# On Nested Justification Systems

Simon Marynissen[1,2], Jesse Heyninck[2,3], **Bart Bogaerts**[2] and Marc Denecker[1]
[1]*KU Leuven*, [2]*Vrije Universiteit Brussel and* [3]*University of Cape Town and CAIR*

International Conference on Logic Programming
August 3rd, 2022

ARTIFICIAL
INTELLIGENCE
RESEARCH GROUP

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t}; \quad q \leftarrow \mathbf{t}; \quad s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{ll} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\} \end{array} \right\}$$

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{matrix} p \leftarrow \mathbf{t}; \quad q \leftarrow \mathbf{t}; \quad s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{matrix} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{matrix} \right\} \end{matrix} \right\}$$

Compression:

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{matrix} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, p, q; & s \leftarrow p, s, p; \\ s \leftarrow p, p, s; & s \leftarrow p, \mathbf{u}; \end{matrix} \right\}$$

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t}; \quad q \leftarrow \mathbf{t}; \quad s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{ll} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\} \end{array} \right\}$$

Compression:

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow \quad p, q; & s \leftarrow \quad s, p; \\ s \leftarrow \quad p, s; & s \leftarrow p, \mathbf{u}; \end{array} \right\}$$

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{\begin{array}{l} p \leftarrow \mathbf{t}; \quad q \leftarrow \mathbf{t}; \quad s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{\begin{array}{ll} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array}\right\} \end{array}\right\}$$

Compression:

$$\mathcal{B}_{\mathrm{st}} : \left\{\begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow \quad p, q; & s \leftarrow \quad s, p; \\ s \leftarrow \quad p, s; & s \leftarrow p, \mathbf{u}; \end{array}\right\}$$

Disadvantages:

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\text{st}} : \left\{ \begin{matrix} p \leftarrow \mathbf{t}; \quad q \leftarrow \mathbf{t}; \quad s \leftarrow p, r; \\ \mathcal{B}_{\text{KK}} : \left\{ \begin{matrix} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{matrix} \right\} \end{matrix} \right\}$$

Compression:

$$\mathcal{B}_{\text{st}} : \left\{ \begin{matrix} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow \quad p, q; & s \leftarrow \quad s, p; \\ s \leftarrow \quad p, s; & s \leftarrow p, \mathbf{u}; \end{matrix} \right\}$$

Disadvantages:

- “Explanations” not in terms of input rules

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{matrix} p \leftarrow \mathbf{t}; \quad q \leftarrow \mathbf{t}; \quad s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{matrix} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{matrix} \right\} \end{matrix} \right\}$$

Compression:

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{matrix} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, q; & s \leftarrow s, p; \\ s \leftarrow p, s; & s \leftarrow p, \mathbf{u}; \end{matrix} \right\}$$

Disadvantages:

- "Explanations" not in terms of input rules
- Only defined for parametric inner systems

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\text{st}} : \left\{ \begin{array}{lll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; & s \leftarrow p, r; \\ \mathcal{B}_{\text{KK}} : & \left\{ \begin{array}{ll} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\} & \end{array} \right\}$$

Compression:

$$\mathcal{B}_{\text{st}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, q; & s \leftarrow s, p; \\ s \leftarrow p, s; & s \leftarrow p, \mathbf{u}; \end{array} \right\}$$

Disadvantages:

- "Explanations" not in terms of input rules
- Only defined for parametric inner systems

Merge:

$$\mathcal{B}_{\text{st.KK}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, r; & \\ r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\}$$

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{lll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; & s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{ll} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\} \end{array} \right\}$$

Compression:

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow \quad p, q; & s \leftarrow \quad s, p; \\ s \leftarrow \quad p, s; & s \leftarrow p, \mathbf{u}; \end{array} \right\}$$

Disadvantages:

- “Explanations” not in terms of input rules
- Only defined for parametric inner systems

Merge:

$$\mathcal{B}_{\mathrm{st.KK}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, r; & \\ r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\}$$

Properties:

# SUMMARY

## Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t}; \quad q \leftarrow \mathbf{t}; \quad s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{ll} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\} \end{array} \right\}$$

## Compression:

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow \quad p, q; & s \leftarrow \quad s, p; \\ s \leftarrow \quad p, s; & s \leftarrow p, \mathbf{u}; \end{array} \right\}$$

## Disadvantages:

- "Explanations" not in terms of input rules
- Only defined for parametric inner systems

## Merge:

$$\mathcal{B}_{\mathrm{st.KK}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, r; & \\ r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\}$$

## Properties:

- Defined for (almost) all branch evaluations

# SUMMARY

## Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t}; \quad q \leftarrow \mathbf{t}; \quad s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{ll} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\} \end{array} \right\}$$

## Compression:

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, q; & s \leftarrow s, p; \\ s \leftarrow p, s; & s \leftarrow p, \mathbf{u}; \end{array} \right\}$$

## Disadvantages:

- "Explanations" not in terms of input rules
- Only defined for parametric inner systems

## Merge:

$$\mathcal{B}_{\mathrm{st.KK}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, r; & \\ r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\}$$

## Properties:

- Defined for (almost) all branch evaluations
- Justifications use original rules

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{lll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; & s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{ll} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\} \end{array} \right\}$$

Compression:

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{llll} p \leftarrow \mathbf{t}; & & q \leftarrow \mathbf{t}; & \\ s \leftarrow & p, q; & s \leftarrow & s, p; \\ s \leftarrow & p, s; & s \leftarrow p, \mathbf{u}; & \end{array} \right\}$$

Disadvantages:

- "Explanations" not in terms of input rules
- Only defined for parametric inner systems

Merge:

$$\mathcal{B}_{\mathrm{st.KK}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, r; & \\ r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\}$$

Properties:

- Defined for (almost) all branch evaluations
- Justifications use original rules
- Uses new branch evaluation

# SUMMARY

Nested Justifications [DBS15]

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t}; \quad q \leftarrow \mathbf{t}; \quad s \leftarrow p, r; \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{ll} r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\} \end{array} \right\}$$

Compression:

$$\mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, q; & s \leftarrow s, p; \\ s \leftarrow p, s; & s \leftarrow p, \mathbf{u}; \end{array} \right\}$$

Disadvantages:

- "Explanations" not in terms of input rules
- Only defined for parametric inner systems

Merge:

$$\mathcal{B}_{\mathrm{st.KK}} : \left\{ \begin{array}{ll} p \leftarrow \mathbf{t}; & q \leftarrow \mathbf{t}; \\ s \leftarrow p, r; & \\ r \leftarrow p, q; & r \leftarrow s, p; \\ r \leftarrow p, s & r \leftarrow r \end{array} \right\}$$

Properties:

- Defined for (almost) all branch evaluations
- Justifications use original rules
- Uses new branch evaluation

**Theorem**

$\mathrm{SV}^t_{\mathrm{Compress}(\mathbb{JS})}(x, I) = \mathrm{SV}^t_{\mathrm{Merge}(\mathbb{JS})}(x, I)$,

*under minor restrictions*

# OUTLINE

# OUTLINE

# JUSTIFICATION THEORY: A UNIFYING FRAMEWORK

Unification of formalisms

- Logic Programming
- Abstract Argumentation
- Nested least and greatest fixpoint definitions

# JUSTIFICATION THEORY: A UNIFYING FRAMEWORK

Unification of formalisms

- Logic Programming
- Abstract Argumentation
- Nested least and greatest fixpoint definitions

Simple and unified way of defining and studying semantics

- Stable
- Well-founded
- Supported
- Kripke-Kleene

The only thing to define is an evaluation of branches

# JUSTIFICATION THEORY: A UNIFYING FRAMEWORK

Unification of formalisms

- Logic Programming
- Abstract Argumentation
- Nested least and greatest fixpoint definitions

Simple and unified way of defining and studying semantics

- Stable
- Well-founded
- Supported
- Kripke-Kleene

The only thing to define is an evaluation of branches

Core idea: semantics is defined in terms of explanations why facts hold.

# JUSTIFICATION THEORY: A UNIFYING FRAMEWORK

Unification of formalisms

- Logic Programming
- Abstract Argumentation
- Nested least and greatest fixpoint definitions

Simple and unified way of defining and studying semantics

- Stable
- Well-founded
- Supported
- Kripke-Kleene

The only thing to define is an evaluation of branches

Core idea: semantics is defined in terms of explanations why facts hold.
Not just relevant for theory; justifications show up in unexpected places

# DEFINITIONS: JUSTIFICATION FRAMES

## Definition

A justification frame $\mathbb{JF}$ is a tuple $\langle \mathbb{F}, \mathbb{F}_d, R \rangle$ with:

fact space $\mathbb{F}$ with $\mathcal{L} = \{\mathbf{t}, \mathbf{f}, \mathbf{u}\} \subseteq \mathbb{F}$.

Involution $\sim$ on $\mathbb{F}$
(with $\sim\mathbf{t} = \mathbf{f}$, $\sim\mathbf{f} = \mathbf{t}$, $\sim\mathbf{u} = \mathbf{u}$)

defined facts $\mathbb{F}_d \subseteq \mathbb{F} \setminus \mathcal{L}$; $\sim\mathbb{F}_d = \mathbb{F}_d$.

rules $R \subseteq \mathbb{F}_d \times 2^{\mathbb{F}}$

## Example

$$\mathbb{F} = \{p, \sim p, q, \sim q, r, \sim r, s, \sim s, \mathbf{t}, \mathbf{f}, \mathbf{u}\}$$

$$\mathbb{F}_d = \{p, \sim p, q, \sim q, r, \sim r\}$$

$$p \leftarrow \sim q$$
$$q \leftarrow \sim p$$
$$p \leftarrow s, \sim r$$
$$r \leftarrow r$$

# DEFINITIONS: JUSTIFICATION FRAMES

## Definition

A justification frame $\mathbb{JF}$ is a tuple $\langle \mathbb{F}, \mathbb{F}_d, R\rangle$ with:

fact space $\mathbb{F}$ with $\mathcal{L} = \{\mathbf{t}, \mathbf{f}, \mathbf{u}\} \subseteq \mathbb{F}$.

Involution $\sim$ on $\mathbb{F}$
(with $\sim\mathbf{t} = \mathbf{f}$, $\sim\mathbf{f} = \mathbf{t}$, $\sim\mathbf{u} = \mathbf{u}$)

defined facts $\mathbb{F}_d \subseteq \mathbb{F} \setminus \mathcal{L}$; $\sim\mathbb{F}_d = \mathbb{F}_d$.

rules $R \subseteq \mathbb{F}_d \times 2^{\mathbb{F}}$

## Example

$$\mathbb{F} = \{p, \sim p, q, \sim q, r, \sim r, s, \sim s, \mathbf{t}, \mathbf{f}, \mathbf{u}\}$$

$$\mathbb{F}_d = \{p, \sim p, q, \sim q, r, \sim r\}$$

Complementation:

| | |
|---|---|
| $p \leftarrow \sim q$ | $\sim p \leftarrow q, r$ |
| $q \leftarrow \sim p$ | $\sim p \leftarrow q, \sim s$ |
| $p \leftarrow s, \sim r$ | $\sim q \leftarrow p$ |
| $r \leftarrow r$ | $\sim r \leftarrow \sim r$ |

# DEFINITIONS: JUSTIFICATIONS

$$
\begin{array}{ll}
p \leftarrow {\sim}q & {\sim}p \leftarrow q, r \\
q \leftarrow {\sim}p & {\sim}p \leftarrow q, {\sim}s \\
p \leftarrow s, {\sim}r & {\sim}q \leftarrow p \\
r \leftarrow r & {\sim}r \leftarrow {\sim}r
\end{array}
$$

## Definition

Let $\mathbb{JF} = \langle \mathbb{F}, \mathbb{F}_d, R \rangle$ be a justification frame. A (tree-like) justification $J$ in $\mathbb{JF}$ is a labeled tree such that the set of children of each node is a case (rule) in $R$ for that node.

$$
\begin{tikzcd}
 & p \arrow[d] & \\
 & {\sim}q \arrow[d] & \\
 & p \arrow[dl] \arrow[dr] & \\
{\sim}r & & s
\end{tikzcd}
$$

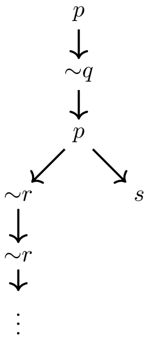
# DEFINITIONS: JUSTIFICATIONS

| | |
|---|---|
| $p \leftarrow {\sim}q$ | ${\sim}p \leftarrow q, r$ |
| $q \leftarrow {\sim}p$ | ${\sim}p \leftarrow q, {\sim}s$ |
| $p \leftarrow s, {\sim}r$ | ${\sim}q \leftarrow p$ |
| $r \leftarrow r$ | ${\sim}r \leftarrow {\sim}r$ |

## Definition

Let $\mathbb{JF} = \langle \mathbb{F}, \mathbb{F}_d, R \rangle$ be a justification frame. A (tree-like) justification $J$ in $\mathbb{JF}$ is a labeled tree such that the set of children of each node is a case (rule) in $R$ for that node.

$J$ is locally complete if all leaves are open



# DEFINITIONS: BRANCH EVALUATION

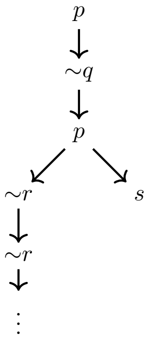
## Definition

A branch evaluation $\mathcal{B}$ maps every branch (sequence of facts) to an element of $\mathbb{F}$.

A justification system $\mathbb{JS}$ is a tuple $\langle \mathbb{F}, \mathbb{F}_d, R, \mathcal{B} \rangle$.

The stable branch evaluation $\mathcal{B}_{\mathrm{st}}$ maps

- finite branches to their last element
- infinite branches with positive tail to $\mathbf{f}$
- infinite branches with negative tail to $\mathbf{t}$
- infinite branches with mixed tail to the element of the first sign switch



# DEFINITIONS: BRANCH EVALUATION

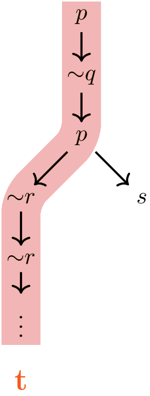
## Definition

A branch evaluation $\mathcal{B}$ maps every branch (sequence of facts) to an element of $\mathbb{F}$.

A justification system $\mathbb{JS}$ is a tuple $\langle \mathbb{F}, \mathbb{F}_d, R, \mathcal{B} \rangle$.

The stable branch evaluation $\mathcal{B}_{\text{st}}$ maps

- finite branches to their last element
- infinite branches with positive tail to $\mathbf{f}$
- infinite branches with negative tail to $\mathbf{t}$
- infinite branches with mixed tail to the element of the first sign switch



# DEFINITIONS: BRANCH EVALUATION

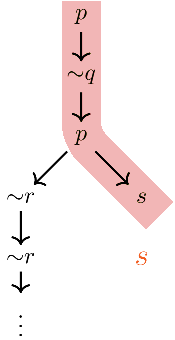
## Definition

A branch evaluation $\mathcal{B}$ maps every branch (sequence of facts) to an element of $\mathbb{F}$.

A justification system $\mathbb{JS}$ is a tuple $\langle \mathbb{F}, \mathbb{F}_d, R, \mathcal{B} \rangle$.

The stable branch evaluation $\mathcal{B}_{\mathrm{st}}$ maps

- finite branches to their last element
- infinite branches with positive tail to $\mathbf{f}$
- infinite branches with negative tail to $\mathbf{t}$
- infinite branches with mixed tail to the element of the first sign switch



# DEFINITIONS: BRANCH EVALUATION

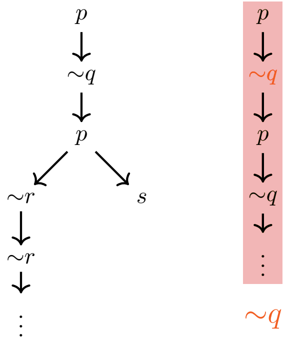
## Definition

A branch evaluation $\mathcal{B}$ maps every branch (sequence of facts) to an element of $\mathbb{F}$.

A justification system $\mathbb{JS}$ is a tuple $\langle \mathbb{F}, \mathbb{F}_d, R, \mathcal{B} \rangle$.

The stable branch evaluation $\mathcal{B}_{\mathrm{st}}$ maps

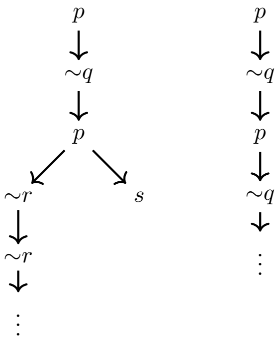
- finite branches to their last element
- infinite branches with positive tail to $\mathbf{f}$
- infinite branches with negative tail to $\mathbf{t}$
- infinite branches with mixed tail to the element of the first sign switch



# DEFINITIONS: BRANCH EVALUATION (2)

The well-founded branch evaluation $\mathcal{B}_{\mathrm{wf}}$ maps

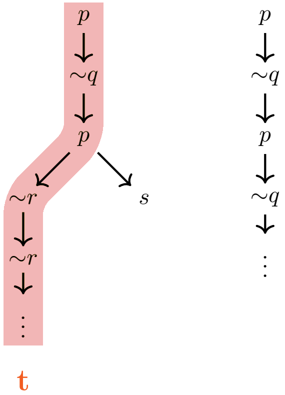
- finite branches to their last element
- infinite branches with positive tail to $\mathbf{f}$
- infinite branches with negative tail to $\mathbf{t}$
- infinite branches with mixed tail to $\mathbf{u}$



# DEFINITIONS: BRANCH EVALUATION (2)

The well-founded branch evaluation $\mathcal{B}_{\mathrm{wf}}$ maps

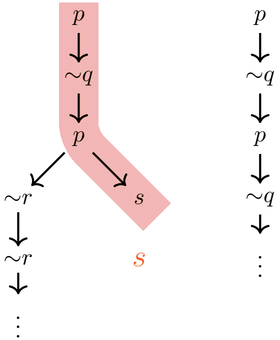
- finite branches to their last element
- infinite branches with positive tail to $\mathbf{f}$
- infinite branches with negative tail to $\mathbf{t}$
- infinite branches with mixed tail to $\mathbf{u}$



# DEFINITIONS: BRANCH EVALUATION (2)

The well-founded branch evaluation $\mathcal{B}_{\mathrm{wf}}$ maps

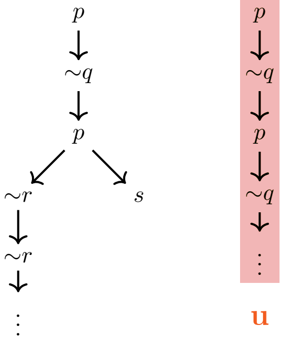
- finite branches to their last element
- infinite branches with positive tail to $\mathbf{f}$
- infinite branches with negative tail to $\mathbf{t}$
- infinite branches with mixed tail to $\mathbf{u}$



# DEFINITIONS: BRANCH EVALUATION (2)

The well-founded branch evaluation $\mathcal{B}_{\mathrm{wf}}$ maps

- finite branches to their last element
- infinite branches with positive tail to $\mathbf{f}$
- infinite branches with negative tail to $\mathbf{t}$
- infinite branches with mixed tail to $\mathbf{u}$



# DEFINITIONS: INTERPRETATIONS

## Definition

An interpretation $I$ maps each fact in $\mathbb{F}$ to a truth value (in $\mathcal{L}$) and

- Commutes with negation: $I({\sim}x) = {\sim}I(x)$
- Identity on $\mathcal{L}$

$$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$$

# DEFINITIONS: INTERPRETATIONS

## Definition

An interpretation $I$ maps each fact in $\mathbb{F}$ to a truth value (in $\mathcal{L}$) and

- Commutes with negation: $I(\sim x) = \sim I(x)$
- Identity on $\mathcal{L}$

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$

And implicitly: $I(\sim p) = \mathbf{f}$, $I(\sim q) = \mathbf{t}$, $I(r) = \mathbf{f}, I(s) = \mathbf{t}$,
$I(\mathbf{t}) = \mathbf{t}$, $I(\mathbf{f}) = \mathbf{f}$, $I(\mathbf{u}) = \mathbf{u}$

# DEFINITIONS: SUPPORTED VALUE

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

## Definition

The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

# DEFINITIONS: SUPPORTED VALUE

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

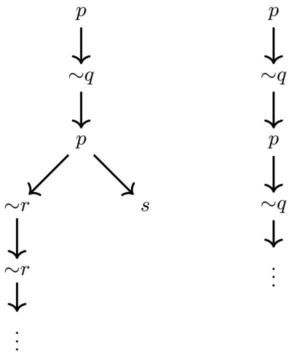
## Definition

The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: SUPPORTED VALUE

## Definition

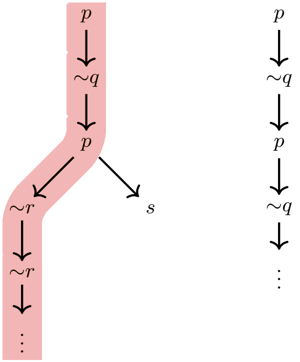
The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$




$I(\mathbf{t}) = \mathbf{t}$

# DEFINITIONS: SUPPORTED VALUE

## Definition

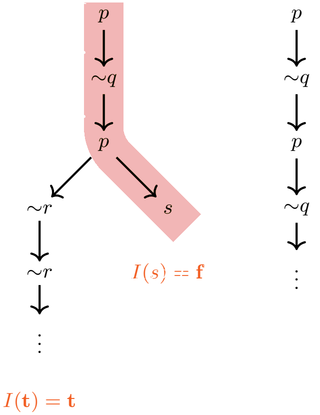
The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: SUPPORTED VALUE

## Definition

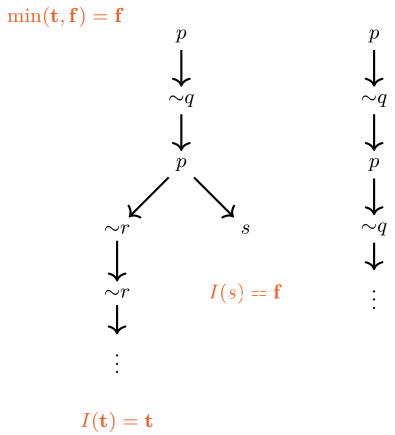
The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: SUPPORTED VALUE

## Definition

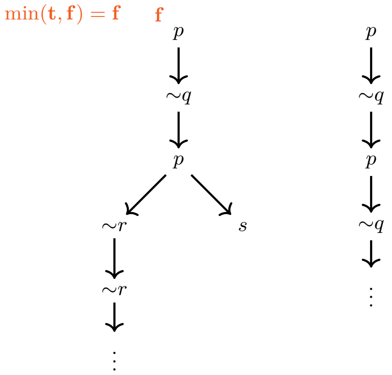
The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: SUPPORTED VALUE

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

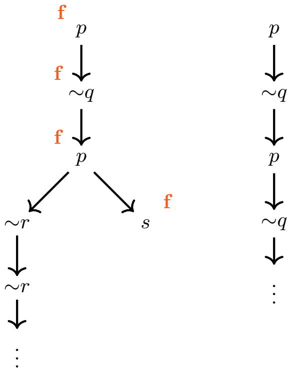
## Definition

The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: SUPPORTED VALUE

## Definition

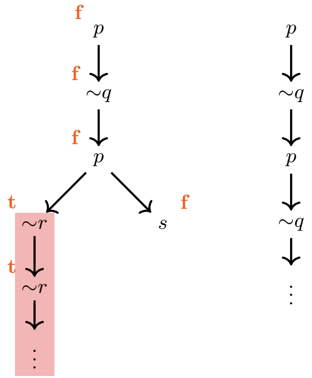
The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: SUPPORTED VALUE

## Definition

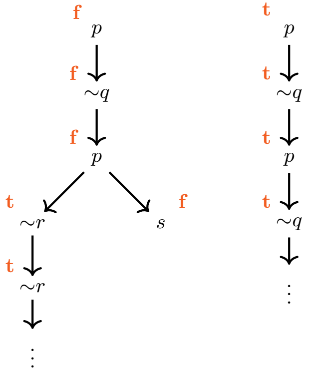
The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

### Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: SUPPORTED VALUE

## Definition

The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

## Definition

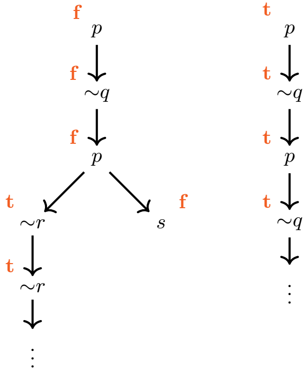
The supported value of $x$ under $I$ is the value of the best justification for $x$:

$$\mathrm{SV}_{\mathcal{B}}(x, I) = \max_{J \in \mathfrak{J}(x)} \mathrm{val}(J, x, I)$$

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: SUPPORTED VALUE

## Definition

The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

## Definition

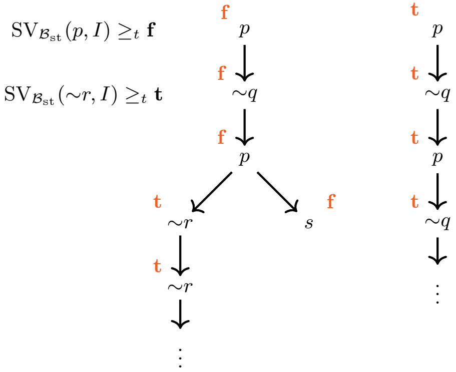
The supported value of $x$ under $I$ is the value of the best justification for $x$:

$$\mathrm{SV}_{\mathcal{B}}(x, I) = \max_{J \in \mathfrak{J}(x)} \mathrm{val}(J, x, I)$$

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: SUPPORTED VALUE

## Definition

The value of node $x$ in $J$ under $I$ is the value of the worst branch starting in $x$:

$$\mathrm{val}_{\mathcal{B}}(J, x, I) = \min_{\mathbf{b} \in B_J(x)} I(\mathcal{B}(\mathbf{b}))$$

## Definition

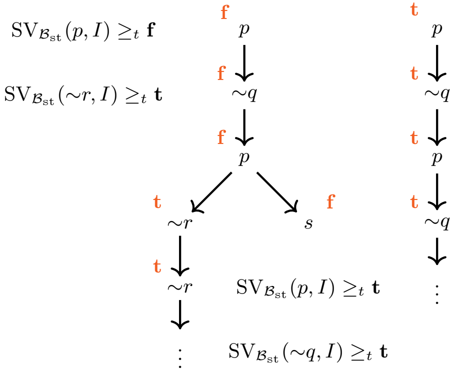
The supported value of $x$ under $I$ is the value of the best justification for $x$:

$$\mathrm{SV}_{\mathcal{B}}(x, I) = \max_{J \in \mathfrak{J}(x)} \mathrm{val}(J, x, I)$$

order: $\mathbf{f} \leq_t \mathbf{u} \leq_t \mathbf{t}$

## Example

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$ using $\mathcal{B}_{\mathrm{st}}$



# DEFINITIONS: MODEL

## Definition

An interpretation $I$ is a $\mathcal{B}$-model if

$$I(x) = \mathrm{SV}_{\mathcal{B}}(x, I)$$

for all defined facts $x$.

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{t}, I(s) = \mathbf{f}$. $I$ is not a $\mathcal{B}_{\mathrm{st}}$-model of

$$\begin{array}{ll} p \leftarrow {\sim}q & {\sim}p \leftarrow q, r \\ q \leftarrow {\sim}p & {\sim}p \leftarrow q, {\sim}s \\ p \leftarrow s, {\sim}r & {\sim}q \leftarrow p \\ r \leftarrow r & {\sim}r \leftarrow {\sim}r \end{array}$$

$r$ only has one justification.

$$\mathbf{t} = I(r) \neq \mathrm{SV}_{\mathcal{B}_{\mathrm{st}}}(r, I) = \mathbf{f}$$

# DEFINITIONS: MODEL

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{f}, I(s) = \mathbf{f}$.
$I$ is a $\mathcal{B}_{\mathrm{st}}$-model of

$$\begin{array}{ll} p \leftarrow {\sim}q & {\sim}p \leftarrow q, r \\ q \leftarrow {\sim}p & {\sim}p \leftarrow q, {\sim}s \\ p \leftarrow s, {\sim}r & {\sim}q \leftarrow p \\ r \leftarrow r & {\sim}r \leftarrow {\sim}r \end{array}$$

## Definition

An interpretation $I$ is a $\mathcal{B}$-model if

$$I(x) = \mathrm{SV}_{\mathcal{B}}(x, I)$$

for all defined facts $x$.

# DEFINITIONS: MODEL

## Definition

An interpretation $I$ is a $\mathcal{B}$-model if

$$I(x) = \mathrm{SV}_{\mathcal{B}}(x, I)$$

for all defined facts $x$.

$I(p) = \mathbf{t}, I(q) = \mathbf{f}, I(r) = \mathbf{f}, I(s) = \mathbf{f}$.
$I$ is a $\mathcal{B}_{\mathrm{st}}$-model of

$$\begin{array}{ll}
p \leftarrow {\sim}q & {\sim}p \leftarrow q, r \\
q \leftarrow {\sim}p & {\sim}p \leftarrow q, {\sim}s \\
p \leftarrow s, {\sim}r & {\sim}q \leftarrow p \\
r \leftarrow r & {\sim}r \leftarrow {\sim}r
\end{array}$$

$$\begin{array}{cc}
{\sim}r & p \\
\downarrow & \downarrow \\
{\sim}r & {\sim}q \\
\downarrow & \downarrow \\
{\sim}r & p \\
\downarrow & \downarrow \\
\vdots & {\sim}q \\
 & \downarrow \\
 & \vdots
\end{array}$$

# DEFINITIONS: SUMMARY

- A justification frame contains a set of rules
- The rules determine which justifications are possible
- A branch evaluation determines which branches are "good"
- A justification is "good" if all its branches are "good"
- An interpretation is a model (according to some semantics, induced by the branch evaluation) if the truth value of each fact equals the value of its best justification

# OUTLINE

# TWO DEFINITIONS OF JUSTIFICATIONS

### Definition ([Den93, MBDH22])

Let $\mathbb{JF} = \langle \mathbb{F}, \mathbb{F}_d, R \rangle$ be a justification frame. A (tree-like) justification $J$ in $\mathbb{JF}$ is a labeled tree such that the set of children of each internal node is a case (rule) in $R$ for that node.

### Definition ([Mar09, DBS15, MPBD18, MBD21])

Let $\mathbb{JF} = \langle \mathbb{F}, \mathbb{F}_d, R \rangle$ be a justification frame. A (graph-like) justification $J$ in $\mathbb{JF}$ is a graph with nodes in $\mathbb{F}$ such that the set of children of each internal node is a case (rule) in $R$ for that node.

# TWO DEFINITIONS OF JUSTIFICATIONS

**Definition ([Den93, MBDH22])**

Let $\mathbb{JF} = \langle \mathbb{F}, \mathbb{F}_d, R \rangle$ be a justification frame. A (tree-like) justification $J$ in $\mathbb{JF}$ is a labeled tree such that the set of children of each internal node is a case (rule) in $R$ for that node.

**Definition ([Mar09, DBS15, MPBD18, MBD21])**

Let $\mathbb{JF} = \langle \mathbb{F}, \mathbb{F}_d, R \rangle$ be a justification frame. A (graph-like) justification $J$ in $\mathbb{JF}$ is a graph with nodes in $\mathbb{F}$ such that the set of children of each internal node is a case (rule) in $R$ for that node.

**Theorem**

*Every graph-like justification $J_g$ can be "unfolded" into a tree-like justification $J_t$ such that $\mathrm{val}_{\mathcal{B}}(J_t, x, I) \geq_t \mathrm{val}_{\mathcal{B}}(J_g, x, I)$ for each $x$*

# THE GRAPH-REDUCIBILITY PROBLEM

## Open Problem (The Graph-Reducibility Problem)

*Under which conditions on $\mathcal{B}$ can every tree-like justification $J_t$ be "reduced" to a graph-like justification $J_g$ with $\mathrm{val}_{\mathcal{B}}(J_g, x, I) \geq_t \mathrm{val}_{\mathcal{B}}(J_t, x, I)$?*

First studied by Marynissen et al [MBD20].

# THE GRAPH-REDUCIBILITY PROBLEM

## Open Problem (The Graph-Reducibility Problem)

*Under which conditions on $\mathcal{B}$ can every tree-like justification $J_t$ be "reduced" to a graph-like justification $J_g$ with $\mathrm{val}_{\mathcal{B}}(J_g, x, I) \geq_t \mathrm{val}_{\mathcal{B}}(J_t, x, I)$?*

First studied by Marynissen et al [MBD20].

### Example

The branch evaluation $\mathcal{B}_{\mathrm{ex}}$ maps:

- finite branches to their last element
- infinite branches with a consistent positive tail to $\mathbf{f}$
- infinite branches with a consistent negative tail to $\mathbf{t}$
- other branches to $\mathbf{u}$

consistent branch: whenever $x_i = x_j$ also $x_{i+1} = x_{j+1}$

# THE GRAPH-REDUCIBILITY PROBLEM: EXAMPLE

$$a \leftarrow b$$

$$a \leftarrow c \qquad \sim a \leftarrow \sim b, \sim c$$

$$b \leftarrow a \qquad \sim b \leftarrow \sim a$$

$$c \leftarrow a \qquad \sim c \leftarrow \sim a$$

---

$\mathcal{B}_{\mathrm{ex}}$ maps:

- finite branch: last element
- consistent positive tail: $\mathbf{f}$
- consistent negative tail: $\mathbf{t}$
- other: $\mathbf{u}$

# THE GRAPH-REDUCIBILITY PROBLEM: EXAMPLE

$a \leftarrow b$

$a \leftarrow c$ $\quad \sim a \leftarrow \sim b, \sim c$

$b \leftarrow a$ $\quad \sim b \leftarrow \sim a$

$c \leftarrow a$ $\quad \sim c \leftarrow \sim a$

Tree-like:

---

$\mathcal{B}_{\mathrm{ex}}$ maps:

- finite branch: last element
- consistent positive tail: $\mathbf{f}$
- consistent negative tail: $\mathbf{t}$
- other: $\mathbf{u}$

# THE GRAPH-REDUCIBILITY PROBLEM: EXAMPLE

$a \leftarrow b$

$a \leftarrow c \qquad {\sim}a \leftarrow {\sim}b, {\sim}c$

$b \leftarrow a \qquad {\sim}b \leftarrow {\sim}a$

$c \leftarrow a \qquad {\sim}c \leftarrow {\sim}a$

---

$\mathcal{B}_{\mathrm{ex}}$ maps:

- finite branch: last element
- consistent positive tail: $\mathbf{f}$
- consistent negative tail: $\mathbf{t}$
- other: $\mathbf{u}$

Tree-like:

$$a \downarrow b \downarrow a \downarrow c \downarrow a \downarrow b \downarrow a \downarrow c \downarrow \vdots$$

# THE GRAPH-REDUCIBILITY PROBLEM: EXAMPLE

$a \leftarrow b$

$a \leftarrow c$ $\quad$ $\sim a \leftarrow \sim b, \sim c$

$b \leftarrow a$ $\quad$ $\sim b \leftarrow \sim a$

$c \leftarrow a$ $\quad$ $\sim c \leftarrow \sim a$

---

$\mathcal{B}_{\text{ex}}$ maps:

- finite branch: last element
- consistent positive tail: $\mathbf{f}$
- consistent negative tail: $\mathbf{t}$
- other: $\mathbf{u}$

Tree-like:

$$\mathrm{SV}_t(a, I) = \mathbf{u}$$

$$a \downarrow b \downarrow a \downarrow c \downarrow a \downarrow b \downarrow a \downarrow c \downarrow \vdots$$

# THE GRAPH-REDUCIBILITY PROBLEM: EXAMPLE

$$
\begin{array}{ll}
a \leftarrow b & \\
a \leftarrow c & \sim a \leftarrow \sim b, \sim c \\
b \leftarrow a & \sim b \leftarrow \sim a \\
c \leftarrow a & \sim c \leftarrow \sim a
\end{array}
$$

---

$\mathcal{B}_{\text{ex}}$ maps:

- finite branch: last element
- consistent positive tail: $\mathbf{f}$
- consistent negative tail: $\mathbf{t}$
- other: $\mathbf{u}$

Tree-like:

$$
\begin{array}{c}
a \\ \downarrow \\ b \\ \downarrow \\ a \\ \downarrow \\ c \\ \downarrow \\ a \\ \downarrow \\ b \\ \downarrow \\ a \\ \downarrow \\ c \\ \downarrow \\ \vdots
\end{array}
$$

$\mathrm{SV}_t(a, I) = \mathbf{u}$
and
$\mathrm{SV}_t(\sim a, I) = \mathbf{u}$

# THE GRAPH-REDUCIBILITY PROBLEM: EXAMPLE

| | |
|---|---|
| $a \leftarrow b$ | |
| $a \leftarrow c$ | $\sim a \leftarrow \sim b, \sim c$ |
| $b \leftarrow a$ | $\sim b \leftarrow \sim a$ |
| $c \leftarrow a$ | $\sim c \leftarrow \sim a$ |

---

$\mathcal{B}_{\text{ex}}$ maps:

- finite branch: last element
- consistent positive tail: $\mathbf{f}$
- consistent negative tail: $\mathbf{t}$
- other: $\mathbf{u}$

Tree-like:

$$\begin{array}{c} a \\ \downarrow \\ b \\ \downarrow \\ a \\ \downarrow \\ c \\ \downarrow \\ a \\ \downarrow \\ b \\ \downarrow \\ a \\ \downarrow \\ c \\ \downarrow \\ \vdots \end{array}$$

$\mathrm{SV}_t(a, I) = \mathbf{u}$
and
$\mathrm{SV}_t(\sim a, I) = \mathbf{u}$

Graph-like:

# THE GRAPH-REDUCIBILITY PROBLEM: EXAMPLE

$$
\begin{array}{ll}
a \leftarrow b & \\
a \leftarrow c & \sim a \leftarrow \sim b, \sim c \\
b \leftarrow a & \sim b \leftarrow \sim a \\
c \leftarrow a & \sim c \leftarrow \sim a
\end{array}
$$

---

$\mathcal{B}_{\mathrm{ex}}$ maps:

- finite branch: last element
- consistent positive tail: $\mathbf{f}$
- consistent negative tail: $\mathbf{t}$
- other: $\mathbf{u}$

Tree-like:

$$
\begin{array}{c}
a \\ \downarrow \\ b \\ \downarrow \\ a \\ \downarrow \\ c \\ \downarrow \\ a \\ \downarrow \\ b \\ \downarrow \\ a \\ \downarrow \\ c \\ \downarrow \\ \vdots
\end{array}
$$

$\mathrm{SV}_t(a, I) = \mathbf{u}$
and
$\mathrm{SV}_t(\sim a, I) = \mathbf{u}$

Graph-like:

$$a \rightleftarrows b$$

$$a \rightleftarrows c$$

# THE GRAPH-REDUCIBILITY PROBLEM: EXAMPLE

$a \leftarrow b$

$a \leftarrow c \qquad \sim a \leftarrow \sim b, \sim c$

$b \leftarrow a \qquad \sim b \leftarrow \sim a$

$c \leftarrow a \qquad \sim c \leftarrow \sim a$

---

$\mathcal{B}_{\text{ex}}$ maps:

- finite branch: last element
- consistent positive tail: $\mathbf{f}$
- consistent negative tail: $\mathbf{t}$
- other: $\mathbf{u}$

Tree-like:

$$\begin{array}{c} a \\ \downarrow \\ b \\ \downarrow \\ a \\ \downarrow \\ c \\ \downarrow \\ a \\ \downarrow \\ b \\ \downarrow \\ a \\ \downarrow \\ c \\ \downarrow \\ \vdots \end{array}$$

$\mathrm{SV}_t(a, I) = \mathbf{u}$
and
$\mathrm{SV}_t(\sim a, I) = \mathbf{u}$

Graph-like:

$$a \rightleftarrows b$$

$$a \rightleftarrows c$$

$\mathrm{SV}_g(a, I) = \mathbf{f}$

# THE GRAPH-REDUCIBILITY PROBLEM: EXAMPLE

| | |
|---|---|
| $a \leftarrow b$ | |
| $a \leftarrow c$ | $\sim a \leftarrow \sim b, \sim c$ |
| $b \leftarrow a$ | $\sim b \leftarrow \sim a$ |
| $c \leftarrow a$ | $\sim c \leftarrow \sim a$ |

$\mathcal{B}_{\text{ex}}$ maps:

- finite branch: last element
- consistent positive tail: $\mathbf{f}$
- consistent negative tail: $\mathbf{t}$
- other: $\mathbf{u}$

Tree-like:

$$a \downarrow b \downarrow a \downarrow c \downarrow a \downarrow b \downarrow a \downarrow c \downarrow \vdots$$

$\mathrm{SV}_t(a, I) = \mathbf{u}$
and
$\mathrm{SV}_t(\sim a, I) = \mathbf{u}$

Graph-like:

$$a \rightleftarrows b$$

$$a \rightleftarrows c$$

$$\sim b \rightleftarrows \sim a \rightleftarrows \sim c$$

$\mathrm{SV}_g(a, I) = \mathbf{f}$
and
$\mathrm{SV}_g(\sim a, I) = \mathbf{u}$

# TREE-LIKE AND GRAPH-LIKE JUSTIFICATIONS

- Our results are only about tree-like justifications

# TREE-LIKE AND GRAPH-LIKE JUSTIFICATIONS

- Our results are only about tree-like justifications
- In examples, I might sometimes draw graph-like justifications, but mean their tree-unfolding

# OUTLINE

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l} \forall x : Q(x) \leftarrow R(x) \\ \left[\begin{array}{l} \forall x, y : R(x) \leftarrow P(x) \land G(x, y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x, y) \land R(x) \end{array}\right] \end{array}\right]$$

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x) \\ \left\lfloor\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \land R(x)\end{array}\right\rfloor\end{array}\right]$$

Fixpoint Computation:

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x) \\ \left\lfloor\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \land R(x)\end{array}\right\rfloor\end{array}\right]$$

Fixpoint Computation:

$Q_0 =$ all nodes

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x) \\ \left\lfloor\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \land R(x)\end{array}\right\rfloor\end{array}\right]$$

Fixpoint Computation:

$Q_0 =$ all nodes

$R_{0,0} = \emptyset$

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x)\\ \left\lfloor\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y)\\ \forall x : R(x) \leftarrow G(x,y) \land R(x)\end{array}\right\rfloor\end{array}\right]$$

Fixpoint Computation:

$Q_0 =$ all nodes

$R_{0,0} = \emptyset$

$R_{0,1} =$ all $P$s with an outgoing edge

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x) \\ \left\lfloor\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \land R(x)\end{array}\right\rfloor\end{array}\right]$$

Fixpoint Computation:

$Q_0 =$ all nodes

$R_{0,0} = \emptyset$

$R_{0,1} =$ all $P$s with an outgoing edge

...

$R_{0,\infty} =$ can reach a $P$ with an outgoing edge

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x) \\ \left[\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \land R(x)\end{array}\right]\end{array}\right]$$

Fixpoint Computation:

$Q_0 =$ all nodes

$R_{0,0} = \emptyset$

$R_{0,1} =$ all $P$s with an outgoing edge

...

$R_{0,\infty} =$ can reach a $P$ with an outgoing edge

$Q_1 =$ can reach a $P$ with an outgoing edge

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x) \\ \left\lfloor\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \land G(x, y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x, y) \land R(x)\end{array}\right\rfloor\end{array}\right]$$

Fixpoint Computation:

$Q_0 =$ all nodes

$R_{0,0} = \emptyset$

$R_{0,1} =$ all $P$s with an outgoing edge

...

$R_{0,\infty} =$ can reach a $P$ with an outgoing edge

$Q_1 =$ can reach a $P$ with an outgoing edge

$R_{1,0} = \emptyset$

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x) \\ \left\lfloor\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \land R(x)\end{array}\right\rfloor\end{array}\right]$$

Fixpoint Computation:

$Q_0 =$ all nodes

$R_{0,0} = \emptyset$

$R_{0,1} =$ all $P$s with an outgoing edge

...

$R_{0,\infty} =$ can reach a $P$ with an outgoing edge

$Q_1 =$ can reach a $P$ with an outgoing edge

$R_{1,0} = \emptyset$

$R_{1,1} =$ all $P$s with outgoing edge to a $Q_1$

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x) \\ \left\lfloor\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \land R(x)\end{array}\right\rfloor\end{array}\right]$$

Fixpoint Computation:

$Q_0 =$ all nodes

$R_{0,0} = \emptyset$

$R_{0,1} =$ all $P$s with an outgoing edge

...

$R_{0,\infty} =$ can reach a $P$ with an outgoing edge

$Q_1 =$ can reach a $P$ with an outgoing edge

$R_{1,0} = \emptyset$

$R_{1,1} =$ all $P$s with outgoing edge to a $Q_1$

...

$R_{1,\infty} =$ outgoing path with two $P$s

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\begin{bmatrix} \forall x : Q(x) \leftarrow R(x) \\ \begin{bmatrix} \forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \land R(x) \end{bmatrix} \end{bmatrix}$$

Fixpoint Computation:

$Q_0 =$ all nodes

$R_{0,0} = \emptyset$

$R_{0,1} =$ all $P$s with an outgoing edge

...

$R_{0,\infty} =$ can reach a $P$ with an outgoing edge

$Q_1 =$ can reach a $P$ with an outgoing edge

$R_{1,0} = \emptyset$

$R_{1,1} =$ all $P$s with outgoing edge to a $Q_1$

...

$R_{1,\infty} =$ outgoing path with two $P$s

$Q_2 =$ outgoing path with two $P$s

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\left[\begin{array}{l}\forall x : Q(x) \leftarrow R(x) \\ \left\lfloor\begin{array}{l}\forall x, y : R(x) \leftarrow P(x) \wedge G(x,y) \wedge Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \wedge R(x)\end{array}\right\rfloor\end{array}\right]$$

Fixpoint Computation:

$Q_0 =$ all nodes

$R_{0,0} = \emptyset$

$R_{0,1} =$ all $P$s with an outgoing edge

...

$R_{0,\infty} =$ can reach a $P$ with an outgoing edge

$Q_1 =$ can reach a $P$ with an outgoing edge

$R_{1,0} = \emptyset$

$R_{1,1} =$ all $P$s with outgoing edge to a $Q_1$

...

$R_{1,\infty} =$ outgoing path with two $P$s

$Q_2 =$ outgoing path with two $P$s

...

$Q_\infty =$ outgoing path with infinitely many $P$s

# NESTED LEAST AND GREATEST FIXPOINT DEFINITIONS

Nested induction and co-induction

## Example

Define $Q$ as the set of nodes in a (finite or infinite) graph with an outgoing path with infinitely many $P$'s.

$$\mathcal{B}_{\text{cwf}}: \left\{ \begin{array}{l} \forall x : Q(x) \leftarrow R(x) \\ \mathcal{B}_{\text{wf}}: \left\{ \begin{array}{l} \forall x, y : R(x) \leftarrow P(x) \land G(x,y) \land Q(y) \\ \forall x : R(x) \leftarrow G(x,y) \land R(x) \end{array} \right\} \end{array} \right\}$$

# AGGREGATES

Logic Programs with Aggregates

$$\left\{\begin{array}{l} p. \\ q. \\ s \leftarrow p \land \#\{p, q, s\} \geq 2. \end{array}\right\}$$

# AGGREGATES

Logic Programs with Aggregates

$$\left\{\begin{array}{l} p. \\ q. \\ s \leftarrow p \wedge \#\{p, q, s\} \geq 2. \end{array}\right\}$$

- Various semantics exist; old problem

# AGGREGATES

Logic Programs with Aggregates

$$\left\{\begin{array}{l} p. \\ q. \\ s \leftarrow p \wedge \#\{p, q, s\} \geq 2. \end{array}\right\}$$

- Various semantics exist; old problem
- Modular definition of the semantics

# AGGREGATES

Logic Programs with Aggregates

$$\begin{Bmatrix} p. \\ q. \\ s \leftarrow p \wedge \#\{p, q, s\} \geq 2. \end{Bmatrix}$$

- Various semantics exist; old problem
- Modular definition of the semantics
- Formal framework for comparison

# AGGREGATES

Logic Programs with Aggregates

$$\left\{\begin{array}{l} p. \\ q. \\ s \leftarrow p \land \#\{p, q, s\} \geq 2. \end{array}\right\}$$

- Various semantics exist; old problem
- Modular definition of the semantics
- Formal framework for comparison

$$\mathbb{JS}_{FLP} = \mathcal{B}_{\text{st}} : \left\{\begin{array}{l} p \leftarrow \mathbf{t} \\ q \leftarrow \mathbf{t} \\ s \leftarrow p, a \\ \mathcal{B}_{\text{KK}} : \left\{\begin{array}{l} a \leftarrow p, q \\ a \leftarrow s, q \\ a \leftarrow p, s \end{array}\right\} \end{array}\right\}$$

# AGGREGATES

Logic Programs with Aggregates

$$\begin{Bmatrix} p. \\ q. \\ s \leftarrow p \wedge \#\{p, q, s\} \geq 2. \end{Bmatrix}$$

- Various semantics exist; old problem
- Modular definition of the semantics
- Formal framework for comparison

$$\mathbb{JS}_{FLP} = \mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t} \\ q \leftarrow \mathbf{t} \\ s \leftarrow p, a \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} a \leftarrow p, q \\ a \leftarrow s, q \\ a \leftarrow p, s \end{array} \right\} \end{array} \right\}$$

$$\mathbb{JS}_{GZ} = \mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t} \\ q \leftarrow \mathbf{t} \\ s \leftarrow p, a \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} a \leftarrow p, q, {\sim} s \\ a \leftarrow s, q, {\sim} p \\ a \leftarrow p, s, {\sim} q \\ a \leftarrow p, q, s \end{array} \right\} \end{array} \right\}$$

## NESTED SYSTEMS

$$\mathbb{JS}_{FLP} = \mathcal{B}_{\text{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t} \\ q \leftarrow \mathbf{t} \\ s \leftarrow p, a \\ \mathcal{B}_{\text{KK}} : \left\{ \begin{array}{l} a \leftarrow p, q \\ a \leftarrow s, q \\ a \leftarrow p, s \end{array} \right\} \end{array} \right\}$$

### Definition ([DBS15])

Let $\mathbb{F}$ be a fact space. A nested justification system on $\mathbb{F}$ is a tuple

$$\left\langle \mathbb{F}, \mathbb{F}_d, \mathbb{F}_{dl}, R, \mathcal{B}, \left\{ \mathbb{JS}^1, \ldots, \mathbb{JS}^k \right\} \right\rangle$$

such that:

# NESTED SYSTEMS

$$\mathbb{JS}_{FLP} = \mathcal{B}_{\text{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t} \\ q \leftarrow \mathbf{t} \\ s \leftarrow p, a \\ \mathcal{B}_{\text{KK}} : \left\{ \begin{array}{l} a \leftarrow p, q \\ a \leftarrow s, q \\ a \leftarrow p, s \end{array} \right\} \end{array} \right\}$$

- $\mathbb{F}_{dl} = \{p, q, s, \sim p, \sim q, \sim s\}$

## Definition ([DBS15])

Let $\mathbb{F}$ be a fact space. A nested justification system on $\mathbb{F}$ is a tuple

$$\left\langle \mathbb{F}, \mathbb{F}_d, \mathbb{F}_{dl}, R, \mathcal{B}, \left\{ \mathbb{JS}^1, \ldots, \mathbb{JS}^k \right\} \right\rangle$$

such that:

1. $\langle \mathbb{F}, \mathbb{F}_{dl}, R, \mathcal{B} \rangle$ is a justification system;

# NESTED SYSTEMS

$$\mathbb{JS}_{FLP} = \mathcal{B}_{\text{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t} \\ q \leftarrow \mathbf{t} \\ s \leftarrow p, a \\ \mathcal{B}_{\text{KK}} : \left\{ \begin{array}{l} a \leftarrow p, q \\ a \leftarrow s, q \\ a \leftarrow p, s \end{array} \right\} \end{array} \right\}$$

- $\mathbb{F}_{dl} = \{p, q, s, \sim p, \sim q, \sim s\}$
- $\mathbb{F}_d^1 = \{a, \sim a\}$

## Definition ([DBS15])

Let $\mathbb{F}$ be a fact space. A nested justification system on $\mathbb{F}$ is a tuple

$$\left\langle \mathbb{F}, \mathbb{F}_d, \mathbb{F}_{dl}, R, \mathcal{B}, \left\{ \mathbb{JS}^1, \ldots, \mathbb{JS}^k \right\} \right\rangle$$

such that:

1. $\langle \mathbb{F}, \mathbb{F}_{dl}, R, \mathcal{B} \rangle$ is a justification system;
2. each $\mathbb{JS}^i$ is a nested justification system $\langle \mathbb{F}^i, \mathbb{F}^i_d, \mathbb{F}^i_{dl}, R^i, \mathcal{B}^i, \ldots \rangle$;

# NESTED SYSTEMS

$$\mathbb{JS}_{FLP} = \mathcal{B}_{\mathrm{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t} \\ q \leftarrow \mathbf{t} \\ s \leftarrow p, a \\ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} a \leftarrow p, q \\ a \leftarrow s, q \\ a \leftarrow p, s \end{array} \right\} \end{array} \right\}$$

- $\mathbb{F}_{dl} = \{p, q, s, \sim p, \sim q, \sim s\}$
- $\mathbb{F}_d^1 = \{a, \sim a\}$
- $\mathbb{F}_d = \mathbb{F}_{dl} \cup \mathbb{F}_d^1$

## Definition ([DBS15])

Let $\mathbb{F}$ be a fact space. A nested justification system on $\mathbb{F}$ is a tuple

$$\left\langle \mathbb{F}, \mathbb{F}_d, \mathbb{F}_{dl}, R, \mathcal{B}, \left\{ \mathbb{JS}^1, \ldots, \mathbb{JS}^k \right\} \right\rangle$$

such that:

1. $\langle \mathbb{F}, \mathbb{F}_{dl}, R, \mathcal{B} \rangle$ is a justification system;
2. each $\mathbb{JS}^i$ is a nested justification system $\langle \mathbb{F}^i, \mathbb{F}_d^i, \mathbb{F}_{dl}^i, R^i, \mathcal{B}^i, \ldots \rangle$;
3. $\mathbb{F}_d$ is partitioned into $\left\{ \mathbb{F}_{dl}, \mathbb{F}_d^1, \ldots, \mathbb{F}_d^k \right\}$;

# NESTED SYSTEMS

$$\mathbb{JS}_{FLP} = \mathcal{B}_{\text{st}} : \left\{\begin{array}{l} p \leftarrow \mathbf{t} \\ q \leftarrow \mathbf{t} \\ s \leftarrow p, a \\ \mathcal{B}_{\text{KK}} : \left\{\begin{array}{l} a \leftarrow p, q \\ a \leftarrow s, q \\ a \leftarrow p, s \end{array}\right\} \end{array}\right\}$$

- $\mathbb{F}_{dl} = \{p, q, s, \sim p, \sim q, \sim s\}$
- $\mathbb{F}^1_d = \{a, \sim a\}$
- $\mathbb{F}_d = \mathbb{F}_{dl} \cup \mathbb{F}^1_d$

## Definition ([DBS15])

Let $\mathbb{F}$ be a fact space. A nested justification system on $\mathbb{F}$ is a tuple

$$\left\langle \mathbb{F}, \mathbb{F}_d, \mathbb{F}_{dl}, R, \mathcal{B}, \left\{\mathbb{JS}^1, \ldots, \mathbb{JS}^k\right\}\right\rangle$$

such that:

1. $\langle \mathbb{F}, \mathbb{F}_{dl}, R, \mathcal{B}\rangle$ is a justification system;
2. each $\mathbb{JS}^i$ is a nested justification system $\langle \mathbb{F}^i, \mathbb{F}^i_d, \mathbb{F}^i_{dl}, R^i, \mathcal{B}^i, \ldots\rangle$;
3. $\mathbb{F}_d$ is partitioned into $\left\{\mathbb{F}_{dl}, \mathbb{F}^1_d, \ldots, \mathbb{F}^k_d\right\}$;
4. $\mathbb{F} = \cup_{i=1}^k \mathbb{F}^k$;

# NESTED SYSTEMS

$$\mathbb{JS}_{FLP} = \mathcal{B}_{\text{st}} : \left\{ \begin{array}{l} p \leftarrow \mathbf{t} \\ q \leftarrow \mathbf{t} \\ s \leftarrow p, a \\ \mathcal{B}_{\text{KK}} : \left\{ \begin{array}{l} a \leftarrow p, q \\ a \leftarrow s, q \\ a \leftarrow p, s \end{array} \right\} \end{array} \right\}$$

- $\mathbb{F}_{dl} = \{p, q, s, {\sim}p, {\sim}q, {\sim}s\}$
- $\mathbb{F}_d^1 = \{a, {\sim}a\}$
- $\mathbb{F}_d = \mathbb{F}_{dl} \cup \mathbb{F}_d^1$

## Definition ([DBS15])

Let $\mathbb{F}$ be a fact space. A nested justification system on $\mathbb{F}$ is a tuple

$$\left\langle \mathbb{F}, \mathbb{F}_d, \mathbb{F}_{dl}, R, \mathcal{B}, \left\{\mathbb{JS}^1, \ldots, \mathbb{JS}^k\right\} \right\rangle$$

such that:

1. $\langle \mathbb{F}, \mathbb{F}_{dl}, R, \mathcal{B} \rangle$ is a justification system;
2. each $\mathbb{JS}^i$ is a nested justification system $\langle \mathbb{F}^i, \mathbb{F}_d^i, \mathbb{F}_{dl}^i, R^i, \mathcal{B}^i, \ldots \rangle$;
3. $\mathbb{F}_d$ is partitioned into $\left\{\mathbb{F}_{dl}, \mathbb{F}_d^1, \ldots, \mathbb{F}_d^k\right\}$;
4. $\mathbb{F} = \cup_{i=1}^k \mathbb{F}^k$;
5. $\mathbb{F}_o^i \subseteq \mathbb{F}_o \cup \mathbb{F}_{dl}$

# OUTLINE

# COMPRESSION SEMANTICS

The original semantics of [DBS15]

# COMPRESSION SEMANTICS

The original semantics of [DBS15]

- For each fact $a$ defined in the inner system

# COMPRESSION SEMANTICS

The original semantics of [DBS15]

- For each fact $a$ defined in the inner system
- and each justification $J$ of $a$

# COMPRESSION SEMANTICS

The original semantics of [DBS15]

- For each fact $a$ defined in the inner system
- and each justification $J$ of $a$
- and each rule (in outer system) with $a$ in its body

# COMPRESSION SEMANTICS

The original semantics of [DBS15]

- For each fact $a$ defined in the inner system
- and each justification $J$ of $a$
- and each rule (in outer system) with $a$ in its body
- make new rule with $a$ replaced by $\mathcal{B}(J)$

# COMPRESSION SEMANTICS

The original semantics of [DBS15]

- For each fact $a$ defined in the inner system
- and each justification $J$ of $a$
- and each rule (in outer system) with $a$ in its body
- make new rule with $a$ replaced by $\mathcal{B}(J)$

Disadvantages:

- "Explanations" not in terms of input rules
- Only defined for parametric inner systems

# COMPRESSION: EXAMPLE

$$\mathbb{JS} =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

# COMPRESSION: EXAMPLE

$$\mathbb{JS} = \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) = \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ \\ {\sim}r \leftarrow {\sim}q \end{array} \right\}$$

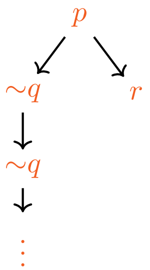
# COMPRESSION: EXAMPLE

$$\mathbb{JS} =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

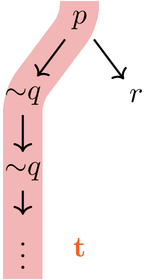
$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ \\ {\sim}r \leftarrow {\sim}q \end{array} \right\}$$



# COMPRESSION: EXAMPLE

$$\mathbb{JS} = \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \color{orange}{\mathcal{B}_{\mathrm{wf}}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) = \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ \\ {\sim}r \leftarrow {\sim}q \end{array} \right\}$$



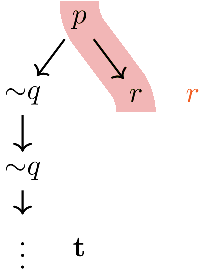
# COMPRESSION: EXAMPLE

$$\mathbb{JS} =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ \sim r \leftarrow \sim p \\ \sim r \leftarrow \sim q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow \sim q, r \\ \sim p \leftarrow q \\ \sim p \leftarrow \sim r \\ q \leftarrow q \\ \sim q \leftarrow \sim q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ \sim r \leftarrow \sim p \\ \\ \sim r \leftarrow \sim q \end{array} \right\}$$



# COMPRESSION: EXAMPLE

$$\mathbb{JS} =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow \cancel{p}\, \mathbf{t}, r, q \\ {\sim}r \leftarrow {\sim}p \\ \\ {\sim}r \leftarrow {\sim}q \end{array} \right\}$$

$$\begin{tikzcd} & p \arrow[dl] \arrow[dr] & & \\ {\sim}q \arrow[d] & & r & r \\ {\sim}q \arrow[d] & & & \\ \vdots & \mathbf{t} & & \end{tikzcd}$$

# COMPRESSION: EXAMPLE

$$\mathbb{JS} =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow \cancel{p}\,\mathbf{t}, r, q \\ {\sim}r \leftarrow {\sim}p \\ \\ {\sim}r \leftarrow {\sim}q \end{array} \right\}$$

$$\begin{array}{c} q \\ \downarrow \\ q \\ \downarrow \\ \vdots \end{array}$$

# COMPRESSION: EXAMPLE

$$\mathbb{JS} =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow \cancel{p}\,\mathbf{t}, r, \cancel{q}\,\mathbf{f} \\ {\sim}r \leftarrow {\sim}p \\ \\ {\sim}r \leftarrow {\sim}q \end{array} \right\}$$

$$\begin{array}{c} q \\ \downarrow \\ q \\ \downarrow \\ \vdots \end{array} \quad \mathbf{f}$$

# COMPRESSION: EXAMPLE

$$
\mathbb{JS} =
\left\{
\begin{array}{l}
\mathcal{B}_{\mathrm{KK}} : \left\{
\begin{array}{l}
r \leftarrow p, q \\
{\sim}r \leftarrow {\sim}p \\
{\sim}r \leftarrow {\sim}q \\
\mathcal{B}_{\mathrm{wf}} : \left\{
\begin{array}{l}
p \leftarrow {\sim}q, r \\
{\sim}p \leftarrow q \\
{\sim}p \leftarrow {\sim}r \\
q \leftarrow q \\
{\sim}q \leftarrow {\sim}q
\end{array}
\right\}
\end{array}
\right.
\end{array}
\right\}
$$

$$
\mathrm{Compress}(\mathbb{JS}) =
\mathcal{B}_{\mathrm{KK}} : \left\{
\begin{array}{l}
r \leftarrow \cancel{p}\,\mathbf{t}, r, \cancel{q}\,\mathbf{f} \\
{\sim}r \leftarrow {\sim}p \\
\\
{\sim}r \leftarrow {\sim}q
\end{array}
\right\}
$$

$$
\begin{array}{ccc}
{\sim}p & \qquad & {\sim}p \\
\downarrow & & \downarrow \\
{\sim}r & & q \\
 & & \downarrow \\
 & & q \\
 & & \downarrow \\
 & & \vdots
\end{array}
$$

# COMPRESSION: EXAMPLE

$$\mathbb{JS} = \left\{ \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\} \right.$$

$$\mathrm{Compress}(\mathbb{JS}) = \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow \cancel{p}\,\mathbf{t}, r, \cancel{q}\,\mathbf{f} \\ {\sim}r \leftarrow {\sim}p \\ \\ {\sim}r \leftarrow {\sim}q \end{array} \right\}$$

$$\boxed{\begin{array}{c} {\sim}p \\ \downarrow \\ {\sim}r \end{array}} \quad {\sim}r \qquad \boxed{\begin{array}{c} {\sim}p \\ \downarrow \\ q \\ \downarrow \\ q \\ \downarrow \\ \vdots \end{array}} \quad \mathbf{f}$$

# COMPRESSION: EXAMPLE

$$
\mathbb{JS} = \\
\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l}
r \leftarrow p, q \\
{\sim}r \leftarrow {\sim}p \\
{\sim}r \leftarrow {\sim}q \\
\mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l}
p \leftarrow {\sim}q, r \\
{\sim}p \leftarrow q \\
{\sim}p \leftarrow {\sim}r \\
q \leftarrow q \\
{\sim}q \leftarrow {\sim}q
\end{array} \right\}
\end{array} \right\}
$$

$$
\mathrm{Compress}(\mathbb{JS}) = \\
\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l}
r \leftarrow \cancel{p}\,\mathbf{t}, r, \cancel{q}\,\mathbf{f} \\
{\sim}r \leftarrow \cancel{{\sim}p}\,{\sim}r \\
{\sim}r \leftarrow \cancel{{\sim}p}\,\mathbf{f} \\
{\sim}r \leftarrow {\sim}q
\end{array} \right\}
$$

$$
\begin{array}{cccc}
{\sim}p & & {\sim}p & \\
\downarrow & & \downarrow & \\
{\sim}r & {\sim}r & q & \\
 & & \downarrow & \\
 & & q & \\
 & & \downarrow & \\
 & & \vdots & \mathbf{f}
\end{array}
$$

# COMPRESSION: EXAMPLE

$$\mathbb{JS} =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow \cancel{p}\,\mathbf{t}, r, \cancel{q}\,\mathbf{f} \\ {\sim}r \leftarrow \cancel{{\sim}p}\,{\sim}r \\ {\sim}r \leftarrow \cancel{{\sim}p}\,\mathbf{f} \\ {\sim}r \leftarrow \cancel{{\sim}q}\,\mathbf{t} \end{array} \right\}$$

$$\begin{array}{cc} {\sim}q & \\ \downarrow & \\ {\sim}q & \\ \downarrow & \\ \vdots & \mathbf{t} \end{array}$$

# COMPRESSION: EXAMPLE

$$\mathbb{JS} =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ \sim r \leftarrow \sim p \\ \sim r \leftarrow \sim q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow \sim q, r \\ \sim p \leftarrow q \\ \sim p \leftarrow \sim r \\ q \leftarrow q \\ \sim q \leftarrow \sim q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ \sim r \leftarrow \sim r \\ \sim r \leftarrow \mathbf{f} \\ \sim r \leftarrow \mathbf{t} \end{array} \right\}$$

# MERGE: EXAMPLE

$$\mathbb{JS} =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

# MERGE: EXAMPLE

$$\mathbb{JS} = \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Merge}(\mathbb{JS}) = \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\}$$

# MERGE: EXAMPLE

$$\mathbb{JS} = \mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ \mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\} \end{array} \right\}$$

$$\mathrm{Merge}(\mathbb{JS}) = \mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\}$$

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^{r}_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^{r}_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?

Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^{r}_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element
2. If it contains infinitely many facts defined at the highest level, project onto that level and use $\mathcal{B}_{\mathrm{KK}}$

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^{r}_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element
2. If it contains infinitely many facts defined at the highest level, project onto that level and use $\mathcal{B}_{\mathrm{KK}}$
3. Otherwise, project onto the lowest level and evaluate with $\mathcal{B}_{\mathrm{wf}}$

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^{r}_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element
2. If it contains infinitely many facts defined at the highest level, project onto that level and use $\mathcal{B}_{\mathrm{KK}}$
3. Otherwise, project onto the lowest level and evaluate with $\mathcal{B}_{\mathrm{wf}}$

## Example

1. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to \mathbf{t}) = \mathbf{t}$

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^r_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element
2. If it contains infinitely many facts defined at the highest level, project onto that level and use $\mathcal{B}_{\mathrm{KK}}$
3. Otherwise, project onto the lowest level and evaluate with $\mathcal{B}_{\mathrm{wf}}$

## Example

1. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to \mathbf{t}) = \mathbf{t}$
2. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to r \to p \to \dots)$

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}_{\mathrm{KK}}^{r}.\mathcal{B}_{\mathrm{wf}}^{p,q}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element
2. If it contains infinitely many facts defined at the highest level, project onto that level and use $\mathcal{B}_{\mathrm{KK}}$
3. Otherwise, project onto the lowest level and evaluate with $\mathcal{B}_{\mathrm{wf}}$

## Example

1. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to \mathbf{t}) = \mathbf{t}$
2. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to r \to p \to \dots) = \mathcal{B}_{\mathrm{KK}}(r \to r \to r \to \dots)$

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^{r}_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element
2. If it contains infinitely many facts defined at the highest level, project onto that level and use $\mathcal{B}_{\mathrm{KK}}$
3. Otherwise, project onto the lowest level and evaluate with $\mathcal{B}_{\mathrm{wf}}$

## Example

1. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to \mathbf{t}) = \mathbf{t}$
2. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to r \to p \to \dots) = \mathcal{B}_{\mathrm{KK}}(r \to r \to r \to \dots) = \mathbf{u}$

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^{r}_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element
2. If it contains infinitely many facts defined at the highest level, project onto that level and use $\mathcal{B}_{\mathrm{KK}}$
3. Otherwise, project onto the lowest level and evaluate with $\mathcal{B}_{\mathrm{wf}}$

## Example

1. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to \mathbf{t}) = \mathbf{t}$
2. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to r \to p \to \dots) = \mathcal{B}_{\mathrm{KK}}(r \to r \to r \to \dots) = \mathbf{u}$
3. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to {\sim}q \to {\sim}q \to {\sim}q \to \dots)$

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^{r}_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element
2. If it contains infinitely many facts defined at the highest level, project onto that level and use $\mathcal{B}_{\mathrm{KK}}$
3. Otherwise, project onto the lowest level and evaluate with $\mathcal{B}_{\mathrm{wf}}$

## Example

1. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to \mathbf{t}) = \mathbf{t}$
2. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to r \to p \to \dots) = \mathcal{B}_{\mathrm{KK}}(r \to r \to r \to \dots) = \mathbf{u}$
3. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to {\sim}q \to {\sim}q \to {\sim}q \to \dots)$
   $= \mathcal{B}_{\mathrm{wf}}(p \to p \to {\sim}q \to {\sim}q \to {\sim}q \to \dots)$

# THE MERGE BRANCH EVALUATION

How is $\mathcal{B}^{r}_{\mathrm{KK}}.\mathcal{B}^{p,q}_{\mathrm{wf}}$ defined?
Take any branch $\mathbf{b}$ (using facts from both levels)

1. If $\mathbf{b}$ is finite, map to its last element
2. If it contains infinitely many facts defined at the highest level, project onto that level and use $\mathcal{B}_{\mathrm{KK}}$
3. Otherwise, project onto the lowest level and evaluate with $\mathcal{B}_{\mathrm{wf}}$

## Example

1. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to \mathbf{t}) = \mathbf{t}$
2. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to r \to p \to \dots) = \mathcal{B}_{\mathrm{KK}}(r \to r \to r \to \dots) = \mathbf{u}$
3. $\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}}\,(r \to p \to r \to p \to {\sim}q \to {\sim}q \to {\sim}q \to \dots)$
   $= \mathcal{B}_{\mathrm{wf}}(p \to p \to {\sim}q \to {\sim}q \to {\sim}q \to \dots) = \mathbf{t}$

# EQUIVALENCE OF $\mathrm{Compress}$ AND $\mathrm{Merge}$

**Theorem**

*For almost all compressible systems,* $\mathrm{Compress}(\mathbb{JS})$ *and* $\mathrm{Merge}(\mathbb{JS})$ *are equivalent.*

# EQUIVALENCE: PROOF IDEA

$$\mathrm{Merge}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \begin{Bmatrix} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{Bmatrix}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \begin{Bmatrix} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{Bmatrix}$$

# EQUIVALENCE: PROOF IDEA

$$\mathrm{Merge}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\}$$

$r$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{array} \right\}$$

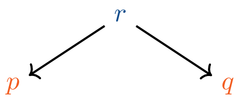
# EQUIVALENCE: PROOF IDEA

$\mathrm{Merge}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \begin{Bmatrix} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{Bmatrix}$$

$\mathrm{Compress}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}} : \begin{Bmatrix} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{Bmatrix}$$



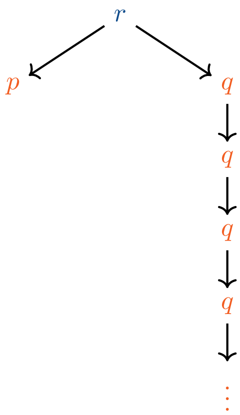
# EQUIVALENCE: PROOF IDEA

$\text{Merge}(\mathbb{JS}) =$

$$\mathcal{B}_{\text{KK}}.\mathcal{B}_{\text{wf}} : \begin{Bmatrix} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{Bmatrix}$$

$\text{Compress}(\mathbb{JS}) =$

$$\mathcal{B}_{\text{KK}} : \begin{Bmatrix} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{Bmatrix}$$



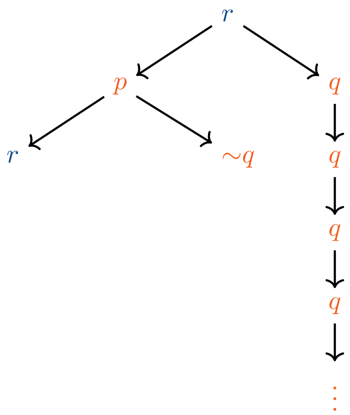
# EQUIVALENCE: PROOF IDEA

$\mathrm{Merge}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \begin{Bmatrix} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{Bmatrix}$$

$\mathrm{Compress}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}} : \begin{Bmatrix} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{Bmatrix}$$



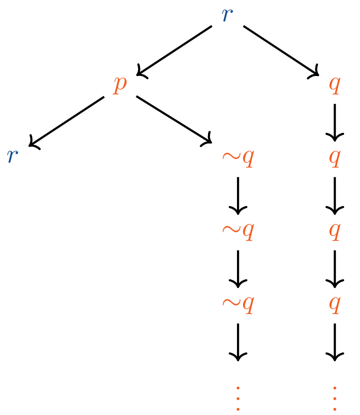
# EQUIVALENCE: PROOF IDEA

Merge($\mathbb{JS}$) =

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \left\{\begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array}\right\}$$

Compress($\mathbb{JS}$) =

$$\mathcal{B}_{\mathrm{KK}} : \left\{\begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{array}\right\}$$



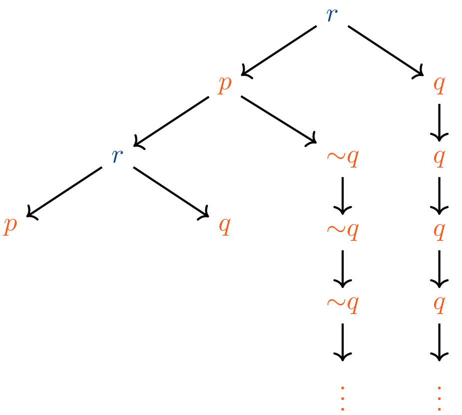
# EQUIVALENCE: PROOF IDEA

$$\mathrm{Merge}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{array} \right\}$$



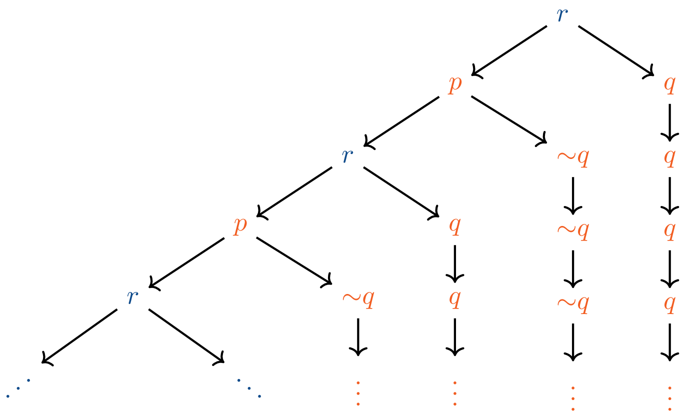
# EQUIVALENCE: PROOF IDEA

$\mathrm{Merge}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \left\{\begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array}\right\}$$

$\mathrm{Compress}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}} : \left\{\begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{array}\right\}$$



# EQUIVALENCE: PROOF IDEA

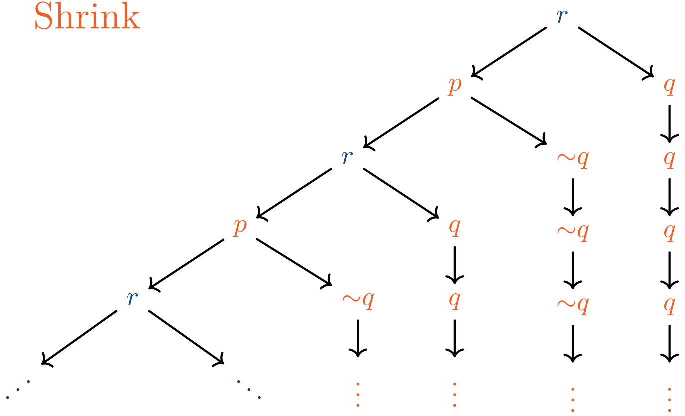
$$\mathrm{Merge}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{ \begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{array} \right\}$$

Shrink



# EQUIVALENCE: PROOF IDEA

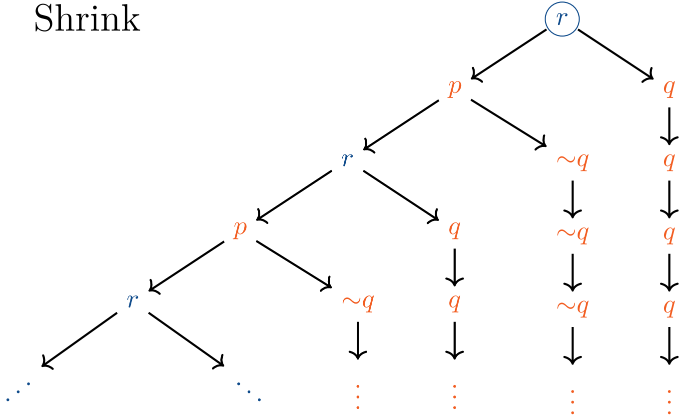
$$\text{Merge}(\mathbb{JS}) =$$

$$\color{blue}{\mathcal{B}_{\text{KK}}}.\color{orange}{\mathcal{B}_{\text{wf}}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\}$$

$$\text{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\text{KK}} : \left\{ \begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{array} \right\}$$

Shrink



# EQUIVALENCE: PROOF IDEA

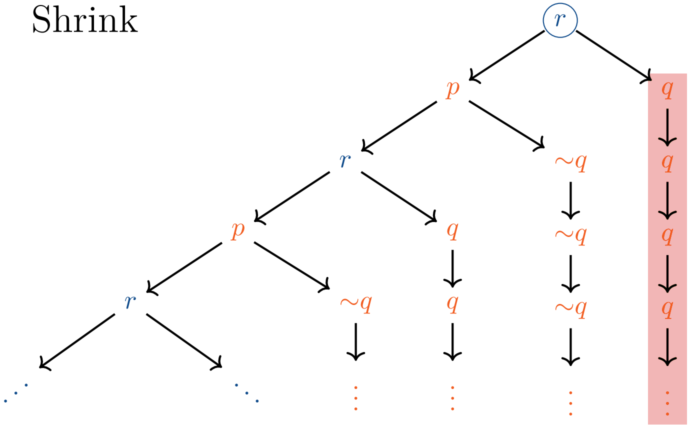
$\text{Merge}(\mathbb{JS}) =$

$$\mathcal{B}_{\text{KK}}.\mathcal{B}_{\text{wf}} : \left\{ \begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array} \right\}$$

$\text{Compress}(\mathbb{JS}) =$

$$\mathcal{B}_{\text{KK}} : \left\{ \begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{array} \right\}$$

Shrink



# EQUIVALENCE: PROOF IDEA

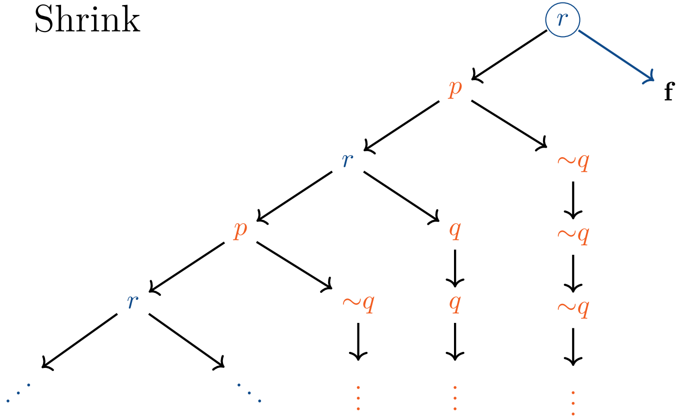
$$\mathrm{Merge}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \begin{Bmatrix} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{Bmatrix}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \begin{Bmatrix} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{Bmatrix}$$

Shrink



# EQUIVALENCE: PROOF IDEA

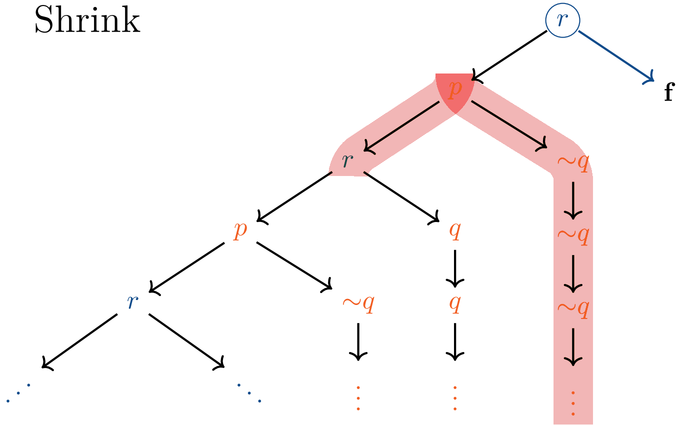
$$\mathrm{Merge}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \left\{\begin{array}{l} r \leftarrow p, q \\ \sim r \leftarrow \sim p \\ \sim r \leftarrow \sim q \\ p \leftarrow \sim q, r \\ \sim p \leftarrow q \\ \sim p \leftarrow \sim r \\ q \leftarrow q \\ \sim q \leftarrow \sim q \end{array}\right\}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \left\{\begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ \sim r \leftarrow \mathbf{t} \\ \sim r \leftarrow \sim r \\ \sim r \leftarrow \mathbf{f} \end{array}\right\}$$

Shrink



# EQUIVALENCE: PROOF IDEA

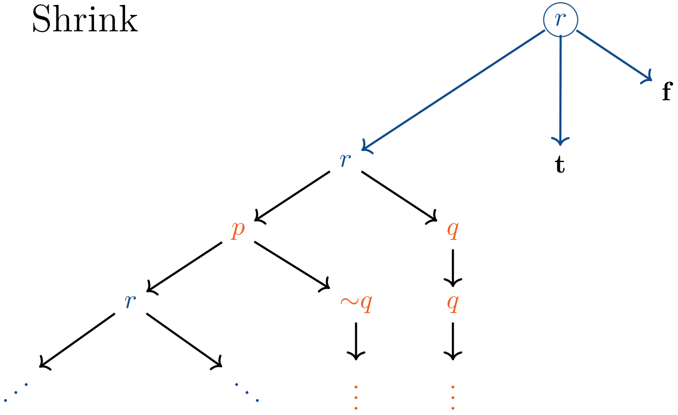
$$\mathrm{Merge}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \begin{Bmatrix} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{Bmatrix}$$

$$\mathrm{Compress}(\mathbb{JS}) =$$

$$\mathcal{B}_{\mathrm{KK}} : \begin{Bmatrix} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{Bmatrix}$$

Shrink



# EQUIVALENCE: PROOF IDEA

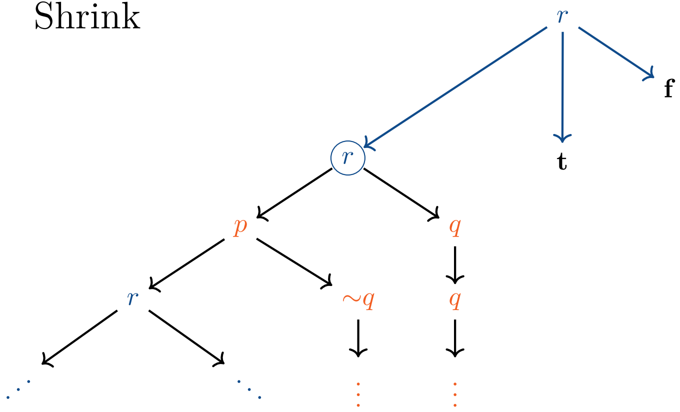
$\mathrm{Merge}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \begin{Bmatrix} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{Bmatrix}$$

$\mathrm{Compress}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}} : \begin{Bmatrix} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{Bmatrix}$$

Shrink



# EQUIVALENCE: PROOF IDEA

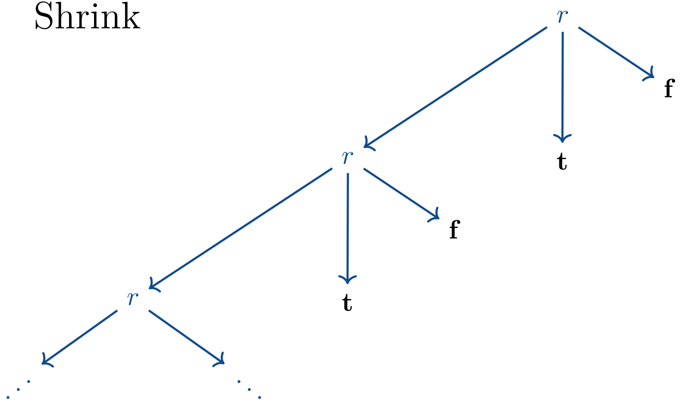
$\mathrm{Merge}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \begin{Bmatrix} r \leftarrow p, q \\ \sim r \leftarrow \sim p \\ \sim r \leftarrow \sim q \\ p \leftarrow \sim q, r \\ \sim p \leftarrow q \\ \sim p \leftarrow \sim r \\ q \leftarrow q \\ \sim q \leftarrow \sim q \end{Bmatrix}$$

$\mathrm{Compress}(\mathbb{JS}) =$

$$\mathcal{B}_{\mathrm{KK}} : \begin{Bmatrix} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ \sim r \leftarrow \mathbf{t} \\ \sim r \leftarrow \sim r \\ \sim r \leftarrow \mathbf{f} \end{Bmatrix}$$

Shrink



# EQUIVALENCE: PROOF IDEA

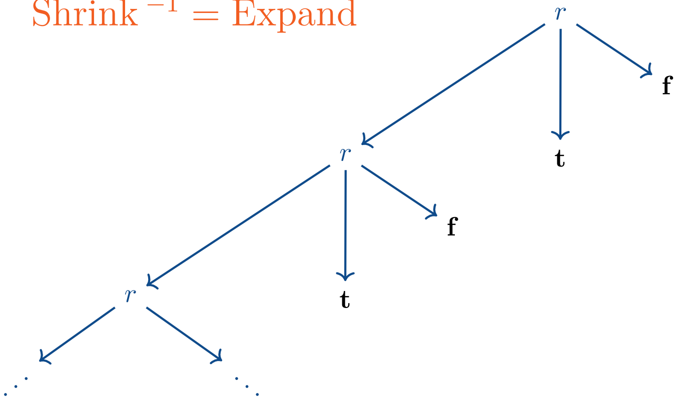
Merge($\mathbb{JS}$) =

$$\mathcal{B}_{\mathrm{KK}}.\mathcal{B}_{\mathrm{wf}} : \left\{\begin{array}{l} r \leftarrow p, q \\ {\sim}r \leftarrow {\sim}p \\ {\sim}r \leftarrow {\sim}q \\ p \leftarrow {\sim}q, r \\ {\sim}p \leftarrow q \\ {\sim}p \leftarrow {\sim}r \\ q \leftarrow q \\ {\sim}q \leftarrow {\sim}q \end{array}\right\}$$

Compress($\mathbb{JS}$) =

$$\mathcal{B}_{\mathrm{KK}} : \left\{\begin{array}{l} r \leftarrow \mathbf{t}, r, \mathbf{f} \\ {\sim}r \leftarrow \mathbf{t} \\ {\sim}r \leftarrow {\sim}r \\ {\sim}r \leftarrow \mathbf{f} \end{array}\right\}$$

$\text{Shrink}^{-1} = \text{Expand}$



# OUTLINE

# CONCLUSION

- Two characterizations of semantics of nested justification systems

# CONCLUSION

- Two characterizations of semantics of nested justification systems
- Important for practical applicability of nesting

# CONCLUSION

- Two characterizations of semantics of nested justification systems
- Important for practical applicability of nesting
- Consistency of nested systems

# CONCLUSION

- Two characterizations of semantics of nested justification systems
- Important for practical applicability of nesting
- Consistency of nested systems

**Open Question**

*Are* Merge *and* Compress *equivalent in the graph-like setting?*

# CONCLUSION

- Two characterizations of semantics of nested justification systems
- Important for practical applicability of nesting
- Consistency of nested systems

**Open Question**

*Are* Merge *and* Compress *equivalent in the graph-like setting?*

# CONCLUSION

- Two characterizations of semantics of nested justification systems
- Important for practical applicability of nesting
- Consistency of nested systems

**Open Question**

*Are* Merge *and* Compress *equivalent in the graph-like setting?*

Interested? I'm looking for good postdocs/PhD students

Thanks for your attention!

# REFERENCES


[DBS15] Marc Denecker, Gerhard Brewka, and Hannes Strass. A formal theory of justifications. In Francesco Calimeri, Giovambattista Ianni, and Mirosław Truszczyński, editors, *Logic Programming and Nonmonotonic Reasoning - 13th International Conference, LPNMR 2015, Lexington, KY, USA, September 27-30, 2015. Proceedings*, volume 9345 of *Lecture Notes in Computer Science*, pages 250–264. Springer, 2015.

[Den93] Marc Denecker. *Knowledge representation and reasoning in incomplete logic programming*. PhD thesis, K.U.Leuven, Leuven, Belgium, September 1993.

[Mar09] Maarten Mariën. *Model Generation for ID-Logic*. PhD thesis, Department of Computer Science, KU Leuven, Belgium, February 2009.

[MBD20] Simon Marynissen, Bart Bogaerts, and Marc Denecker. Exploiting game theory for analysing justifications. *Theory Pract. Log. Program.*, 20(6):880–894, 2020.

[MBD21] Simon Marynissen, Bart Bogaerts, and Marc Denecker. On the relation between approximation fixpoint theory and justification theory. In Zhi-Hua Zhou, editor, *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI 2021, Virtual Event / Montreal, Canada, 19-27 August 2021*, pages 1973–1980. ijcai.org, 2021.

# REFERENCES

[MBDH22] Simon Marynissen, Bart Bogaerts, Marc Denecker, and Jesse Heyninck. On nested justification systems. *Theory Pract. Log. Program.*, 22, 2022. To appear (Accepted for ICLP 2022 special issue in TPLP).

[MPBD18] Simon Marynissen, Niko Passchyn, Bart Bogaerts, and Marc Denecker. Consistency in justification theory. In *Proceedings of 17th International Workshop on Non-Monotonic Reasoning (NMR 2018), Tempe, Arizona, USA, Oct. 27-29, 2018*, pages 41–52. AAAI Press 2018, 2018.