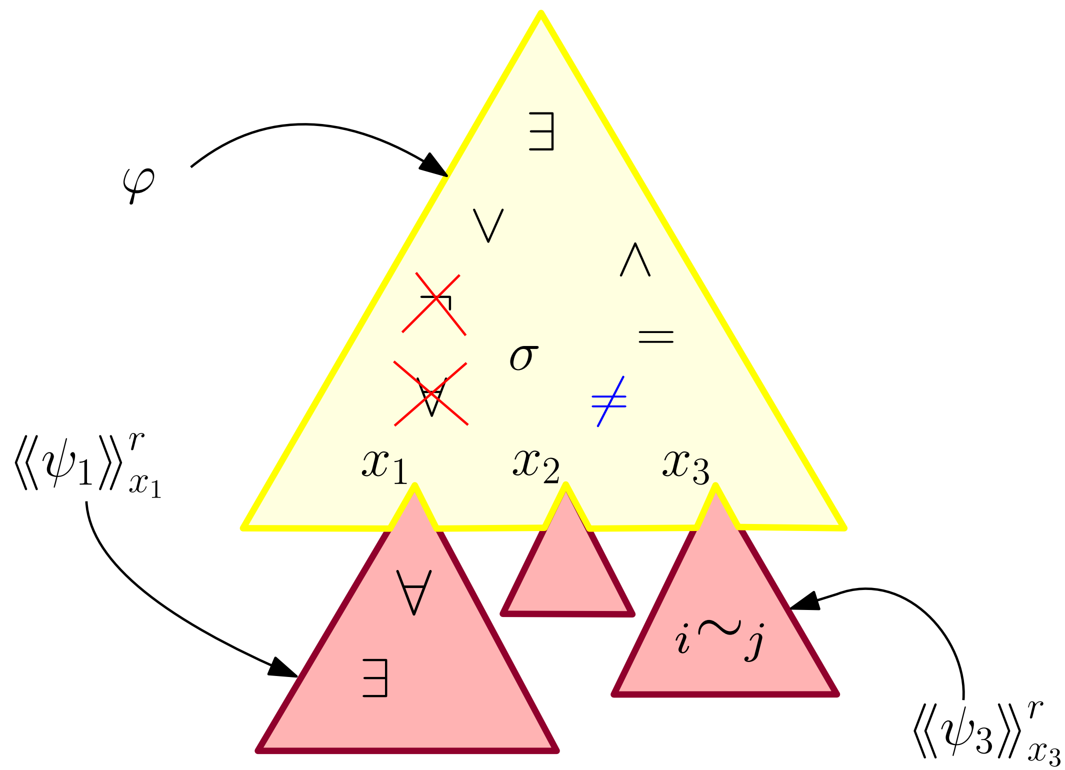
- Σ finite set of unary relation symbols
- $\kappa > 0$ an integer
- $r \geq 0$ an integer

Definition

The logic $\exists$-r-$\mathsf{LF}_\kappa[\Sigma]$ is given as follows:

$$\varphi \ ::= \ \langle\!\langle \psi \rangle\!\rangle_x^r \ \mid \ x = y \ \mid \ x \neq y \ \mid \ \exists x.\varphi \ \mid \ \varphi \vee \varphi \ \mid \ \varphi \wedge \varphi$$

where $\psi \in \mathsf{FO}_D[\Sigma]$.



Remark:

- No restriction on the quantifiers in ψ located in $\langle\!\langle \psi \rangle\!\rangle_x^r$

The Existential Fragment - Positive Results (1)

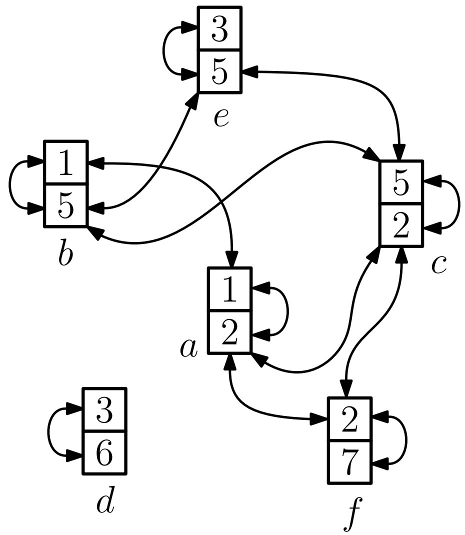
$\textsc{Sat}(\exists\text{-}2\text{-}\mathsf{LF}_2)$ is N2EXP-complete.

Proof. Reduction to $\textsc{Sat}(\mathsf{FO}_1)$:

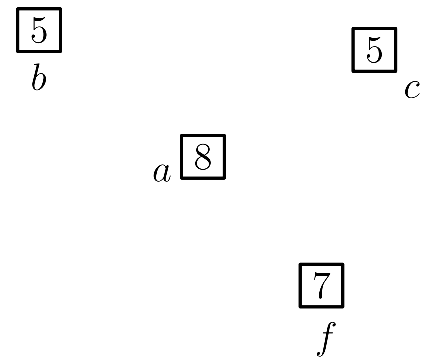
- Take a formula in prenex normal form
$$\exists x_1 \ldots . \exists x_n . \phi_{qf}(x_1, \ldots, x_n)$$
- Take n elements $a_1, \ldots, a_n$ of the structure
- Encode the relation with the data of a_i's by unary predicates
- Keep only the data in the views of a_i's not in relation with a_i's (at most 1 per element because we have 2-views)
- Ensure 1-data structures are well-formed $\square$

The Existential Fragment - Positive Results (2)

From 2-data structure to 1-data structure



$\mathfrak{A}$



$P_{a[1,1]} = \{a, b\}$
$P_{a[2,2]} = \{a, c\}$
$P_{a[1,2]} = \emptyset$
$P_{a[2,1]} = \{f\}$

$[\![\mathfrak{A}]\!]_{(a)}$

- Each element in $B_2^{\mathfrak{A}}(a)$ shares at least one data with a
- $P_{a[i,j]}$: the element has its j-th data equals to the i-th data of a

The Existential Fragment - Positive Results (3)

Translating the formula $\phi_{qf}(x_1, \ldots, x_n)$

- We want to translate $\phi_{qf}(x_1, \ldots, x_n)$ into $[\![\phi_{qf}]\!](x_1, \ldots, x_n)$ of $\mathsf{FO}_1[\Sigma']$
- Main issue: formulas of the form $y\,{}_j{\sim}_k\,z$

The Existential Fragment - Positive Results (3)

Translating the formula $\phi_{qf}(x_1, \ldots, x_n)$

- We want to translate $\phi_{qf}(x_1, \ldots, x_n)$ into $[\![\phi_{qf}]\!](x_1, \ldots, x_n)$ of $\mathsf{FO}_1[\Sigma']$
- Main issue: formulas of the form $y \,_j{\sim}_k\, z$

Trick to solve this case when located in the subformula $\langle\!\langle\psi\rangle\!\rangle_x^2$:

1. the j-th data of y is **in the radius 1 ball** around x:
   $\rightarrow$ y have to be labeled with $P_{a[i,j]}$ and z with $P_{a[i,k]}$ for $i \in \{1, 2\}$
2. the j-th data of y is **in the radius 2 ball** but **not in the radius 1 ball** around x:
   $\rightarrow$ y and z have the same data in the translated data structure
3. the j-th data of y is **not in the radius 2 ball** around x:
   $\rightarrow$ $y \,_j{\sim}_k\, z$ cannot hold in $\langle\!\langle\psi\rangle\!\rangle_x^2$

The Existential Fragment - Positive Results (3)

Translating the formula $\phi_{qf}(x_1, \ldots, x_n)$

- We want to translate $\phi_{qf}(x_1, \ldots, x_n)$ into $[\![\phi_{qf}]\!](x_1, \ldots, x_n)$ of $\mathsf{FO}_1[\Sigma']$
- Main issue: formulas of the form $y\,{}_j{\sim}_k\,z$

Trick to solve this case when located in the subformula $\langle\!\langle\psi\rangle\!\rangle_x^2$:

1. the j-th data of y is **in the radius 1 ball** around x:
   $\rightarrow$ y have to be labeled with $P_{a[i,j]}$ and z with $P_{a[i,k]}$ for $i \in \{1, 2\}$
2. the j-th data of y is **in the radius 2 ball** but **not in the radius 1 ball** around x:
   $\rightarrow$ y and z have the same data in the translated data structure
3. the j-th data of y is **not in the radius 2 ball** around x:
   $\rightarrow$ $y\,{}_j{\sim}_k\,z$ cannot hold in $\langle\!\langle\psi\rangle\!\rangle_x^2$

- We finally obtain:
  $\mathfrak{A} \models \phi_{qf}(a_1, \ldots, a_n)$ **iff** $[\![\mathfrak{A}]\!]_{(a_1,\ldots,a_n)} \models [\![\phi_{qf}]\!](a_1, \ldots, a_n)$

The Existential Fragment - Positive Results (4)

Ensuring 1-data structure are well-formed

- To finish, we need to find a 1-data structure $\mathfrak{B}$ such that

1. $\mathfrak{B}$ has n elements $a_1, \ldots, a_n$
2. $\mathfrak{B} \models [\![\phi_{qf}]\!](a_1, \ldots, a_n)$
3. $\mathfrak{B} = [\![\mathfrak{A}]\!]_{(a_1, \ldots, a_n)}$ for some 2-data structure $\mathfrak{A}$.

The Existential Fragment - Positive Results (4)
Ensuring 1-data structure are well-formed

- To finish, we need to find a 1-data structure $\mathfrak{B}$ such that

1. $\mathfrak{B}$ has n elements $a_1, \ldots, a_n$
2. $\mathfrak{B} \models [\![\phi_{qf}]\!](a_1, \ldots, a_n)$
3. $\mathfrak{B} = [\![\mathfrak{A}]\!]_{(a_1, \ldots, a_n)}$ for some 2-data structure $\mathfrak{A}$.

- For this last point, we ensure with formulas in $\mathsf{FO}_1[\Sigma']$ that
  - the labeling by $P_{a_k[i,j]}$ is consistent
  - if a node is is not labelled by any $P_{a_k[i,j]}$, its value is unique
  - the same holds for nodes labelled by $P_{a_k[i,1]}$ and $P_{a_\ell[j,2]}$

The Existential Fragment - Positive Results (4)

Ensuring 1-data structure are well-formed

- To finish, we need to find a 1-data structure $\mathfrak{B}$ such that

1. $\mathfrak{B}$ has n elements $a_1, \ldots, a_n$
2. $\mathfrak{B} \models [\![\phi_{qf}]\!](a_1, \ldots, a_n)$
3. $\mathfrak{B} = [\![\mathfrak{A}]\!]_{(a_1, \ldots, a_n)}$ for some 2-data structure $\mathfrak{A}$.

- For this last point, we ensure with formulas in $\mathsf{FO}_1[\Sigma']$ that
  - the labeling by $P_{a_k[i,j]}$ is consistent
  - if a node is is not labelled by any $P_{a_k[i,j]}$, its value is unique
  - the same holds for nodes labelled by $P_{a_k[i,1]}$ and $P_{a_\ell[j,2]}$

- This gives rise to a formula $\phi_{wf}(x_1, \ldots, x_n)$

The Existential Fragment - Positive Results (4)
Ensuring 1-data structure are well-formed

- To finish, we need to find a 1-data structure $\mathfrak{B}$ such that

1. $\mathfrak{B}$ has n elements $a_1, \ldots, a_n$
2. $\mathfrak{B} \models [\![\phi_{qf}]\!](a_1, \ldots, a_n)$
3. $\mathfrak{B} = [\![\mathfrak{A}]\!]_{(a_1,\ldots,a_n)}$ for some 2-data structure $\mathfrak{A}$.

- For this last point, we ensure with formulas in $\mathsf{FO}_1[\Sigma']$ that
  - the labeling by $P_{a_k[i,j]}$ is consistent
  - if a node is is not labelled by any $P_{a_k[i,j]}$, its value is unique
  - the same holds for nodes labelled by $P_{a_k[i,1]}$ and $P_{a_\ell[j,2]}$

- This gives rise to a formula $\phi_{wf}(x_1, \ldots, x_n)$

- Finally, $\exists x_1. \ldots. \exists x_n. \phi_{qf}(x_1, \ldots, x_n)$ **is satisfiable iff** $\exists x_1. \ldots. \exists x_n. [\![\phi_{qf}]\!](x_1, \ldots, x_n) \land \phi_{wf}(x_1, \ldots, x_n)$ **is satisfiable**. $\square$

Theorem

For all $\kappa \geq 1$, $\textsc{Sat}(\exists\text{-}1\text{-}\mathsf{LF}_\kappa)$ is NEXP-complete.

Proof. Reduction to $\textsc{Sat}(\mathsf{FO}_0)$:

- Take a formula in prenex normal form $\exists x_1 \ldots \exists x_n . \phi_{qf}(x_1, \ldots, x_n)$
- Take n elements $a_1, \ldots, a_n$ of the structures
- Encode the relation with the data of a_i's by unary predicates
- ~~Keep only the data in the views of a_i's not in relation with a_i's (at most 1 per element because we have 2-neighborhoods)~~
- Ensure 0-data structures are well-formed $\square$

The Existential Fragment - Negative Results

Radius 3 and two data values

Theorem

$\textsc{Sat}(\exists\text{-}3\text{-}\mathsf{LF}_2)$ is undecidable.

The Existential Fragment - Negative Results

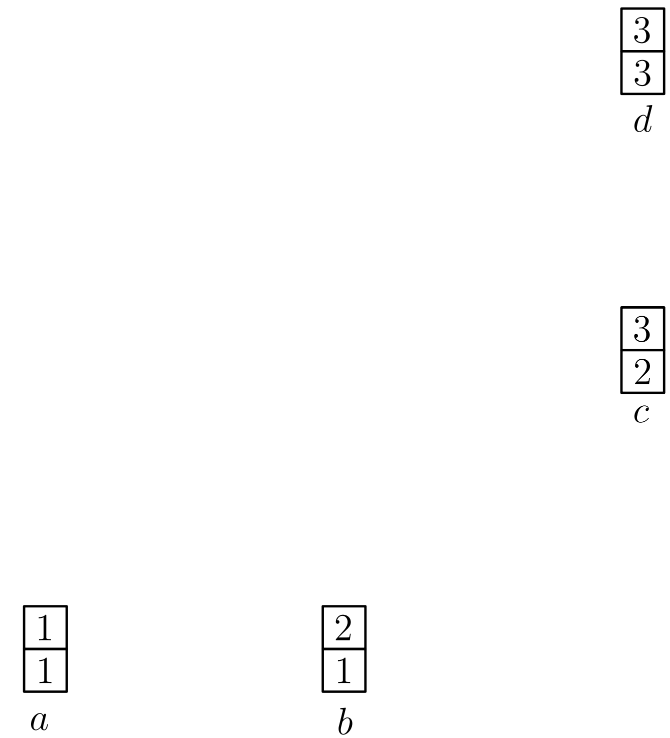
Radius 3 and two data values

Theorem

$\textsc{Sat}(\exists\text{-}3\text{-}\mathsf{LF}_2)$ is undecidable.

Key idea for the proof:

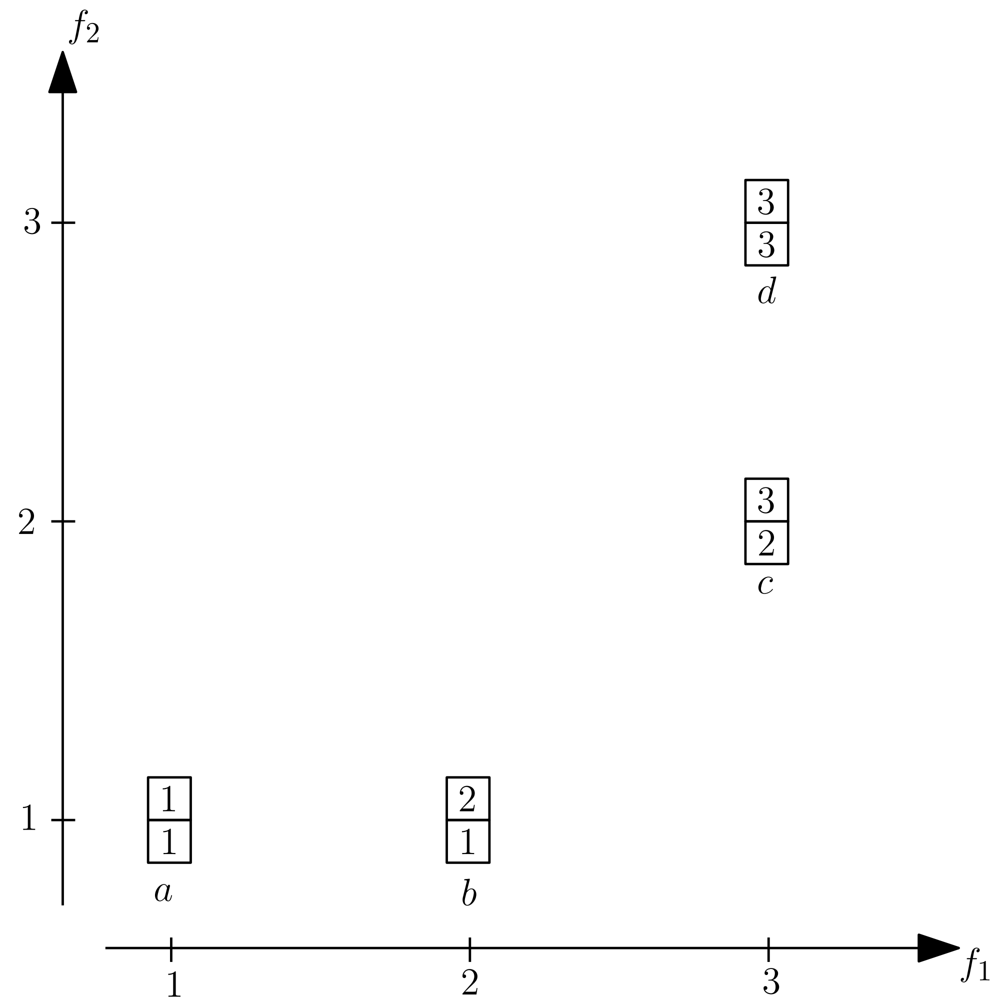
- Some elements can be far apart (e.g. a and d)



Radius 3 and two data values

Theorem

SAT($\exists$-3-LF_2) is undecidable.



Key idea for the proof:

- Some elements can be far apart (e.g. a and d)

The Existential Fragment - Negative Results

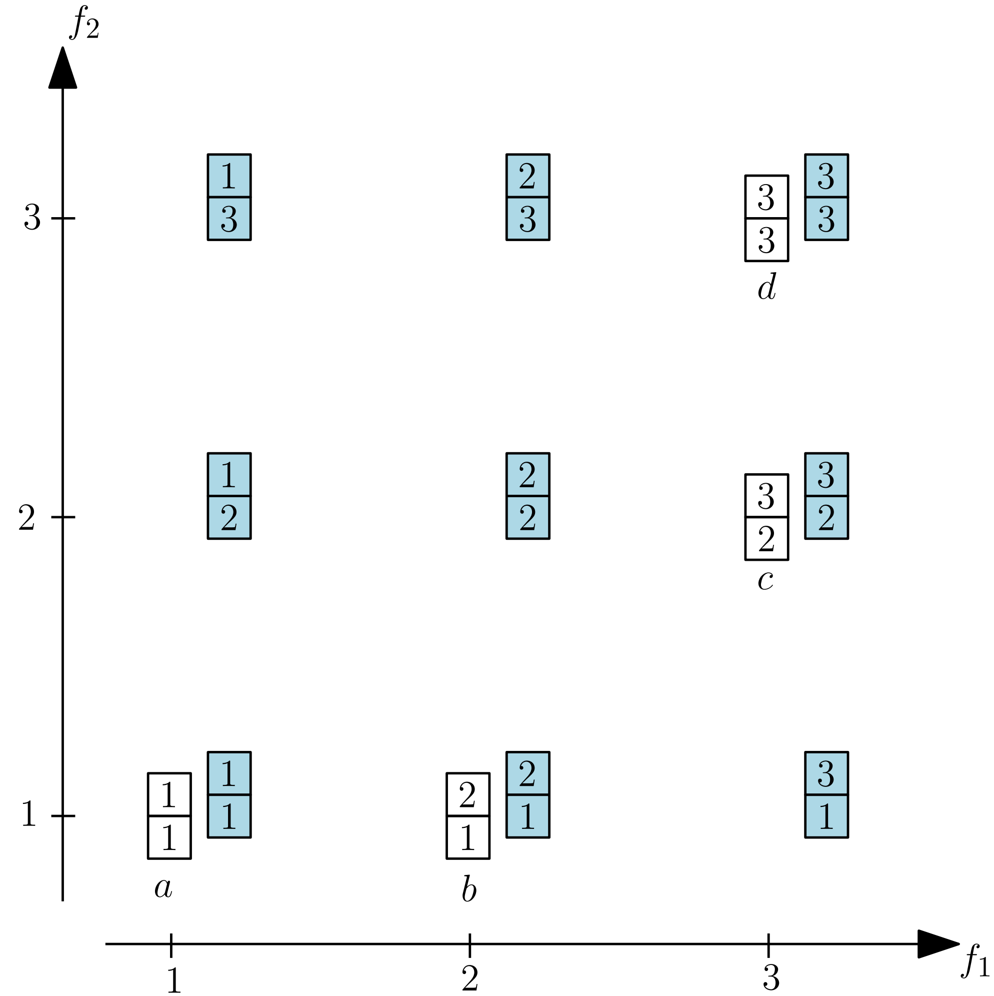
Radius 3 and two data values

Theorem

$\textsc{Sat}(\exists\text{-}3\text{-}\mathsf{LF}_2)$ is undecidable.

Key idea for the proof:

- Some elements can be far apart (e.g. a and d)
- We can always add element so the distance is ≤ 3



The Existential Fragment - Negative Results

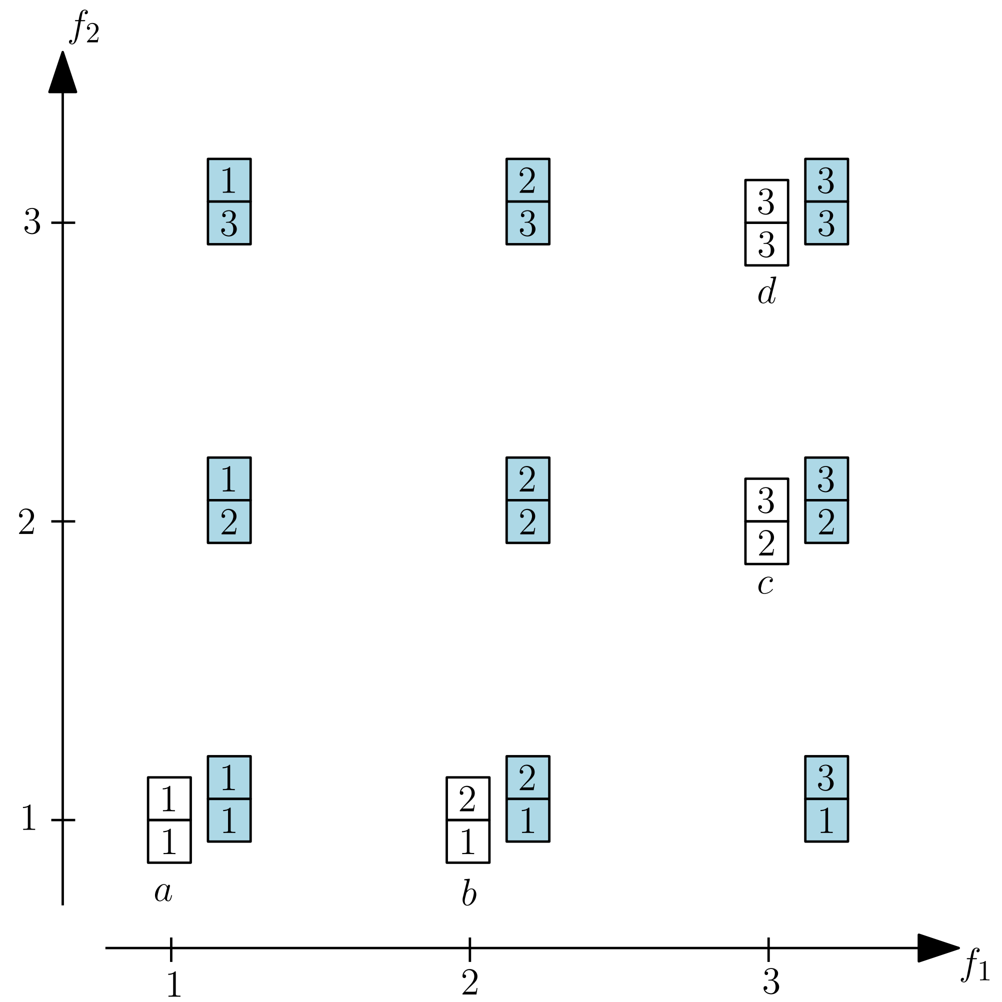
Radius 3 and two data values

Theorem

$\textsc{Sat}(\exists\text{-}3\text{-}\mathsf{LF}_2)$ is undecidable.

Key idea for the proof:

- Some elements can be far apart (e.g. a and d)
- We can always add element so the distance is ≤ 3
- The 3-view is the whole structure

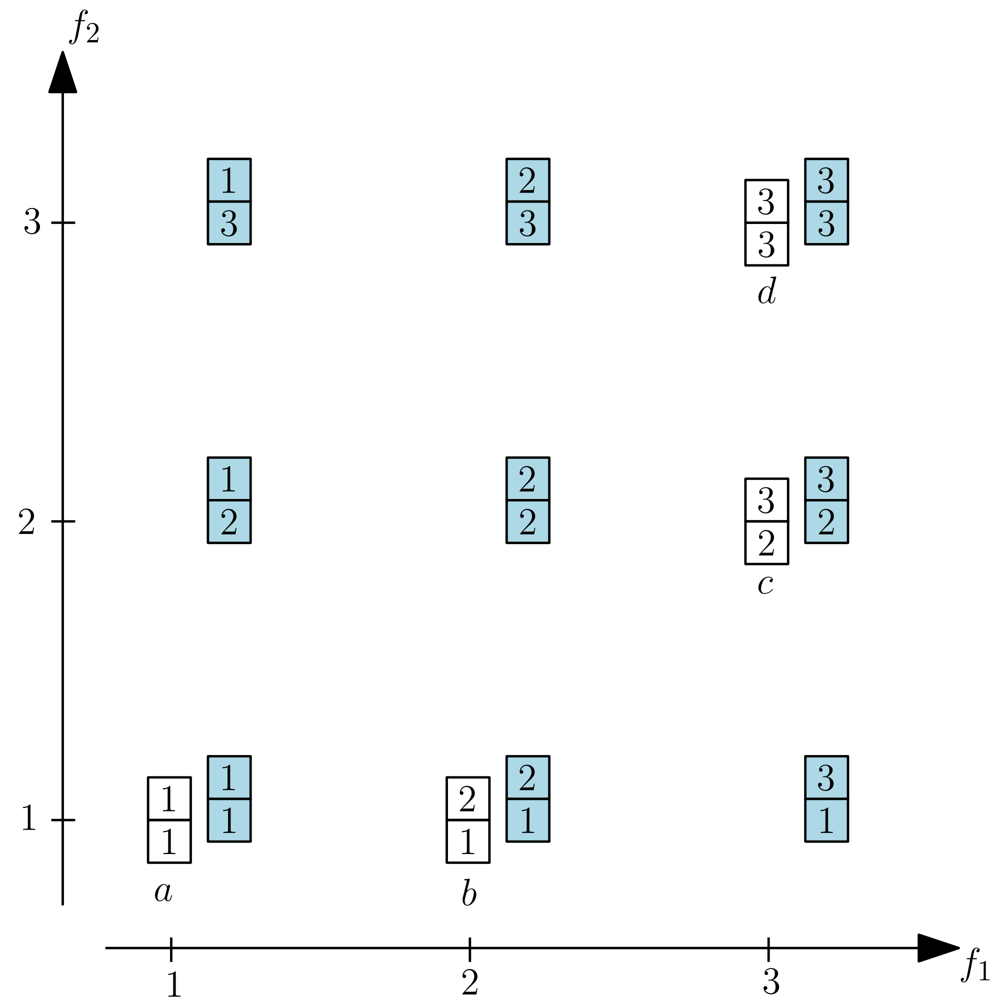


The Existential Fragment - Negative Results

Radius 3 and two data values

Theorem

SAT($\exists$-3-LF_2) is undecidable.



Key idea for the proof:

- Some elements can be far apart (e.g. a and d)
- We can always add element so the distance is ≤ 3
- The 3-view is the whole structure

Radius 2 and three data values

Theorem

SAT($\exists$-2-LF_3) is undecidable.

The Existential Fragment - Conclusion

⋮	⋮	⋮	⋮	⋮	
4	✓	✗	✗	✗	...
3	✓	✗	✗	✗	...
2	✓	✓	✗	✗	...
1	✓	✓	✓	✓	...
Radius / *Data*	1	2	3	4	...

Outline :

Conclusion & Outlook

On this work:

- Resolve the case of $\textsc{Sat}(1\text{-}\mathsf{LF}_2)$ without restriction
- Try to apply this work for verification
- Study effective decidability and approximation methods

Broadly:

- Many theoretical results claim that (almost) nothing is possible in distributed computing, while in practice, it's ubiquitous in our everyday life.
  $\rightarrow$ try to decrease the size of this gap
  $\rightarrow$ find better paradigms

Recap

Logic	**r**	κ	Relations	**Decidability status**
FO_κ	$-$	0	$\emptyset$	NEXP-complete
	$-$	1	$\{_1\sim_1\}$	N2EXP-complete
	$-$	2	$\{_1\sim_1, _2\sim_2\}$	Undecidable
r-LF_κ	1	2	$\{_1\sim_1, _2\sim_2, _1\sim_2\}$	Decidable
	1	2	$\{_1\sim_1, _2\sim_2, _2\sim_1\}$	Decidable
	2	2	$\{_1\sim_1, _2\sim_2, _1\sim_2\}$	Undecidable
	3	2	$\{_1\sim_1, _2\sim_2\}$	Undecidable
$\exists$-r-LF_κ	1	≥ 1	$\{_1\sim_1\}$	NEXP-complete
	2	2	$\{_1\sim_1, _2\sim_2, _1\sim_2, _2\sim_1\}$	N2EXP-complete
	3	2	$\{_1\sim_1, _2\sim_2, _1\sim_2, _2\sim_1\}$	Undecidable
	2	3	All_3	Undecidable