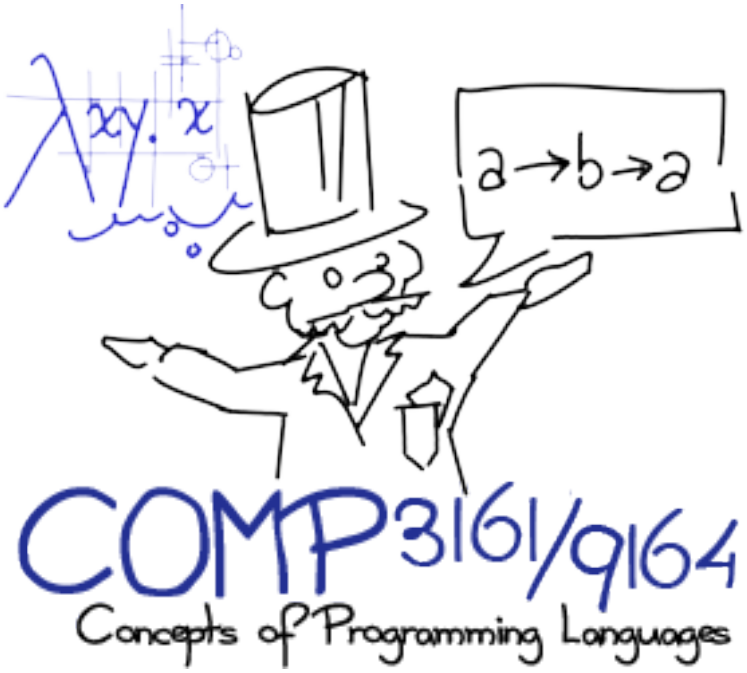

Hindley-Milner Type Inference

Rob Sison
UNSW
Term 3 2025

Implicitly Typed MinHS

Explicitly typed languages are awkward to use[1]. Ideally, we'd like the compiler to determine the types for us.

Example

What is the type of this function?

$$\textbf{recfun}\ f\ x = \text{fst}\ x + 1$$

We want the compiler to infer the most general type.

[1] See Java

Implicitly Typed MinHS

Start with our polymorphic MinHS, then:

- remove type signatures from **recfun**, **let**, etc.
- remove explicit **type** abstractions, and type applications (the @ operator).
- keep $\forall$-quantified types.
- remove recursive types, as we can't infer types for them.

Typing Rules

$$\frac{x : \tau \in \Gamma}{\Gamma \vdash x : \tau}\textsc{Var}$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \to \tau_2 \quad \Gamma \vdash e_2 : \tau_1}{\Gamma \vdash e_1\ e_2 : \tau_2}\textsc{App}$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (\texttt{Pair}\ e_1\ e_2) : \tau_1 \times \tau_2}\textsc{Conj}_I$$

$$\frac{\Gamma \vdash e_1 : \texttt{Bool} \quad \Gamma \vdash e_2 : \tau \quad \Gamma \vdash e_3 : \tau}{\Gamma \vdash (\texttt{If}\ e_1\ e_2\ e_3) : \tau}\textsc{If}$$

Primitive Operators

For convenience, we treat prim ops as functions, and place their types in the environment.

$$(+) : \texttt{Int} \to \texttt{Int} \to \texttt{Int}, \Gamma \vdash (\texttt{App}\,(\texttt{App}\,(+)\,(\texttt{Num}\,2))\,(\texttt{Num}\,1)) : \texttt{Int}$$

Functions

$$\frac{x : \tau_1, f : \tau_1 \to \tau_2, \Gamma \vdash e : \tau_2}{\Gamma \vdash (\texttt{Recfun}\ (f.x.\ e)) : \tau_1 \to \tau_2}\ \textsc{Func}$$

Sum Types

$$\frac{\Gamma \vdash e : \tau_1}{\Gamma \vdash \mathsf{InL}\ e : \tau_1 + \tau_2}\ \textsc{Disj}_{\text{I1}}$$

$$\frac{\Gamma \vdash e : \tau_2}{\Gamma \vdash \mathsf{InR}\ e : \tau_1 + \tau_2}\ \textsc{Disj}_{\text{I2}}$$

Note that we allow the other side of the sum to be any type.

Polymorphism

If we have a polymorphic type, we can instantiate it to any type:

$$\frac{\Gamma \vdash e : \forall a.\tau}{\Gamma \vdash e : \tau[a := \rho]}\,\textsc{All}_{\textsc{E}}$$

We can quantify over any variable that has not already been used.

$$\frac{\Gamma \vdash e : \tau \quad a \notin TV(\Gamma)}{\Gamma \vdash e : \forall a.\ \tau}\,\textsc{All}_{\textsc{I}}$$

(Where $TV(\Gamma)$ here is all type variables occurring free in the types of variables in Γ)

The Goal

We want *an algorithm* for type inference:

- With a clear input and output
- Which *terminates*.
- Which is fully *deterministic*.

Typing Rules

$$\frac{\Gamma \vdash e_1 : \tau_1 \quad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (\texttt{Pair}\ e_1\ e_2) : \tau_1 \times \tau_2}$$

Can we use the existing typing rules as our algorithm?

$$\texttt{infer} :: \texttt{Context} \rightarrow \texttt{Expr} \rightarrow \texttt{Type}$$

This approach can work for monomorphic types, but not polymorphic ones. Why not?

First Problem

$$\frac{\Gamma \vdash e : \forall a.\tau}{\Gamma \vdash e : \tau[a := \rho]}\,\textsc{All}_{\textsc{E}}$$

The rule to add a $\forall$-quantifier can always be applied:

$$\dfrac{\dfrac{\dfrac{\vdots}{\Gamma \vdash (\texttt{Num 5}) : \forall a.\ \forall b.\ \texttt{Int}}\,\textsc{All}_{\textsc{E}}}{\Gamma \vdash (\texttt{Num 5}) : \forall b.\ \texttt{Int}}\,\textsc{All}_{\textsc{E}}}{\Gamma \vdash (\texttt{Num 5}) : \texttt{Int}}\,\textsc{All}_{\textsc{E}}$$

Read as an algorithm, the rules are non-deterministic – there are many possible rules for a given input. A depth-first search strategy may end up attempting infinite derivations.

Another Problem

$$\frac{\Gamma \vdash e : \forall a.\tau}{\Gamma \vdash e : \tau[a := \rho]}\,\textsc{All}_{\textsc{E}}$$

The above rule can be applied at any time to a polymorphic type, even if it would break later typing derivations:

$$\frac{\dfrac{\Gamma \vdash \mathsf{fst} : \forall a.\ \forall b.\ (a \times b) \to a}{\Gamma \vdash \mathsf{fst} : (\mathtt{Bool} \times \mathtt{Bool}) \to \mathtt{Bool}} \quad \dfrac{\cdots}{\Gamma \vdash (\mathtt{Pair\ 1\ True}) : (\mathtt{Int} \times \mathtt{Bool})}}{\Gamma \vdash (\mathtt{Apply\ fst\ (Pair\ 1\ True)}) : ???}$$

We need a way to capture the dependency between the parts of this derivation!

Yet Another Problem

The rule for **recfun** mentions τ_2 in both input and output positions.

$$\frac{x : \tau_1, f : \tau_1 \to \tau_2, \Gamma \vdash e : \tau_2}{\Gamma \vdash (\texttt{Recfun}\ (f.x.\ e)) : \tau_1 \to \tau_2}\ \textsc{Func}$$

In order to infer τ_2 we must provide a context that includes τ_2 — this is circular. Any guess we make for τ_2 could be wrong.

Solution: Unknowns and Unification

We allow types to include *unknowns*, also known as *unification variables* or *schematic variables*. We will call them flexible (type) variables. These are placeholders for types that we haven't worked out yet. We shall use α, β etc. for names of type variables, $\alpha^{\mathtt{F}}$, $\beta^{\mathtt{F}}$ for flexible type variables, and $\alpha^{\mathtt{R}}$, $\beta^{\mathtt{R}}$ for rigid type variables (bound by $\forall$-quantifiers).

Example

$(\mathtt{Int} \times \alpha^{\mathtt{F}}) \to \beta^{\mathtt{F}}$ is the type of a function from tuples where the left side is `Int`, but no other details of the type have been determined yet.

As we encounter situations where two types should be equal, we *unify* the two types to determine what the unknown variables should be.

Unification

Our rules for unification will be specified by the judgement:

$$\Gamma_1 \vdash \tau_1 \sim \tau_2 \Longrightarrow \Gamma_2$$

which are defined such that:

1. Γ_1 and Γ_2 contain the same type variables;
2. Γ_2 is more informative than Γ_1 in the sense that declared type variables have been given definitions in order for $\tau_1 \sim \tau_2$ to hold: $\Gamma_1 \sqsubseteq \Gamma_2$
3. The information increase is minimal (most general) in the sense that it makes the least commitment in order to solve the equation: any other solution $\Gamma_1 \sqsubseteq \Gamma'$ factors through $\Gamma_1 \sqsubseteq \Gamma_2$.

Back to Type Inference

To keep track of the solutions to unification problems in context, we will decompose the typing judgement to allow for an additional output — an updated typing context which represents the *minimal information increase* over the input context (obtained via unification rules!) in order to infer the type of the expression.

Inputs Expression, Context

Outputs Type, Context

We will write this as $\Gamma \vdash e \Longrightarrow \tau \dashv \Gamma$, to make clear what are the inputs and outputs.

One More Concern: Generalisation

$$\frac{\Gamma \vdash e : \tau \quad a \notin TV(\Gamma)}{\Gamma \vdash e : \forall a.\ \tau}\ \textsc{All}_{\textsc{I}}$$

We can generalise a type to a polymorphic type by introducing a $\forall$ at any point. We want to restrict this to only occur in a *syntax-directed* way.
Consider this example:

$$\textbf{let } f = (\textbf{recfun } f\ x = (x, x)) \textbf{ in } (\text{fst } (f\ 4), \text{fst } (f\ \texttt{True}))$$

Where should generalisation happen?

Solution: Let-generalisation

To make type inference tractable, we will generalise only in **let** expressions.

This means that **let** expressions are now not just sugar for a function application. They actually play a vital role, as the place where generalisation happens.

The New Type Inference Judgement

$$\Gamma_1 \vdash e \Longrightarrow \tau \dashv \Gamma_2$$

Purpose Keep track of the solutions to unification problems in context;

Output context represents *minimal information increase* over the input context in order to infer the type of the expression.

Inputs Expression, Context

Outputs Type, Context

Inference Example

$$\textsc{App}\ \frac{?}{\Gamma \vdash \texttt{Apply fst (Pair 1 True)} \Longrightarrow\ ?\ \dashv\ ?}$$

Example Part 1

$$\textsc{Prim}\ \frac{}{\Gamma \vdash \mathsf{fst} \Longrightarrow ? \dashv ?}$$

Example Part 1

$$\textsc{Prim}\ \frac{\mathsf{primOpType}(\mathsf{fst}) = \forall \alpha\, \beta.(\alpha^{\mathtt{R}} \times \beta^{\mathtt{R}}) \to \alpha^{\mathtt{R}}}{\Gamma \vdash \mathsf{fst} \Longrightarrow ? \dashv ?}$$

Example Part 1

$$\textsc{Prim}\ \frac{\mathsf{primOpType}(\mathsf{fst}) = \forall \alpha\, \beta.(\alpha^{\mathtt{R}} \times \beta^{\mathtt{R}}) \to \alpha^{\mathtt{R}}}{\Gamma \vdash \mathsf{fst} \Longrightarrow (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \dashv \Gamma \cdot \alpha \cdot \beta}$$

NB: Must introduce *fresh* names for the flexible type variables (α, β reused here just for convenience!)

Example Part 2

$$\textsc{Prim}\ \frac{\mathsf{primOpType}(\mathsf{fst}) = \forall \alpha\, \beta.(\alpha^{\mathtt{R}} \times \beta^{\mathtt{R}}) \to \alpha^{\mathtt{R}}}{\Gamma \vdash \mathsf{fst} \Longrightarrow (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \dashv \Gamma \cdot \alpha \cdot \beta}$$

$$\Gamma_2 = \Gamma \cdot \alpha \cdot \beta$$

$$\textsc{Prod}\ \frac{\Gamma_2 \vdash 1 \Longrightarrow \mathsf{Int} \dashv \Gamma_2 \quad \Gamma_2 \vdash \mathsf{True} \Longrightarrow \mathsf{Bool} \dashv \Gamma_2 \qquad \Gamma_2 \cdot \rho \vdash \rho^{\mathtt{F}} \sim \mathsf{Int} \times \mathsf{Bool} \Longrightarrow \Gamma_2 \cdot \rho := \mathsf{Int} \times \mathsf{Bool}}{\Gamma_2 \vdash (\mathtt{Pair}\ 1\ \mathsf{True}) \Longrightarrow \rho^{\mathtt{F}} \dashv \Gamma_2 \cdot \rho := \mathsf{Int} \times \mathsf{Bool}}$$

Example Part 3

$$\Gamma_3 = \Gamma_2 \cdot \rho := \mathsf{Int} \times \mathsf{Bool}$$

$$\textsc{App}\ \frac{\Gamma \vdash \mathsf{fst} \Longrightarrow (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \dashv \Gamma_2 \quad \Gamma_2 \vdash (\mathtt{Pair}\ 1\ \mathsf{True}) \Longrightarrow \rho^{\mathtt{F}} \dashv \Gamma_3 \quad \Gamma_3 \cdot \omega \vdash (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \sim \rho^{\mathtt{F}} \to \omega^{\mathtt{F}} \Longrightarrow ?}{\Gamma \vdash \mathtt{Apply}\ \mathsf{fst}\ (\mathtt{Pair}\ 1\ \mathsf{True}) \Longrightarrow ? \dashv ?}$$

Example Part 3

$$\Gamma_3 = \Gamma_2 \cdot \rho := \mathsf{Int} \times \mathsf{Bool}$$

$$\frac{\Gamma_3 \cdot \omega \vdash \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \sim \rho^{\mathtt{F}} \Longrightarrow ? \quad ? \vdash \alpha^{\mathtt{F}} \sim \omega^{\mathtt{F}} \Longrightarrow ?}{\Gamma_3 \cdot \omega \vdash (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \sim \rho^{\mathtt{F}} \to \omega^{\mathtt{F}} \Longrightarrow ?}$$

Example Part 3A

$$
\begin{array}{lcl}
\Gamma_2 & = & \Gamma \cdot \alpha \cdot \beta \\
\Gamma_3 & = & \Gamma_2 \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \\
\Gamma_4 & = & \Gamma_3 \cdot \omega
\end{array}
$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\Gamma_2 \vdash \mathsf{Int} \sim \alpha^{\mathtt{F}} \Longrightarrow ? \quad ? \vdash \mathsf{Bool} \sim \beta^{\mathtt{F}} \Longrightarrow ?}{\Gamma_2 \cdot [\,] \vdash \mathsf{Int} \times \mathsf{Bool} \sim \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \Longrightarrow ?}}{\Gamma_3 \mid [\,] \vdash \rho^{\mathtt{F}} \sim \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \Longrightarrow ?}\ \textsc{Subst}}{\Gamma_4 \mid [\,] \vdash \rho^{\mathtt{F}} \sim \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \Longrightarrow ?}\ \textsc{Skip-Ty}}{\Gamma_4 \vdash \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \sim \rho^{\mathtt{F}} \Longrightarrow ?}\ \textsc{Inst}
$$

Example Part 3A

$$
\begin{aligned}
\Gamma_2 &= \Gamma \cdot \alpha \cdot \beta \\
\Gamma_3 &= \Gamma_2 \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \\
\Gamma_4 &= \Gamma_3 \cdot \omega
\end{aligned}
$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\Gamma_2 \vdash \mathsf{Int} \sim \alpha^{\mathbf{F}} \Longrightarrow ? \quad ? \vdash \mathsf{Bool} \sim \beta^{\mathbf{F}} \Longrightarrow ?}{\Gamma_2 \vdash \mathsf{Int} \times \mathsf{Bool} \sim \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \Longrightarrow ?}}{\Gamma_3 \mid [\,] \vdash \rho^{\mathbf{F}} \sim \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \Longrightarrow ?}}{\Gamma_4 \mid [\,] \vdash \rho^{\mathbf{F}} \sim \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \Longrightarrow ?}}{\Gamma_4 \vdash \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \sim \rho^{\mathbf{F}} \Longrightarrow ?}
$$

Example Part 3A

$$\Gamma_2 = \Gamma \cdot \alpha \cdot \beta \qquad \Gamma_5 = \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta$$
$$\Gamma_3 = \Gamma_2 \cdot \rho := \mathsf{Int} \times \mathsf{Bool}$$
$$\Gamma_4 = \Gamma_3 \cdot \omega$$

$$\dfrac{\dfrac{\dfrac{\dfrac{\Gamma_2 \vdash \mathsf{Int} \sim \alpha^{\mathtt{F}} \Longrightarrow \Gamma_5 \quad ? \vdash \mathsf{Bool} \sim \beta^{\mathtt{F}} \Longrightarrow ?}{\Gamma_2 \vdash \mathsf{Int} \times \mathsf{Bool} \sim \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \Longrightarrow ?}}{\Gamma_3 \mid [\,] \vdash \rho^{\mathtt{F}} \sim \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \Longrightarrow ?}}{\Gamma_4 \mid [\,] \vdash \rho^{\mathtt{F}} \sim \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \Longrightarrow ?}}{\Gamma_4 \vdash \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \sim \rho^{\mathtt{F}} \Longrightarrow ?}$$

Example Part 3A

$$
\begin{array}{lcl}
\Gamma_2 & = & \Gamma \cdot \alpha \cdot \beta \\
\Gamma_3 & = & \Gamma_2 \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \\
\Gamma_4 & = & \Gamma_3 \cdot \omega
\end{array}
\qquad
\begin{array}{lcl}
\Gamma_5 & = & \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta \\
\Gamma_6 & = & \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool}
\end{array}
$$

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\Gamma_2 \vdash \mathsf{Int} \sim \alpha^{\mathtt{F}} \Longrightarrow \Gamma_5 \quad \Gamma_5 \vdash \mathsf{Bool} \sim \beta^{\mathtt{F}} \Longrightarrow \Gamma_6
}{
\Gamma_2 \vdash \mathsf{Int} \times \mathsf{Bool} \sim \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \Longrightarrow \Gamma_6
}
}{
\Gamma_3 \mid [\,] \vdash \rho^{\mathtt{F}} \sim \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \Longrightarrow ?
}
}{
\Gamma_4 \mid [\,] \vdash \rho^{\mathtt{F}} \sim \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \Longrightarrow ?
}
}{
\Gamma_4 \vdash \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \sim \rho^{\mathtt{F}} \Longrightarrow ?
}
$$

Example Part 3A

$$
\begin{array}{lcllcl}
\Gamma_2 & = & \Gamma \cdot \alpha \cdot \beta & \Gamma_5 & = & \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta \\
\Gamma_3 & = & \Gamma_2 \cdot \rho := \mathsf{Int} \times \mathsf{Bool} & \Gamma_6 & = & \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \\
\Gamma_4 & = & \Gamma_3 \cdot \omega & \Gamma_7 & = & \Gamma_6 \cdot \rho := \mathsf{Int} \times \mathsf{Bool}
\end{array}
$$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\Gamma_2 \vdash \mathsf{Int} \sim \alpha^{\mathbf{F}} \Longrightarrow \Gamma_5 \quad \Gamma_5 \vdash \mathsf{Bool} \sim \beta^{\mathbf{F}} \Longrightarrow \Gamma_6}{\Gamma_2 \vdash \mathsf{Int} \times \mathsf{Bool} \sim \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \Longrightarrow \Gamma_6}}{\Gamma_3 \mid [\,] \vdash \rho^{\mathbf{F}} \sim \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \Longrightarrow \Gamma_7}}{\Gamma_4 \mid [\,] \vdash \rho^{\mathbf{F}} \sim \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \Longrightarrow \Gamma_7 \cdot \omega}}{\Gamma_4 \vdash \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \sim \rho^{\mathbf{F}} \Longrightarrow \Gamma_7 \cdot \omega}
$$

Example Part 3

$$
\begin{aligned}
\Gamma_4 &= \Gamma_3 \cdot \omega \\
\Gamma_6 &= \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \\
\Gamma_7 &= \Gamma_6 \cdot \rho := \mathsf{Int} \times \mathsf{Bool}
\end{aligned}
$$

$$
\frac{\Gamma_4 \vdash \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \sim \rho^{\mathbf{F}} \Longrightarrow \Gamma_7 \cdot \omega \quad ? \vdash \alpha^{\mathbf{F}} \sim \omega^{\mathbf{F}} \Longrightarrow ?}{\Gamma_4 \vdash (\alpha^{\mathbf{F}} \times \beta^{\mathbf{F}}) \to \alpha^{\mathbf{F}} \sim \rho^{\mathbf{F}} \to \omega^{\mathbf{F}} \Longrightarrow ?}
$$

Example Part 3B

$$
\begin{aligned}
\Gamma_4 &= \Gamma_3 \cdot \omega \\
\Gamma_6 &= \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \\
\Gamma_7 &= \Gamma_6 \cdot \rho := \mathsf{Int} \times \mathsf{Bool}
\end{aligned}
$$

$$
\frac{\Gamma_4 \vdash \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \sim \rho^{\mathbf{F}} \Longrightarrow \Gamma_7 \cdot \omega \quad \Gamma_7 \cdot \omega \vdash \alpha^{\mathbf{F}} \sim \omega^{\mathbf{F}} \Longrightarrow ?}{\Gamma_4 \vdash (\alpha^{\mathbf{F}} \times \beta^{\mathbf{F}}) \to \alpha^{\mathbf{F}} \sim \rho^{\mathbf{F}} \to \omega^{\mathbf{F}} \Longrightarrow ?}
$$

Example Part 3B

$$
\begin{array}{lcl}
\Gamma_4 & = & \Gamma_3 \cdot \omega \\
\Gamma_6 & = & \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \\
\Gamma_7 & = & \Gamma_6 \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \\
\Gamma_8 & = & \Gamma_7 \cdot \omega := \alpha^{\mathbf{F}}
\end{array}
$$

$$
\frac{\Gamma_4 \vdash \alpha^{\mathbf{F}} \times \beta^{\mathbf{F}} \sim \rho^{\mathbf{F}} \Longrightarrow \Gamma_7 \cdot \omega \quad \Gamma_7 \cdot \omega \vdash \alpha^{\mathbf{F}} \sim \omega^{\mathbf{F}} \Longrightarrow \Gamma_8}{\Gamma_4 \vdash (\alpha^{\mathbf{F}} \times \beta^{\mathbf{F}}) \to \alpha^{\mathbf{F}} \sim \rho^{\mathbf{F}} \to \omega^{\mathbf{F}} \Longrightarrow \Gamma_8}
$$

Example Part 3

$$
\begin{aligned}
\Gamma_4 &= \Gamma_3 \cdot \omega \\
\Gamma_6 &= \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \\
\Gamma_7 &= \Gamma_6 \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \\
\Gamma_8 &= \Gamma_7 \cdot \omega := \alpha^{\mathtt{F}}
\end{aligned}
$$

$$
\frac{\Gamma_4 \vdash \alpha^{\mathtt{F}} \times \beta^{\mathtt{F}} \sim \rho^{\mathtt{F}} \Longrightarrow \Gamma_7 \cdot \omega \quad \Gamma_7 \cdot \omega \vdash \alpha^{\mathtt{F}} \sim \omega^{\mathtt{F}} \Longrightarrow \Gamma_8}{\Gamma_4 \vdash (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \sim \rho^{\mathtt{F}} \to \omega^{\mathtt{F}} \Longrightarrow \Gamma_8}
$$

$$
\Gamma_8 = \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \cdot \omega := \alpha^{\mathtt{F}}
$$

Example

$$
\begin{array}{lcl}
\Gamma_2 & = & \Gamma \cdot \alpha \cdot \beta \\
\Gamma_3 & = & \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \\
\Gamma_8 & = & \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \cdot \omega := \alpha^{\mathtt{F}}
\end{array}
$$

$$
\textsc{App}\ \frac{\Gamma \vdash \mathsf{fst} \Longrightarrow (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \dashv \Gamma_2 \quad \Gamma_2 \vdash (\mathtt{Pair}\ 1\ \mathsf{True}) \Longrightarrow \rho^{\mathtt{F}} \dashv \Gamma_3 \quad \Gamma_3 \cdot \omega \vdash (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \sim \rho^{\mathtt{F}} \to \omega^{\mathtt{F}} \Longrightarrow \Gamma_8}{\Gamma \vdash \mathtt{Apply}\ \mathsf{fst}\ (\mathtt{Pair}\ 1\ \mathsf{True}) \Longrightarrow ? \dashv ?}
$$

Example

$$
\begin{array}{lcl}
\Gamma_2 & = & \Gamma \cdot \alpha \cdot \beta \\
\Gamma_3 & = & \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \\
\Gamma_8 & = & \Gamma \cdot \alpha := \mathsf{Int} \cdot \beta := \mathsf{Bool} \cdot \rho := \mathsf{Int} \times \mathsf{Bool} \cdot \omega := \alpha^{\mathtt{F}}
\end{array}
$$

App

$$
\frac{\begin{array}{c}\Gamma \vdash \mathsf{fst} \Longrightarrow (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \dashv \Gamma_2 \quad \Gamma_2 \vdash (\mathtt{Pair}\ 1\ \mathsf{True}) \Longrightarrow \rho^{\mathtt{F}} \dashv \Gamma_3 \\ \Gamma_3 \cdot \omega \vdash (\alpha^{\mathtt{F}} \times \beta^{\mathtt{F}}) \to \alpha^{\mathtt{F}} \sim \rho^{\mathtt{F}} \to \omega^{\mathtt{F}} \Longrightarrow \Gamma_8\end{array}}{\Gamma \vdash \mathtt{Apply}\ \mathsf{fst}\ (\mathtt{Pair}\ 1\ \mathsf{True}) \Longrightarrow \omega^{\mathtt{F}} \dashv \Gamma_8}
$$

Unification Example

Demo: See Notes on course website after the lecture.

Summary

- We've started examining a variant of algorithm $\mathcal{W}$ (originally due to Damas & Milner, variant thanks to Gundry) for type inference which tracks flexible variables and their instantiations using typing contexts;
- This algorithm is restricted to the Hindley-Milner subset of decidable polymorphic instantiations, and requires that polymorphism is top-level — polymorphic functions are not first class;
- The rest of the rules will be given in the specification for Assignment 2 — out soon.