Two-variable first-order logic on ordered structures

Thomas Zeume
Ruhr University Bochum
Part I: Introduction
How to organize grandma’s 80th birthday... (1/2)

- Planning of grandma’s 80th birthday dinner
- Guests from subfamilies: Red, Yellow and Blue

Banquet

Thomas Zeume

Two-variable first-order logic on ordered structures
How to organize grandma’s 80th birthday... (1/2)

- Planning of grandma’s 80th birthday dinner
- Guests from subfamilies: Red, Yellow and Blue

Conditions on banquet placement

- Mixing of subfamilies

Banquet
How to organize grandma’s 80th birthday... (1/2)

- Planning of grandma’s 80th birthday dinner
- Guests from subfamilies: Red, Yellow and Blue

### Conditions on banquet placement

- **Mixing of subfamilies**
  - The blue and yellow subfamilies do not like each other

- **Youth protection**
  - Ages: 80, 79, 80, 78, 10, 10

---

Thomas Zeume

Two-variable first-order logic on ordered structures
How to organize grandma’s 80th birthday...

- Planning of grandma’s 80th birthday dinner
- Guests from subfamilies: Red, Yellow and Blue

### Conditions on banquet placement

- **Mixing of subfamilies**
  - Red (r) and Red (r) are not mixed.
  - Blue (b) and Blue (b) are not mixed.
  - Yellow (y) and Yellow (y) are not mixed.

- **Youth protection**
  - No one under 80 is mixed with someone over 80.
  - No one under 10 is mixed with someone over 10.

- The blue and yellow subfamilies do not like each other.
How to organize grandma’s 80th birthday... (2/2)

- **Question:** Can we find a guest list such that there is a placement that makes all persons happy?

- What if conditions get more complex?
How to organize grandma’s 80th birthday... (2/2)

• Question: Can we find a guest list such that there is a placement that makes all persons happy?

What if conditions get more complex?

Guest list problem

• Given:

- A set of placement conditions

• Question: Is there a guest list such that there is a placement making all guests happy?

• Meta-Problem: How to apply for research funding for solving grandma’s problems?

Logic to the rescue!
**Question:** Can we find a guest list such that there is a placement that makes all persons happy?

<table>
<thead>
<tr>
<th>10 b</th>
<th>12 r</th>
</tr>
</thead>
<tbody>
<tr>
<td>17 r</td>
<td>12 r</td>
</tr>
<tr>
<td>40 b</td>
<td>25 b</td>
</tr>
<tr>
<td>40 r</td>
<td>43 b</td>
</tr>
<tr>
<td>65 y</td>
<td>77 r</td>
</tr>
<tr>
<td>80 r</td>
<td>78 y</td>
</tr>
<tr>
<td>80 b</td>
<td>80 r</td>
</tr>
</tbody>
</table>

**What if conditions get more complex?**
How to organize grandma’s 80th birthday... (2/2)

• Question: Can we find a guest list such that there is a placement that makes all persons happy?

Guest list problem

• Given: A set of placement conditions
• Question: Is there a guest list such that there is a placement making all guests happy?

What if conditions get more complex?
• **Question:** Can we find a guest list such that there is a placement that makes all persons happy?

**Guest list problem**

- **Given:** A set of placement conditions
- **Question:** Is there a guest list such that there is a placement making all guests happy?

- **Meta-Problem:** How to apply for research funding for solving grandma’s problems?

● What if conditions get more complex?
How to organize grandma’s 80th birthday... (2/2)

- **Question:** Can we find a guest list such that there is a placement that makes all persons happy?

**Guest list problem**

- **Given:** A set of placement conditions
- **Question:** Is there a guest list such that there is a placement making all guests happy?

- **Meta-Problem:** How to apply for research funding for solving grandma’s problems?
  - Logic to the rescue!

---

Thomas Zeume

Two-variable first-order logic on ordered structures
A simple model: Ordered structures

Banquets as structures

Ordered structures for us:
- Orders
- possibly successor relations
- Additional uninterpreted relations
- Orders: linear orders and/or preorders
- Basic variants in 2-dimensions:
  - $\leq_2 \preceq_2 \preceq_1 \leq_1$
- $(\leq_1; \preceq_2)$-structures can be seen as ordered data strings [Bouyer et al. '03]

Question: How to model conditions?
- First try: First-order logic
  - Problem: Too strong
- Second try: Two-variable fragment of first-order logic
A simple model: Ordered structures

Banquets as structures

Ordered structures for us:

- Orders
- Possibly successor relations
- Additional uninterpreted relations

Orders:
- Linear orders and/or preorders

Basic variants in 2-dimensions:

- (≤₁, ≺₂)
  - Structures can be seen as (or-)ordered data strings [Bouyer et al. ’03]

Question: How to model conditions?

First try:
- First-order logic
  - Problem: Too strong

Second try:
- Two-variable fragment of first-order logic
A simple model: Ordered structures

Banquets as structures

Orders:
- linear orders
- preorders

Basic variants in 2-dimensions:

\((\leq_1; \prec_2)\) - structures can be seen as ordered data strings [Bouyer et al. '03]

Question: How to model conditions?

First try: First-order logic

Problem: Too strong

Second try: Two-variable fragment of first-order logic
A simple model: Ordered structures

Banquets as structures

- Ordered structures for us:
  - Orders + possibly successor relations
  - Additional uninterpreted relations
  - Orders: **linear orders** and/or **preorders**

Two-variable first-order logic on ordered structures
A simple model: Ordered structures

Banquets as structures

Ordered structures for us:
- Orders + possibly successor relations
- Additional uninterpreted relations
- Orders: linear orders and/or preorders

Basic variants in 2-dimensions:

orders: (≤₁, ≤₂) and (≤₂, ≤₁)

Question: How to model conditions?
First try: First-order logic
Problem: Too strong
Second try: Two-variable fragment of first-order logic

Thomas Zeume
Two-variable first-order logic on ordered structures
A simple model: Ordered structures

**Banquets as structures**

<table>
<thead>
<tr>
<th>80</th>
<th>80</th>
<th>65</th>
<th>40</th>
<th>40</th>
<th>17</th>
<th>10</th>
</tr>
</thead>
</table>

Ordered structures for us:
- Orders + possibly successor relations
- Additional uninterpreted relations

Orders: **linear orders** and/or **preorders**

Basic variants in 2-dimensions:

- \( \leq_2 \)
- \( \prec_2 \)

\((\leq_1; \prec_2)\)-structures can be seen as (ordered) data strings [Bouyer et al. '03]

**Question:** How to model conditions?
- First try: First-order logic
- Problem: Too strong
- Second try: Two-variable fragment of first-order logic

Thomas Zeume

Two-variable first-order logic on ordered structures
A simple model: Ordered structures

- Ordered structures for us:
  - Orders + possibly successor relations
  - Additional uninterpreted relations
- Orders: linear orders and/or preorders

- Basic variants in 2-dimensions:

- $(\leq_1; \prec_2)$-structures can be seen as (ordered) data strings [Bouyer et al. '03]

- Question: How to model conditions?
A simple model: Ordered structures

Banquets as structures

80 80 65 40 40 17 10
80 78 77 43 25 12 12

Ordered structures for us:
- Orders + possibly successor relations
- Additional uninterpreted relations
- Orders: linear orders and/or preorders

Basic variants in 2-dimensions:

(≤₂; <₂)-structures can be seen as (ordered) data strings [Bouyer et al. '03]

Question: How to model conditions?
- First try: First-order logic
A simple model: Ordered structures

Banquets as structures

80  80  65  40  40  17  10

80  78  77  43  25  12  12

Ordered structures for us:
• Orders + possibly successor relations
• Additional uninterpreted relations
• Orders: linear orders and/or preorders

Basic variants in 2-dimensions:
• \( \leq_2 \)
• \( \prec_2 \)
• \( \leq_1 \)
• \( \prec_1 \)

\((\leq_1; \prec_2)\)-structures can be seen as (ordered) data strings \([\text{Bouyer et al. '03}]\)

Question: How to model conditions?
• First try: First-order logic
⇒ Problem: Too strong
A simple model: Ordered structures

- Ordered structures for us:
  - Orders + possibly successor relations
  - Additional uninterpreted relations
- Orders: linear orders and/or preorders

- Basic variants in 2-dimensions:
  - \((\leq_1; \prec_2)\)-structures can be seen as (ordered) data strings \cite{BouyerEtAl2003}

- Question: How to model conditions?
  - First try: First-order logic
    - Problem: Too strong
  - Second try: Two-variable fragment of first-order logic
Two-variable first-order logic

- **Two-variable first-order logic (FO$^2$):**
  First-order logic restricted to two variables $x$ and $y$ (that can be reused)

Example

- "From all red nodes one can reach a blue node in three steps":
  $$\forall x \exists y \exists x \exists y \ldots$$

- FO$^2$ cannot express
  - A structure has at least three elements
  - A binary relation is transitive
  - Proof: via Ehrenfeucht-Fra"ıssé games with two pebbles
  - FO$^2$ with a linear order $<$ is more expressive
  - "An ordered structure has at least three elements":
    $$\exists x \exists y (x < y \land \exists x (y < x))$$
Two-variable first-order logic

- **Two-variable first-order logic (FO²):**
  First-order logic restricted to two variables \(x\) and \(y\) (that can be reused)

**Example**
- “From all red nodes one can reach a blue node in three steps”:
Two-variable first-order logic

- **Two-variable first-order logic (FO²):**
  First-order logic restricted to two variables \(x\) and \(y\) (that can be reused)

**Example**

- “From all red nodes one can reach a blue node in three steps”:

\[
\forall x \exists y \exists x \exists y \exists y \exists y \exists y \exists y \exists y \exists y \\
\exists x \exists y (x < y \land \exists x (y < x))
\]
Two-variable first-order logic

- **Two-variable first-order logic (\(\text{FO}^2\)):** First-order logic restricted to two variables \(x\) and \(y\) (that can be reused)

**Example**

- “From all red nodes one can reach a blue node in three steps”:

\[
\forall x \quad \exists y \quad \exists x \quad \exists y \quad \bullet
\]

\[\exists x \quad \exists y \quad (x < y \land \exists x) \quad (y < x)\]
**Two-variable first-order logic**

- **Two-variable first-order logic (FO$^2$):**
  
  First-order logic restricted to two variables $x$ and $y$ (that can be reused)

**Example**

- “From all red nodes one can reach a blue node in three steps”:

  \[
  \forall x \exists y \exists x \exists y \left( R(x) \rightarrow \exists y (E(x, y) \wedge \exists x (E(x, y) \wedge \exists y (E(x, y) \wedge B(y)))) \right)
  \]

- FO$^2$ cannot express

  - A structure has at least three elements
  - A binary relation is transitive

- Proof: via Ehrenfeucht–Fraïssé games with two pebbles

- FO$^2$ with a linear order $<$ is more expressive

- “An ordered structure has at least three elements”:

  \[
  \exists x \exists y (x < y \wedge \exists x (y < x))
  \]
Two-variable first-order logic

- **Two-variable first-order logic (FO²):** First-order logic restricted to two variables $x$ and $y$ (that can be reused)

**Example**

- "From all red nodes one can reach a blue node in three steps":

\[
\forall x \exists y \exists x \exists y (R(x) \rightarrow \exists y (E(x,y) \land \exists x (E(x,y) \land B(y))))
\]

FO² cannot express
- A structure has at least three elements
- A binary relation is transitive

Proof: via Ehrenfeucht–Fraïssé games with two pebbles

FO² with a linear order $<$ is more expressive

"An ordered structure has at least three elements":

\[
\exists x \exists y (x < y \land \exists x (y < x))
\]
Two-variable first-order logic (FO²):

First-order logic restricted to two variables $x$ and $y$ (that can be reused)

Example

“From all red nodes one can reach a blue node in three steps”:

As formula:

$$\forall x \exists y \exists x \exists y (R(x) \rightarrow \exists y (E(x, y) \land \exists x (E(x, y) \land \exists y (E(x, y) \land B(y))))$$

• FO² cannot express
  • A structure has at least three elements
  • A binary relation is transitive

Proof: via Ehrenfeucht–Fraïssé games with two pebbles

• FO² with a linear order $<$ is more expressive

“An ordered structure has at least three elements”:

$$\exists x \exists y (x < y \land \exists x (y < x))$$
Two-variable first-order logic

- **Two-variable first-order logic (FO²):**
  First-order logic restricted to two variables $x$ and $y$ (that can be reused)

**Example**

- “From all red nodes one can reach a blue node in three steps”:

  $\forall x \exists y \exists x \exists y \exists y$  
  $\exists y (E(x, y) \land \exists x (E(x, y) \land B(y))))$  

- As formula:

  $\forall x (R(x) \rightarrow \exists y (E(x, y) \land \exists x (E(x, y) \land B(y))))$
Two-variable first-order logic

- **Two-variable first-order logic (FO\(^2\)):**
  First-order logic restricted to two variables \(x\) and \(y\) (that can be reused)

**Example**

- “From all red nodes one can reach a blue node in three steps”:

  \[
  \forall x \exists y \exists x \exists y (E(x, y) \land E(x, y) \land E(x, y) \land B(y))
  \]

- **As formula:**

  \[
  \forall x (R(x) \rightarrow \exists y (E(x, y) \land \exists x (E(x, y) \land \exists y (E(x, y) \land B(y)))) )
  \]

- **FO\(^2\) cannot express**
  - A structure has at least three elements
  - A binary relation is transitive
Two-variable first-order logic

- **Two-variable first-order logic ($\text{FO}^2$):** First-order logic restricted to two variables $x$ and $y$ (that can be reused)

**Example**
- “From all red nodes one can reach a blue node in three steps”:

  ![Diagram](image)

  - As formula:
    \[
    \forall x \left( R(x) \rightarrow \exists y \left( E(x, y) \land \exists x \left( E(x, y) \land \exists y \left( E(x, y) \land B(y) \right) \right) \right) \right)
    \]

**FO$^2$ cannot express**
- A structure has at least three elements
- A binary relation is transitive
- Proof: via Ehrenfeucht-Fraïssé games with two pebbles

Two-variable first-order logic on ordered structures
Two-variable first-order logic

- **Two-variable first-order logic (FO²):**
  - First-order logic restricted to two variables $x$ and $y$ (that can be reused)

**Example**

- “From all red nodes one can reach a blue node in three steps”:

$$\forall x \exists y \exists x \exists y (\exists x)$$

- As formula:

$$\forall x \left( R(x) \rightarrow \exists y \left( E(x, y) \wedge \exists y \left( E(x, y) \wedge B(y) \right) \right) \right)$$

- **FO² cannot express**
  - A structure has at least three elements
  - A binary relation is transitive
  - Proof: via Ehrenfeucht-Fraïssé games with two pebbles

- **FO² with a linear order $<$ is more expressive**
  - “An ordered structure has at least three elements”:

$$\exists x \exists y \left( x < y \wedge \exists x (y < x) \right)$$
Part II:

Satisfiability of $\mathbf{FO}^2$
on ordered structures
Satisfiability of two-variable logic

**Theorem 1** [Grädel, Kolaitis, Vardi '97]

(Finite) Satisfiability of $\text{FO}^2$ on general structures is $\text{NEXPTIME}$-complete.
Satisfiability of two-variable logic

**Theorem 1**  
[Grädel, Kolaitis, Vardi '97]  
(Finite) Satisfiability of $\text{FO}^2$ on general structures is $\text{NEXPTIME}$-complete

**Question:** Is this the end of the story?
Satisfiability of two-variable logic

**Theorem 1** [Grädel, Kolaitis, Vardi '97]

(Finite) Satisfiability of $\text{FO}^2$ on general structures is $\text{NEXPTIME}$-complete

- **Question:** Is this the end of the story?
  
  - *No!* Decidable satisfiability of $\text{FO}^2$ on *general structures* does not transfer to *restricted structures*

---

Thomas Zeume

Two-variable first-order logic on ordered structures
Satisfiability of two-variable logic

**Theorem 1** [Grädel, Kolaitis, Vardi '97]

(Finite) Satisfiability of $\text{FO}^2$ on general structures is $\text{NEXPTIME}$-complete

- **Question:** Is this the end of the story?
  - **No!** Decidable satisfiability of $\text{FO}^2$ on **general structures** does not transfer to **restricted structures**
  - **The crux in our case:** Transitivity cannot be axiomatized in $\text{FO}^2$
Theorem 1 [Grädel, Kolaitis, Vardi '97]

(Finite) Satisfiability of $\mathsf{FO}^2$ on general structures is $\mathsf{NEXPTIME}$-complete

• **Question:** Is this the end of the story?
  
  ➔ **No!** Decidable satisfiability of $\mathsf{FO}^2$
  
  on **general structures** does not transfer to **restricted structures**

  ➔ **The crux in our case:** Transitivity cannot be axiomatized in $\mathsf{FO}^2$

• This lead to an extensive study of the satisfiability problem for special structures

Satisfiability of two-variable logic
Theorem 1 [Grädel, Kolaitis, Vardi '97]

(Finite) Satisfiability of $\mathbf{FO}^2$ on general structures is $\text{NEXPTIME}$-complete

- **Question:** Is this the end of the story?
  - **No!** Decidable satisfiability of $\mathbf{FO}^2$ on general structures does not transfer to restricted structures
  - **The crux in our case:** Transitivity cannot be axiomatized in $\mathbf{FO}^2$

- This lead to an extensive study of the satisfiability problem for special structures

Theorem 2: [Kieronski, Otto '05]

$\mathbf{FO}^2$ with equivalence relations

(Finite) Satisfiability of $\mathbf{FO}^2$ on structures

- with two equivalence relations: decidable
- with three equivalence relations: undecidable

Theorem 3: [Kieronski '11]

$\mathbf{FO}^2$ with linear orders

(Finite) Satisfiability of $\mathbf{FO}^2$ on structures

- with two linear orders: decidable
- with three linear orders: undecidable

[Schwentick, Z. '10; Torunczyk, Z. '20]
Satisfiability of two-variable logic

**Theorem 1**  
[Grädel, Kolaitis, Vardi '97]  
(Finite) Satisfiability of $\text{FO}^2$ on general structures is $\text{NEXPTIME}$-complete

- **Question:** Is this the end of the story?  
  → No! Decidable satisfiability of $\text{FO}^2$ on general structures does not transfer to restricted structures  
  → The crux in our case: Transitivity cannot be axiomatized in $\text{FO}^2$

- This lead to an extensive study of the satisfiability problem for special structures

**Theorem 2:**  
[Kieroński, Otto '05]  
$\text{FO}^2$ with equivalence relations  
(Finite) Satisfiability of $\text{FO}^2$ on structures  
- with two equivalence relations: decidable  
- with three equivalence relations: undecidable

**Theorem 3:**  
$\text{FO}^2$ with linear orders  
(Finite) Satisfiability of $\text{FO}^2$ on structures  
- with two linear orders: decidable  
  [Schwentick, Z. '10; Torunczyk, Z. '20]  
- with three linear orders: undecidable  
  [Kieronski '11]

Thomas Zeume  
Two-variable first-order logic on ordered structures
### Satisfiability of $\text{FO}^2$ on ordered structures

#### Finite Satisfiability:

<table>
<thead>
<tr>
<th></th>
<th>$S_2$</th>
<th>$S_2, &lt;_2$</th>
<th>$&lt;_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_2$</td>
<td>decidable</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>[Charatonik, Witkowski '13] [Manuel '10]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$S_2, &lt;_2$</td>
<td>decidable</td>
<td>undecidable</td>
<td></td>
</tr>
<tr>
<td></td>
<td>[Z., Harwath '16] [Manuel, Z. '16]</td>
<td>[Manuel '10]</td>
<td></td>
</tr>
<tr>
<td>$&lt;_2$</td>
<td>decidable</td>
<td>decidable</td>
<td>decidable</td>
</tr>
<tr>
<td></td>
<td>[Z., Harwath '16] [Manuel, Z. '13]</td>
<td>[Z., Harwath '16] [Schwentick, Z. '10]</td>
<td>[Z., Harwath '16] [Schwentick, Z. '10]</td>
</tr>
</tbody>
</table>

General Satisfiability: $\text{satisfiability of } \text{FO}^2(<_1, <_2)$ is decidable [Torunczyk, Z. '20]
# Satisfiability of $\text{FO}^2$ on ordered structures

## Finite Satisfiability:

<table>
<thead>
<tr>
<th></th>
<th>$S_2$</th>
<th>$S_2, &lt;_2$</th>
<th>$&lt;_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_2$</td>
<td>decidable</td>
<td>[Charatonik, Witkowski '13] [Manuel '10]</td>
<td>[Z., Harwath '16] [Manuel, Z. '16]</td>
</tr>
<tr>
<td>$S_2, &lt;_2$</td>
<td>decidable</td>
<td>undecidable</td>
<td>[Manuel '10]</td>
</tr>
<tr>
<td>$&lt;_2$</td>
<td>decidable</td>
<td>decidable</td>
<td>decidable</td>
</tr>
<tr>
<td></td>
<td>[Z., Harwath '16] [Manuel, Z. '13]</td>
<td>[Z., Harwath '16] [Schwentick, Z. '10]</td>
<td>[Z., Harwath '16] [Schwentick, Z. '10]</td>
</tr>
</tbody>
</table>

## General Satisfiability:

Satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

[Torunczyk, Z. '20]
## Proof techniques: A toy example

<table>
<thead>
<tr>
<th>Theorem 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finite satisfiability of $\text{FO}^2(S, &lt;)$ is in $\text{NEXPTIME}$</td>
</tr>
</tbody>
</table>

### Proof

- Given a $\text{FO}^2(S, <)$-formula
### Theorem 4

Finite satisfiability of \( \text{FO}^2(S, <) \) is in \( \text{NEXPTIME} \)

### Proof

- Given a \( \text{FO}^2(S, <) \)-formula
- **Step 1:** Translation into exponentially many simple constraints
### Theorem 4
Finite satisfiability of $\text{FO}^2(S, <)$ is in $\text{NEXPTIME}$

#### Proof
- Given a $\text{FO}^2(S, <)$-formula

- **Step 1**: Translation into exponentially many simple constraints
  - Two types of constraints:
    - $\forall \exists$: For every $\bullet$-point there is a $\bullet$-point in direction $d$ connected via a yellow edge
    - $\forall \forall$: There is no $\bullet$-point in direction $d$ of a $\bullet$-point connected via a yellow edge
# Proof techniques: A toy example

## Theorem 4

Finite satisfiability of \( \text{FO}^2(S, <) \) is in \( \text{NEXPTIME} \)

### Proof

- Given a \( \text{FO}^2(S, <) \)-formula

- **Step 1:** Translation into exponentially many simple constraints

- Two types of constraints:
  - \( \forall \exists \): For every \( \bullet \)-point there is a \( \bullet \)-point in direction \( d \) connected via a yellow edge
  - \( \forall \forall \): There is no \( \bullet \)-point in direction \( d \) of a \( \bullet \)-point connected via a yellow edge

- Directions: \( d \in \{ \rightarrow, \leftrightarrow, \leftarrow, \rightarrow \} \)
Proof techniques: A toy example

**Theorem 4**

Finite satisfiability of $\text{FO}^2(S, <)$ is in $\text{NEXPTIME}$

**Proof**

- Given a $\text{FO}^2(S, <)$-formula

- **Step 1:** Translation into exponentially many simple constraints
  - Two types of constraints:
    - $\forall \exists$: For every blue point there is a red point in direction $d$ connected via a yellow edge
    - $\forall \forall$: There is no blue point in direction $d$ of a red point connected via a yellow edge
  - Directions: $d \in \{S, \rightarrow, S, \leftrightarrow\}$

- **Step 2:** If there is a model, then there is a small model
**Theorem 4**

Finite satisfiability of $\mathbf{FO}^2(S, <)$ is in $\text{NEXPTIME}$

**Proof**

- Given a $\mathbf{FO}^2(S, <)$-formula

**Step 1:** Translation into exponentially many simple constraints

- Two types of constraints:
  - $\forall \exists$: For every blue point there is a red point in direction $d$ connected via a yellow edge
  - $\forall \forall$: There is no blue point in direction $d$ of a red point connected via a yellow edge

- Directions: $d \in \{\rightarrow, \rightarrow, \leftarrow, \leftarrow\}$

**Step 2:** If there is a model, then there is a small model

**Example**

- $\forall \exists : \bullet \rightarrow \circ$
- $\forall \forall : \bullet \leftarrow \circ$
Proof techniques: A toy example

**Theorem 4**

Finite satisfiability of $\text{FO}^2(S, <)$ is in \text{NEXPTIME}

**Proof**

- Given a $\text{FO}^2(S, <)$-formula
- **Step 1:** Translation into exponentially many simple constraints
  - Two types of constraints:
    - $\forall \exists$: For every blue point there is a red point in direction $d$ connected via a yellow edge
    - $\forall \forall$: There is no blue point in direction $d$ of a red point connected via a yellow edge
  - directions: $d \in \{\rightarrow, \to, \leftarrow, \leftrightarrow\}$
- **Step 2:** If there is a model, then there is a small model

**Example**

- $\forall \exists \rightarrow$
- $\forall \exists \leftarrow$

**Goal:** Small model in two steps
Proof techniques: A toy example

**Theorem 4**

Finite satisfiability of $\text{FO}^2(S, <)$ is in $\text{NEXPTIME}$

**Proof**

- **Given a $\text{FO}^2(S, <)$-formula**

  **Step 1:** Translation into exponentially many simple constraints
  - Two types of constraints:
    - $\forall\exists$: For every $\bullet$-point there is a $\circ$-point in direction $d$ connected via a yellow edge
    - $\forall\forall$: There is no $\bullet$-point in direction $d$ of a $\circ$-point connected via a yellow edge
  - directions: $d \in \{\rightarrow, \rightarrow, \leftarrow, \leftarrow\}$

- **Step 2:** If there is a model, then there is a small model

**Example**

- $\forall\exists\circ : \bullet \rightarrow \circ$
- $\forall\forall\bullet : \bullet \leftarrow \circ$

**Goal:** Small model in two steps
- Rewire witnesses
Proof techniques: A toy example

**Theorem 4**
Finite satisfiability of $\text{FO}^2(S, <)$ is in NEXPTIME

**Proof**
- Given a $\text{FO}^2(S, <)$-formula

**Step 1:** Translation into exponentially many simple constraints
- Two types of constraints:
  - $\forall \exists$: For every blue point there is a red point in direction $d$ connected via a yellow edge
  - $\forall \exists$: There is no blue point in direction $d$ of a red point connected via a yellow edge
- Directions: $d \in \{S, \rightarrow, S, \leftarrow\}$

**Step 2:** If there is a model, then there is a small model

**Example**
- $\forall \exists \circ \rightarrow \bullet$
- $\forall \exists \circ \leftarrow \bullet$

**Goal:** Small model in two steps
- Rewire witnesses

Thomas Zeume
Two-variable first-order logic on ordered structures
Proof techniques: A toy example

**Theorem 4**

Finite satisfiability of $\text{FO}^2(S, <)$ is in \text{NEXPTIME}

**Proof**

- **Given a $\text{FO}^2(S, <)$-formula**
- **Step 1:** Translation into exponentially many simple constraints
  - Two types of constraints:
    - $\forall \exists$: For every blue-point there is a red-point in direction $d$ connected via a yellow edge
    - $\forall \forall$: There is no blue-point in direction $d$ of a red-point connected via a yellow edge
  - directions: $d \in \{\rightarrow, \leftarrow, \leftarrow, \rightarrow\}$
- **Step 2:** If there is a model, then there is a small model
  - Example
    - $\forall \exists$:
    - $\forall \forall$:

**Goal:** Small model in two steps

- Rewire witnesses
- Shrink the model using pumping
Proof methods: Reductions, small models, and automata

- **Undecidability**
  via reductions from PCP, 2-counter machines,…
Proof methods: Reductions, small models, and automata

- **Undecidability**
  via reductions from PCP, 2-counter machines,…
- **Decidability** via
  - Small models
Proof methods: Reductions, small models, and automata

- **Undecidability** via reductions from PCP, 2-counter machines,…

- **Decidability** via

  - Small models
  - Automata
Proof methods: Reductions, small models, and automata

- **Undecidability** via reductions from PCP, 2-counter machines, ...
- **Decidability** via

![Diagram](image_url)
Proof methods: Reductions, small models, and automata

- **Undecidability**
  via reductions from PCP, 2-counter machines,…

- **Decidability** via
  
  ![Diagram with Small models and Automata]

- **Now**: A glimpse at the proofs for decidability of
  
  - finite satisfiability of $\text{FO}^2(<_1, <_2)$ via small models
  - (general) satisfiability of $\text{FO}^2(<_1, <_2)$ via automata
Interlude: Simplified signatures for FO(\(<1, <2\))

- **Observation 1:** Signatures can be restricted to symbols of arity \(\leq 2\)
Interlude: Simplified signatures for $\text{FO}(\prec_1, \prec_2)$

- **Observation 1:** Signatures can be restricted to symbols of arity $\leq 2$
- **Observation 2:** Sometimes even restriction to arity $\leq 1$ is possible, for non-special symbols!
Interlude: Simplified signatures for FO($\lt_1, \lt_2$)

- **Observation 1**: Signatures can be restricted to symbols of arity $\leq 2$
- **Observation 2**: Sometimes even restriction to arity $\leq 1$ is possible, for non-special symbols!

**Example: Cloning**

- Making a copy of an element:
Interlude: Simplified signatures for FO($<_1, <_2$)

- **Observation 1:** Signatures can be restricted to symbols of arity $\leq 2$
- **Observation 2:** Sometimes even restriction to arity $\leq 1$ is possible, for non-special symbols!

Example: Cloning
- Making a copy of an element:
**Interlude: Simplified signatures for FO(\(<_1, <_2\))**

- **Observation 1:** Signatures can be restricted to symbols of arity \(\leq 2\)
- **Observation 2:** Sometimes even restriction to arity \(\leq 1\) is possible, for non-special symbols!

**Example: Cloning**

- Making a copy of an element:

  \[
  <_1 \quad <_2
  \]

  \[
  a \quad a'
  \]

- Satisfiability translates:
  - \(\forall\forall\)-constraints:
  - \(\forall\exists\)-constraints:
Interlude: Simplified signatures for FO(\(<_1, <_2\))

- **Observation 1:** Signatures can be restricted to symbols of arity \(\leq 2\)
- **Observation 2:** Sometimes even restriction to arity \(\leq 1\) is possible, for non-special symbols!

**Example: Cloning**

- Making a copy of an element:

\[
\begin{align*}
\langle 2 \rangle & \quad \langle 2 \rangle \\
\langle 1 \rangle & \quad \langle 1 \rangle
\end{align*}
\]

\[
\begin{array}{c}
\bullet a \\
\bullet a'
\end{array}
\]

- Satisfiability translates:
  - \(\forall\forall\)-constraints: ✓
  - \(\forall\exists\)-constraints:
**Interlude: Simplified signatures for FO(<1, <2>)**

- **Observation 1:** Signatures can be restricted to symbols of arity ≤ 2
- **Observation 2:** Sometimes even restriction to arity ≤ 1 is possible, for non-special symbols!

**Example: Cloning**

- Making a copy of an element:

![Diagram showing cloning](image)

- Satisfiability translates:
  - ∀∀-constraints: √
  - ∀∃-constraints: √
Interlude: Simplified signatures for $\text{FO}(<_1, <_2)$

- **Observation 1**: Signatures can be restricted to symbols of arity $\leq 2$
- **Observation 2**: Sometimes even restriction to arity $\leq 1$ is possible, for non-special symbols!

**Example: Cloning**
- Making a copy of an element:
  - $<_1$ $<_2$

- Satisfiability translates:
  - $\forall \forall$-constraints: ✓
  - $\forall \exists$-constraints: ✓

**Lemma 5** [Pratt-Hartmann ’18; Torunczyk, Z. ’22]
- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$
Interlude: Simplified signatures for FO$(\langle 1, \langle 2 \rangle)$

- **Observation 1**: Signatures can be restricted to symbols of arity $\leq 2$
- **Observation 2**: Sometimes even restriction to arity $\leq 1$ is possible, for non-special symbols!

Example: Cloning

- Making a copy of an element:

  \[
  \begin{align*}
  \langle 2 \rangle
  &\quad
  \begin{array}{c}
  \begin{array}{c}
  \langle 1 \rangle
  \end{array}
  \end{array}
  \\
  \langle 2 \rangle
  &\quad
  \begin{array}{c}
  \begin{array}{c}
  \langle 1 \rangle
  \end{array}
  \end{array}
  \\
  a
  &\quad
  a'
  \\
  \end{align*}
  \]

- Satisfiability translates:
  - $\forall\forall$-constraints: ✓
  - $\forall\exists$-constraints: ✓

Lemma 5 [Pratt-Hartmann '18; Torunczyk, Z. '22]

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$
- If $\mathcal{K}$ allows “cloning”, then

Two-variable first-order logic on ordered structures
Interlude: Simplified signatures for $\text{FO}(\prec_1, \prec_2)$

- **Observation 1**: Signatures can be restricted to symbols of arity $\leq 2$
- **Observation 2**: Sometimes even restriction to arity $\leq 1$ is possible, for non-special symbols!

**Example: Cloning**

- Making a copy of an element:

  \[
  \begin{array}{c}
  \prec_2 \\
  \downarrow \\
  \prec_1 \\
  \end{array}
  \quad
  \begin{array}{c}
  \prec_2 \\
  \downarrow \\
  \prec_1 \\
  \end{array}
  \]

  \[
  \begin{array}{c}
  a \\
  \end{array}
  \quad
  \begin{array}{c}
  a' \\
  \end{array}
  \]

- Satisfiability translates:
  - $\forall\forall$-constraints: ✓
  - $\forall\exists$-constraints: ✓

**Lemma 5** [Pratt-Hartmann ’18; Torunczyk, Z. ’22]

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$
- If $\mathcal{K}$ allows “cloning”, then
  - from an $\text{FO}^2(\mathcal{K})$ formula over $\Delta \cup \Theta$
Interlude: Simplified signatures for $\text{FO}(\prec_1, \prec_2)$

- **Observation 1**: Signatures can be restricted to symbols of arity $\leq 2$
- **Observation 2**: Sometimes even restriction to arity $\leq 1$ is possible, for non-special symbols!

**Example: Cloning**

- Making a copy of an element:
  - $\prec_2$
  - $\prec_1$

- Satisfiability translates:
  - $\forall \forall$-constraints: ✓
  - $\forall \exists$-constraints: ✓

**Lemma 5** [Pratt-Hartmann '18; Torunczyk, Z. '22]

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$
- If $\mathcal{K}$ allows “cloning”, then
  - from an $\text{FO}^2(\mathcal{K})$ formula over $\Delta \uplus \Theta$
  - an equisatisfiable $\text{FO}^2(\mathcal{K})$ formula over signature $\Delta \uplus \Theta'$ can be constructed with unary $\Theta'$

Two-variable first-order logic on ordered structures
Interlude: Simplified signatures for \( \text{FO}(\prec_1, \prec_2) \)

- **Observation 1**: Signatures can be restricted to symbols of arity \( \leq 2 \)
- **Observation 2**: Sometimes even restriction to arity \( \leq 1 \) is possible, for non-special symbols!

**Example: Cloning**
- Making a copy of an element:

  - Satisfiability translates:
    - \( \forall \forall \)-constraints: ✓
    - \( \forall \exists \)-constraints: ✓

**Lemma 5** [Pratt-Hartmann '18; Torunczyk, Z. '22]
- Suppose \( \mathcal{K} \) is a class of structures with signature \( \Delta \)
- If \( \mathcal{K} \) allows “cloning”, then
  - from an \( \text{FO}^2(\mathcal{K}) \) formula over \( \Delta \cup \Theta \)
  - an equisatisfiable \( \text{FO}^2(\mathcal{K}) \) formula over signature \( \Delta \cup \Theta' \) can be constructed with unary \( \Theta' \)
- For us: \( \Delta = \{ \prec_1, \prec_2 \} \)
Interlude: Simplified signatures for $\text{FO}(\langle 1, 2 \rangle)$

- **Observation 1:** Signatures can be restricted to symbols of arity $\leq 2$
- **Observation 2:** Sometimes even restriction to arity $\leq 1$ is possible, for non-special symbols!

### Example: Cloning
- Making a copy of an element:

- Satisfiability translates:
  - $\forall\forall$-constraints: ✓
  - $\forall\exists$-constraints: ✓

---

**Lemma 5** [Pratt-Hartmann ’18; Torunczyk, Z. ’22]

- Suppose $\mathcal{K}$ is a class of structures with signature $\Delta$
- If $\mathcal{K}$ allows “cloning”, then
  - from an $\text{FO}^2(\mathcal{K})$ formula over $\Delta \cup \Theta$
  - an equisatisfiable $\text{FO}^2(\mathcal{K})$ formula over signature $\Delta \cup \Theta'$ can be constructed with unary $\Theta'$

- For us: $\Delta = \{ \langle 1, 2 \rangle \}$
- **Consequence:** Studying satisfiability of $\text{FO}^2(\langle 1, 2 \rangle)$ over unary signatures (apart from $\langle 1 \rangle$ and $\langle 2 \rangle$) suffices
Part Ila:

Decidability via small models
Theorem 6

Finite satisfiability of $\mathsf{FO}^2(<_1,<_2)$ is decidable

[Schwentick, Z. ’10]
**Theorem 6**  
[Schwentick, Z. ’10]  
Finite satisfiability of $\text{FO}^2(<_1,<_2)$ is is **decidable**  

**Proof idea**  
- **Step 1:** Translate $\text{FO}^2$-formulas into $\forall\exists$- and $\forall\forall$-constraints
Theorem 6 [Schwentick, Z. ’10]

Finite satisfiability of $\text{FO}^2(<_1,<_2)$ is decidable

Proof idea

- **Step 1:** Translate $\text{FO}^2$-formulas into $\forall\exists$- and $\forall\forall$-constraints

\[
\forall \exists \Omega A : \quad \downarrow
\]
**Theorem 6**  
[Schwentick, Z. ’10]  

Finite satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

**Proof idea**

- **Step 1:** Translate $\text{FO}^2$-formulas into $\forall \exists$- and $\forall \forall$-constraints

![Diagram](attachment:image.png)
**Theorem 6** [Schwentick, Z. ’10]

Finite satisfiability of $\text{FO}^2(\prec_1, \prec_2)$ is decidable.

**Proof idea**

- **Step 1:** Translate $\text{FO}^2$-formulas into $\forall\exists$- and $\forall\forall$-constraints.

**Example**

[Diagram showing the translation process and an example structure]
**Theorem 6** [Schwentick, Z. ’10]

Finite satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

**Proof idea**

- **Step 1:** Translate $\text{FO}^2$-formulas into $\forall\exists$- and $\forall\forall$-constraints

**Example**

- **Step 2:** Show that if there is a model, then there is a small model
Theorem 6 [Schwentick, Z. ’10]

Finite satisfiability of $\text{FO}^2(<_1,<_2)$ is decidable

Proof idea

- Step 1: Translate $\text{FO}^2$-formulas into $\forall\exists$- and $\forall\forall$-constraints

Example

- Step 2: Show that
  If there is a model, then there is a small model
**Theorem 6**  
[Schwentick, Z. ’10]

Finite satisfiability of $\mathsf{FO}^2(<_1, <_2)$ is decidable

**Proof idea**

- **Step 1:** Translate $\mathsf{FO}^2$-formulas into $\forall \exists$- and $\forall \forall$-constraints

$$\forall \exists \forall : \quad \neg \forall \forall :$$

**Example**

- **Step 2:** Show that  
  If there is a model, then there is a small model
Example (simplified)

- The profile of an element uses the following elements:
  - \( u \)
  - For every unary type:
    - the left and right most occurrences above \( u \)
    - the left and right most occurrences below \( u \)
  - The witnesses of those elements
- The profile is the sequence of those elements ordered by their x-coordinate.
\( \text{FO}^2 \) with two linear orders: Finite satisfiability (2/3)

**Example (simplified)**

- The **profile** of an element \( u \) uses the following elements:
  - \( u \),
  - For every unary type:
    - the \( 3|\Gamma| + 1 \) left and right most occurrences above \( u \)
    - the \( 3|\Gamma| + 1 \) left and right most occurrences below \( u \)
  - The witnesses of those elements
The profile of an element $u$ uses the following elements:

- $u$,
- For every unary type:
  - the $3|\Gamma| + 1$ left and right most occurrences above $u$
  - the $3|\Gamma| + 1$ left and right most occurrences below $u$
- The witnesses of those elements
The **profile** of an element $u$ uses the following elements:
- $u$,
- For every unary type:
  - the $3|\Gamma| + 1$ left and right most occurrences above $u$
  - the $3|\Gamma| + 1$ left and right most occurrences below $u$
- The witnesses of those elements
- The profile is the sequence of those elements ordered by their $x$-coordinate
The profile of an element $u$ uses the following elements:

- $u$,
- For every unary type:
  - the $3|\Gamma| + 1$ left and right most occurrences above $u$
  - the $3|\Gamma| + 1$ left and right most occurrences below $u$
- The witnesses of those elements

The profile is the sequence of those elements ordered by their $x$-coordinate

$(\circ, \uparrow)(\circ, \downarrow)(\circ, \downarrow)(\circ, \uparrow)(\circ, \downarrow)(\circ, \uparrow)(\circ, \uparrow)(\circ, \uparrow)$
A pumping argument:

If two elements \( u \) and \( v \) do have the same profile...
$\text{FO}^2$ with two linear orders: Finite satisfiability (3/3)

Example (simplified)

- A pumping argument:
  - If two elements $u$ and $v$ do have the same profile...
**FO² with two linear orders: Finite satisfiability (3/3)**

*Example (simplified)*

- **A pumping argument:**
  - If two elements $u$ and $v$ do have the same profile...
  - ...then everything between them can be removed
    - possibly some elements have to be shifted horizontally afterwards
  - possibly some witnesses have to be rewired
Example (simplified)

- A pumping argument:
  - If two elements $u$ and $v$ do have the same profile...
  - ...then everything between them can be removed
    - possibly some elements have to be shifted horizontally afterwards
    - possibly some witnesses have to be rewired
  - The obtained structure is again a model
\( \textbf{FO}^2 \text{ with two linear orders: Finite satisfiability} \ (3/3) \)

**Example (simplified)**

- A **pumping argument**:
  - If two elements \( u \) and \( v \) do have the same profile...
  - ...then everything between them can be removed
    - possibly some elements have to be shifted horizontally afterwards
    - possibly some witnesses have to be rewired
  - The obtained structure is again a model

\[ \Rightarrow \text{If there is a model, then there is a model with “few” elements} \]
Part IIb:

Decidability via automata
Theorem 7 [Torunczyk, Z. ’20]

Satisfiability of $\text{FO}^2(\prec_1, \prec_2)$ is decidable
**Theorem 7**  \[\text{[Torunczyk, Z. '20]}\]

Satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

**Proof**

- **Key idea 1:** Use register automata
Theorem 7 [Torunczyk, Z. '20]
Satisfiability of $\mathbf{FO}^2(\prec_1, \prec_2)$ is decidable

Proof
• Key idea 1: Use register automata

• Register automata:
  • Reads data words over $\Sigma \times D$
  • Finite state automata + registers
  • Transitions depend on
    • label and data value at current position
    • current state and register contents
**FO$^2$ with two linear orders: General satisfiability**

**Theorem 7** [Torunczyk, Z. ’20]

Satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

**Proof**

- Key idea 1: Use register automata

- Register automata:
  - Reads data words over $\Sigma \times D$
  - Finite state automata + registers
  - Transitions depend on
    - label and data value at current position
    - current state and register contents

**Example**

- $(<_1, <_2)$-structure:
Theorem 7 [Torunczyk, Z. ’20]

Satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

Proof

- **Key idea 1:** Use register automata
- **Register automata:**
  - Reads data words over $\Sigma \times D$
  - Finite state automata + registers
  - Transitions depend on
    - label and data value at current position
    - current state and register contents

Example

- $(<_1, <_2)$-structure:
  - As a data word:

```
  y y b r y r r
  5 6 3 7 1 4 2
```
**FO² with two linear orders: General satisfiability**

**Theorem 7** [Torunczyk, Z. ’20]

Satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

**Proof**

- **Key idea 1:** Use register automata

- **Register automata:**
  - Reads data words over $\Sigma \times \mathcal{D}$
  - Finite state automata + registers
  - Transitions depend on
    - label and data value at current position
    - current state and register contents

- For checking $\forall\exists$-/$\forall\forall$-constraints:
  Guess/verify maxima and minima of past and future values

**Example**

- $(<_1, <_2)$-structure:

- As a data word:

<table>
<thead>
<tr>
<th>y</th>
<th>y</th>
<th>b</th>
<th>r</th>
<th>y</th>
<th>r</th>
<th>r</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>
Theorem 7 [Torunczyk, Z. ’20]
Satisfiability of $\mathsf{FO}^2(\prec_1, \prec_2)$ is decidable

Proof

- **Key idea 1**: Use register automata

- **Register automata:**
  - Reads data words over $\Sigma \times D$
  - Finite state automata + registers
  - Transitions depend on
    - label and data value at current position
    - current state and register contents

- For checking $\exists/\forall\forall$-constraints:
  - Guess/verify maxima and minima of past and future values

Example

- $(\prec_1, \prec_2)$-structure:
  - As a data word:
    - $y \ y \ b \ r \ y \ r \ r$
    - $5 \ 6 \ 3 \ 7 \ 1 \ 4 \ 2$

- Checking a constraint $\forall\exists$:
  - Guess and verify the maximal value for $\bullet$-positions in the future
  - Check consistency at $\bullet$-positions

Thomas Zeume

Two-variable first-order logic on ordered structures
Theorem 7 [Torunczyk, Z. ’20]

Satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

Proof

• Key idea 1: Use register automata

• Register automata:
  • Reads data words over $\Sigma \times D$
  • Finite state automata + registers
  • Transitions depend on
    • label and data value at current position
    • current state and register contents

• For checking $\forall \exists$/\(\forall \forall\)-constraints:
  Guess/verify maxima and minima of past and future values

Example

• $(<_1, <_2)$-structure:

• As a data word:
  \[
  \begin{array}{cccccccc}
  y & y & b & r & y & r & r & r \\
  5 & 6 & 3 & 7 & 1 & 4 & 2 \\
  \end{array}
  \]

• Checking a constraint $\forall \exists \forall$:
  • Guess and verify the maximal value for $\bullet$-positions in the future
  • Check consistency at $\circ$-positions

Input:

\[
\begin{array}{cccccccc}
  y & y & b & r & y & r & r & r \\
  5 & 6 & 3 & 7 & 1 & 4 & 2 \\
\end{array}
\]
**Theorem 7** [Torunczyk, Z. ’20]

Satisfiability of $\mathbf{FO}^2(<_1, <_2)$ is decidable

**Proof**

- **Key idea 1:** Use register automata

- **Register automata:**
  - Reads data words over $\Sigma \times \Delta$
  - Finite state automata + registers
  - Transitions depend on
    - label and data value at current position
    - current state and register contents

- For checking $\forall \exists-/\forall \forall$-constraints:
  Guess/verify maxima and minima of past and future values

**Example**

- $(<_1, <_2)$-structure:
- As a data word:
  $y \ y \ b \ r \ y \ r \ r \ 5 \ 6 \ 3 \ 7 \ 1 \ 4 \ 2$

- Checking a constraint $\forall \exists \bigtriangleup$:
  - Guess and verify the maximal value for $\bullet$-positions in the future
  - Check consistency at $\bigtriangleup$-positions

**Input:**

<table>
<thead>
<tr>
<th></th>
<th>$y$</th>
<th>$y$</th>
<th>$y$</th>
<th>$b$</th>
<th>$r$</th>
<th>$r$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$R_{max,r,\rightarrow}$</td>
<td>$5$</td>
<td>$6$</td>
<td>$3$</td>
<td>$7$</td>
<td>$1$</td>
<td>$4$</td>
<td>$2$</td>
</tr>
</tbody>
</table>
Theorem 7 [Torunczyk, Z. '20]
Satisfiability of $\mathsf{FO}^2(<_1, <_2)$ is decidable

Proof

• **Key idea 1:** Use register automata

• **Register automata:**
  • Reads **data words** over $\Sigma \times D$
  • Finite state automata + registers
  • Transitions depend on
    • label and data value at current position
    • current state and register contents

• For checking $\forall\exists$-/ $\forall\forall$-constraints:
  Guess/verify maxima and minima of past and future values

Example

• $(<_1, <_2)$-structure:

• As a data word:
  $y \ y \ b \ r \ y \ r \ r$
  $5 \ 6 \ 3 \ 7 \ 1 \ 4 \ 2$

• Checking a constraint $\forall\exists$:
  • Guess and verify the maximal value for $\bigcirc$-positions in the future
  • Check consistency at $\square$-positions

Input:

\[
\begin{array}{cccccccc}
\text{y} & \text{y} & \text{b} & \text{r} & \text{y} & \text{r} & \text{r} \\
5 & 6 & 3 & 7 & 1 & 4 & 2 \\
\end{array}
\]

\[
R_{\max,r} \rightarrow 7
\]
Theorem 7 [Torunczyk, Z. ’20]

Satisfiability of $\textbf{FO}^2(\prec_1, \prec_2)$ is decidable

Proof

- **Key idea 1:** Use register automata

**Register automata:**
- Reads **data words** over $\Sigma \times D$
- Finite state automata + registers
- Transitions depend on
  - label and data value at current position
  - current state and register contents

- For checking $\forall \exists$-/$\forall \forall$-constraints: Guess/verify maxima and minima of past and future values

Example

- $(\prec_1, \prec_2)$-structure:
  - As a data word:
    - $y \ y \ b \ r \ y \ r \ r$
    - $5 \ 6 \ 3 \ 7 \ 1 \ 4 \ 2$

- Checking a constraint $\forall \exists$:
  - Guess and verify the maximal value for $\bullet$-positions in the future
  - Check consistency at $\circ$-positions

Input:

<table>
<thead>
<tr>
<th></th>
<th>$y$</th>
<th>$y$</th>
<th>$b$</th>
<th>$r$</th>
<th>$y$</th>
<th>$r$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>6</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>2</td>
<td></td>
</tr>
</tbody>
</table>

$R_{\text{max}, r} \rightarrow 7$ $7$

Thomas Zeume Two-variable first-order logic on ordered structures
**FO² with two linear orders: General satisfiability**

**Theorem 7** [Torunczyk, Z. ’20]

Satisfiability of $\mathbf{FO}^2(<_1, <_2)$ is decidable

**Proof**

- **Key idea 1:** Use register automata

- **Register automata:**
  - Reads **data words** over $\Sigma \times D$
  - Finite state automata + registers
  - Transitions depend on
    - label and data value at current position
    - current state and register contents

- For checking $\forall \exists-/\forall \forall$-constraints:
  - Guess/verify maxima and minima of past and future values

**Example**

- $(<_1, <_2)$-structure:
  - As a data word:
    - $y \ y \ b \ r \ y \ r \ r$
    - $5 \ 6 \ 3 \ 7 \ 1 \ 4 \ 2$
  - Checking a constraint $\forall \exists$:
    - Guess and verify the maximal value for $\bigcirc$-positions in the future
    - Check consistency at $\bullet$-positions

- Input:
  - $y \ y \ b \ r \ y \ r \ r$
  - $5 \ 6 \ 3 \ 7 \ 1 \ 4 \ 2$

- $R_{\text{max},r} \rightarrow$
  - $7 \ 7 \ 7$
**FO² with two linear orders: General satisfiability**

**Theorem 7** [Torunczyk, Z. ’20]

Satisfiability of $\text{FO}^2(<_1,<_2)$ is decidable

**Proof**

- **Key idea 1:** Use register automata

- **Register automata:**
  - Reads **data words** over $\Sigma \times D$
  - Finite state automata + registers
  - Transitions depend on
    - label and data value at current position
    - current state and register contents

- For checking $\forall \exists$-/ $\forall \forall$-constraints:
  - Guess/verify maxima and minima of past and future values

**Example**

- **$\langle <_1,<_2 \rangle$-structure:**
  - As a data word:
    - $y$ $y$ $b$ $r$ $y$ $r$ $r$
    - $5$ $6$ $3$ $7$ $1$ $4$ $2$

- Checking a constraint $\forall \exists$:
  - Guess and verify the maximal value for $\bullet$-positions in the future
  - Check consistency at $\circ$-positions

**Input:**

<table>
<thead>
<tr>
<th>$y$ $y$ $b$ $r$ $y$ $r$ $r$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$5$ $6$ $3$ $7$ $1$ $4$ $2$</td>
</tr>
</tbody>
</table>

$R_{\max,r, \rightarrow}$

| $7$ $7$ $7$ $4$ |
**Theorem 7** [Torunczyk, Z. '20]  
Satisfiability of $\text{FO}^2(\prec_1, \prec_2)$ is decidable

**Proof**

- **Key idea 1:** Use register automata
- **Register automata:**
  - Reads *data words* over $\Sigma \times \mathcal{D}$
  - Finite state automata + registers
  - Transitions depend on
    - label and data value at current position
    - current state and register contents
- For checking $\forall \exists$-/\$\forall \forall$-constraints: Guess/verify maxima and minima of past and future values

**Example**

- $(\prec_1, \prec_2)$-structure:
  - As a data word:
    - $y \ y \ b \ r \ y \ r \ r$
    - $5 \ 6 \ 3 \ 7 \ 1 \ 4 \ 2$
  - Checking a constraint $\forall \exists$:
    - Guess and verify the maximal value for $\bullet$-positions in the future
    - Check consistency at $\circ$-positions

Input:

<table>
<thead>
<tr>
<th>$\mathcal{R}_{\max,r}$</th>
<th>7</th>
<th>7</th>
<th>7</th>
<th>4</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$y \ y \ b \ r \ y \ r \ r$</td>
<td>5</td>
<td>6</td>
<td>3</td>
<td>7</td>
<td>1</td>
</tr>
</tbody>
</table>
Theorem 7 [Torunczyk, Z. ’20]

Satisfiability of $\text{FO}^2(\prec_1, \prec_2)$ is decidable

Proof

• Key idea 1: Use register automata

• Register automata:
  • Reads data words over $\Sigma \times \mathcal{D}$
  • Finite state automata + registers
  • Transitions depend on
    • label and data value at current position
    • current state and register contents

• For checking $\forall \exists$-/ $\forall \forall$-constraints:
  Guess/verify maxima and minima of past and future values

Example

• $(\prec_1, \prec_2)$-structure:

  <\ 2

  \begin{center}
  \begin{tikzpicture}
    \draw[->] (0,0) -- (2,0);
    \draw[->] (0,0) -- (0,2);
  \end{tikzpicture}
  \end{center}

• As a data word:

  \begin{center}
  y \hspace{0.5em} y \hspace{0.5em} b \hspace{0.5em} r \hspace{0.5em} y \hspace{0.5em} r \hspace{0.5em} r \hspace{0.5em} 5 \hspace{0.5em} 6 \hspace{0.5em} 3 \hspace{0.5em} 7 \hspace{0.5em} 1 \hspace{0.5em} 4 \hspace{0.5em} 2
  \end{center}

• Checking a constraint $\forall \exists$:

  • Guess and verify the maximal value for $\bullet$-positions in the future
  • Check consistency at $\circ$-positions

Input:

\begin{center}
\begin{array}{cccccccc}
  \ & \ & \ & \ & \ & \ & \ & \ \\
  y & y & b & r & y & r & r & \\
  5 & 6 & 3 & 7 & 1 & 4 & 2 & \\
\end{array}
\end{center}

$R_{\text{max},r,\rightarrow}$

\begin{center}
\begin{array}{cccccccc}
  \ & \ & \ & \ & \ & \ & \ & \ \\
  7 & 7 & 7 & 4 & 4 & 2 & \ \\
\end{array}
\end{center}
Theorem 7 [Torunczyk, Z. '20]

Satisfiability of $\mathbf{FO}^2(\prec_1, \prec_2)$ is decidable

Proof

• **Key idea 1:** Use register automata

• **Register automata:**
  • Reads **data words** over $\Sigma \times \mathcal{D}$
  • Finite state automata + registers
  • Transitions depend on
    • label and data value at current position
    • current state and register contents

• For checking $\forall\exists$/-$\forall\forall$-constraints:
  Guess/verify maxima and minima of past and future values

Example

• $(\prec_1, \prec_2)$-structure:

  \[
  \begin{array}{c}
  2 \\
  1
  \end{array}
  \]

• As a data word:

  \[
  y \quad y \quad b \quad r \quad y \quad r \quad r \\
  5 \quad 6 \quad 3 \quad 7 \quad 1 \quad 4 \quad 2
  \]

• Checking a constraint $\forall \exists$:

  • Guess and verify the maximal value for $\bullet$-positions in the future
  • Check consistency at $\circ$-positions

Input:

<table>
<thead>
<tr>
<th></th>
<th>$y$</th>
<th>$y$</th>
<th>$b$</th>
<th>$r$</th>
<th>$y$</th>
<th>$r$</th>
<th>$r$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>5</td>
<td>6</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

$R_{\max,r} \rightarrow$

|   | 7   | 7   | 7   | 4   | 4   | 2   | 2   |
**FO² with two linear orders: General satisfiability**

<table>
<thead>
<tr>
<th>Theorem 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>[Torunczyk, Z. '20]</td>
</tr>
<tr>
<td>Satisfiability of $\text{FO}^2(&lt;_1, &lt;_2)$ is decidable</td>
</tr>
</tbody>
</table>

**Proof (continued)**

- **Key idea 1:** Use register automata
\( \text{FO}^2 \) with two linear orders: General satisfiability

<table>
<thead>
<tr>
<th>Theorem 7</th>
<th>[\text{Torunczyk, Z. '20}]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Satisfiability of ( \text{FO}^2(&lt;_1, &lt;_2) ) is decidable</td>
<td></td>
</tr>
</tbody>
</table>

**Proof (continued)**

- **Key idea 1:** Use register automata
- **Problem:** In infinite words, a maximum/minimum may not exist

- **Key idea 2:** Extend register automata by infima- and suprema-conditions
  - **Problem:** Satisfying structures may not be encodable as words
    - E.g. because elements are "dense" wrt. \( <_1 \)

- **Key idea 3:** Use register automata on data trees
Theorem 7  [Torunczyk, Z. '20]

Satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

Proof (continued)

- **Key idea 1:** Use register automata

- **Problem:** In infinite words, a maximum/minimum may not exist

- **Key idea 2:** Extend register automata by infima- and suprema-conditions
**Theorem 7** [Torunczyk, Z. ’20]

Satisfiability of $\text{FO}^2(<_1,<_2)$ is decidable

**Proof (continued)**

- **Key idea 1:** Use register automata

- **Problem:** In infinite words, a maximum/minimum may not exist

- **Key idea 2:** Extend register automata by infima- and suprema-conditions

- **Problem:** Satisfying structures may not be encodable as words
  - E.g. because elements are “dense” wrt. $<_1$
**FO^2 with two linear orders: General satisfiability**

**Theorem 7**  
[Torunczyk, Z. '20]  
Satisfiability of $\text{FO}^2(\prec_1, \prec_2)$ is decidable

**Proof (continued)**

- **Key idea 1:** Use register automata

- **Problem:** In infinite words, a maximum/minimum may not exist

- **Key idea 2:** Extend register automata by infima- and suprema-conditions

- **Problem:** Satisfying structures may not be encodable as words  
  ➞ E.g. because elements are “dense” wrt. $\prec_1$

- **Key idea 3:** Use register automata on data trees
Part III:

Perspectives and summary
Perspective: From linear orders to preorders

• We have seen: (Finite) satisfiability of $\mathbf{FO}^2(<_1,<_2)$ is decidable
Perspective: From linear orders to preorders

- **We have seen:** (Finite) satisfiability of FO²(<₁, <₂) is decidable
- **Question:** What about preorders <₁ and <₂?
- **Preorder:** Equivalence relation whose equivalence classes are ordered
Perspective: From linear orders to preorders

- **We have seen:** (Finite) satisfiability of $\mathbf{FO}^2(\prec_1, \prec_2)$ is decidable

- **Question:** What about preorders $\prec_1$ and $\prec_2$?

- **Preorder:** Equivalence relation whose equivalence classes are ordered

- **Observation:** (Finite) satisfiability of $\mathbf{FO}^2(\prec, \prec)$ is decidable
  - Same proof techniques...
Perspective: From linear orders to preorders

- **We have seen:** (Finite) satisfiability of $\text{FO}^2(\prec_1, \prec_2)$ is decidable

- **Question:** What about preorders $\prec_1$ and $\prec_2$?

- **Observation:** (Finite) satisfiability of $\text{FO}^2(\prec, \bowtie)$ is decidable
  → Same proof techniques...

---

**Theorem 8** [Schwentick, Z. ’10]

- Finite satisfiability of $\text{FO}^2(\prec_1, \bowtie_2)$ is **undecidable**
Perspective: From linear orders to preorders

- We have seen: (Finite) satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable
- Question: What about preorders $<_1$ and $<_2$?
- Preorder: Equivalence relation whose equivalence classes are ordered

- Observation: (Finite) satisfiability of $\text{FO}^2(<, \prec)$ is decidable
  ➔ Same proof techniques...

Theorem 8 [Schwentick, Z. ’10]
- Finite satisfiability of $\text{FO}^2(<_1, \prec_2)$ is undecidable

Proof sketch
- Idea: Reduction from PCP
**Perspective: From linear orders to preorders**

- **We have seen:** (Finite) satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable
- **Question:** What about preorders $<_1$ and $<_2$?
- **Observation:** (Finite) satisfiability of $\text{FO}^2(<, \prec)$ is decidable
  - Same proof techniques…

**Theorem 8** [Schwentick, Z. ’10]

- Finite satisfiability of $\text{FO}^2(<_1, \prec_2)$ is **undecidable**

**Proof sketch**

- **Idea:** Reduction from PCP
- **Example instance:**
  - $u = ab|cdef|g$
  - $v = a|bcd|efg$
Perspective: From linear orders to preorders

- **We have seen:** (Finite) satisfiability of $\text{FO}^2(<_1, <_2)$ is decidable

- **Question:** What about preorders $\prec_1$ and $\prec_2$?

- **Preorder:** Equivalence relation whose equivalence classes are ordered

- **Observation:** (Finite) satisfiability of $\text{FO}^2(<, \prec)$ is decidable
  - Same proof techniques...

---

**Theorem 8** [Schwentick, Z. ’10]

- Finite satisfiability of $\text{FO}^2(\prec_1, \prec_2)$ is **undecidable**

---

**Proof sketch**

- **Idea:** Reduction from PCP

- **Example instance:**
  - $u = ab|cdef|g$
  - $v = a|bcd|efg$

---

Example instance:

```
  u = ab|cdef|g
  v = a|bcd|efg
```

---

Example instance:

```
  a a b
  b c c d d e f
  e f g g
```
Perspective: From linear orders to preorders

- We have seen: (Finite) satisfiability of $\text{FO}^2(\prec_1, \prec_2)$ is decidable
- Question: What about preorders $\prec_1$ and $\prec_2$?
- Preorder: Equivalence relation whose equivalence classes are ordered
- Observation: (Finite) satisfiability of $\text{FO}^2(\prec, \prec)$ is decidable
  - Same proof techniques...

Theorem 8 [Schwentick, Z. ’10]
- Finite satisfiability of $\text{FO}^2(\prec_1, \prec_2)$ is undecidable

Proof sketch
- Idea: Reduction from PCP
- Example instance:
  - $u = ab|cdef|g$
  - $v = a|bcd|efg$
  - Positions from substring pairs with same index: equivalent wrt. $\prec_2$ (rows)
  - Corresponding positions: equivalent wrt. $\prec_1$ (columns)
Perspective: Many linear orders, many successors

- Two linear orders, two successors:

  \[
  \begin{array}{ccc}
  S_2 & S_1 & <_1 \\
  \text{decidable} & [\text{Charatonik, Witkowski '13}]
  & [\text{Manuel '10}] \\
  \text{decidable} & [\text{Z., Harwath '16}]
  & [\text{Manuel, Z. '16}] \\
  \text{decidable} & [\text{Z., Harwath '16}]
  & [\text{Z., Harwath '16}]
  \\
  \text{undecidable} & [\text{Manuel '10}]& [\text{Schwentick, Z. '10}] \\
  \end{array}
  \]

Theorem 9: \( \text{FO}_2 \) with linear orders

- (Finite) Satisfiability of \( \text{FO}_2 \):
  - with two linear orders: decidable
    - [Schwentick, Z. '10; Torunczyk, Z. '20]
  - with three linear orders: undecidable
    - [Kieronski '11]

Open question

- Is finite satisfiability of \( \text{FO}_2 \) with \( k \) successors decidable for all \( k \)?
- Equivalent: Is finite satisfiability of \( \text{FO}_2 \) with \( k \) permutations with one cycle each decidable for all \( k \)?
- Decidable: Finite satisfiability of \( \text{FO}_2 \) with \( k \) permutations (with arbitrary many cycles)
### Perspective: Many linear orders, many successors

- **Two linear orders, two successors:**

<table>
<thead>
<tr>
<th>$S_2$</th>
<th>$S_2, &lt;_2$</th>
<th>$&lt;_2$</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>decidable</strong></td>
<td><strong>undecidable</strong></td>
<td><strong>decidable</strong></td>
</tr>
<tr>
<td>[Charatonik, Witkowski '13] [Manuel '10]</td>
<td>[Manuel '10]</td>
<td>[Z., Harwath '16] [Manuel, Z. '16] [Z., Harwath '16] [Schwentick, Z. '10]</td>
</tr>
</tbody>
</table>

- **Question:** What happens for
  - many linear orders?
  - many successors?
Perspective: Many linear orders, many successors

- Two linear orders, two successors:
  - $S_2$: decidable
    - [Charatonik, Witkowski '13]
    - [Manuel '10]
  - $S_2, <_2$: decidable
    - [Z., Harwath '16]
    - [Manuel, Z. '16]

- Three linear orders, one successor:
  - $<_2$: decidable
    - [Z., Harwath '16]
    - [Manuel, Z. '13]
    - [Z., Harwath '16]
    - [Schwentick, Z. '10]

- Three linear orders, one successor:
  - $S_1$: decidable
    - [Z., Harwath '16]
    - [Manuel, Z. '13]
  - $S_1, <_1$: decidable
    - [Z., Harwath '16]
    - [Schwentick, Z. '10]
  - $<_1$: decidable
    - [Z., Harwath '16]
    - [Schwentick, Z. '10]

Theorem 9: $\text{FO}^2$ with linear orders

(Finite) Satisfiability of $\text{FO}^2$

- with two linear orders: decidable
  - [Schwentick, Z. '10; Torunczyk, Z. '20]
- with three linear orders: undecidable
  - [Kieronski '11]

Question: What happens for
- many linear orders?
- many successors?
Perspective: Many linear orders, many successors

- Two linear orders, two successors:
  - $S_2$ decidable
    - [Charatonik, Witkowski '13]
    - [Manuel '10]
  - $S_2, \prec_2$ decidable
    - [Z., Harwath '16]
    - [Manuel, Z. '16]
    - [Manuel '10]
  - $\prec_2$ decidable
    - [Z., Harwath '16]
    - [Manuel, Z. '13]
    - [Schwentick, Z. '10]

- Question: What happens for
  - many linear orders?
  - many successors?

Theorem 9: $\text{FO}^2$ with linear orders

(Finite) Satisfiability of $\text{FO}^2$

- with two linear orders: decidable
  - [Schwentick, Z. '10; Torunczyk, Z. '20]
- with three linear orders: undecidable
  - [Kieronski '11]

Open question

- Is finite satisfiability of $\text{FO}^2$ with $k$ successors decidable for all $k$?
Perspective: Many linear orders, many successors

- Two linear orders, two successors:
  
  \[
  S_2 \quad \text{decidable} \quad \text{Charatonik, Witkowski '13} \quad \text{Manuel '10}
  \]

  \[
  S_2, <_2 \quad \text{decidable} \quad \text{undecidable} \quad \text{Z., Harwath '16} \quad \text{Manuel '10} \quad \text{Manuel, Z. '16}
  \]

  \[
  <_2 \quad \text{decidable} \quad \text{decidable} \quad \text{decidable} \quad \text{Z., Harwath '16} \quad \text{Manuel, Z. '13} \quad \text{Z., Harwath '16} \quad \text{Schwentick, Z. '10} \quad \text{Schwentick, Z. '10}
  \]

  \[
  S_1 \quad S_1, <_1 \quad <_1
  \]

  \[
  S_1 \quad S_1, <_1 \quad <_1
  \]

- Question: What happens for
  - many linear orders?
  - many successors?

Theorem 9: \( \text{FO}^2 \) with linear orders

(Finite) Satisfiability of \( \text{FO}^2 \)

- with two linear orders: \text{decidable} [Schwentick, Z. '10; Torunczyk, Z. '20]

- with three linear orders: \text{undecidable} [Kieronski '11]

Open question

- Is finite satisfiability of \( \text{FO}^2 \) with \( k \) successors decidable for all \( k \)?

- Equivalent: Is finite satisfiability of \( \text{FO}^2 \) with \( k \) permutations with one cycle each decidable for all \( k \)?
Perspective: Many linear orders, many successors

- Two linear orders, two successors:

<table>
<thead>
<tr>
<th>S₂</th>
<th>S₂, &lt;₂</th>
<th>&lt;₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>decidable</td>
<td>decidable</td>
<td>decidable</td>
</tr>
<tr>
<td>[Charatonik, Witkowski '13]</td>
<td>[Manuel '10]</td>
<td>[Z., Harwath '16]</td>
</tr>
</tbody>
</table>

- Open question

  - Is finite satisfiability of FO² with k successors decidable for all k?

  - Equivalent: Is finite satisfiability of FO² with k permutations with one cycle each decidable for all k?

  - Decidable: Finite satisfiability of FO² with k permutations (with arbitrary many cycles)

Theorem 9: FO² with linear orders

(Finite) Satisfiability of FO²

- with two linear orders: decidable
  [Schwentick, Z. '10; Torunczyk, Z. '20]
- with three linear orders: undecidable
  [Kieronski '11]

Question: What happens for
- many linear orders?
- many successors?
Summary

Finite Satisfiability:

- \( S_2 \)
  - Decidable
  - \([\text{Charatonik, Witkowski '13}, \text{Manuel '10}]\)

- \( S_2, \prec \)
  - Decidable
  - \([\text{Z., Harwath '16}, \text{Manuel, Z. '16}]\)
  - Undecidable
  - \([\text{Manuel '10}]\)

- \( \prec \)
  - Decidable
  - \([\text{Z., Harwath '16}, \text{Manuel, Z. '13}]\)
  - Decidable
  - \([\text{Z., Harwath '16}, \text{Schwentick, Z. '10}]\)

General Satisfiability:

- \( FO^2(\prec_1, \prec_2) \) is decidable
  - \([\text{Torunczyk, Z. '20}]\)

Approaches for decidability:
- Small model approach
- Automata-based approach

Approach for undecidability: Reductions

Open question
- Is finite satisfiability of \( FO^2 \) with \( k \) successors decidable for all \( k \)?

Thank You!