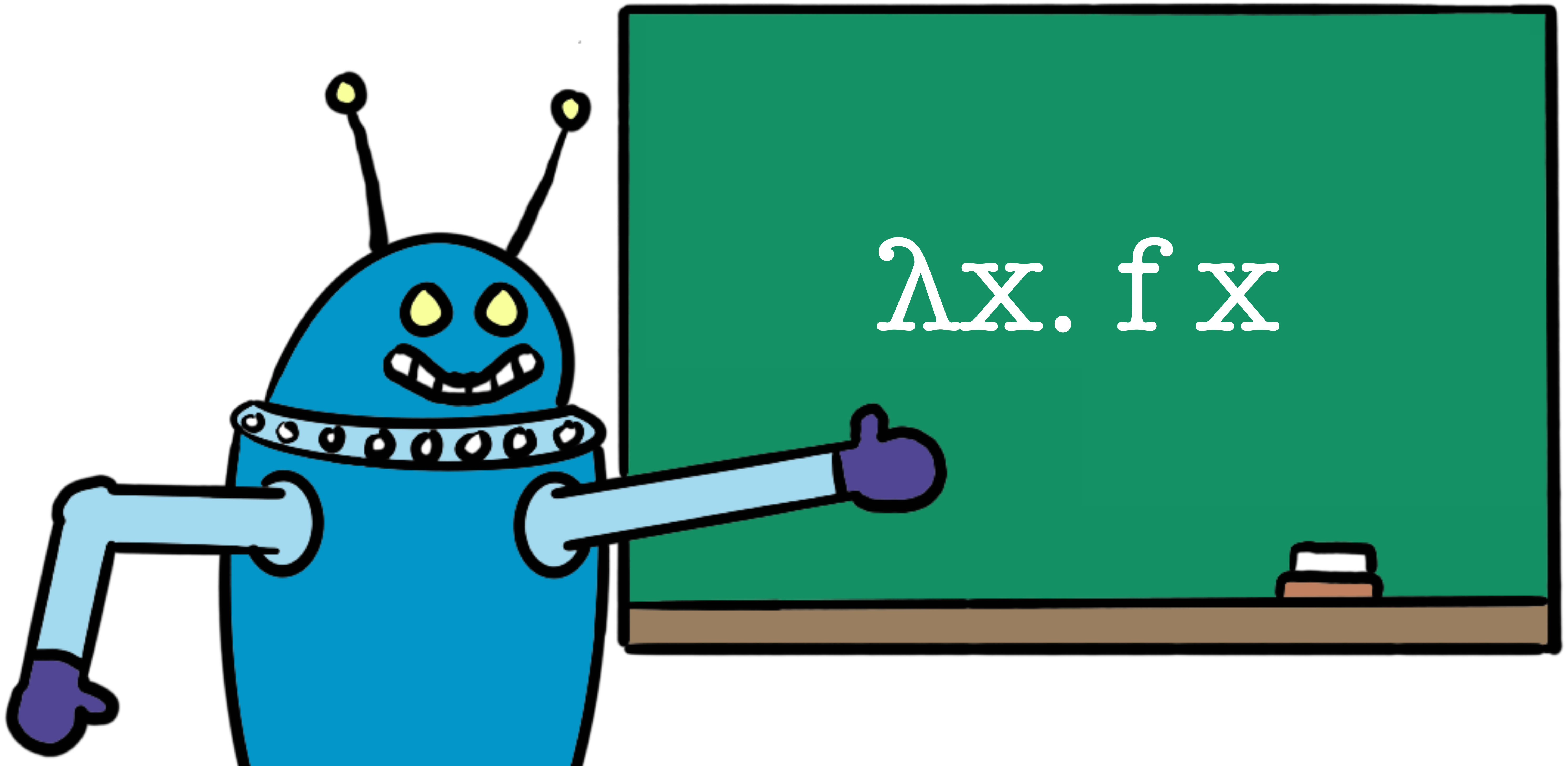
Provers & Solvers

Lecture 3: λ-Superposition

Jasmin Blanchette

LMU Munich

Partly based on slides by
Alexander Bentkamp

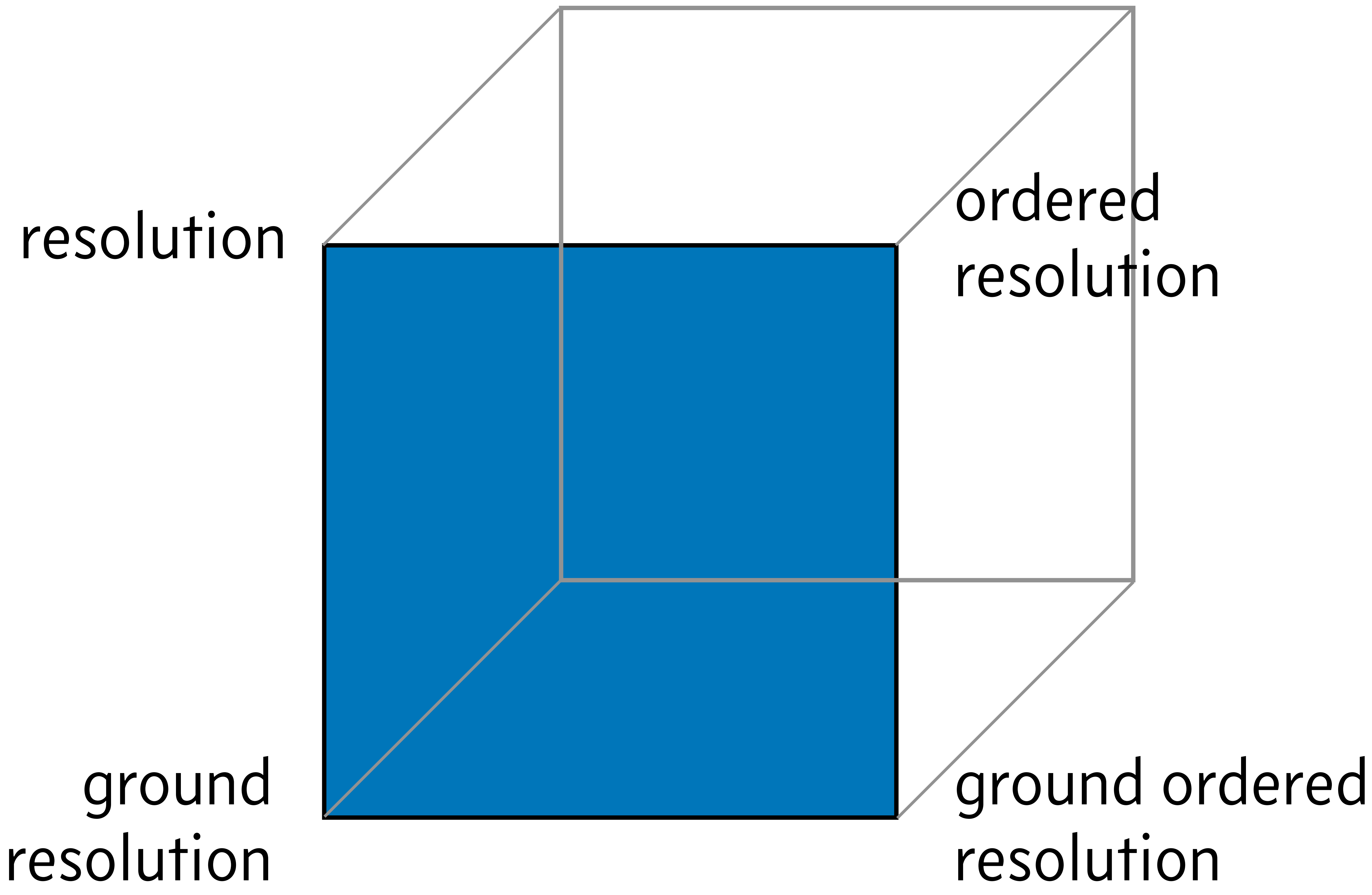


Outline of These Lectures

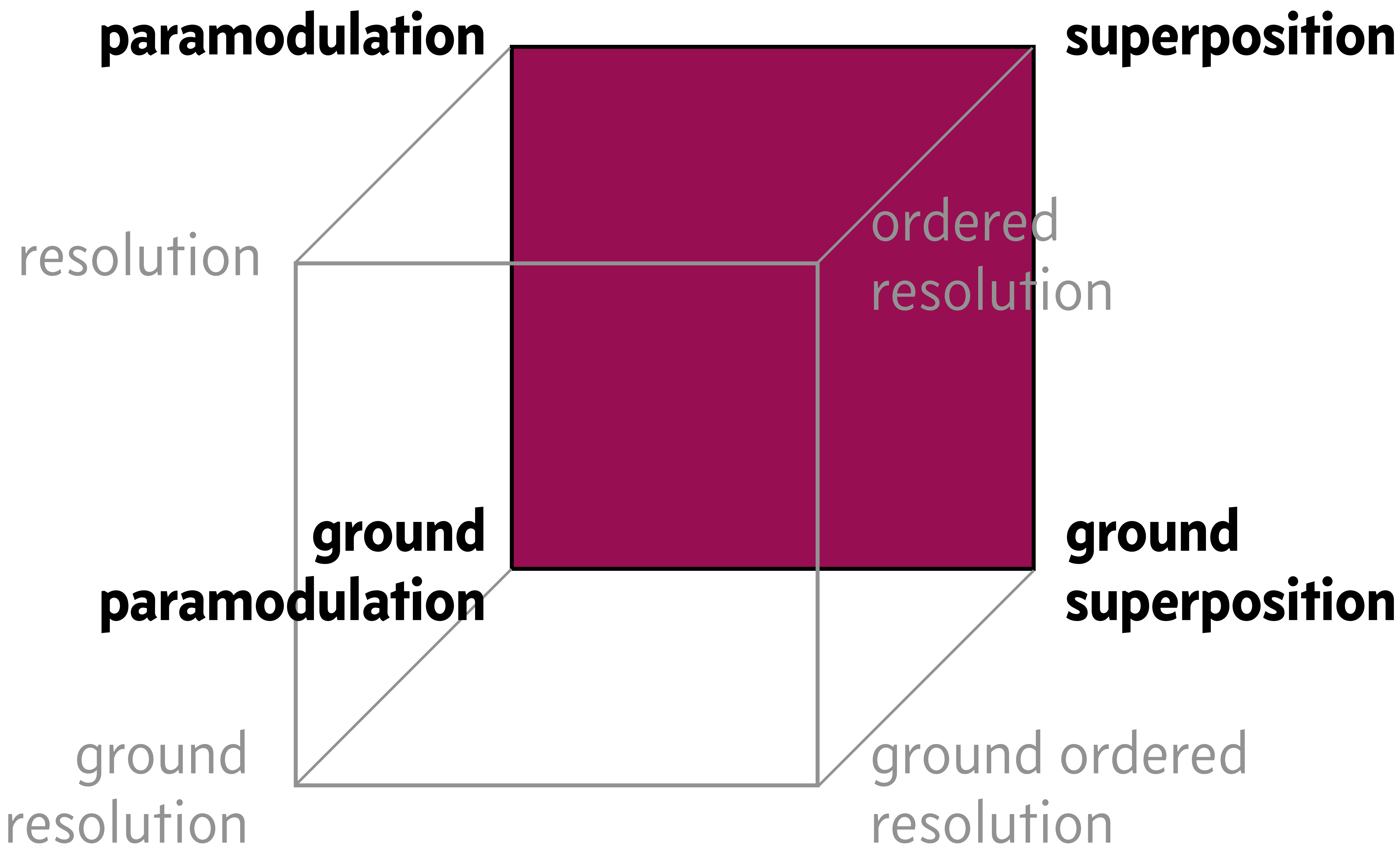
1. Resolution
2. Superposition
3. **λ-Superposition**
4. CDCL and CDCL(T)
5. AVATAR

Disclaimer

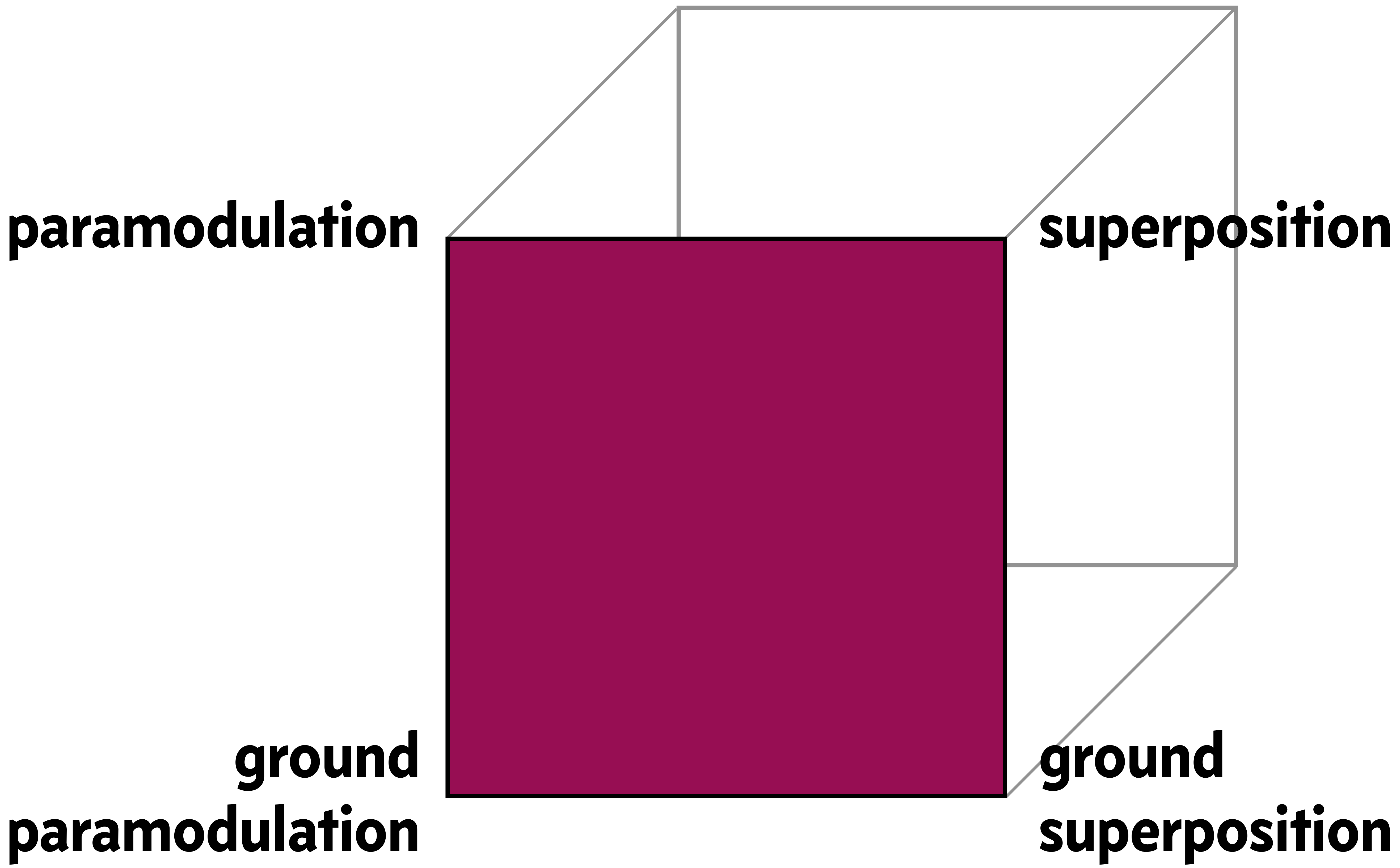
Relations between the Calculi



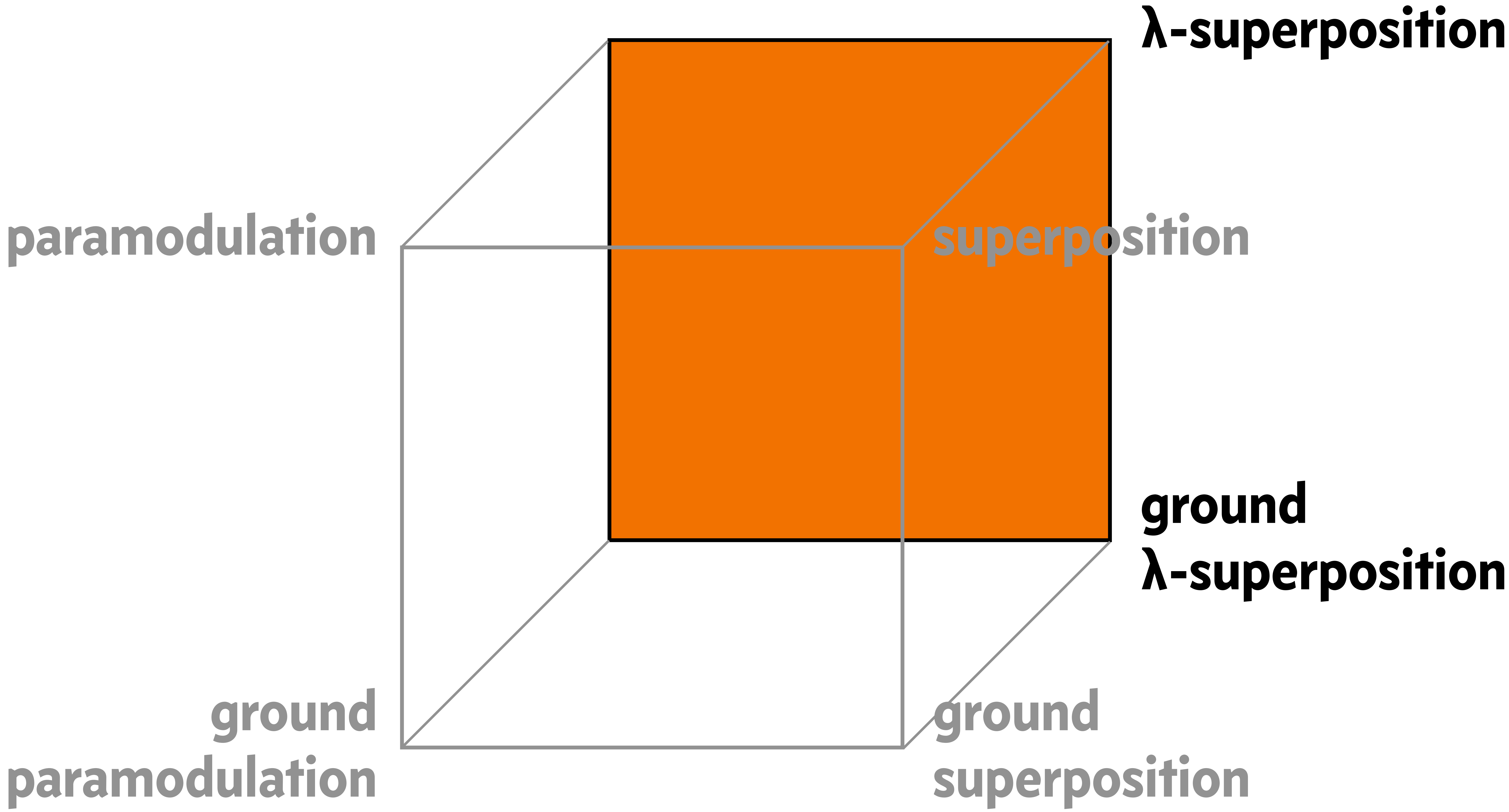
Relations between the Calculi



Relations between the Calculi



Relations between the Calculi



A Higher-Order Proof Goal

Proof assistant

show $\left(\sum_{i=1}^{n} i^2 + 2i + 1\right) = \left(\sum_{i=1}^{n} i^2\right) + \left(\sum_{i=1}^{n} 2i\right) + \left(\sum_{i=1}^{n} 1\right)$

Find Proof

A Higher-Order Proof Goal

Proof assistant

show $\left(\sum_{i=1}^{n} i^2 + 2i + 1\right) = \left(\sum_{i=1}^{n} i^2\right) + \left(\sum_{i=1}^{n} 2i\right) + \left(\sum_{i=1}^{n} 1\right)$

Find Proof

Lost in Translation

$$\left(\sum_{i=1}^{n} i^2 + 2i + 1\right) = \left(\sum_{i=1}^{n} i^2\right) + \left(\sum_{i=1}^{n} 2i\right) + \left(\sum_{i=1}^{n} 1\right)$$

Lost in Translation

$$\left(\sum_{i=1}^{n} i^2 + 2i + 1\right) = \left(\sum_{i=1}^{n} i^2\right) + \left(\sum_{i=1}^{n} 2i\right) + \left(\sum_{i=1}^{n} 1\right)$$

sum(1, n, C(B(plus, S(B(plus, C(power, 2)), app(times, 2))), 1))
= app(app(plus, app(app(plus, sum(1, n, C(power, 2))),
sum(1, n, app(times, 2)))), sum(1, n, K(1)))

Design Goal for λ-Superposition

A sound, complete, **graceful generalization**
of first-order superposition

Syntax of Higher-Order Logic

Given a **signature**, consisting of

- atomic types (e.g., bool, nat)
- symbols (e.g., 1, ·, gcd) declared with their types

Syntax of Higher-Order Logic

Given a **signature**, consisting of

- atomic types (e.g., bool, nat)
- symbols (e.g., 1, ·, gcd) declared with their types

The **terms** are defined by these rules:

- A variable x declared with type τ is a term of type τ
- A symbol f declared with type τ is a term of type τ
- If t has type τ, an abstraction $\lambda x : \sigma.\ t$ is a term of type σ → τ
- If t has type σ → τ and t' has type σ, an application $t\ t'$ is a term of type τ

Syntax of Higher-Order Logic

Given a **signature**, consisting of

- atomic types (e.g., bool, nat)
- symbols (e.g., 1, ·, gcd) declared with their types

Formulas are terms of type bool

The **terms** are defined by these rules:

- A variable x declared with type τ is a term of type τ
- A symbol f declared with type τ is a term of type τ
- If t has type τ, an abstraction $\lambda x : \sigma.\ t$ is a term of type σ → τ
- If t has type σ → τ and t' has type σ, an application $t\ t'$ is a term of type τ

Examples: Higher-Order Terms

g a x

f (g a x) b y

p f (f a) (f a b)

$\lambda x.\ \lambda y.$ g y x

1 + 2

αβη-Equivalence

Terms are considered equal up to the following three rules:

(α) $(\lambda x.\ t(x)) = (\lambda y.\ t(y))$

(β) $(\lambda x.\ t(x))\ u = t(u)$ if x does not occur free in u

(η) $(\lambda x. \mathrm{t}\ \mathrm{x}) = t$ if x does not occur free in t

Examples: αβη-Equivalence

$$g\ (\lambda x.\ f\ x\ x) = g\ (\lambda z.\ f\ z\ z)$$

$$(\lambda x.\ f\ x\ x)\ a = f\ a\ a$$

$$g\ (f\ a\ a) = g\ (\lambda y.\ f\ a\ a\ y)$$

Syntax of Clausal Higher-Order Logic

The **atoms** are defined by this rule:

- If t_1 and t_2 are terms, then $t_1 = t_2$ (viewed as an unordered pair) is an atom

The **literals** are defined by this rule:

- If A is an atom, then A and $\neg A$ are literals

The **clauses** are defined by this rule:

- If $L_1, \ldots, L_n$ are literals, then $L_1 \vee \cdots \vee L_n$ (viewed as a multiset) is a clause

We write $\bot$ if $n = 0$

Syntax of Clausal Higher-Order Logic

The **atoms** are defined by this rule:

- If t_1 and t_2 are terms, then $t_1 = t_2$ (viewed as an unordered pair) is an atom

The **literals** are defined by this rule:

- If A is an atom, then A and $\neg A$ are literals

The **clauses** are defined by this rule:

- If L_1, ..., L_n are literals, then $L_1 \lor \cdots \lor L_n$ (viewed as a multiset) is a clause

We write $\bot$ if $n = 0$

- $\top$, $\land$, $\Rightarrow$, $\forall$, $\exists$ can appear in terms
- Variables are understood as "for all"

Example: Summations

Goal:

$$\left(\sum_{i=1}^{n} i^2 + 2i + 1\right) = \left(\sum_{i=1}^{n} i^2\right) + \left(\sum_{i=1}^{n} 2i\right) + \left(\sum_{i=1}^{n} 1\right)$$

Distributivity lemma:

$$\forall f, g\,.\ \sum_{i=1}^{n} \left(f\ i + g\ i\right) = \sum_{i=1}^{n} f\ i + \sum_{i=1}^{n} g\ i$$

Proof idea:

$$\left(\sum_{i=1}^{n} i^2 + 2i + 1\right) = \sum_{i=1}^{n} \left(i^2 + 2i\right) + \sum_{i=1}^{n} 1 = \left(\sum_{i=1}^{n} i^2\right) + \left(\sum_{i=1}^{n} 2i\right) + \left(\sum_{i=1}^{n} 1\right)$$

Example: Summations

Goal:

$$\left(\sum_{i=1}^{n} i^2 + 2i + 1\right) = \left(\sum_{i=1}^{n} i^2\right) + \left(\sum_{i=1}^{n} 2i\right) + \left(\sum_{i=1}^{n} 1\right)$$

sum 1 n (λ*i*. *i*² + 2*i* + 1) = sum 1 n (λ*i*. *i*²) + sum 1 n (λ*i*. 2*i*) + sum 1 n (λ*i*. 1)

Distributivity lemma:

$$\forall f, g\,.\ \sum_{i=1}^{n} (f\,i + g\,i) = \sum_{i=1}^{n} f\,i + \sum_{i=1}^{n} g\,i$$

sum 1 n (λ*i*. *f i* + *g i*) = sum 1 n (λ*i*. *f i*) + sum 1 n (λ*i*. *g i*)

Proof idea:

$$\left(\sum_{i=1}^{n} i^2 + 2i + 1\right) = \sum_{i=1}^{n} \left(i^2 + 2i\right) + \sum_{i=1}^{n} 1 = \left(\sum_{i=1}^{n} i^2\right) + \left(\sum_{i=1}^{n} 2i\right) + \left(\sum_{i=1}^{n} 1\right)$$

Example: Summations

$$\text{sum } 1 \text{ n } (\lambda i.\, f\, i + g\, i)$$
$$= \text{sum } 1 \text{ n } (\lambda i.\, f\, i) + \text{sum } 1 \text{ n } (\lambda i.\, g\, i)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$
$$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2) + \text{sum } 1 \text{ n } (\lambda i.\, 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

Example: Summations

sum 1 n ($\lambda i.\, f\, i + g\, i$)
= sum 1 n ($\lambda i.\, f\, i$) + sum 1 n ($\lambda i.\, g\, i$)

sum 1 n ($\lambda i.\, i^2 + 2i + 1$)
≠ sum 1 n ($\lambda i.\, i^2$) + sum 1 n ($\lambda i.\, 2i$) + sum 1 n ($\lambda i.\, 1$)

sum 1 n ($\lambda i.\, i^2 + 2i + 1$)
≠ sum 1 n ($\lambda i.\, i^2 + 2i$) + sum 1 n ($\lambda i.\, 1$)

Example: Summations

$$\text{sum } 1 \text{ n } (\lambda i.\, f\, i + g\, i)$$
$$= \text{sum } 1 \text{ n } (\lambda i.\, f\, i) + \text{sum } 1 \text{ n } (\lambda i.\, g\, i)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$
$$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2) + \text{sum } 1 \text{ n } (\lambda i.\, 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$
$$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

Example: Summations

$\text{sum } 1 \text{ n } (\lambda i.\, f\, i + g\, i)$
$= \text{sum } 1 \text{ n } (\lambda i.\, f\, i) + \text{sum } 1 \text{ n } (\lambda i.\, g\, i)$

$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$
$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2) + \text{sum } 1 \text{ n } (\lambda i.\, 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$

$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$
$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$

Example: Summations

$$\text{sum } 1 \text{ n } (\lambda i.\, f\, i + g\, i) = \text{sum } 1 \text{ n } (\lambda i.\, f\, i) + \text{sum } 1 \text{ n } (\lambda i.\, g\, i)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1) \neq \text{sum } 1 \text{ n } (\lambda i.\, i^2) + \text{sum } 1 \text{ n } (\lambda i.\, 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1) \neq \text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

Example: Summations

$$\text{sum } 1 \text{ n } (\lambda i.\, f\, i + g\, i)$$
$$= \text{sum } 1 \text{ n } (\lambda i.\, f\, i) + \text{sum } 1 \text{ n } (\lambda i.\, g\, i)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$
$$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2) + \text{sum } 1 \text{ n } (\lambda i.\, 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$
$$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$
$$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$

Example: Summations

$$\text{sum } 1\ n\ (\lambda i.\ f\,i + g\,i)$$
$$= \text{sum } 1\ n\ (\lambda i.\ f\,i) + \text{sum } 1\ n\ (\lambda i.\ g\,i)$$

$$\text{sum } 1\ n\ (\lambda i.\ i^2 + 2i + 1)$$
$$\neq \text{sum } 1\ n\ (\lambda i.\ i^2) + \text{sum } 1\ n\ (\lambda i.\ 2i) + \text{sum } 1\ n\ (\lambda i.\ 1)$$

$$\text{sum } 1\ n\ (\lambda i.\ i^2 + 2i + 1)$$
$$\neq \text{sum } 1\ n\ (\lambda i.\ i^2 + 2i) + \text{sum } 1\ n\ (\lambda i.\ 1)$$

$$\text{sum } 1\ n\ (\lambda i.\ i^2 + 2i + 1)$$
$$\neq \text{sum } 1\ n\ (\lambda i.\ i^2 + 2i + 1)$$

Example: Summations

$$\text{sum } 1\ n\ (\lambda i.\ f\,i + g\,i) = \text{sum } 1\ n\ (\lambda i.\ f\,i) + \text{sum } 1\ n\ (\lambda i.\ g\,i)$$

$$\text{sum } 1\ n\ (\lambda i.\ i^2 + 2i + 1) \neq \text{sum } 1\ n\ (\lambda i.\ i^2) + \text{sum } 1\ n\ (\lambda i.\ 2i) + \text{sum } 1\ n\ (\lambda i.\ 1)$$

$$\text{sum } 1\ n\ (\lambda i.\ i^2 + 2i + 1) \neq \text{sum } 1\ n\ (\lambda i.\ i^2 + 2i) + \text{sum } 1\ n\ (\lambda i.\ 1)$$

$$\text{sum } 1\ n\ (\lambda i.\ i^2 + 2i + 1) \neq \text{sum } 1\ n\ (\lambda i.\ i^2 + 2i + 1)$$

Example: Summations

$$\begin{aligned} &\text{sum } 1 \text{ n } (\lambda i.\, f\, i + g\, i) \\ =\ &\text{sum } 1 \text{ n } (\lambda i.\, f\, i) + \text{sum } 1 \text{ n } (\lambda i.\, g\, i) \end{aligned}$$

$$\begin{aligned} &\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1) \\ \neq\ &\text{sum } 1 \text{ n } (\lambda i.\, i^2) + \text{sum } 1 \text{ n } (\lambda i.\, 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1) \end{aligned}$$

$$\begin{aligned} &\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1) \\ \neq\ &\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1) \end{aligned}$$

$$\begin{aligned} &\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1) \\ \neq\ &\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1) \end{aligned}$$

Example: Summations

$$\text{sum } 1 \text{ n } (\lambda i.\, f\,i + g\,i)$$
$$= \text{sum } 1 \text{ n } (\lambda i.\, f\,i) + \text{sum } 1 \text{ n } (\lambda i.\, g\,i)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$
$$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2) + \text{sum } 1 \text{ n } (\lambda i.\, 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$
$$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$
$$\neq \text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$

$\bot$

Example: Summations

$$\begin{aligned} &\mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ f\, i + g\, i) \\ = &\ \mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ f\, i) + \mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ g\, i) \end{aligned}$$

$$\begin{aligned} &\mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ i^2 + 2i + 1) \\ \neq &\ \mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ i^2) + \mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ 2i) + \mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ 1) \end{aligned}$$

$$\begin{aligned} &\mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ i^2 + 2i + 1) \\ \neq &\ \mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ i^2 + 2i) + \mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ 1) \end{aligned}$$

$$\begin{aligned} &\mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ i^2 + 2i + 1) \\ \neq &\ \mathrm{sum}\ 1\ \mathrm{n}\ (\lambda i.\ i^2 + 2i + 1) \end{aligned}$$

$\bot$

Example: Summations

$$\text{sum } 1 \text{ n } (\lambda i.\, f\, i + g\, i) = \text{sum } 1 \text{ n } (\lambda i.\, f\, i) + \text{sum } 1 \text{ n } (\lambda i.\, g\, i)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1) \neq \text{sum } 1 \text{ n } (\lambda i.\, i^2) + \text{sum } 1 \text{ n } (\lambda i.\, 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1) \neq \text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i) + \text{sum } 1 \text{ n } (\lambda i.\, 1)$$

$$\text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1) \neq \text{sum } 1 \text{ n } (\lambda i.\, i^2 + 2i + 1)$$

$\bot$

Design Challenges for λ-Superposition

1. The term order
2. Unification
3. Booleans

Challenge 1: The Term Order

In first-order superposition:

- A subterm must be smaller than the whole term:
  e.g. $c < g(c) < f(g(c))$
- Putting two terms in the same context should preserve the orientation:
  e.g. if $b > a$ then $f(b, c) > f(a, c)$

Challenge 1: The Term Order

In first-order superposition:

- A subterm must be smaller than the whole term:
  e.g. $c < g(c) < f(g(c))$
- Putting two terms in the same context should preserve the orientation:
  e.g. if $b > a$ then $f(b, c) > f(a, c)$

Issue in higher-order:

$$(\lambda y.\ f\ (\lambda z.\ z)) \quad > \quad (\lambda z.\ z)$$

Challenge 1: The Term Order

In first-order superposition:

- A subterm must be smaller than the whole term:
  e.g. $c < g(c) < f(g(c))$
- Putting two terms in the same context should preserve the orientation:
  e.g. if $b > a$ then $f(b, c) > f(a, c)$

Issue in higher-order:

$$(\lambda y.\ f\ (\lambda z.\ z)) \quad > \quad (\lambda z.\ z)$$

$$[\,]\ (g\ (f\ (\lambda z.\ z))) \qquad [\,]\ (g\ (f\ (\lambda z.\ z)))$$

Challenge 1: The Term Order

In first-order superposition:

- A subterm must be smaller than the whole term:
  e.g. $\mathrm{c} < \mathrm{g(c)} < \mathrm{f(g(c))}$
- Putting two terms in the same context should preserve the orientation:
  e.g. if $\mathrm{b} > \mathrm{a}$ then $\mathrm{f(b, c)} > \mathrm{f(a, c)}$

Issue in higher-order:

$(\lambda y.\ \mathrm{f}\ (\lambda z.\ z)) \quad > \quad (\lambda z.\ z)$

$[\,]\ (\mathrm{g}\ (\mathrm{f}\ (\lambda z.\ z))) \qquad [\,]\ (\mathrm{g}\ (\mathrm{f}\ (\lambda z.\ z)))$

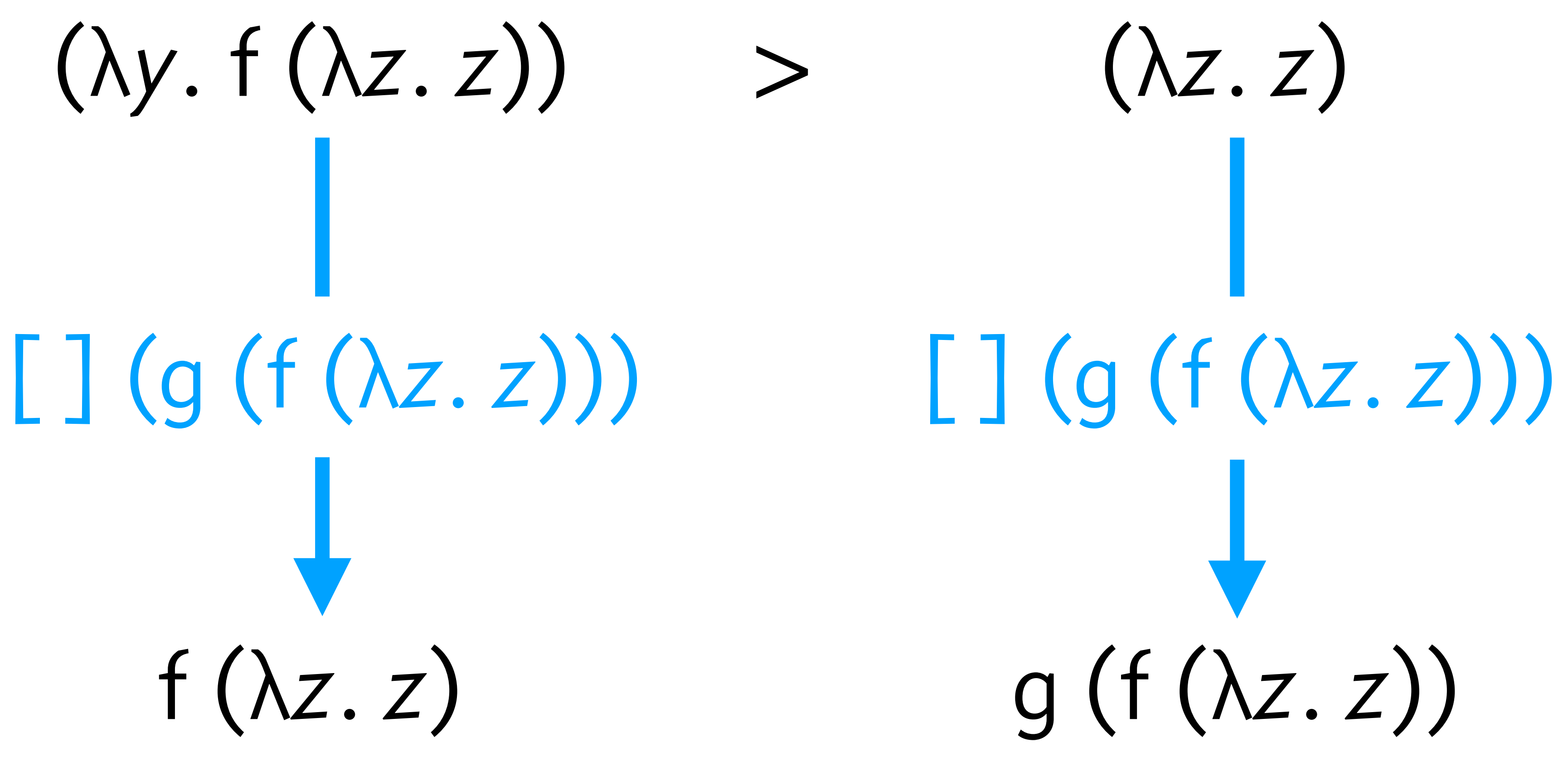
$\mathrm{f}\ (\lambda z.\ z)$

Challenge 1: The Term Order

In first-order superposition:

- A subterm must be smaller than the whole term:
  e.g. $c < g(c) < f(g(c))$
- Putting two terms in the same context should preserve the orientation:
  e.g. if $b > a$ then $f(b, c) > f(a, c)$

Issue in higher-order:



Challenge 1: The Term Order

In first-order superposition:

- A subterm must be smaller than the whole term:
  e.g. $\mathrm{c} < \mathrm{g(c)} < \mathrm{f(g(c))}$
- Putting two terms in the same context should preserve the orientation:
  e.g. if $\mathrm{b} > \mathrm{a}$ then $\mathrm{f(b, c)} > \mathrm{f(a, c)}$

Issue in higher-order:

$$(\lambda y.\, \mathrm{f}\,(\lambda z.\, z)) \quad > \quad (\lambda z.\, z)$$

$$[\,]\,(\mathrm{g}\,(\mathrm{f}\,(\lambda z.\, z))) \qquad [\,]\,(\mathrm{g}\,(\mathrm{f}\,(\lambda z.\, z)))$$

$$\mathrm{f}\,(\lambda z.\, z) \quad < \quad \mathrm{g}\,(\mathrm{f}\,(\lambda z.\, z))$$

Challenge 1: The Term Order

In first-order superposition:

- A subterm must be smaller than the whole term:
  e.g. $c < g(c) < f(g(c))$
- Putting two terms in the same context should preserve the orientation:
  e.g. if $b > a$ then $f(b, c) > f(a, c)$

Issue in higher-order:

$$(\lambda y.\ \mathrm{f}\ (\lambda z.\ z)) \quad > \quad (\lambda z.\ z)$$

$$\downarrow\ [\,]\ (\mathrm{g}\ (\mathrm{f}\ (\lambda z.\ z))) \qquad\qquad \downarrow\ [\,]\ (\mathrm{g}\ (\mathrm{f}\ (\lambda z.\ z)))$$

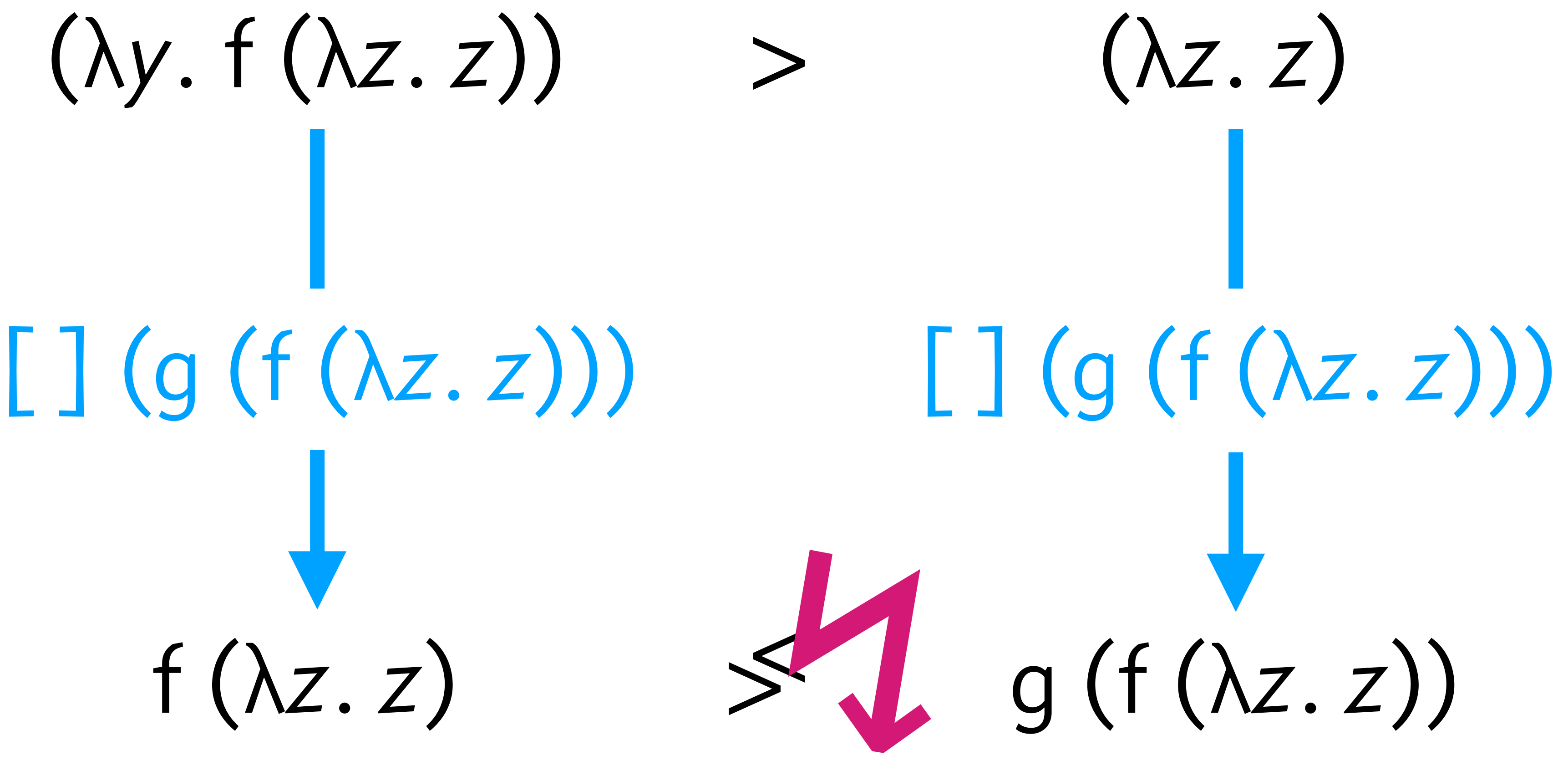
$$\mathrm{f}\ (\lambda z.\ z) \quad \not> \quad \mathrm{g}\ (\mathrm{f}\ (\lambda z.\ z))$$

Challenge 1: The Term Order

In first-order superposition:

- A subterm must be smaller than the whole term:
  e.g. $c < g(c) < f(g(c))$
- Putting two terms in the same context should preserve the orientation:
  e.g. if $b > a$ then $f(b, c) > f(a, c)$

Issue in higher-order:



Challenge 1: The Term Order

Solution:

- Weaken the second requirement: only **good contexts** must preserve the orientation
- The main superposition rule acts only on **good contexts**
- Compensate with an **extra calculus rule**

Challenge 1: The Term Order

Solution:

- Weaken the second requirement: only **good contexts** must preserve the orientation
- The main superposition rule acts only on **good contexts**
- Compensate with an **extra calculus rule**

Extra rule:

If the clause

$$C \vee t = t'$$

is contained in $\mathbf{F}$
and t, t' are functions,
then add the clause

$$C \vee t\ x = t'\ x$$

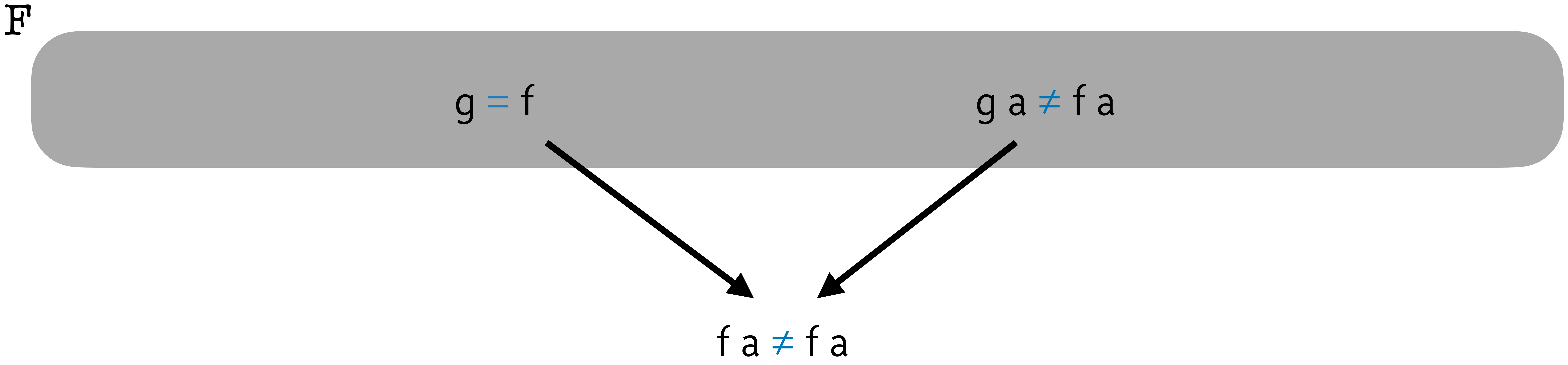
to $\mathbf{F}$

Example: The Term Order

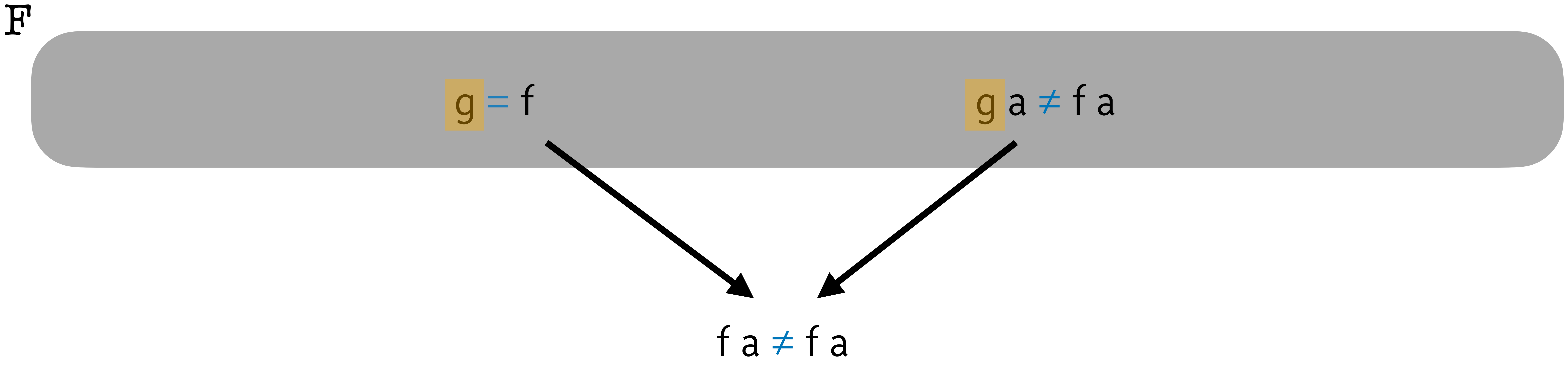
F

$g = f$ $\quad$ $g\ a \neq f\ a$

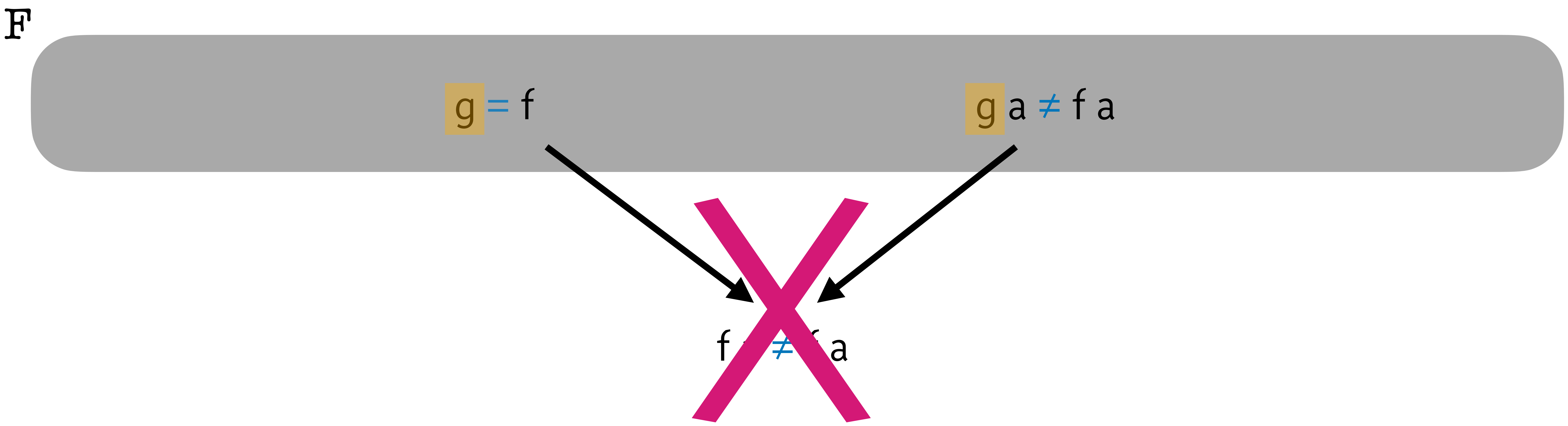
Example: The Term Order



Example: The Term Order



Example: The Term Order



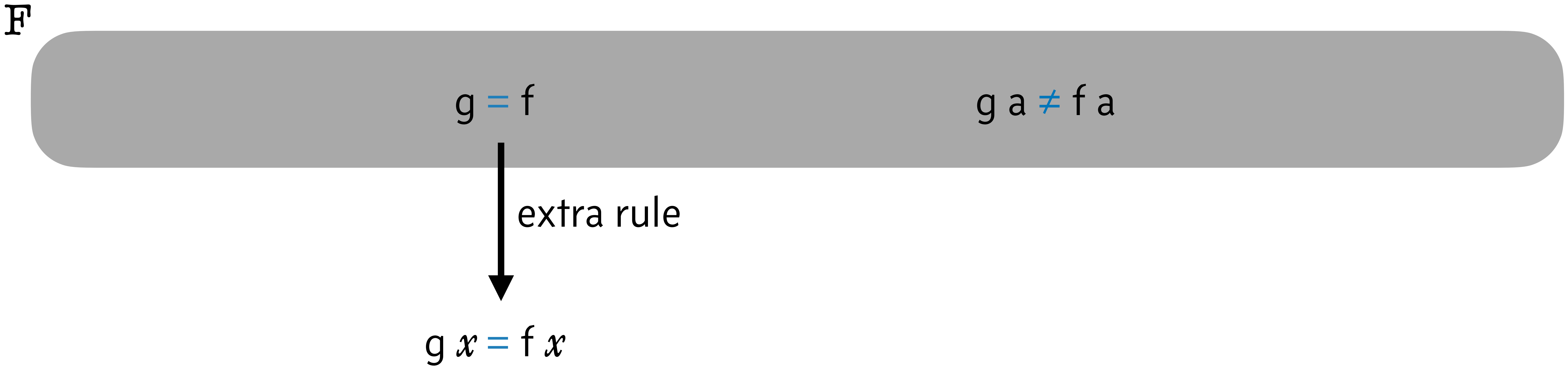
disallowed because [ ] a is not a good context

Example: The Term Order

F

$g = f \qquad g\ a \neq f\ a$

Example: The Term Order



Example: The Term Order

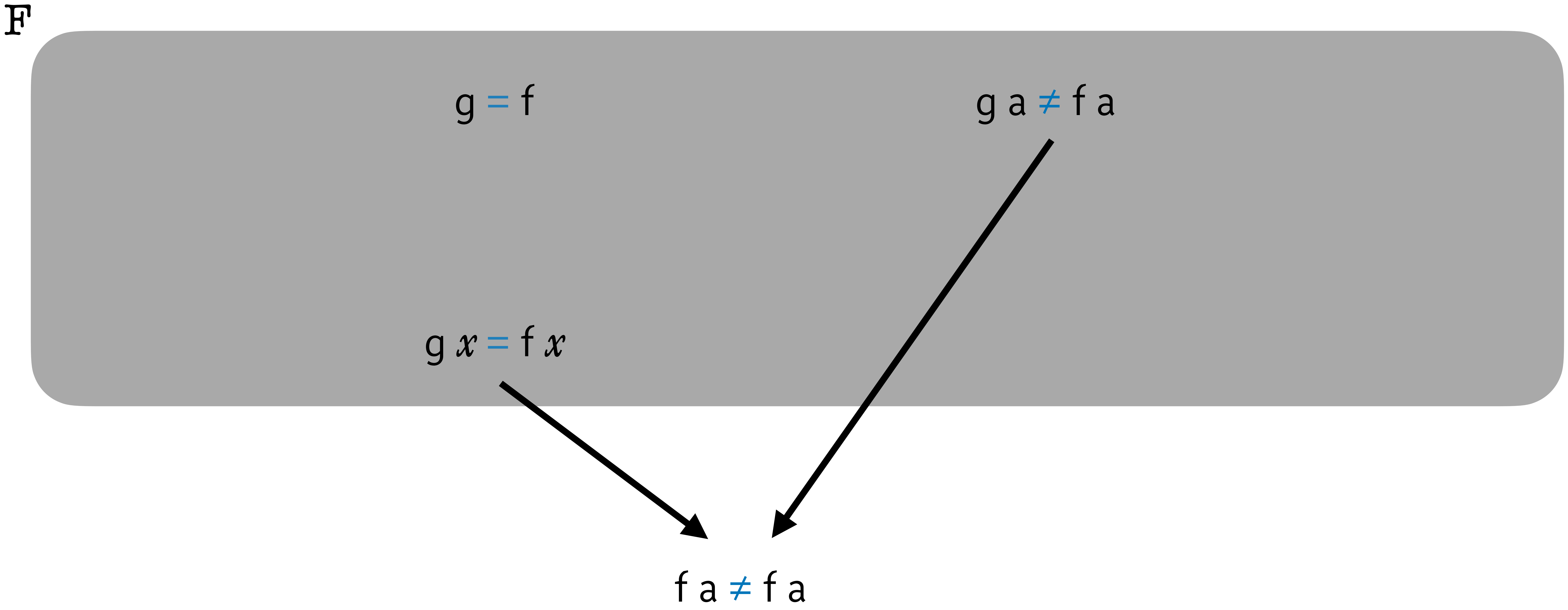
$\mathtt{F}$

$\mathsf{g} = \mathsf{f}$

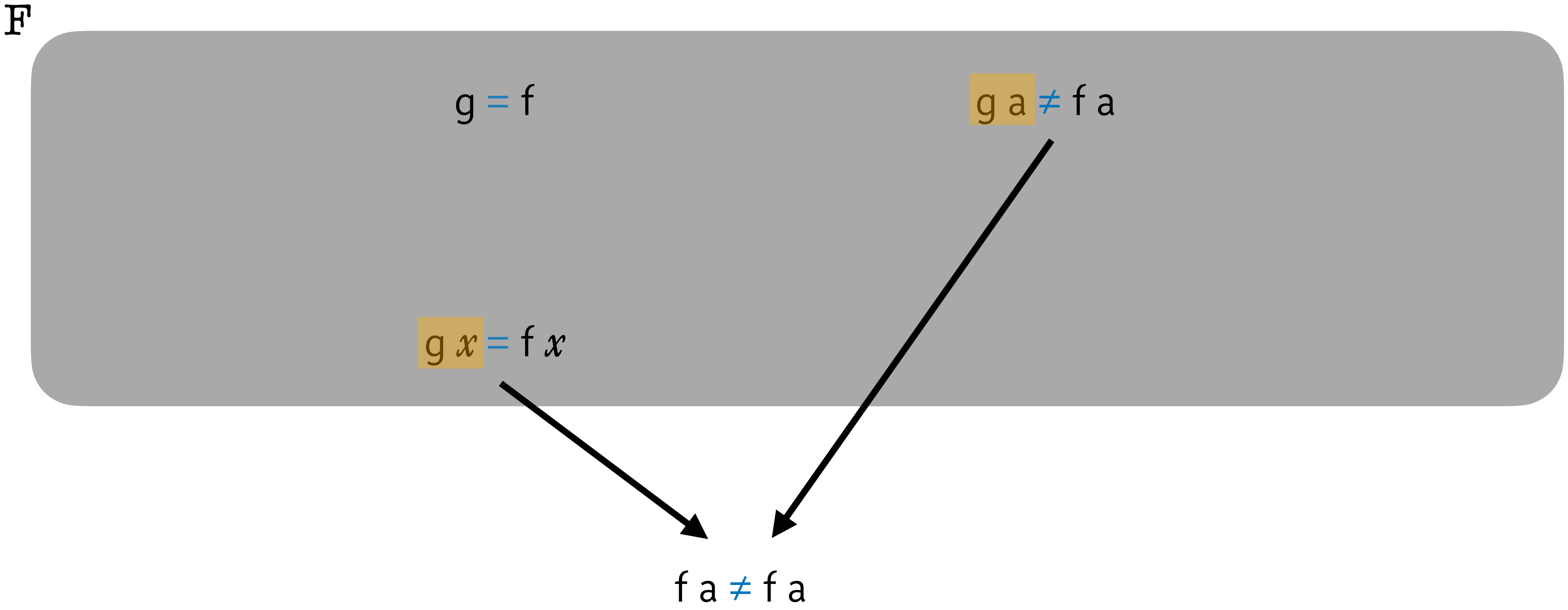
$\mathsf{g}\ \mathsf{a} \neq \mathsf{f}\ \mathsf{a}$

$\mathsf{g}\ x = \mathsf{f}\ x$

Example: The Term Order



Example: The Term Order



Example: The Term Order

$\mathtt{F}$

$\mathsf{g} = \mathsf{f}$

$\mathsf{g}\ \mathsf{a} \neq \mathsf{f}\ \mathsf{a}$

$\mathsf{g}\ x = \mathsf{f}\ x$

$\mathsf{f}\ \mathsf{a} \neq \mathsf{f}\ \mathsf{a}$

Challenge 1bis: The Redundancy Criterion

In first-order logic:

- A ground clause C is redundant w.r.t. ground $\mathtt{F}$ if $C_1, \ldots, C_n \in \mathtt{F}$ are all smaller than C and they together entail C

Challenge 1bis: The Redundancy Criterion

In first-order logic:

- A ground clause C is redundant w.r.t. ground $\mathtt{F}$ if $C_1, \ldots, C_n \in \mathtt{F}$ are all smaller than C and they together entail C

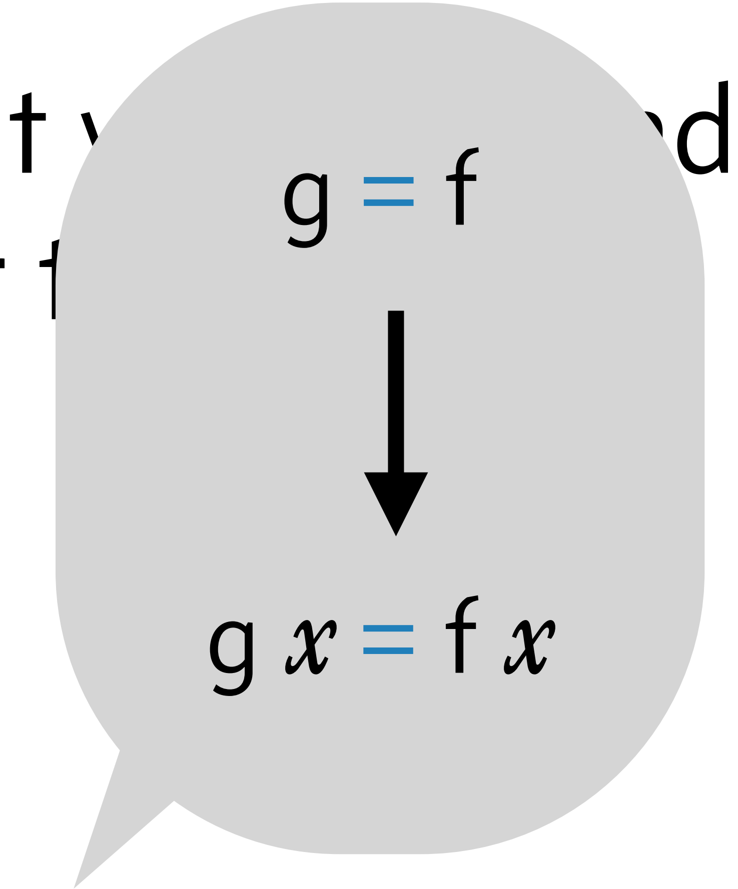
Higher-order issue:

- The conclusion of the extra inference rule would be redundant w.r.t. the premise

Challenge 1bis: The Redundancy Criterion

In first-order logic:

- A ground clause C is redundant ᐧ d $\mathbf{F}$ if $C_1, \ldots, C_n \in \mathbf{F}$ are all smaller ᐧ they together entail C



Higher-order issue:

- The conclusion of the extra inference rule would be redundant w.r.t. the premise

Challenge 1bis: The Redundancy Criterion

Solution:
- Use weaker notion of entailment

Examples:
- b = a makes f b = f a redundant
- g = f **does not** make g a = f a redundant

Challenge 1bis: The Redundancy Criterion

Solution:

- Use weaker notion of entailment

Examples:

- b = a makes f b = f a redundant
- $g_0 = f_0$ **does not** make $g_1 a_0 = f_1 a_0$ redundant

Nonground Redundancy Criterion

A nonground clause C is redundant w.r.t. nonground $\mathtt{F}$
if each clause in ground(C) is redundant w.r.t. ground($\mathtt{F}$)

Nonground Redundancy Criterion

A nonground clause C is redundant w.r.t. nonground $\mathtt{F}$
if each clause in ground(C) is redundant w.r.t. ground($\mathtt{F}$)

Examples:

- b x = a makes f (b x) = f a redundant
- g = f **does not** make g x = f x redundant

Example of a Redundancy Rule

Argument pruning:

If a clause of the form

$$C(y)$$

is contained in $\mathtt{F}$,
where one of y's arguments is computable from the others,
then **remove** the argument

Example of a Redundancy Rule

Argument pruning:

If a clause of the form

$C(y)$

is contained in $\mathtt{F}$,
where one of y's arguments is computable from the others,
then **remove** the argument

$y\ \mathrm{b}\ \mathrm{b} \neq y\ \mathrm{a}\ \mathrm{a}$

Example of a Redundancy Rule

Argument pruning:

If a clause of the form

$C(y)$

is contained in $\mathtt{F}$,
where one of y's arguments is computable from the others,
then **remove** the argument

~~$y\ b\ b \neq y\ a\ a$~~ $y\ b \neq y\ a$

Example of a Redundancy Rule

Argument pruning:

If a clause of the form

$C(y)$

is contained in $\mathtt{F}$,
where one of y's arguments is computable from the others,
then **remove** the argument

~~y b b ≠ y a a~~ y b ≠ y a

y a c ≠ y a b

Example of a Redundancy Rule

Argument pruning:

If a clause of the form

$C(y)$

is contained in $\mathtt{F}$,
where one of y's arguments is computable from the others,
then **remove** the argument

~~$y\ \mathrm{b}\ \mathrm{b} \neq y\ \mathrm{a}\ \mathrm{a}$~~ $y\ \mathrm{b} \neq y\ \mathrm{a}$

~~$y\ \mathrm{a}\ \mathrm{c} \neq y\ \mathrm{a}\ \mathrm{b}$~~ $y\ \mathrm{c} \neq y\ \mathrm{b}$

Example of a Redundancy Rule

Argument pruning:

If a clause of the form

$C(y)$

is contained in $\mathtt{F}$,
where one of y's arguments is computable from the others,
then **remove** the argument

~~y b b $\neq$ y a a~~ y b $\neq$ y a

~~y a c $\neq$ y a b~~ y c $\neq$ y b

y a c $\neq$ z b d

Example of a Redundancy Rule

Argument pruning:

If a clause of the form

$C(y)$

is contained in $\mathtt{F}$,
where one of y's arguments is computable from the others,
then **remove** the argument

~~$y\ \mathrm{b}\ \mathrm{b} \neq y\ \mathrm{a}\ \mathrm{a}$~~ $y\ \mathrm{b} \neq y\ \mathrm{a}$

~~$y\ \mathrm{a}\ \mathrm{c} \neq y\ \mathrm{a}\ \mathrm{b}$~~ $y\ \mathrm{c} \neq y\ \mathrm{b}$

~~$y\ \mathrm{a}\ \mathrm{c} \neq z\ \mathrm{b}\ \mathrm{d}$~~ $y \neq z$

Challenge 2: Unification

In first-order logic:

- Most general unifiers always exist and can be computed
  e.g. unifying f(a, y) with f(x, b) yields the mgu $\{x \mapsto \mathrm{a},\ y \mapsto \mathrm{b}\}$

Challenge 2: Unification

In first-order logic:

- Most general unifiers always exist and can be computed
  e.g. unifying f(a, y) with f(x, b) yields the mgu $\{x \mapsto \mathrm{a},\ y \mapsto \mathrm{b}\}$

In higher-order logic:

A. Most general unifiers **do not** always exist
   e.g. unifying f (y a) and y (f a) yields infinitely many unifiers
   $\{y \mapsto \lambda x.\ x\}$ $\{y \mapsto \lambda x.\ \mathrm{f}\ x\}$ $\{y \mapsto \lambda x.\ \mathrm{f}\ (\mathrm{f}\ x)\}$ …

B. Unification is undecidable

C. Applied variables can hide positions where inferences should be made

Challenge 2: Unification

Solutions:

A. Use a (possibly infinite) sequence of unifiers instead of mgu
B. Interweave unification and inferences
C. Introduce a special "fluid" version of the main inference

Example: Fluid Inference

$\mathcal{F}$

$f\ a = c \qquad h\ (y\ b)\ (y\ a) \neq h\ (g\ (f\ b))\ (g\ c)$

Example: Fluid Inference

$\mathtt{F}$

$f\ a = c \qquad h\ (y\ b)\ (y\ a) \neq h\ (g\ (f\ b))\ (g\ c)$

$\{y \mapsto \lambda x.\ g\ (f\ x)\}$

$h\ (g\ (f\ b))\ (g\ (f\ a)) \neq h\ (g\ (f\ b))\ (g\ c)$

Example: Fluid Inference

F

$f\ a = c$ $\qquad$ $h\ (y\ b)\ (y\ a) \neq h\ (g\ (f\ b))\ (g\ c)$

$\{y \mapsto \lambda x.\ g\ (f\ x)\}$

$h\ (g\ (f\ b))\ (g\ (f\ a)) \neq h\ (g\ (f\ b))\ (g\ c)$

$h\ (g\ (f\ b))\ (g\ c) \neq h\ (g\ (f\ b))\ (g\ c)$

Example: Fluid Inference

$\mathcal{F}$

$$\mathsf{f\ a} = \mathsf{c}$$

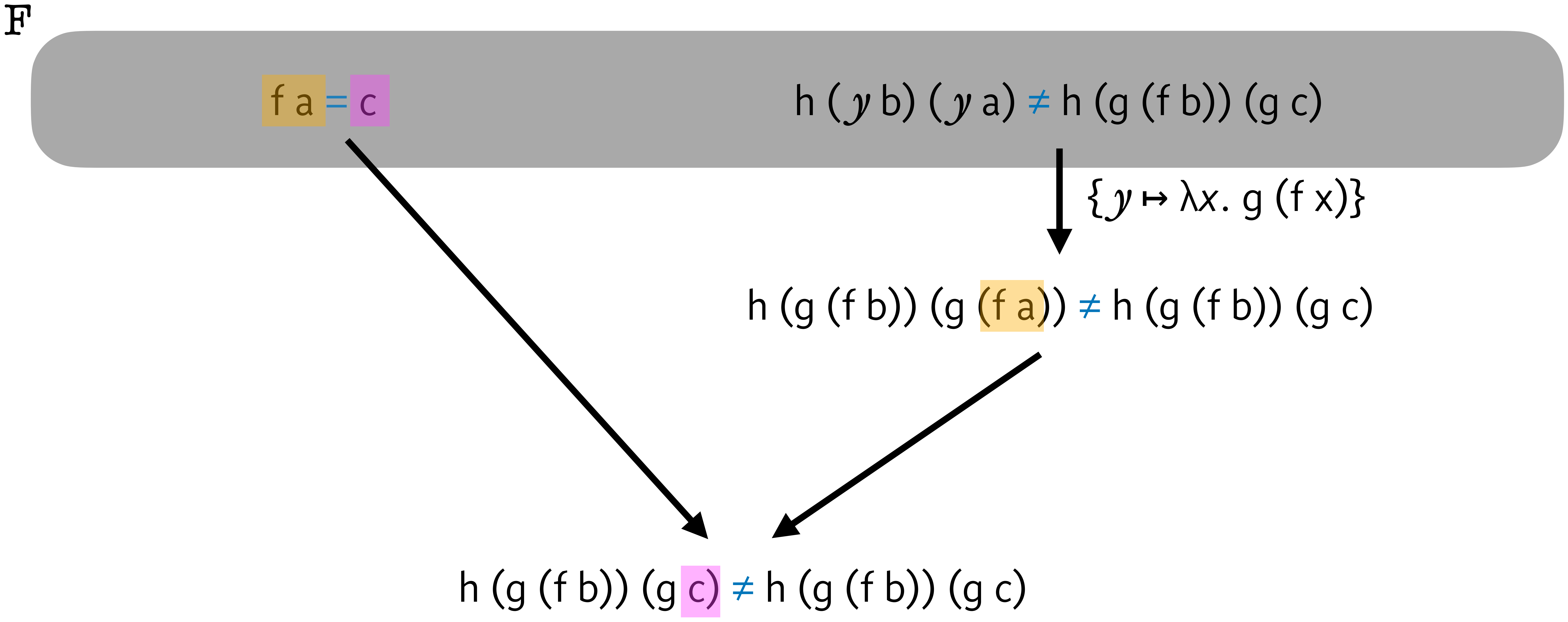
$$\mathsf{h}\ (y\ \mathsf{b})\ (y\ \mathsf{a}) \neq \mathsf{h}\ (\mathsf{g}\ (\mathsf{f\ b}))\ (\mathsf{g\ c})$$

$$\{y \mapsto \lambda x.\ \mathsf{g}\ (\mathsf{f}\ x)\}$$

$$\mathsf{h}\ (\mathsf{g}\ (\mathsf{f\ b}))\ (\mathsf{g}\ (\mathsf{f\ a})) \neq \mathsf{h}\ (\mathsf{g}\ (\mathsf{f\ b}))\ (\mathsf{g\ c})$$

$$\mathsf{h}\ (\mathsf{g}\ (\mathsf{f\ b}))\ (\mathsf{g\ c}) \neq \mathsf{h}\ (\mathsf{g}\ (\mathsf{f\ b}))\ (\mathsf{g\ c})$$

Example: Fluid Inference



Example: Fluid Inference

$\mathtt{F}$

$z\,(\mathsf{f}\ \mathsf{a}) = z\ \mathsf{c}$

$\mathsf{h}\ (y\ \mathsf{b})\ (y\ \mathsf{a}) \neq \mathsf{h}\ (\mathsf{g}\ (\mathsf{f}\ \mathsf{b}))\ (\mathsf{g}\ \mathsf{c})$

Example: Fluid Inference

$\mathbf{F}$

$z\ (\mathsf{f\ a}) = z\ \mathsf{c}$ $\qquad$ $\mathsf{h}\ (y\ \mathsf{b})\ (y\ \mathsf{a}) \neq \mathsf{h}\ (\mathsf{g}\ (\mathsf{f\ b}))\ (\mathsf{g\ c})$

z (f a) is unified
with y a

Example: Fluid Inference

F

$z\ (\mathsf{f\ a}) = z\ \mathsf{c}$ $\qquad$ $\mathsf{h}\ (y\ \mathsf{b})\ (y\ \mathsf{a}) \neq \mathsf{h}\ (\mathsf{g}\ (\mathsf{f\ b}))\ (\mathsf{g\ c})$

$z\ (\mathsf{f\ a})$ is unified
with $y\ \mathsf{a}$
$\{z \mapsto \mathsf{g},\ y \mapsto \lambda x.\ \mathsf{g}\ (\mathsf{f}\ x)\}$

$\mathsf{h}\ (\mathsf{g}\ (\mathsf{f\ b}))\ (\mathsf{g\ c}) \neq \mathsf{h}\ (\mathsf{g}\ (\mathsf{f\ b}))\ (\mathsf{g\ c})$

Challenge 3: Boolean Expressions

In first-order logic:

- Terms and formulas are distinct syntactic entities
- Clausification is simple and focuses on the outer skeleton

Challenge 3: Boolean Expressions

In first-order logic:

- Terms and formulas are distinct syntactic entities
- Clausification is simple and focuses on the outer skeleton

Higher-order issues:

A. Formulas can appear nested in terms, including under λ's
   e.g. (λx. if $\exists y$. p x y then a else b)

B. We cannot perform clausification entirely in preprocessing

Challenge 3: Boolean Expressions

Solution:

- We introduce dedicated inference rules to clausify dynamically

Soundness of λ-Superposition

The inference rules are easy to show sound

In particular, the extra rule is justified by **argument congruence**:
if $[\![g]\!]^{\mathcal{J}} = [\![f]\!]^{\mathcal{J}}$, then $[\![g\ a]\!]^{\mathcal{J}} = [\![f\ a]\!]^{\mathcal{J}}$ for any a

Completeness of λ-Superposition

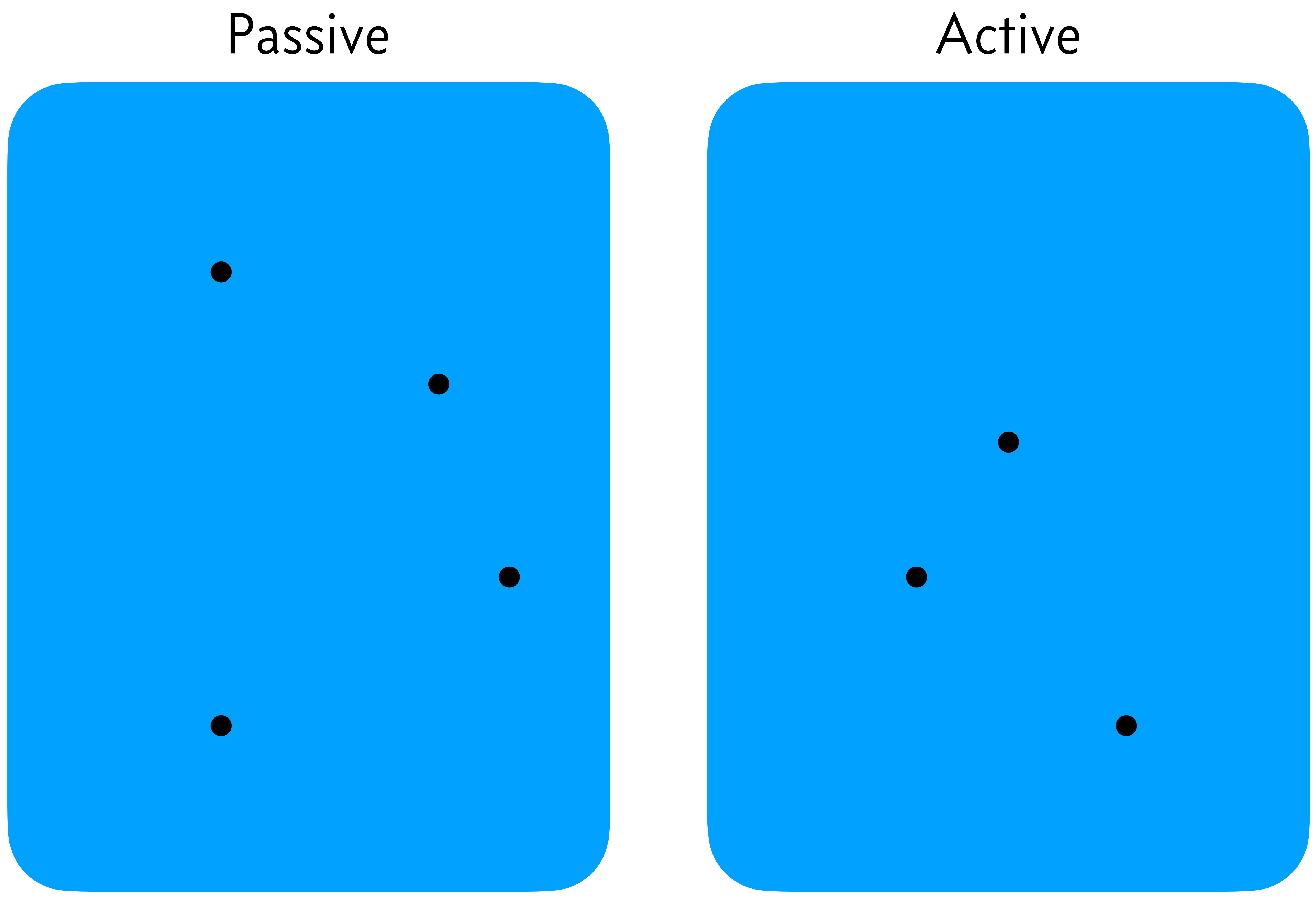
If the clause set $\mathtt{F}$ is initially unsatisfiable*
and inferences are performed fairly,
then $\mathtt{F}$ will eventually contain $\bot$

Completeness of λ-Superposition

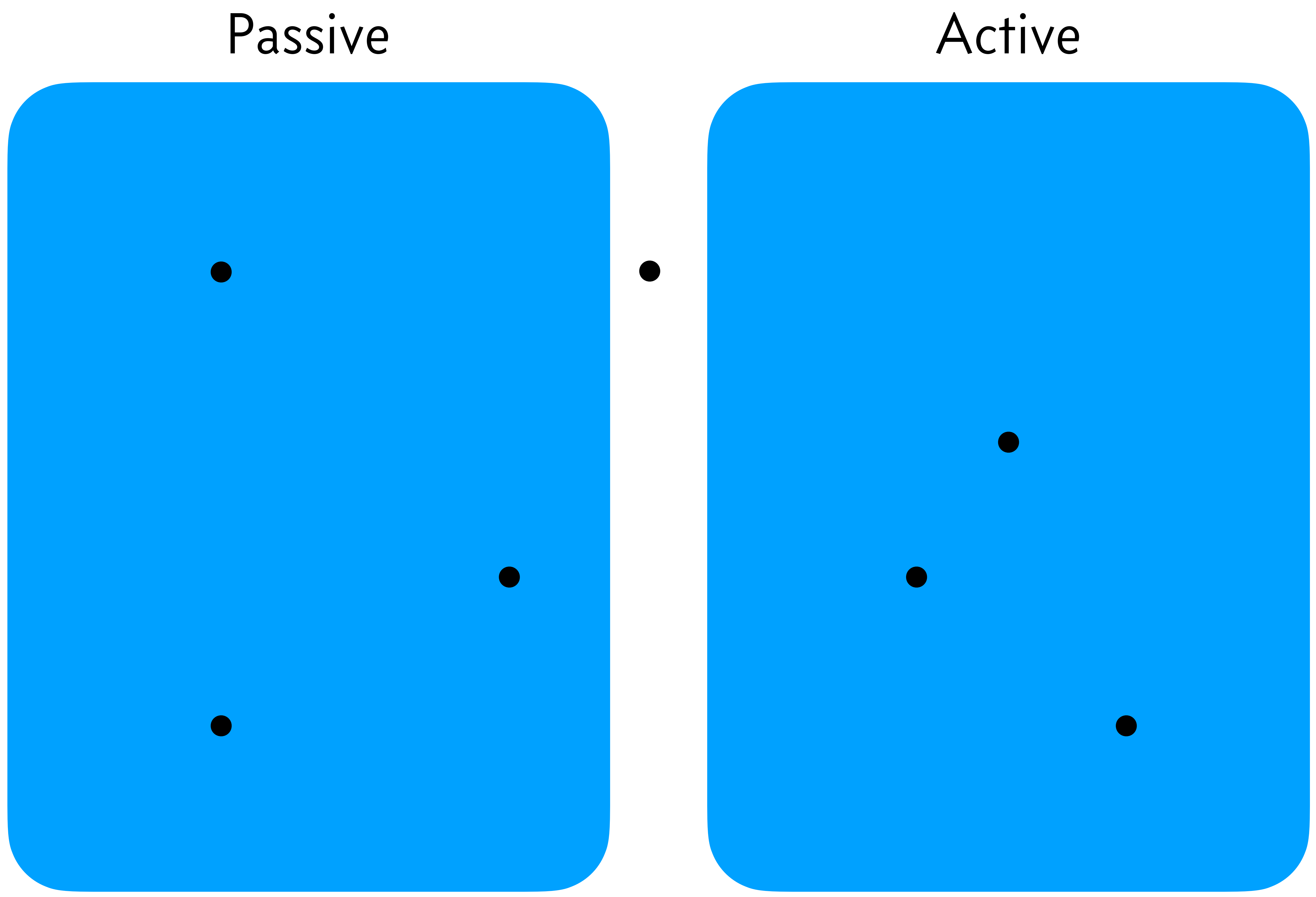
If the clause set $\mathtt{F}$ is initially unsatisfiable*
and inferences are performed fairly,
then $\mathtt{F}$ will eventually contain $\bot$

* with respect to the so-called Henkin semantics

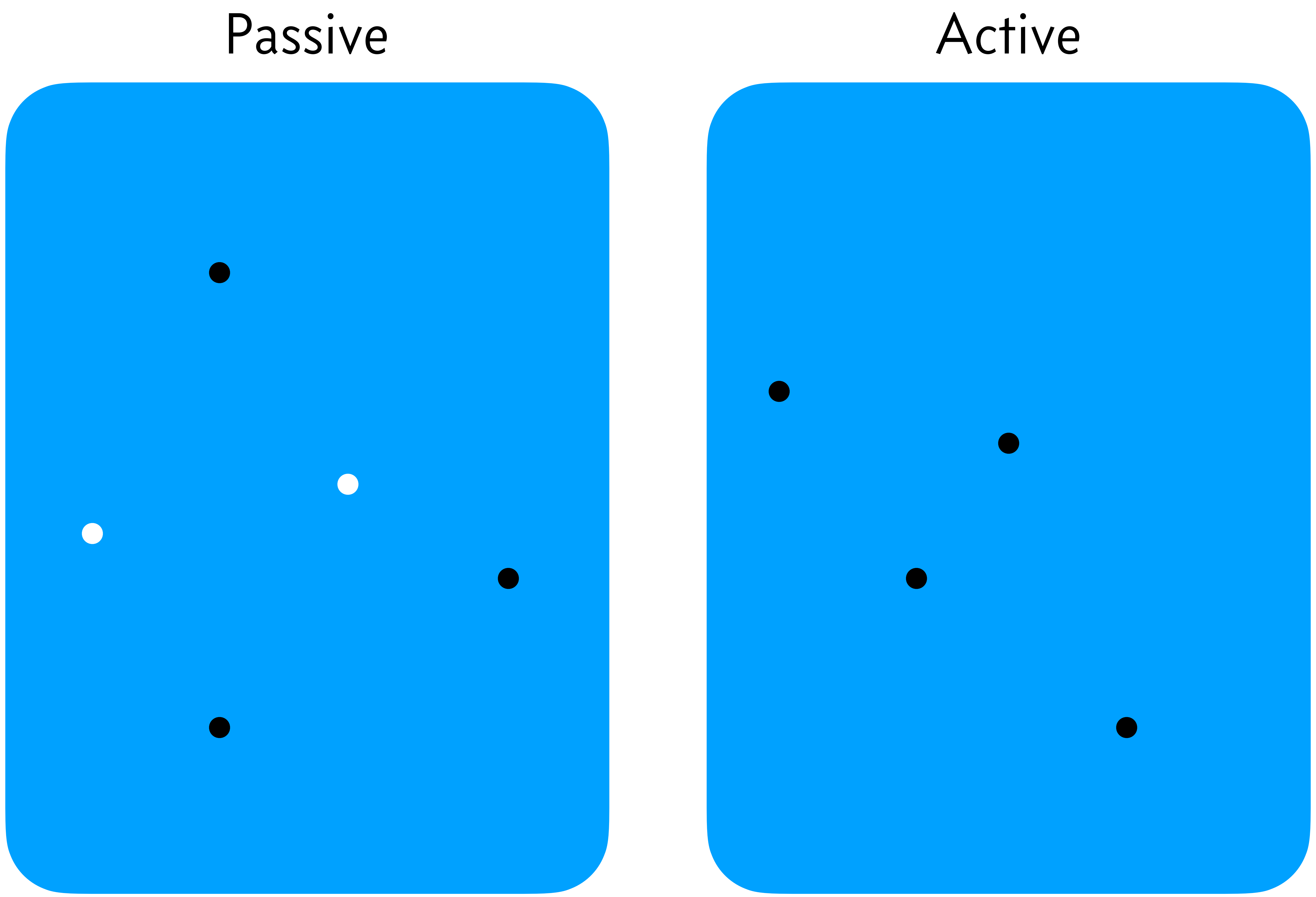
Standard Saturation Loop



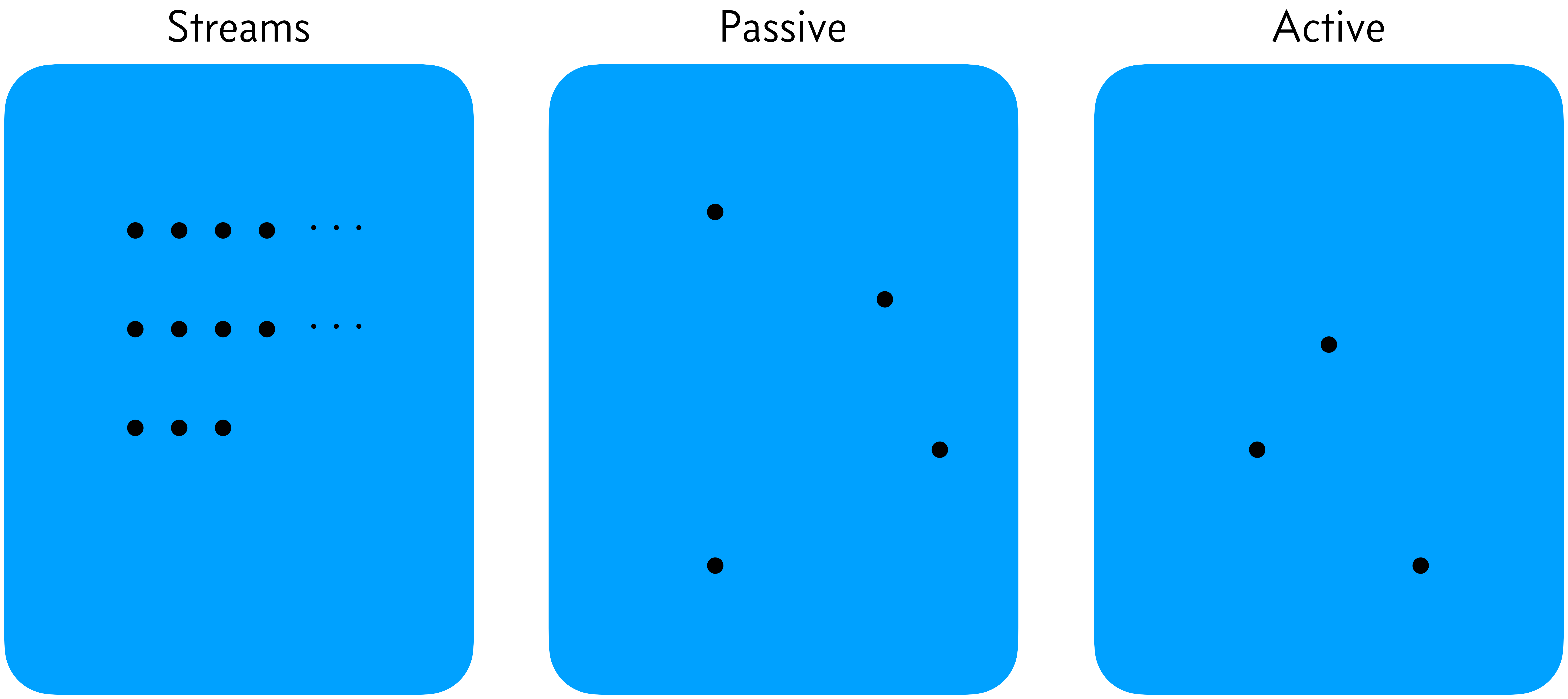
Standard Saturation Loop



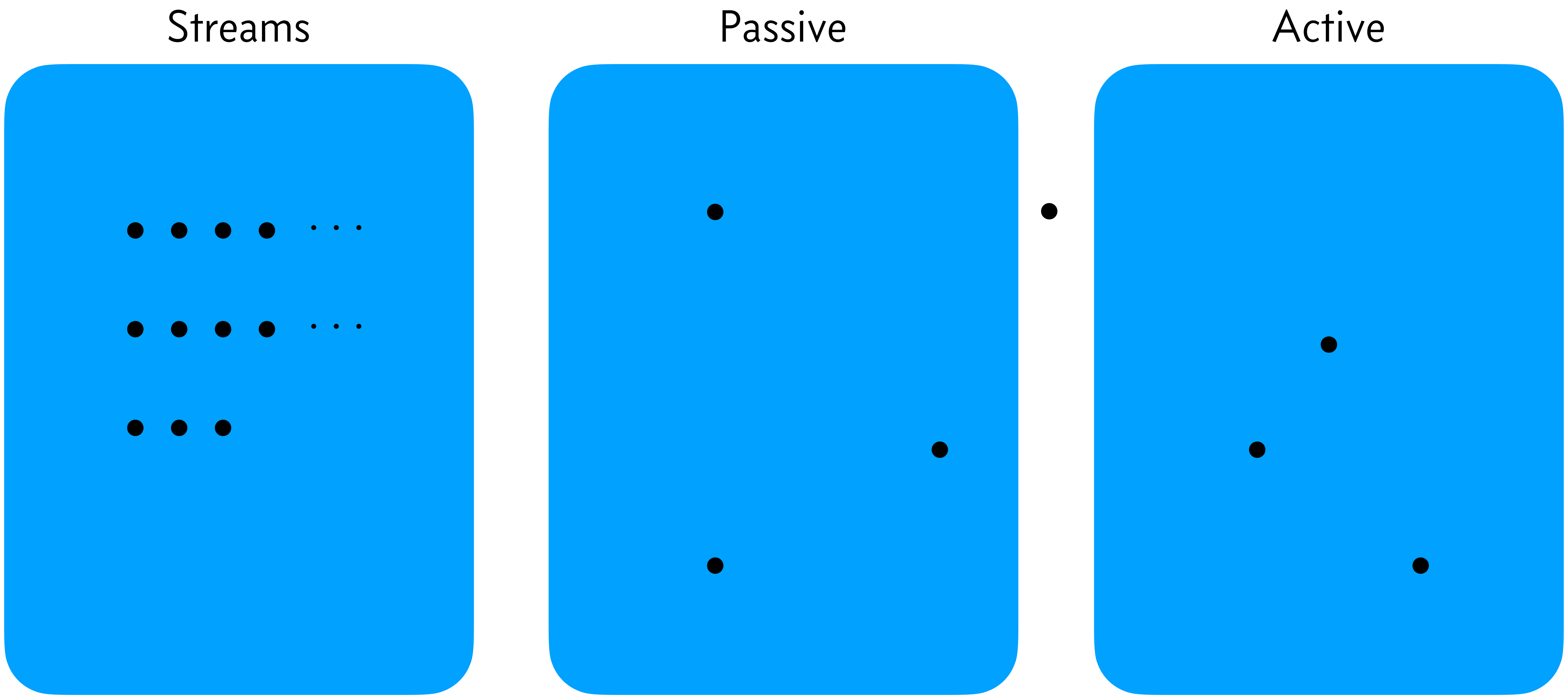
Standard Saturation Loop



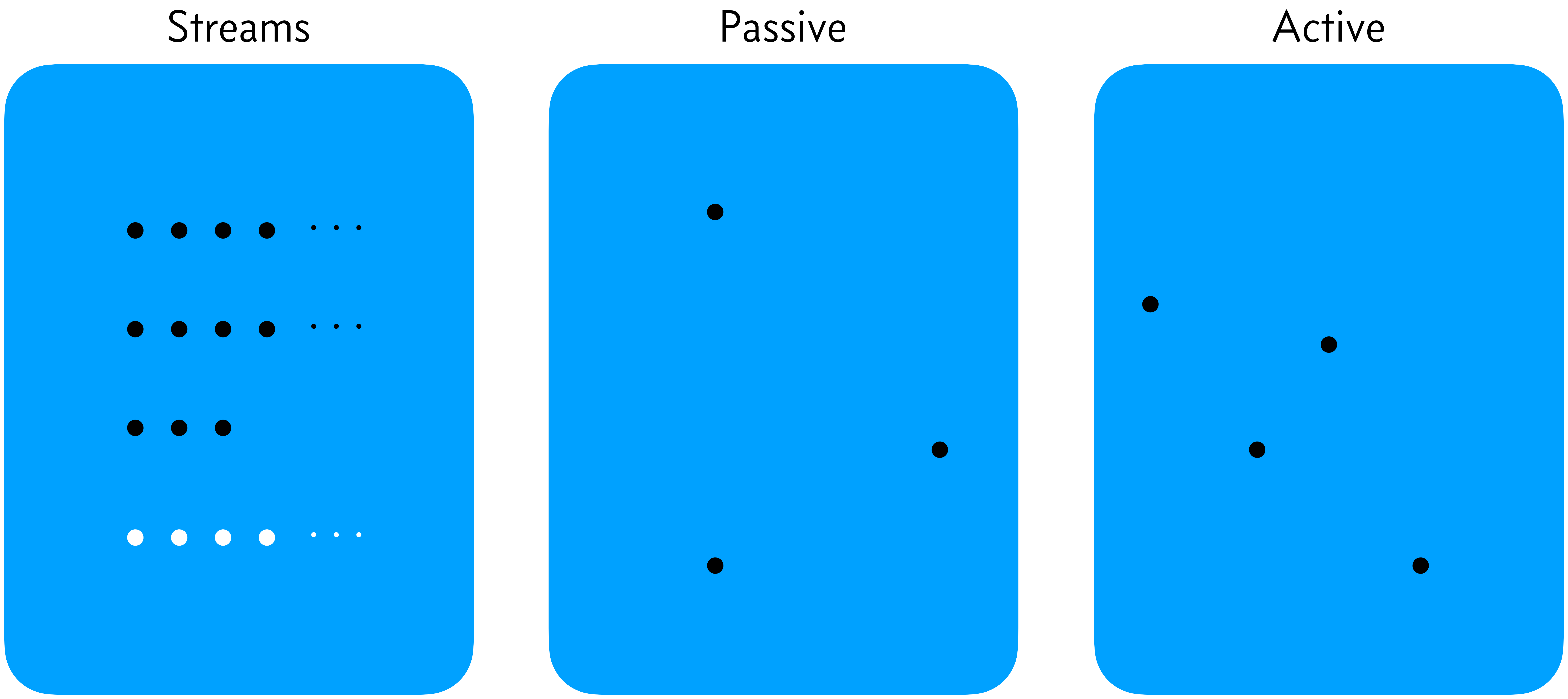
Generalized Saturation Loop



Generalized Saturation Loop



Generalized Saturation Loop



Competition Results (CASC 2023)

Higher-order Theorems	Vampire 4.8	Zipperpin 2.1.9999	*Zipperpin 2.1.999*	E 3.1	Leo-III 1.7.8	Satallax 3.4	cvc5 1.0.5	Lash 1.13	LEO-II 1.7.0	Duper 1.0
Solved/500	452/500	440/500	438/500	407/500	302/500	268/500	258/500	208/500	58/500	36/500
Solutions	452 90%	440 88%	438 87%	407 81%	302 60%	268 53%	258 51%	196 39%	58 11%	36 7%

SLedgeHammer Theorems	*E 3.0*	E 3.1	Zipperpin 2.1.9999	Vampire 4.8	cvc5 1.0.5	Satallax 3.4	Lash 1.13	Leo-III 1.7.8	Duper 1.0
Solved/1000	467/1000	467/1000	462/1000	454/1000	362/1000	278/1000	219/1000	125/1000	51/1000
Solutions	467 46%	467 46%	462 46%	454 45%	362 36%	278 27%	219 21%	125 12%	51 5%

Competition Results (CASC 2023)

Higher-order Theorems	Vampire 4.8	Zipperpin 2.1.9999	*Zipperpin 2.1.999*	E 3.1	Leo-III 1.7.8	Satallax 3.4	cvc5 1.0.5	Lash 1.13	LEO-II 1.7.0	Duper 1.0
Solved/500	452/500	440/500	438/500	407/500	302/500	268/500	258/500	208/500	58/500	36/500
Solutions	452 90%	440 88%	438 87%	407 81%	302 60%	268 53%	258 51%	196 39%	58 11%	36 7%

SLedgeHammer Theorems	*E 3.0*	E 3.1	Zipperpin 2.1.9999	Vampire 4.8	cvc5 1.0.5	Satallax 3.4	Lash 1.13	Leo-III 1.7.8	Duper 1.0
Solved/1000	467/1000	467/1000	462/1000	454/1000	362/1000	278/1000	219/1000	125/1000	51/1000
Solutions	467 46%	467 46%	462 46%	454 45%	362 36%	278 27%	219 21%	125 12%	51 5%

References

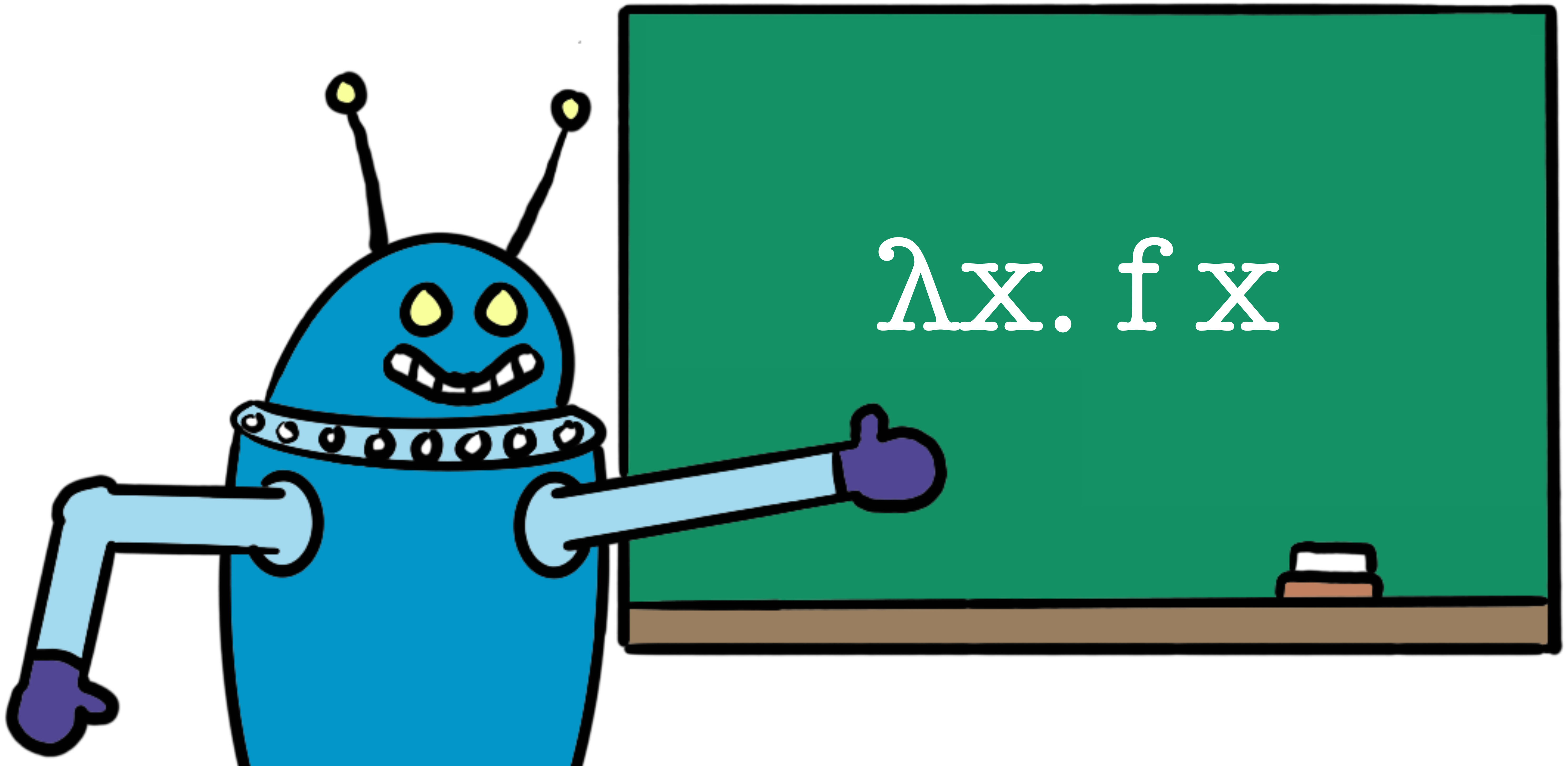
Superposition with Lambdas
A. Bentkamp, J. Blanchette, S. Tourret, P. Vukmirović, and U. Waldmann
Journal of Automated Reasoning 65(7), 2021

Superposition for Higher-Order Logic
A. Bentkamp, J. Blanchette, S. Tourret, and P. Vukmirović
Journal of Automated Reasoning 67, article number 10, 2023

Mechanical Mathematicians
A. Bentkamp, J. Blanchette, V. Nummelin, S. Tourret, P. Vukmirović, and U. Waldmann
Communications of the ACM 66(4), 2023

Provers & Solvers

Lecture 3: λ-Superposition



Jasmin Blanchette
LMU Munich

Partly based on slides by
Alexander Bentkamp

Blah