Hinting
A nice presentation of algorithmic typing

Wojciech Kołowski

STLC with Hints

STLC with Hints is a flavour of STLC inspired by Bidirectional STLC. The main insight behind it is that in Bidirectional STLC, we have a hard time deciding whether a rule should be in checking mode or in inference mode, so why not both? This way, we would have some input type that guides us, but also produce an output type, which is in some sense "better". This is a bit silly if we already have the correct type as input, but we can make this idea work by introducing hints, which are types with holes, and insisting that the input is not a type, but merely a hint.

Hints

$$H ::= \mathbf{?} \mid H_1 \to H_2 \mid H_1 \times H_2 \mid H_1 + H_2 \mid \mathbf{1} \mid \mathbf{0}$$

Intuitively, hints are partial types. They are built like types, except that there's one additional constructor, **?**, which can be read as "hole" or "unknown".

We use the letter H for hints. When we use letters like A, B, C which usually stand in for types, it means that the hint **is** a type, i.e. it doesn't contain any **?**s.

Order on hints

$$\frac{}{\mathbf{?} \sqsubseteq H}$$

$$\frac{H_1 \sqsubseteq H_1' \quad H_2 \sqsubseteq H_2'}{H_1 \to H_2 \sqsubseteq H_1' \to H_2'}$$

$$\frac{H_1 \sqsubseteq H_1' \quad H_2 \sqsubseteq H_2'}{H_1 \times H_2 \sqsubseteq H_1' \times H_2'}$$

$$\frac{H_1 \sqsubseteq H_1' \quad H_2 \sqsubseteq H_2'}{H_1 + H_2 \sqsubseteq H_1' + H_2'}$$

$$\frac{}{\mathbf{1} \sqsubseteq \mathbf{1}} \qquad \frac{}{\mathbf{0} \sqsubseteq \mathbf{0}}$$

Order on hints – intuition

The order can be intuitively interpreted as information increase: $H_1 \sqsubseteq H_2$ means that hint H_2 is more informative than H_1, but in a compatible way. In other words, H_1 and H_2 have the same structure, but some **?**s from H_1 were possibly refined to something more informative in H_2.

Order on hints – properties

$\sqsubseteq$ is a partial order with a least element.

- Reflexivity: $H \sqsubseteq H$
- Transitivity: $H_1 \sqsubseteq H_2 \implies H_2 \sqsubseteq H_3 \implies H_1 \sqsubseteq H_3$
- Weak antisymmetry: $H_1 \sqsubseteq H_2 \implies H_2 \sqsubseteq H_1 \implies H_1 = H_2$
- Least element: $\mathbf{?} \sqsubseteq H$

Least upper bound of hints

$$
\begin{aligned}
\mathbf{?} \sqcup H &= H \\
H \sqcup \mathbf{?} &= H \\
(H_1 \to H_2) \sqcup (H_1' \to H_2') &= (H_1 \sqcup H_1') \to (H_2 \sqcup H_2') \\
(H_1 \times H_2) \sqcup (H_1' \times H_2') &= (H_1 \sqcup H_1') \times (H_2 \sqcup H_2') \\
(H_1 + H_2) \sqcup (H_1' + H_2') &= (H_1 \sqcup H_1') + (H_2 \sqcup H_2') \\
\mathbf{1} \sqcup \mathbf{1} &= \mathbf{1} \\
\mathbf{0} \sqcup \mathbf{0} &= \mathbf{0}
\end{aligned}
$$

The order on hints induces a partial operation $\sqcup$, which computes the least upper bound of two hints when it exists. Intuitively, $\sqcup$ combines two hints which share the same structure, filling the $\mathbf{?}$s in the leaves with something more informative coming from the other argument. For hints with incompatible structure the result is undefined.

Least upper bound of hints – properties

If all relevant results are defined, then:

- $(H_1 \sqcup H_2) \sqcup H_3 = H_1 \sqcup (H_2 \sqcup H_3)$
- $H_1 \sqcup H_2 = H_2 \sqcup H_1$
- $\mathbf{?} \sqcup H = H = H \sqcup \mathbf{?}$
- $H \sqcup H = H$

If $\sqcup$ were not partial, $(H, \sqcup, \mathbf{?})$ would be a commutative idempotent monoid. But since it is partial, meh...

Greatest lower bound of hints

$$
\begin{aligned}
\mathbf{?} \sqcap H &= \mathbf{?} \\
H \sqcap \mathbf{?} &= \mathbf{?} \\
(H_1 \to H_2) \sqcap (H_1' \to H_2') &= (H_1 \sqcap H_1') \to (H_2 \sqcap H_2') \\
(H_1 \times H_2) \sqcap (H_1' \times H_2') &= (H_1 \sqcap H_1') \times (H_2 \sqcap H_2') \\
(H_1 + H_2) \sqcap (H_1' + H_2') &= (H_1 \sqcap H_1') + (H_2 \sqcap H_2') \\
\mathbf{1} \sqcap \mathbf{1} &= \mathbf{1} \\
\mathbf{0} \sqcap \mathbf{0} &= \mathbf{0}
\end{aligned}
$$

The order on hints induces a partial operation $\sqcap$, which computes the greatest lower bound of two hints when it exists. Intuitively, $\sqcap$ combines two hints which share the same structure, replacing any subhints with the $\mathbf{?}$s if it appears in the other argument. For hints with incompatible structure the result is undefined.

Greatest lower bound of hints – properties

If all relevant results are defined, then:

- $(H_1 \sqcap H_2) \sqcap H_3 = H_1 \sqcap (H_2 \sqcap H_3)$
- $H_1 \sqcap H_2 = H_2 \sqcap H_1$
- $\mathbf{?} \sqcap H = \mathbf{?} = H \sqcap \mathbf{?}$
- $H \sqcap H = H$

If $\sqcap$ were not partial, $(H, \sqcap, \mathbf{?})$ would be a commutative idempotent monoid. But since it is partial, meh...

Hint subtraction

$$
\begin{gathered}
\mathbf{?} \setminus H = \mathbf{?} \\
H \setminus \mathbf{?} = H \\
H \setminus H = \mathbf{?} \\
(H_1 \to H_2) \setminus (H_1' \to H_2') = (H_1 \setminus H_1') \to (H_2 \setminus H_2') \\
(H_1 \times H_2) \setminus (H_1' \times H_2') = (H_1 \setminus H_1') \times (H_2 \setminus H_2') \\
(H_1 + H_2) \setminus (H_1' + H_2') = (H_1 \setminus H_1') + (H_2 \setminus H_2')
\end{gathered}
$$

Hint subtraction is a partial operation which we'll need when proving minimality. When subtracting a hint from itself or from **?**, the result is **?**, and subtracting **?** changes nothing. In the remaining cases, when the hints are not equal but have the same structure, the subtraction proceeds recursively.

Hint subtraction – properties

- If $H_1 \sqsubseteq H_2$, then $H_1 \setminus H \sqsubseteq H_2 \setminus H$

Information order on terms

The order on hints induces an order on terms: it is the smallest relation which preserves term constructors and subsumes hint ordering on annotated terms.

$$\frac{e_1 \sqsubseteq e_2 \quad H_1 \sqsubseteq H_2}{(e_1 : H_1) \sqsubseteq (e_2 : H_2)}$$

Intuitively, $e_1 \sqsubseteq e_2$ holds when e_2 has more informative hints than e_1.

Information order on terms – rules

$$\frac{}{x \sqsubseteq x} \qquad \frac{e_1 \sqsubseteq e_2}{\lambda x.\, e_1 \sqsubseteq \lambda x.\, e_2} \qquad \frac{f_1 \sqsubseteq f_2 \quad a_1 \sqsubseteq a_2}{f_1\ a_1 \sqsubseteq f_2\ a_2}$$

$$\frac{a_1 \sqsubseteq a_2 \quad b_1 \sqsubseteq b_2}{(a_1, b_1) \sqsubseteq (a_2, b_2)} \qquad \frac{e_1 \sqsubseteq e_2}{\texttt{outl}\ e_1 \sqsubseteq \texttt{outl}\ e_2} \qquad \frac{e_1 \sqsubseteq e_2}{\texttt{outl}\ e_1 \sqsubseteq \texttt{outr}\ e_2}$$

$$\frac{e_1 \sqsubseteq e_2}{\texttt{inl}\ e_1 \sqsubseteq \texttt{inl}\ e_2} \qquad \frac{e_1 \sqsubseteq e_2}{\texttt{inr}\ e_1 \sqsubseteq \texttt{inr}\ e_2}$$

$$\frac{e_1 \sqsubseteq e_2 \quad f_1 \sqsubseteq f_2 \quad g_1 \sqsubseteq g_2}{\texttt{case}\ e_1\ \texttt{of}\ (f_1, g_1) \sqsubseteq \texttt{case}\ e_2\ \texttt{of}\ (f_2, g_2)}$$

$$\frac{}{\texttt{unit} \sqsubseteq \texttt{unit}} \qquad \frac{e_1 \sqsubseteq e_2}{\mathbf{0}\texttt{-elim}\ e_1 \sqsubseteq \mathbf{0}\texttt{-elim}\ e_2}$$

Terms

Terms:

$$
\begin{aligned}
e ::= \\
& x \mid (e : H) \mid \\
& \lambda x.\, e \mid e_1\ e_2 \mid \\
& (e_1, e_2) \mid \texttt{outl}\ e \mid \texttt{outr}\ e \mid \\
& \texttt{inl}\ e \mid \texttt{inr}\ e \mid \texttt{case}\ e\ \texttt{of}\ (e_1, e_2) \mid \\
& \texttt{unit} \mid \mathbf{0}\texttt{-elim}\ e
\end{aligned}
$$

Note: red color marks differences from Bidirectional STLC.

Judgements:

$\Gamma \vdash e \Leftarrow H \Rightarrow A$ – in context Γ, term e checks with hint H and infers type A

Declarative typing – differences

$$\frac{\Gamma \vdash e : A \quad H \sqsubseteq A}{\Gamma \vdash (e : H) : A}\textsc{Annot}$$

Algorithmic typing – basic rules

$$\frac{(x : A) \in \Gamma \quad H \sqsubseteq A}{\Gamma \vdash x \Leftarrow H \Rightarrow A}\textsc{Var}$$

$$\frac{\Gamma \vdash e \Leftarrow H_1 \sqcup H_2 \Rightarrow A}{\Gamma \vdash (e : H_1) \Leftarrow H_2 \Rightarrow A}\textsc{Annot}$$

Algorithmic typing – type-directed rules

$$\frac{\Gamma, x : A \vdash e \Leftarrow H \Rightarrow B}{\Gamma \vdash \lambda x.\, e \Leftarrow A \to H \Rightarrow A \to B}$$

$$\frac{\Gamma \vdash f \Leftarrow \mathbf{?} \to H \Rightarrow A \to B \quad \Gamma \vdash a \Leftarrow A \Rightarrow A}{\Gamma \vdash f\ a \Leftarrow H \Rightarrow B}$$

$$\frac{H \sqcup \mathbf{?} \times \mathbf{?} = H_A \times H_B \quad \Gamma \vdash a \Leftarrow H_A \Rightarrow A \quad \Gamma \vdash b \Leftarrow H_B \Rightarrow B}{\Gamma \vdash (a, b) \Leftarrow H \Rightarrow A \times B}$$

$$\frac{\Gamma \vdash e \Leftarrow H \times \mathbf{?} \Rightarrow A \times B}{\Gamma \vdash \mathtt{outl}\ e \Leftarrow H \Rightarrow A} \qquad \frac{\Gamma \vdash e \Leftarrow \mathbf{?} \times H \Rightarrow A \times B}{\Gamma \vdash \mathtt{outr}\ e \Leftarrow H \Rightarrow B}$$

Algorithmic typing – type-directed rules

$$\frac{\Gamma \vdash e \Leftarrow H \Rightarrow A}{\Gamma \vdash \texttt{inl}\ e \Leftarrow H + B \Rightarrow A + B}$$

$$\frac{\Gamma \vdash e \Leftarrow H \Rightarrow B}{\Gamma \vdash \texttt{inr}\ e \Leftarrow A + H \Rightarrow A + B}$$

$$\frac{\Gamma \vdash e \Leftarrow \mathbf{?} + \mathbf{?} \Rightarrow A + B \qquad \begin{array}{c}\Gamma \vdash f \Leftarrow A \to H \Rightarrow A \to C \\ \Gamma \vdash g \Leftarrow B \to C \Rightarrow B \to C\end{array}}{\Gamma \vdash \mathsf{case}\ e\ \mathsf{of}\ (f, g) \Leftarrow H \Rightarrow C}$$

$$\frac{H \sqsubseteq \mathbf{1}}{\Gamma \vdash \texttt{unit} \Leftarrow H \Rightarrow \mathbf{1}} \qquad \frac{\Gamma \vdash e \Leftarrow \mathbf{0} \Rightarrow \mathbf{0}}{\Gamma \vdash \mathbf{0}\texttt{-elim}\ e \Leftarrow A \Rightarrow A}$$

Algorithmic typing – alternative rules

$$\frac{\Gamma \vdash a \Leftarrow \mathbf{?} \Rightarrow A \quad \Gamma \vdash f \Leftarrow A \to H \Rightarrow A \to B}{\Gamma \vdash f\ a \Leftarrow H \Rightarrow B}\ \textsc{AltApp}$$

$$\frac{\begin{array}{c}\Gamma \vdash f \Leftarrow \mathbf{?} \to H \Rightarrow A \to C \\ \Gamma \vdash g \Leftarrow \mathbf{?} \to C \Rightarrow B \to C\end{array} \quad \Gamma \vdash e \Leftarrow A + B \Rightarrow A + B}{\Gamma \vdash \mathsf{case}\ e\ \mathsf{of}\ (f, g) \Leftarrow H \Rightarrow C}\ \textsc{AltCase}$$

$$\frac{\Gamma \vdash e \Leftarrow \mathbf{?} + \mathbf{?} \Rightarrow A + B \quad \begin{array}{c}\Gamma \vdash f \Leftarrow A \to H \Rightarrow A \to C_1 \\ \Gamma \vdash g \Leftarrow B \to H \Rightarrow B \to C_2\end{array}}{\Gamma \vdash \mathsf{case}\ e\ \mathsf{of}\ (f, g) \Leftarrow H \Rightarrow C_1 \sqcup C_2}\ \textsc{SymCase}$$

Algorithmic typing – discussion of alternative rules

We could have made some different choices.

- For application, we could try to infer the argument type first and then feed it to the function as a hint.
- For case, we could try to infer domains of the branches first, then feed these as hints when checking the discriminee. This is what the rule AltCase does.
- But there's also a third possibility for case: we could treat the branches more symmetrically, checking both with the input hint H, and only combining the output types C_1 and C_2 at the end. This is what the rule SymCase does.

Metatheory – basics

If $\Gamma \vdash e \Leftarrow H \Rightarrow A$ then:

- (Soundness) $\Gamma \vdash e : A$ (proof: induction)
- (Compatibility) $H \sqsubseteq A$ (proof: induction)
- (Squeeze) If $H \sqsubseteq H'$ and $H' \sqsubseteq A$, then $\Gamma \vdash e \Leftarrow H' \Rightarrow A$ (proof: induction)
- (Determinism) If $\Gamma \vdash e \Leftarrow H \Rightarrow B$, then $A = B$.

(Decidability) For Γ, e, H it is decidable whether there exists A such that $\Gamma \vdash e \Leftarrow H \Rightarrow A$ (proof: the rules are literally the algorithm)

Metatheory – greatest lower bound

If $\Gamma \vdash e \Leftarrow H_1 \Rightarrow A$ and $\Gamma \vdash e \Leftarrow H_2 \Rightarrow A$, then $\Gamma \vdash e \Leftarrow H_1 \sqcap H_2 \Rightarrow A$ (TODO: not proven, probably not true; fails for App, but works for AltApp; also fails for OutL and OutR)

Metatheory – minimality

(Minimality) There exists $H' \sqsubseteq H$ such that $\Gamma \vdash e \Leftarrow H' \Rightarrow A$ and for all $H'' \sqsubset H'$ it is not the case that $\Gamma \vdash e \Leftarrow H'' \Rightarrow A$ (proof: induction, the only hard case is Annot)

Proof: we need $\Gamma \vdash (e : H_1) \Leftarrow H' \Rightarrow A$ for minimal H'. From the induction hypothesis we have minimal $H' \sqsubseteq H_1 \sqcup H_2$ such that $\Gamma \vdash e \Leftarrow H' \Rightarrow A$. Our minimal hint will be $H' \setminus H_1$, so we need $\Gamma \vdash (e : H_1) \Leftarrow H' \setminus H_1 \Rightarrow A$. Since $H_1 \sqcup (H' \setminus H_1) = H_1 \sqcup H'$, it suffices to show $\Gamma \vdash e \Leftarrow H_1 \sqcup H' \Rightarrow A$, which follows from squeezing and IH, because $H' \sqsubseteq H_1 \sqcup H' \sqsubseteq H_1 \sqcup H_2$. Of course we also have $H' \setminus H_1 \sqsubseteq H_2 \setminus H_1 \sqsubseteq H_2$.

Metatheory – minimality, cont.

Now assume $H'' \sqsubset H' \setminus H_1$ and $\Gamma \vdash (e : H_1) \Leftarrow H'' \Rightarrow A$. We have $H'' \sqsubset H'$ and so $H'' \sqsubset H_2$.

TODO

Metatheory – minimality v2

(Minimality v2) There exists $M \sqsubseteq H$ such that $\Gamma \vdash e \Leftarrow M \Rightarrow A$ and for all $M' \sqsubseteq M$ such that $\Gamma \vdash e \Leftarrow M' \Rightarrow A$ we have $M' = M$ (proof: TODO)

Metatheory 2

If $\Gamma \vdash e : A$, then:

- (Annotability, checking) There exists e' such that $e \sqsubseteq e'$ and $\Gamma \vdash e' \Leftarrow A \Rightarrow A$ (proof: induction; annotations need to be put on eliminators, like in Dual Intrinsic STLC)
- (Annotability, inference) There exists e' such that $e \sqsubseteq e'$ and $\Gamma \vdash e' \Leftarrow \mathbf{?} \Rightarrow A$ (proof: induction; annotations need to be put on constructors, like in Intrinsic STLC)
- (Minimal annotability) There exists e' such that $e \sqsubseteq e'$ and $\Gamma \vdash e' \Leftarrow A \Rightarrow A$ and for all e'' such that $e \sqsubseteq e''$ and $e'' \sqsubseteq e'$ it is not the case that $\Gamma \vdash e'' \Leftarrow A \Rightarrow A$
- There exist $e \sqsubseteq e_1$ and $H_1 \sqsubseteq A$ such that $\Gamma \vdash e_1 \Leftarrow H_1 \Rightarrow A$ and for all $e \sqsubseteq e_2 \sqsubseteq e_1$ and $H \sqsubseteq H' \sqsubseteq A$ it is not the case that $\Gamma \vdash e_2 \Leftarrow H_2 \Rightarrow A$

(Non)uniqueness of typing

Similarly to Extrinsic STLC, STLC with Hints does not enjoy uniqueness of typing. This is because we still can have terms like $\lambda x.\, x$ with hint **?**, which can be typed with any type of the form $A \to A$. However, if the hint is informative enough, then the type is unique. Moreover, every typable term can be given a hint which makes its type unique.

Embedding Bidirectional STLC

We will now turn our attention towards showing that Bidirectional STLC can be in some sense "embedded" into our hint-based system.

First, checking and inference modes are definable:

- $\Gamma \vdash e \Leftarrow A$ is defined as $\Gamma \vdash e \Leftarrow A \Rightarrow A$
- $\Gamma \vdash e \Rightarrow A$ is defined as $\Gamma \vdash e \Leftarrow \mathbf{?} \Rightarrow A$

Second, we will show that bidirectional typing rules are admissible in the hint-based system.

Embedding Bidirectional STLC – variables and annotations

$$\frac{(x : A) \in \Gamma}{\Gamma \vdash x \Rightarrow A}\textsc{Var}$$

$$\frac{\Gamma \vdash e \Leftarrow A}{\Gamma \vdash (e : A) \Rightarrow A}\textsc{Annot}$$

Thanks to our definitions, the bidirectional rules Var and Annot are special cases of our hint-based Var and Annot.

Embedding Bidirectional STLC – subsumption

$$\frac{\Gamma \vdash e \Rightarrow B \quad A = B}{\Gamma \vdash e \Leftarrow A}\textsc{Sub}$$

Subsumption is a bit harder to prove, but it boils down to our squeeze theorem. After rewriting $A = B$, we have $\Gamma \vdash e \Leftarrow \mathbf{?} \Rightarrow A$ and we need to show $\Gamma \vdash e \Leftarrow A \Rightarrow A$, but this follows from squeeze theorem because $\mathbf{?} \sqsubseteq A \sqsubseteq A$.

Embedding Bidirectional STLC – checked rules

$$\frac{\Gamma, x : A \vdash e \Leftarrow B}{\Gamma \vdash \lambda x.\, e \Leftarrow A \to B}$$

$$\frac{\Gamma \vdash a \Leftarrow A \quad \Gamma \vdash b \Leftarrow B}{\Gamma \vdash (a, b) \Leftarrow A \times B}$$

$$\frac{\Gamma \vdash e \Leftarrow A}{\Gamma \vdash \mathtt{inl}\ e \Leftarrow A + B} \qquad \frac{\Gamma \vdash e \Leftarrow B}{\Gamma \vdash \mathtt{inr}\ e \Leftarrow A + B}$$

$$\frac{}{\Gamma \vdash \mathtt{unit} \Leftarrow \mathbf{1}} \qquad \frac{\Gamma \vdash e \Leftarrow \mathbf{0}}{\Gamma \vdash \mathbf{0}\text{-}\mathtt{elim}\ e \Leftarrow A}$$

Checked rules are special cases of the corresponding hint-based rules.

Embedding Bidirectional STLC – rules with inference

$$\frac{\Gamma \vdash f \Rightarrow A \to B \quad \Gamma \vdash a \Leftarrow A}{\Gamma \vdash f\ a \Rightarrow B}$$

$$\frac{\Gamma \vdash e \Rightarrow A \times B}{\Gamma \vdash \texttt{outl}\ e \Rightarrow A} \qquad \frac{\Gamma \vdash e \Rightarrow A \times B}{\Gamma \vdash \texttt{outr}\ e \Rightarrow B}$$

$$\frac{\Gamma \vdash e \Rightarrow A + B \quad \Gamma \vdash f \Leftarrow A \to C \quad \Gamma \vdash g \Leftarrow B \to C}{\Gamma \vdash \mathtt{case}\ e\ \mathtt{of}\ (f, g) \Leftarrow C}$$

To prove rules which make use of inference mode, we need to use the squeeze theorem. For example, assume $\Gamma \vdash f \Leftarrow \mathbf{?} \Rightarrow A \to B$ and $\Gamma \vdash a \Leftarrow A \Rightarrow A$. From the squeeze theorem, we also have $\Gamma \vdash f \Leftarrow \mathbf{?} \to \mathbf{?} \Rightarrow A \to B$, so that $\Gamma \vdash f\ a \Leftarrow \mathbf{?} \Rightarrow B$ follows from hint-based rule App.

Embedding Bidirectional STLC – additional rules

$$\frac{\Gamma \vdash e \Rightarrow A + B \quad \Gamma \vdash f \Rightarrow A \to C \quad \Gamma \vdash g \Rightarrow B \to C}{\Gamma \vdash \texttt{case}\ e\ \texttt{of}\ (f, g) \Rightarrow C}$$

$$\frac{}{\Gamma \vdash \texttt{unit} \Rightarrow \mathbf{1}}$$

$$\frac{\Gamma \vdash a \Rightarrow A \quad \Gamma \vdash b \Rightarrow B}{\Gamma \vdash (a, b) \Rightarrow A \times B}$$

The inferred rule for case also requires using the squeeze theorem, but this time we need to do it three times, once for each premise. As for the inferred rules for unit and pairs, they follow from the corresponding hint-based rules.

Embedding Intrinsic STLC – terms

We can also embed Intrinsic STLC. First, we define the missing terms using partial annotations:

- $\lambda x : A.\, e :\equiv (\lambda x.\, e : A \to \mathbf{?})$
- $\texttt{inl}_B\ e :\equiv (\texttt{inl}\ e : \mathbf{?} + B)$
- $\texttt{inr}_A\ e :\equiv (\texttt{inr}\ e : A + \mathbf{?})$
- $\mathbf{0}\texttt{-elim}_A\ e :\equiv (\mathbf{0}\texttt{-elim}\ e : A)$

Second, we will show that the typing rules are admissible in the hint-based system.

Embedding Intrinsic STLC – rules

$$\frac{\Gamma, x : A \vdash e \Rightarrow B}{\Gamma \vdash \lambda x : A.\, e \Rightarrow A \to B}$$

$$\frac{\Gamma \vdash e \Rightarrow A}{\Gamma \vdash \texttt{inl}_B\ e \Rightarrow A + B} \qquad \frac{\Gamma \vdash e \Rightarrow B}{\Gamma \vdash \texttt{inr}_A\ e \Rightarrow A + B}$$

$$\frac{\Gamma \vdash e \Rightarrow \mathbf{0}}{\Gamma \vdash \mathbf{0}\texttt{-elim}_A\ e \Rightarrow A}$$

Rules for lambdas and sum constructors can be derived by first using the rule for annotations and then the corresponding hint-based rule. For $\mathbf{0}$-`elim` we also need to use the squeeze theorem on the premise.

Embedding Dual Intrinsic STLC – terms

We can also embed Dual Intrinsic STLC. First, we define the missing terms using partial annotations:

- $\mathtt{app}_A\ f\ a :\equiv (f : A \to \mathbf{?})\ a$
- $\mathtt{outl}_B\ e :\equiv \mathtt{outl}\ (e : \mathbf{?} \times B)$
- $\mathtt{outr}_A\ e :\equiv \mathtt{outr}\ (e : A \times \mathbf{?})$
- $\mathtt{case}_{A,B}\ e\ \mathtt{of}\ (f, g) :\equiv \mathtt{case}\ (e : A + B)\ \mathtt{of}\ (f, g)$

Second, we will show that the typing rules are admissible in the hint-based system.

Embedding Dual Intrinsic STLC – rules

$$\frac{\Gamma \vdash f \Leftarrow A \to B \quad \Gamma \vdash a \Leftarrow A}{\Gamma \vdash \mathrm{app}_A\ f\ a \Leftarrow B}$$

$$\frac{\Gamma \vdash e \Leftarrow A \times B}{\Gamma \vdash \mathtt{outl}_B\ e \Leftarrow A} \qquad \frac{\Gamma \vdash e \Leftarrow A \times B}{\Gamma \vdash \mathtt{outr}_A\ e \Leftarrow B}$$

$$\frac{\Gamma \vdash e \Leftarrow A + B \quad \Gamma \vdash f \Leftarrow A \to C \quad \Gamma \vdash g \Leftarrow B \to C}{\Gamma \vdash \mathrm{case}_{A,B}\ e\ \mathrm{of}\ (f, g) \Leftarrow C}$$

To prove the rules, we first need to use the corresponding hint-based rule and then the annotation rule in one of the premises.