▶ APLAS 2003 — Beijing, China

27–29 November 2003

Type Inference with structural Subtyping: the faithful description of an efficient constraint solver

Vincent Simonet

Vincent.Simonet@inria.fr
http://cristal.inria.fr/~simonet/

INRIA ROCQUENCOURT

Talk Outline

- Structural subtyping
- Solving constraints
- Simplifying constraints
- Experimental results

Structural subtyping

What is subtyping?

- A partial order $\leq$ on types and a subsumption rule:

$$\frac{\Gamma \vdash e : t_1 \qquad t_1 \leq t_2}{\Gamma \vdash e : t_2}$$

- Widely used: object-oriented languages, static analysis, ...
- The definition of the subtyping order itself varies, depending on the application.

Structural subtyping: an example

Ground types

$$t ::= \mathsf{num} \quad \mid t \rightarrow t \mid t \times t$$

Structural subtyping: an example

Atoms

$$a \in \begin{tikzcd}[column sep=small] & complex & \\ real \arrow[ur] & \leq_{\mathcal{A}} & fermat \arrow[ul] \\ & nat \arrow[ul] \arrow[ur] & \end{tikzcd}$$

Ground types

$$t ::= \mathsf{num}\ a \mid t \to t \mid t \times t$$

Subtyping order

$$\frac{a \leq_{\mathcal{A}} a'}{\mathsf{num}\ a \leq \mathsf{num}\ a'} \qquad \frac{t_1' \leq t_1 \qquad t_2 \leq t_2'}{t_1 \to t_2 \leq t_1' \to t_2'} \qquad \frac{t_1 \leq t_1' \qquad t_2 \leq t_2'}{t_1 \times t_2 \leq t_1' \times t_2'}$$

Structural subtyping (1/2)

In structural subtyping, comparable types must have the same shape an can only differ by their atomic leaves.

- A poset of atoms $a \in (\mathcal{A}, \leq_{\mathcal{A}})$, supposed to be a lattice.
- A set of type constructors $c \in \mathcal{C}$.
  Every type constructor is given with an arity, each parameter must be either covariant or contravariant.
- Ground types are defined by the following grammar:

$$t ::= a \mid c(t_1, \ldots, t_n)$$

Structural subtyping (2/2)

The subtyping order $\leq$ is defined by lifting the atomic order $\leq_{\mathcal{A}}$ along the type structure:

$$\frac{a \leq_{\mathcal{A}} a'}{a \leq a'} \qquad \frac{\begin{array}{c}\text{if } c\text{'s } i^{\text{th}} \text{ parameter is covariant then } t_i \leq t'_i \\ \text{if } c\text{'s } i^{\text{th}} \text{ parameter is contravariant then } t'_i \leq t_i\end{array}}{c(t_1, \ldots, t_n) \leq c(t'_1, \ldots, t'_n)}$$

We also define $t \approx t'$ (read: t has the same shape as t'):

$$a \approx a' \qquad \frac{\forall i \;\; t_i \approx t'_i}{c(t_1, \ldots, t_n) \approx c(t'_1, \ldots, t'_n)}$$

$\approx$ is the symmetric transitive closure of $\leq$.

Type inference for structural subtyping has been widely studied

Mitchell (1981–1984) Fuh and Mishra (1988-1989)	First algorithms
Tiuryn (1992)	Solving atomic constraints is PSPACE-hard in the general case, but linear in a lattice
Hoang and Mitchell (1995)	Type inference is equivalent to constraint resolution
Rehof (1997)	Minimal typings
Kuncak and Rinard (2003)	The first-order theory of structural subtyping is decidable

Our work

However, most of the implementation techniques remains in folklore. This paper proposes:

- A complete account about efficient techniques for constraint resolution in the case of structural subtyping.
- A generic implementation of an efficient constraint solver for structural subtyping.

Constraint-based type inference

Type inference systems for subtyping are constraint-based:

- Every piece of code is described by a constrained type scheme, which is computed by a syntactic analysis:

$$
\begin{array}{rll}
\sigma ::= & \forall\bar{\alpha}[C].\tau & \text{(scheme)} \\
\tau ::= & \alpha \mid a \mid c(\tau_1, \ldots, \tau_n) & \text{(type)} \\
C ::= & \tau \leq \tau' \mid C_1 \wedge C_2 \mid \exists\alpha.C & \text{(constraint)}
\end{array}
$$

- It is well-typed if and only if the type scheme has an instance, i.e. its constraint is satisfiable

$\boxed{\text{type inference is reduced to constraint solving}}$

Example

$$\begin{aligned}\mathsf{let}\ f = \lambda x.\lambda y\big(&\mathsf{bind}\ s = x + y\ \mathsf{in}\\ &\mathsf{bind}\ p = x \times y\ \mathsf{in}\\ &(p + s, p - s)\big)\end{aligned}$$

$$\forall\alpha\left[\begin{array}{l} \quad\exists\alpha_1\alpha_2\alpha_3.(\alpha_1 = \alpha_2 \to \alpha_3 \wedge \alpha_1 \leq \alpha \\ \wedge\ \exists\alpha_4\alpha_5\alpha_6.(\alpha_4 = \alpha_5 \to \alpha_6 \wedge \alpha_4 \leq \alpha_3 \\ \quad\wedge\ \exists\beta_1\gamma_1.(\gamma_1 = \mathsf{num}\ \beta_1 \wedge \alpha_2 \leq \gamma_1 \wedge \alpha_5 \leq \gamma_1 \\ \quad\wedge\ \exists\beta_2\gamma_2.(\gamma_2 = \mathsf{num}\ \beta_2 \wedge \alpha_2 \leq \gamma_2 \wedge \alpha_5 \leq \gamma_2 \\ \quad\wedge\ \exists\beta_3\gamma_3.(\gamma_3 = \mathsf{num}\ \beta_3 \wedge \gamma_1 \leq \gamma_3 \wedge \gamma_2 \leq \gamma_3 \\ \quad\wedge\ \exists\beta_4\gamma_4.(\gamma_4 = \mathsf{num}\ \beta_4 \wedge \gamma_1 \leq \gamma_4 \wedge \gamma_2 \leq \gamma_4 \\ \wedge\ \exists\alpha_7\alpha_8\gamma_5.(\gamma_3 \leq \alpha_7 \wedge \gamma_4 \leq \alpha_8 \\ \qquad\qquad \wedge\gamma_5 = \alpha_7 \times \alpha_8 \wedge \gamma_5 \leq \alpha_6) \cdots) \end{array}\right].\alpha$$

Example

$$\text{let } f = \lambda x.\lambda y\Big(\text{bind } s = x + y \text{ in}$$
$$\text{bind } p = x \times y \text{ in}$$
$$(p + s, p - s)\Big)$$

$$\forall\alpha\big[\exists\beta.(\alpha = \text{num } \beta \to \text{num } \beta \to \text{num } \beta \times \text{num } \beta)\big].\alpha$$

$$\forall\beta.\text{num } \beta \to \text{num } \beta \to \text{num } \beta \times \text{num } \beta$$

Solving constraints

Overview

The solving procedure consists in three steps

- Unification: discovering sharing between type structures
- Expansion and decomposition: making type structure explicit and decomposing inequalities
- Solving the resulting atomic problem.

Example: unification (1/2)

Two unification algorithms are simultaneously performed:

- One for type equality ($=$)
- One for type structure ($\approx$)

$$\begin{array}{c} \alpha_1 = \alpha_2 \to \alpha_3 \\ \alpha_1 \leq \alpha \end{array} \longrightarrow \boxed{\dfrac{\langle \alpha_1 = \alpha_2 \to \alpha_3 \rangle \approx \alpha}{\alpha_1 \leq \alpha}}$$

Example: unification (2/2)

$$\frac{\langle \alpha_1 = \alpha_2 \rightarrow \alpha_3 \rangle \approx \alpha}{\alpha_1 \leq \alpha}$$

$$\frac{\langle \alpha_4 = \alpha_5 \rightarrow \alpha_6 \rangle \approx \alpha_3}{\alpha_4 \leq \alpha_3}$$

$$\frac{\langle \gamma_5 = \alpha_7 \times \alpha_8 \rangle \approx \alpha_6}{\gamma_5 \leq \alpha_6}$$

$$\frac{\begin{array}{c}\langle \gamma_1 = \mathsf{num}\ \beta_1 \rangle \approx \langle \gamma_2 = \mathsf{num}\ \beta_2 \rangle \approx \langle \gamma_3 = \mathsf{num}\ \beta_3 \rangle \approx \langle \gamma_4 = \mathsf{num}\ \beta_4 \rangle \approx \\ \alpha_2 \approx \alpha_5 \approx \alpha_7 \approx \alpha_8\end{array}}{\alpha_2, \alpha_5 \leq \gamma_1, \gamma_2 \wedge \gamma_1, \gamma_2 \leq \gamma_3, \gamma_4 \wedge \gamma_3 \leq \alpha_7 \wedge \gamma_4 \leq \alpha_8}$$

$$\frac{\beta_1 \approx \beta_2 \approx \beta_3 \approx \beta_4}{\quad}$$

Example: expansion and decomposition

$$\frac{\langle \alpha_1 = \alpha_2 \to \alpha_3 \rangle \approx \langle \alpha = \alpha_9 \to \alpha_{10} \rangle}{\cancel{\alpha_1 \leq \alpha}}$$

$$\frac{\langle \alpha_4 = \alpha_5 \to \alpha_6 \rangle \approx \alpha_3 \approx \alpha_{10}}{\alpha_3 \leq \alpha_{10} \wedge \alpha_4 \leq \alpha_3}$$

$$\frac{\langle \gamma_5 = \alpha_7 \times \alpha_8 \rangle \approx \alpha_6}{\gamma_5 \leq \alpha_6}$$

$$\frac{\begin{array}{c}\langle \gamma_1 = \mathsf{num}\ \beta_1 \rangle \approx \langle \gamma_2 = \mathsf{num}\ \beta_2 \rangle \approx \langle \gamma_3 = \mathsf{num}\ \beta_3 \rangle \approx \langle \gamma_4 = \mathsf{num}\ \beta_4 \rangle \approx \\ \alpha_2 \approx \alpha_5 \approx \alpha_7 \approx \alpha_8 \approx \alpha_9\end{array}}{\alpha_9 \leq \alpha_2 \wedge \alpha_2, \alpha_5 \leq \gamma_1, \gamma_2 \wedge \gamma_1, \gamma_2 \leq \gamma_3, \gamma_4 \wedge \gamma_3 \leq \alpha_7 \wedge \gamma_4 \leq \alpha_8}$$

$$\frac{\beta_1 \approx \beta_2 \approx \beta_3 \approx \beta_4}{\quad}$$

Example: expansion and decomposition

$$\frac{\langle \alpha_1 = \alpha_2 \to \alpha_3 \rangle \approx \langle \alpha = \alpha_9 \to \alpha_{10} \rangle}{}$$

$$\frac{\langle \alpha_4 = \alpha_5 \to \alpha_6 \rangle \approx \langle \alpha_3 = \alpha_{11} \to \alpha_{12} \rangle \approx \langle \alpha_{10} = \alpha_{13} \to \alpha_{14} \rangle}{\cancel{\alpha_3 \leq \alpha_{10}} \wedge \cancel{\alpha_4 \leq \alpha_3}}$$

$$\frac{\langle \gamma_5 = \alpha_7 \times \alpha_8 \rangle \approx \alpha_6 \approx \alpha_{12} \approx \alpha_{14}}{\gamma_5 \leq \alpha_6 \wedge \alpha_{12} \leq \alpha_{14} \wedge \alpha_6 \leq \alpha_{12}}$$

$$\frac{\langle \gamma_1 = \mathsf{num}\ \beta_1 \rangle \approx \langle \gamma_2 = \mathsf{num}\ \beta_2 \rangle \approx \langle \gamma_3 = \mathsf{num}\ \beta_3 \rangle \approx \langle \gamma_4 = \mathsf{num}\ \beta_4 \rangle \approx \\ \alpha_2 \approx \alpha_5 \approx \alpha_7 \approx \alpha_8 \approx \alpha_9 \approx \alpha_{11} \approx \alpha_{13}}{\alpha_9 \leq \alpha_2 \wedge \alpha_{13} \leq \alpha_{11} \wedge \alpha_{11} \leq \alpha_5 \wedge \alpha_2, \alpha_5 \leq \gamma_1, \gamma_2 \wedge \gamma_1, \gamma_2 \leq \gamma_3, \gamma_4 \wedge \gamma_3 \leq \alpha_7 \wedge \gamma_4 \leq \alpha_8}$$

$$\frac{\beta_1 \approx \beta_2 \approx \beta_3 \approx \beta_4}{}$$

Complexity

This strategy is theoretically exponential, but has a linear complexity under the hypothesis of types of bounded height.

We have to simplify constraints throughout the solving process:

- To improve its efficiency,
- To obtain human readable results.

Simplifying constraints

Overview

Simplification is a subtle problem:

- It must be correct!
- It cannot be time consuming.

Simplification can be performed:

- Throughout the expansion process, in order to reduce the potential number of expanded variables.
- At the atomic level: useful in presence of let-polymorphism.

We are not interested by optimality (in size) of simplification techniques.

Polarities

[Fuh & Mishra (1989), Trifonov & Smith, Pottier]

Polarities assigned to type variables in a type scheme gives some information about their use:

A $\frac{\text{negative}}{\text{positive}}$ type variable represents an $\frac{\text{input}}{\text{output}}$ of the expression

Example:

$$\forall \alpha_1 \alpha_2 \beta_1 \beta_2 [\alpha_1 \leq \beta_1 \wedge \alpha_2 \leq \beta_1 \wedge \alpha_2 \leq \beta_2].\alpha_1 \rightarrow \alpha_2 \rightarrow \beta_1 \times \beta_2$$

A $\frac{\text{negative}}{\text{positive}}$ type variable can receive new $\frac{\text{lower bounds}}{\text{upper bounds}}$

Chains reduction

[Eifrig et al. (1995), Aiken and Fähndrich (1996)]

Polarities can be exploited throughout expansion to eliminate variables:

A $\dfrac{\text{non-positive}}{\text{non-negative}}$ variable can be unified with its unique $\dfrac{\text{upper bound}}{\text{lower bound}}$

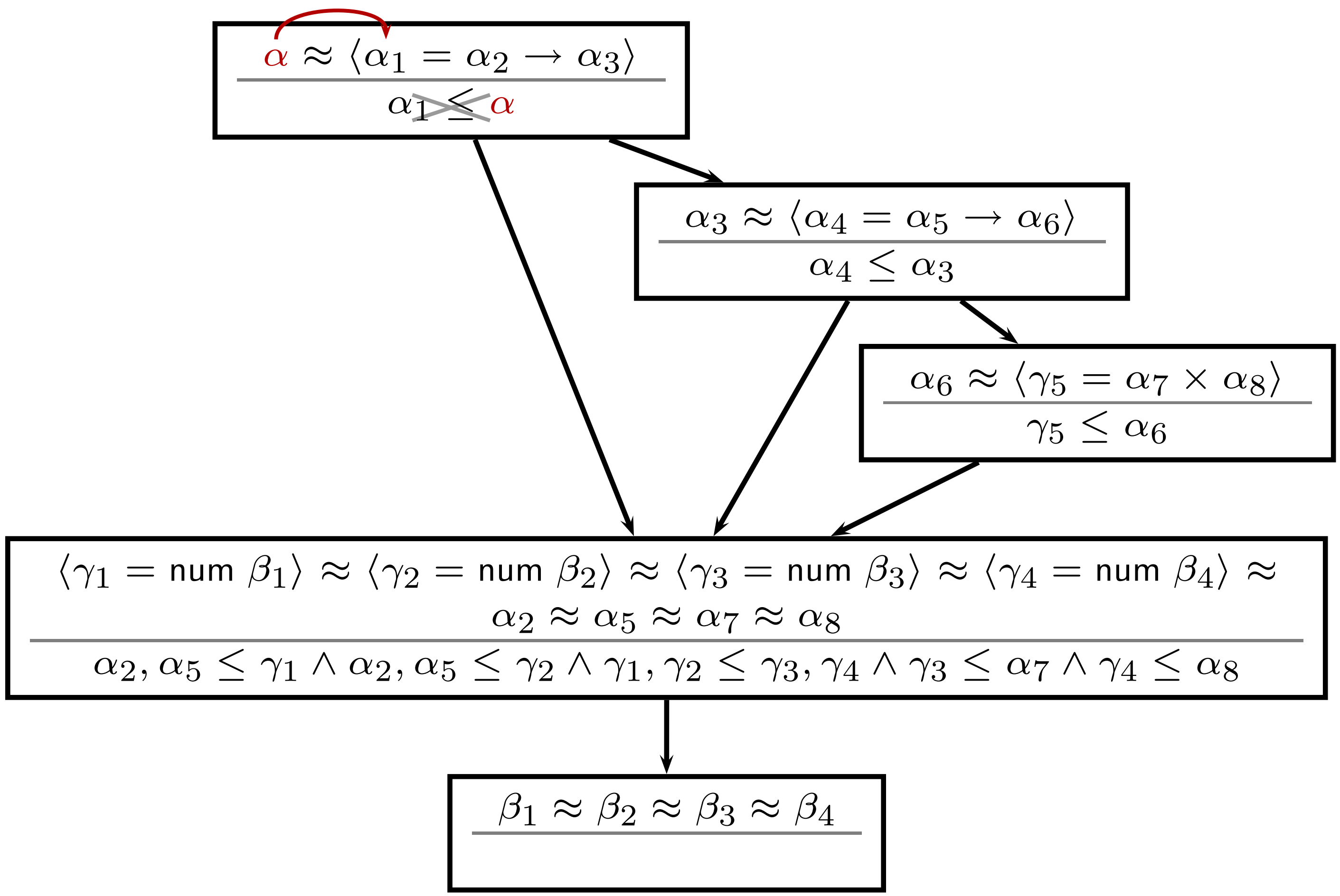
Example:

$$\forall \alpha_1 \alpha_2 \beta_1 \beta_2 [\alpha_1 \leq \beta_1 \wedge \alpha_2 \leq \beta_1 \wedge \alpha_2 \leq \beta_2].\alpha_1 \rightarrow \alpha_2 \rightarrow \beta_1 \times \beta_2$$

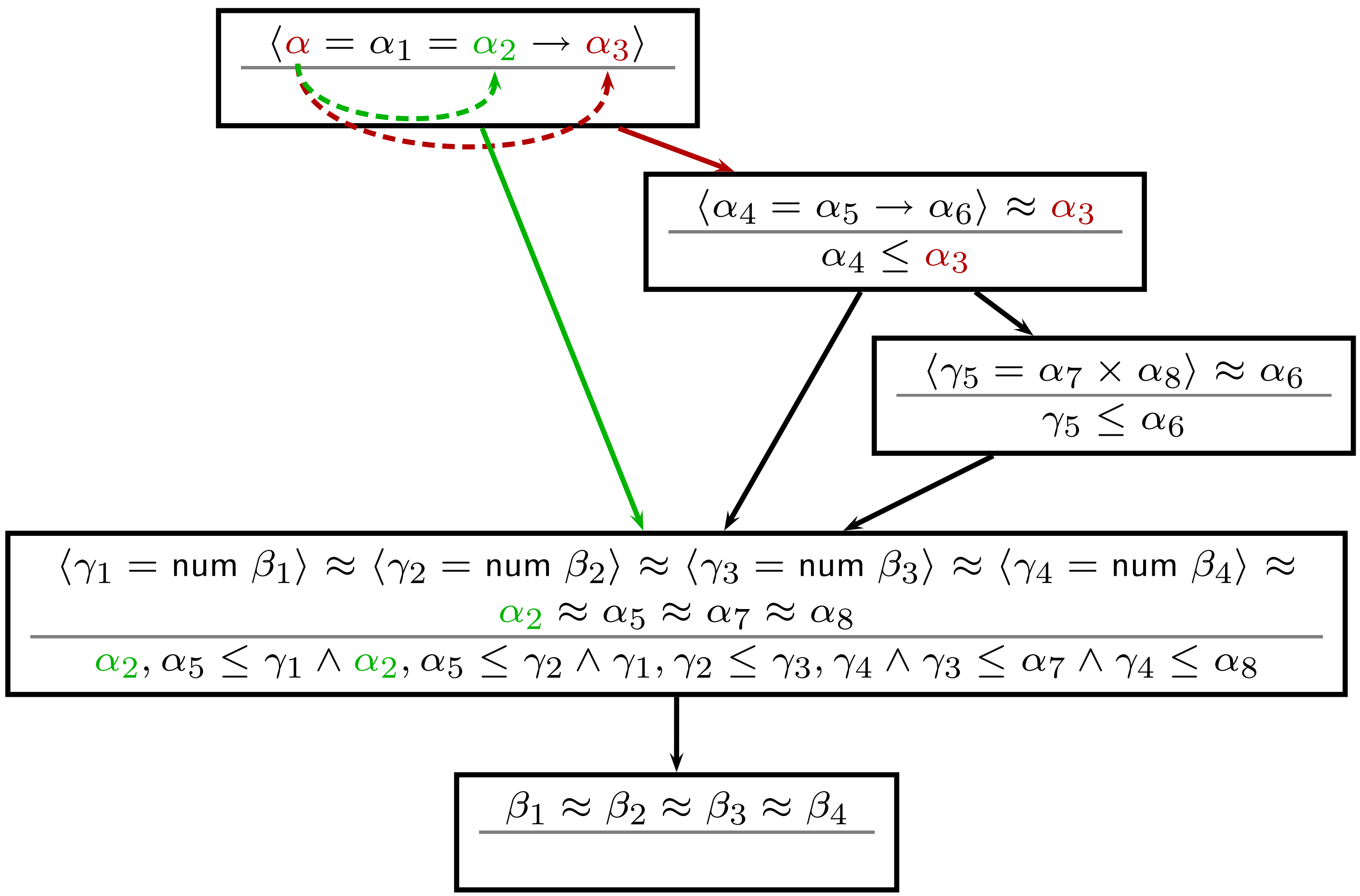
is equivalent to

$$\forall \alpha_1 \alpha_2 \beta_1 \beta_2 [\alpha_1 \leq \beta_1 \wedge \alpha_2 \leq \beta_1].\alpha_1 \rightarrow \alpha_2 \rightarrow \beta_1 \times \alpha_2$$

Example: reducing a chain



Example: propagating polarities



Example: removing a non-polar variable

$$\frac{\langle \alpha = \cancel{\alpha_1} = \alpha_2 \to \alpha_3 \rangle}{}$$

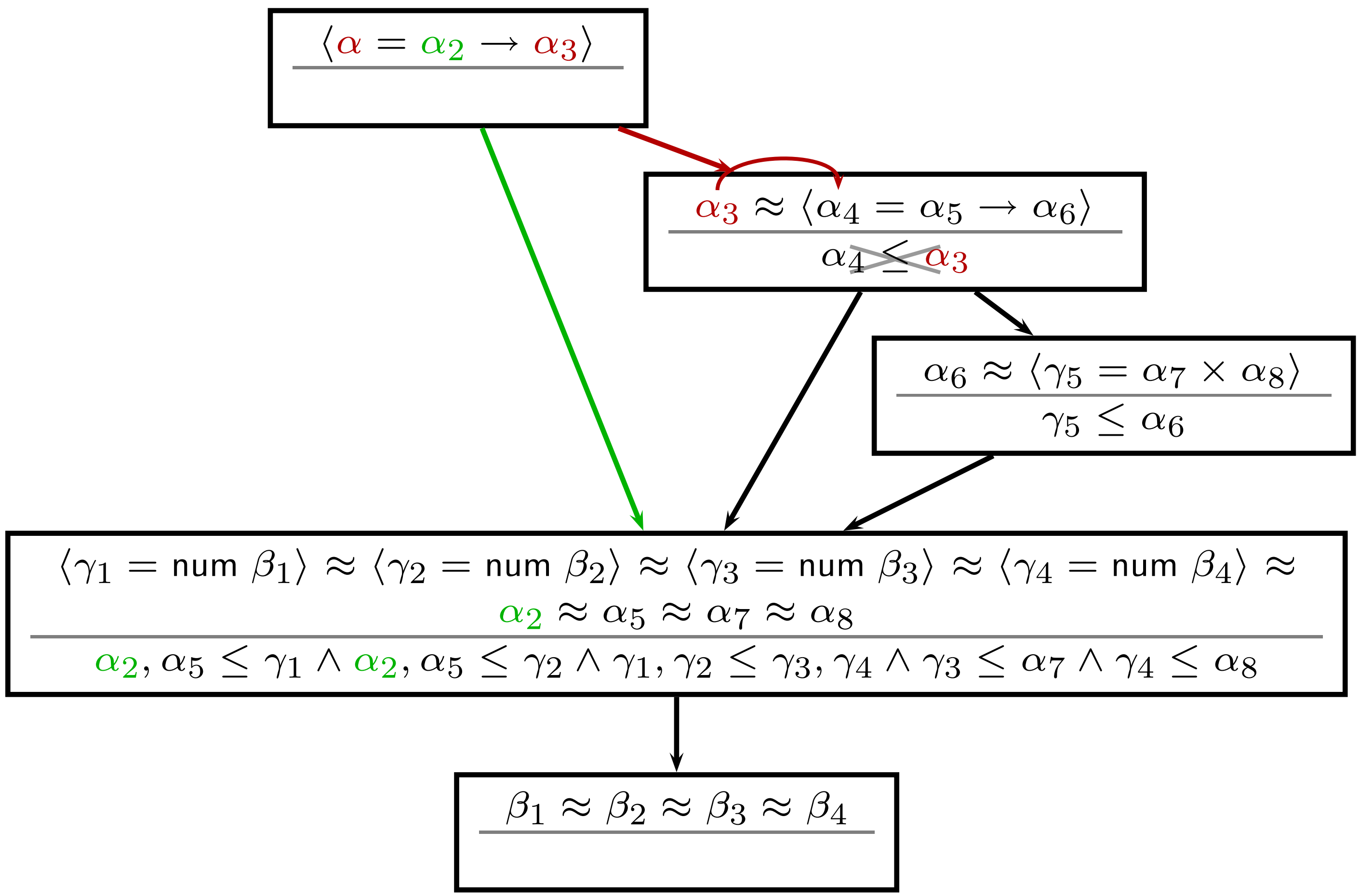
$$\frac{\alpha_3 \approx \langle \alpha_4 = \alpha_5 \to \alpha_6 \rangle}{\alpha_4 \leq \alpha_3}$$

$$\frac{\alpha_6 \approx \langle \gamma_5 = \alpha_7 \times \alpha_8 \rangle}{\gamma_5 \leq \alpha_6}$$

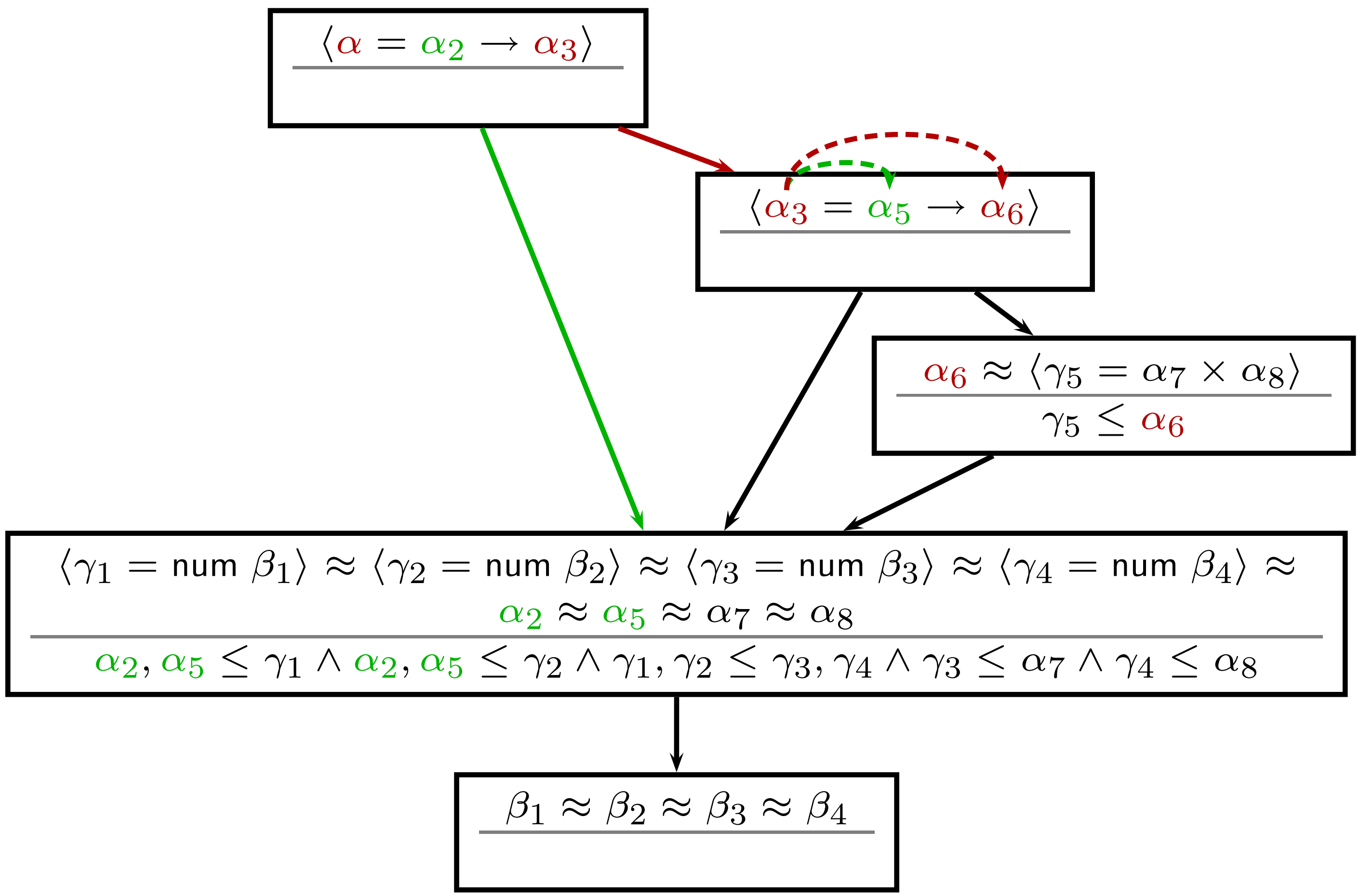
$$\frac{\langle \gamma_1 = \textsf{num}\ \beta_1 \rangle \approx \langle \gamma_2 = \textsf{num}\ \beta_2 \rangle \approx \langle \gamma_3 = \textsf{num}\ \beta_3 \rangle \approx \langle \gamma_4 = \textsf{num}\ \beta_4 \rangle \approx \alpha_2 \approx \alpha_5 \approx \alpha_7 \approx \alpha_8}{\alpha_2, \alpha_5 \leq \gamma_1 \wedge \alpha_2, \alpha_5 \leq \gamma_2 \wedge \gamma_1, \gamma_2 \leq \gamma_3, \gamma_4 \wedge \gamma_3 \leq \alpha_7 \wedge \gamma_4 \leq \alpha_8}$$

$$\frac{\beta_1 \approx \beta_2 \approx \beta_3 \approx \beta_4}{}$$

Example: reducing a chain



Example: reducing a chain



Reducing two chains

$$\frac{\langle \alpha = \alpha_2 \rightarrow \alpha_3 \rangle}{}$$

$$\frac{\langle \alpha_3 = \alpha_5 \rightarrow \alpha_6 \rangle}{}$$

$$\frac{\langle \alpha_6 = \alpha_7 \times \alpha_8 \rangle}{}$$

$$\frac{\langle \gamma_1 = \text{num } \beta_1 \rangle \approx \langle \gamma_2 = \text{num } \beta_2 \rangle \approx \langle \gamma_3 = \text{num } \beta_3 \rangle \approx \langle \gamma_4 = \text{num } \beta_4 \rangle \approx \alpha_2 \approx \alpha_5 \approx \alpha_7 \approx \alpha_8}{\alpha_2, \alpha_5 \leq \gamma_1, \gamma_2 \wedge \gamma_1, \gamma_2 \leq \gamma_3, \gamma_4 \wedge \gamma_3 \cancel{\leq} \alpha_7 \wedge \gamma_4 \cancel{\leq} \alpha_8}$$

$$\downarrow$$

$$\frac{\beta_1 \approx \beta_2 \approx \beta_3 \approx \beta_4}{}$$

Example: reducing two chains

$$\frac{\langle \alpha = \alpha_2 \to \alpha_3 \rangle}{}$$

$$\frac{\langle \alpha_3 = \alpha_5 \to \alpha_6 \rangle}{}$$

$$\frac{\langle \alpha_6 = \alpha_7 \times \alpha_8 \rangle}{}$$

$$\frac{\langle \gamma_1 = \mathsf{num}\ \beta_1 \rangle \approx \langle \gamma_2 = \mathsf{num}\ \beta_2 \rangle \approx \langle \gamma_3 = \alpha_7 = \mathsf{num}\ \beta_3 \rangle \approx \langle \gamma_4 = \alpha_8 = \mathsf{num}\ \beta_4 \rangle \approx \langle \alpha_2 \rangle \approx \langle \alpha_5 \rangle}{\alpha_2, \alpha_5 \leq \gamma_1, \gamma_2 \wedge \gamma_1, \gamma_2 \leq \gamma_3, \gamma_4}$$

$\downarrow$

$$\frac{\beta_1 \approx \beta_2 \approx \beta_3 \approx \beta_4}{}$$

Example: expanding variables

$$\frac{\langle \alpha = \alpha_2 \rightarrow \alpha_3 \rangle}{}$$

$$\frac{\langle \alpha_3 = \alpha_5 \rightarrow \alpha_6 \rangle}{}$$

$$\frac{\langle \alpha_6 = \alpha_7 \times \alpha_8 \rangle}{}$$

$$\frac{\langle \gamma_1 = \mathsf{num}\ \beta_1 \rangle \approx \langle \gamma_2 = \mathsf{num}\ \beta_2 \rangle \approx \langle \gamma_3 = \alpha_7 = \mathsf{num}\ \beta_3 \rangle \approx \langle \gamma_4 = \alpha_8 = \mathsf{num}\ \beta_4 \rangle \approx \langle \alpha_2 = \mathsf{num}\ \beta_5 \rangle \approx \langle \alpha_5 = \mathsf{num}\ \beta_6 \rangle}{\alpha_2, \alpha_5 \leq \gamma_1, \gamma_2 \wedge \gamma_1, \gamma_2 \leq \gamma_3, \gamma_4}$$

$$\downarrow$$

$$\frac{\beta_1 \approx \beta_2 \approx \beta_3 \approx \beta_4 \approx \beta_5 \approx \beta_6}{}$$

Example: decomposing inequalities

$$\dfrac{\langle \alpha = \alpha_2 \to \alpha_3 \rangle}{}$$

$$\dfrac{\langle \alpha_3 = \alpha_5 \to \alpha_6 \rangle}{}$$

$$\dfrac{\langle \alpha_6 = \alpha_7 \times \alpha_8 \rangle}{}$$

$$\dfrac{\langle \gamma_1 \cancel{= \mathsf{num}} \ \beta_1 \rangle \approx \langle \gamma_2 \cancel{= \mathsf{num}} \ \beta_2 \rangle \approx \langle \cancel{\gamma_3} = \alpha_7 = \mathsf{num}\ \beta_3 \rangle \approx \langle \cancel{\gamma_4} = \alpha_8 = \mathsf{num}\ \beta_4 \rangle \approx \langle \alpha_2 = \mathsf{num}\ \beta_5 \rangle \approx \langle \alpha_5 = \mathsf{num}\ \beta_6 \rangle}{\alpha_2, \cancel{\alpha_5 \le \gamma_1}, \gamma_2 \wedge \gamma_1, \cancel{\gamma_2 \le \gamma_3}, \gamma_4}$$

$$\downarrow$$

$$\dfrac{\beta_1 \approx \beta_2 \approx \beta_3 \approx \beta_4 \approx \beta_5 \approx \beta_6}{\beta_5, \beta_6 \le \gamma_1, \gamma_2 \wedge \gamma_1, \gamma_2 \le \beta_3, \beta_4}$$

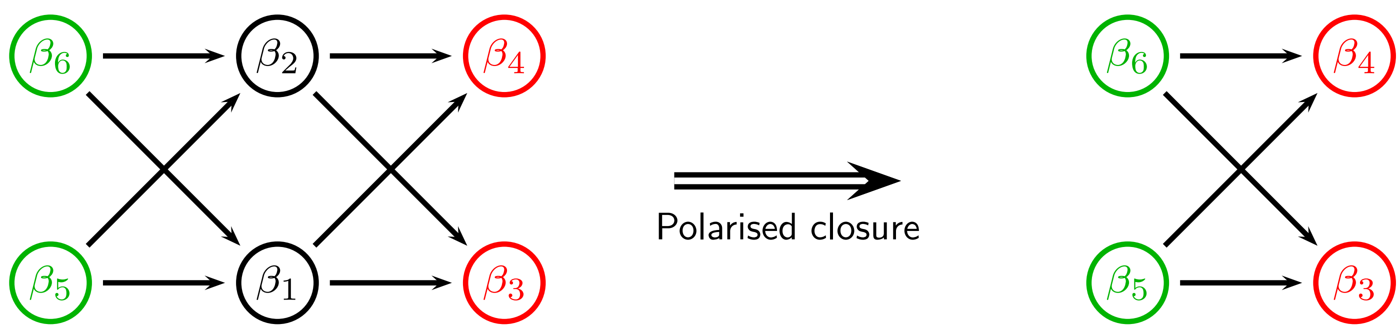
Summary of expansion

- On this example, expansion introduces only 2 variables.
- In the absence of simplification, they would have been 25.
- It remains to simplify the atomic graph.

Example: polarised closure

[Fuh & Mishra (1989)]

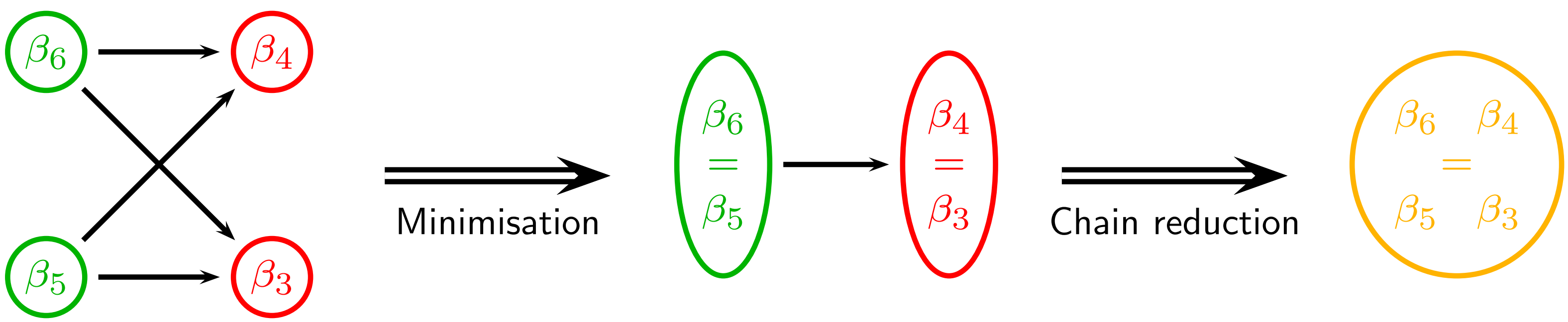
Polarised closure removes non-polar variables in the atomic graph, and keeps only paths from negative variables to positive ones.



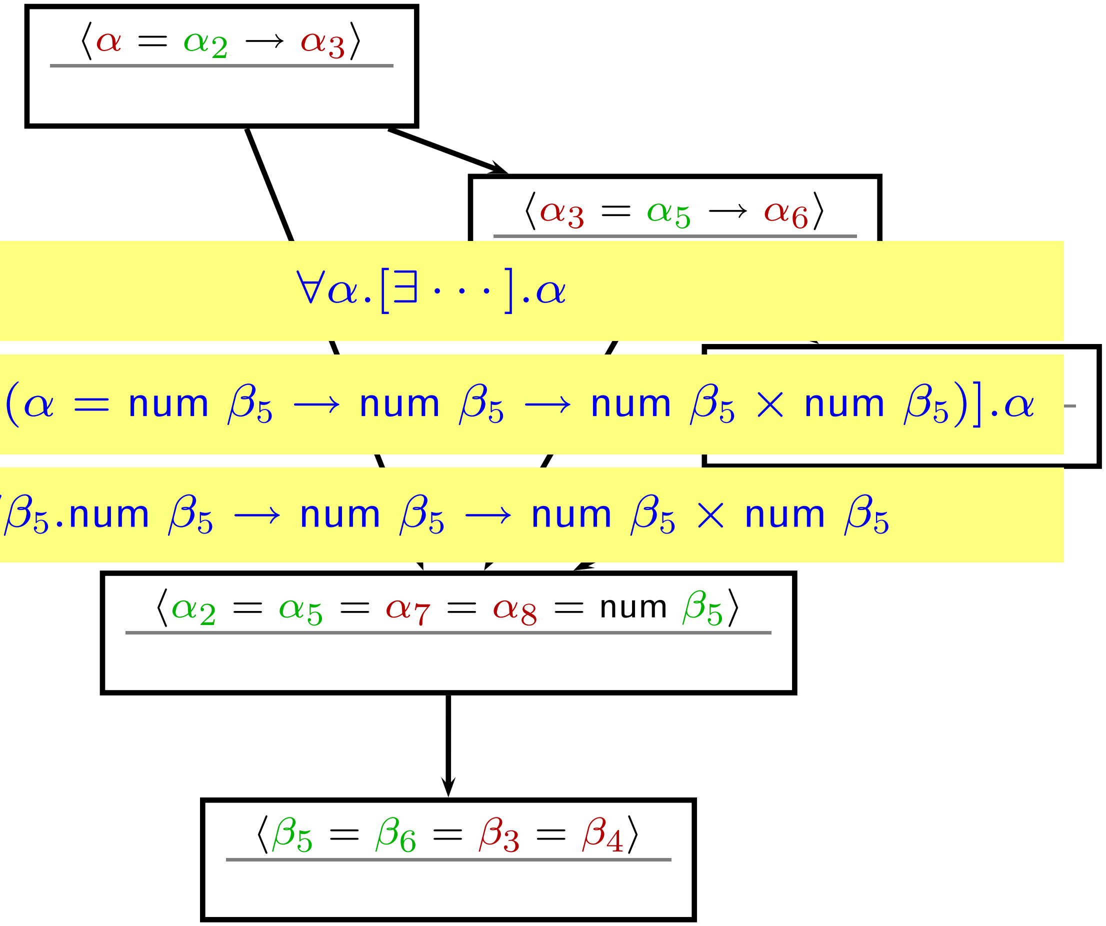
Example: minimisation

[Felleisen & Flanagan (1995), Pottier (1998)]

Two $\frac{\text{negative}}{\text{positive}}$ variables with the same $\frac{\text{successors}}{\text{predecessors}}$ are equivalent



Example: final result



$$\forall \alpha . [\exists \cdots] . \alpha$$

$$\forall \alpha [\exists \beta_5 . (\alpha = \mathsf{num}\ \beta_5 \longrightarrow \mathsf{num}\ \beta_5 \longrightarrow \mathsf{num}\ \beta_5 \times \mathsf{num}\ \beta_5)] . \alpha$$

$$\forall \beta_5 . \mathsf{num}\ \beta_5 \longrightarrow \mathsf{num}\ \beta_5 \longrightarrow \mathsf{num}\ \beta_5 \times \mathsf{num}\ \beta_5$$

Implementation

The Dalton Library

- The Dalton Library is a real-size implementation in Objective Caml of these algorithms.

  `http://cristal.inria.fr/~simonet/soft/dalton/`

- It comes as a functor parametrized by a series of modules describing the client's type system.

- It has been used within the Flow Caml System, an information flow analyzer for the Caml Language

  `http://cristal.inria.fr/~simonet/soft/flowcaml/`

Experimental results

	Typing the code of Caml Light	
	library	compiler
A.s.t. nodes	14002	22996
Unification solver	0.346 s	0.954 s
Structural subtyping solver	0.966 s	2.213 s
/	2.79	2.31

Conclusion

- An efficient implementation type inference engine for structural subtyping
- whose correctness is proved.