

Judgment

Analytic judgments are those that become evident by conceptual analysis.

$$\frac{J_1 \quad \dots \quad J_n}{J_c}$$

1. **Syntax:** well-formed

E.g.

$$\frac{}{T \text{ wf}} \quad \frac{A \text{ wf} \quad B \text{ wf}}{A \circ B \text{ wf}}$$

2. **Semantics:** true

1. **I:** *introduce* a connective
2. **E:** *extract* info out of a pure connective

E.g.

$$\frac{A \quad B}{A \wedge B} \wedge I \quad \frac{A \wedge B}{A} \wedge E_1$$

Hypothetical Reasoning

Given P , Q holds.

1. $\Rightarrow I$: Internalizing a reasoning process.

E.g.

$$\frac{\frac{P_1 \quad P_2}{\dots}}{P_1, P_2 \Rightarrow Q}$$

Note that P s and Q s cease to exist in the conclusion, thus “internalized”.

1. $\Rightarrow E$: modus ponens

Properties of assumptions (structural)

Weakening: assumptions can be unused

$$\frac{\frac{\frac{}{A} u \quad (\text{unused}) \frac{}{B} w}{B \Rightarrow A} \Rightarrow I^w}{A \Rightarrow B \Rightarrow A} \Rightarrow I^u$$

Contraction: assumptions can be used multiple times

Substitution: TODO

Context

Notice the two dimensional rule of $\Rightarrow I$:

$$\frac{\frac{P_1 \quad P_2}{\dots}}{Q}$$

$$P_1, P_2 \Rightarrow Q$$

It's kind of awkward to keep this hypothetical structure around. Instead, we write

$$\frac{\Gamma, h_1 : P_1, h_2 : P_2 \vdash Q}{\Gamma \vdash P_1, P_2 \Rightarrow Q}$$

Note that we declare

$$\frac{x : A \in \Gamma}{\Gamma \vdash A} x$$

By following the same principle, we can rewrite **Weakening** and **Contraction**.

Weakening:

$$\frac{\Gamma \vdash B}{\Gamma, x : A \vdash B}$$

Contraction:

$$\frac{\Gamma, x : A \vdash B}{\Gamma, x : A, y : A \vdash B}$$

Substitution:

$$\frac{\Gamma, x : A \vdash C \quad \Gamma \vdash A}{\Gamma \vdash C}$$

Local Soundness/Completeness

Local in the sense that these properties only discuss a **single connective**, not the whole system. It's a weak witness that this system makes sense.

1. **Local Soundness:** the combination of intro and elim is not too strong, i.e. they don't allow us to infer more than what's already known.

E.g. the proof

$$\frac{\frac{\Gamma \vdash A \quad \Gamma \vdash B}{\Gamma \vdash A \times B} \times I}{\Gamma \vdash A} \times E_1$$

collapses to

$$\frac{\Gamma \vdash A}{\Gamma \vdash A}$$

so there's no additional information provided by the intro and elim rules.

2. **Local Completeness:** the combination of intro and elim is sufficiently strong, in terms of rebuilding the info we have.

E.g. there's no information loss in the rebuild process of the connective $A \times B$

$$\frac{\frac{\Gamma \vdash A \times B}{\Gamma \vdash A} \times E_1 \quad \frac{\Gamma \vdash A \times B}{\Gamma \vdash B} \times E_2}{\Gamma \vdash A \times B} \times I$$

C.H.

- Propositions - Types
- Proof - Programs
- Nat. Ded. - λ -calculus

Importance:

1. Guiding program language design
2. Basis of Type Theory
3. Proving logic consistency by looking at programs

Typing Judgment

$$\boxed{\Gamma \vdash M : A}$$

Given some assumptions in the context Γ ,

- M is a proof term corresponding to the proposition A .
- M is a program that has a type A .

C.H.: $\Gamma \vdash A$ iff $\Gamma \vdash M : A$

Example: conjunction

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N \rangle : A \wedge B} \wedge I \quad \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash M.1 : A} \quad \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash M.2 : B}$$

Example: implication

$$\frac{\Gamma, x : A \vdash \lambda x : A. M : B}{\Gamma \vdash M : A \rightarrow B} \Rightarrow I^x \quad \frac{\Gamma \vdash M : A \rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash MN : B} \Rightarrow E$$

Thm (Local Soundness)

$$\frac{\Gamma \vdash M : A \quad \Gamma \vdash N : B}{\Gamma \vdash \langle M, N \rangle : A \wedge B} \quad \frac{\Gamma \vdash \langle M, N \rangle : A \wedge B}{\Gamma \vdash \langle M, N \rangle.1 : A} \text{ collapses to } \frac{}{\Gamma \vdash M : A}$$

$$\frac{\Gamma, x : A \vdash M : B}{\Gamma \vdash \lambda x : A. M : A \rightarrow B} \Rightarrow I^x \quad \frac{\Gamma \vdash \lambda x : A. M : A \rightarrow B \quad \Gamma \vdash N : A}{\Gamma \vdash (\lambda x : A. M)N : B} \Rightarrow E \text{ collapses to } \frac{}{\Gamma \vdash M[N/x] : B}$$

1. **Logic:** not gaining any information through this intro and elim detour.
2. **Program:** type is **preserved** through this detour.

The “collapse” can then be written as:

Reduction $\langle M, N \rangle.1 \Rightarrow M$ and $(\lambda x : A. M)N \Rightarrow M[N/x]$.

Thm (Subject Reduction)

If $\Gamma \vdash M : A$ and $M \Rightarrow M'$, then $\Gamma \vdash M' : A$

Thm (Local Completeness)

$$\begin{array}{c}
 \frac{}{\Gamma \vdash M : A \wedge B \text{ expands to}} \quad \frac{\frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash M.1 : A} \wedge E_1 \quad \frac{\Gamma \vdash M : A \wedge B}{\Gamma \vdash M.2 : B} \wedge E_2}{\Gamma \vdash \langle M.1, M.2 \rangle : A \wedge B} \wedge I \\
 \\
 \frac{}{\Gamma \vdash M : A \rightarrow B \text{ } (\eta)\text{-expands to}} \quad \frac{\frac{\Gamma \vdash M : A \rightarrow B}{\Gamma, x : A \vdash M : A \rightarrow B} \quad \frac{\Gamma, x : A \vdash x : A}{\Gamma, x : A \vdash Mx : B} \Rightarrow E}{\Gamma \vdash \lambda x : A. Mx : A \rightarrow B} \Rightarrow I^x
 \end{array}$$

This **exhibits the actual structure** (i.e. **expands its internal term/proof structure**) of M while preserving the type.

Insight:

- Proof Reduction - Program Reduction
- Normalizing Proofs - Normalizing Programs

Compare both proofs

TODO fill in the blanks

$$\frac{}{\vdash \lambda x : A \wedge A. (\lambda y : A. y)x.2 : A \wedge A \rightarrow A}$$

Disjunction

Let's now expand the previous example to disjunction.

Rules

$$\begin{array}{c}
 \frac{\Gamma \vdash M : A}{\Gamma \vdash \iota_1 M : A \vee B} \vee I_1 \quad \frac{\Gamma \vdash M : B}{\Gamma \vdash \iota_2 M : A \vee B} \vee I_2 \\
 \\
 \frac{\Gamma \vdash M : A \vee B \quad \Gamma, x : A \vdash N_1 : C \quad \Gamma, x : B \vdash N_2 : C}{\Gamma \vdash \text{cases } M \text{ of } \iota_1 x \Rightarrow N_1 \mid \iota_2 x \Rightarrow N_2 : C} \vee E
 \end{array}$$

Local Soundness

TODO Fill in the blanks

$$\frac{}{\Gamma \vdash \text{case}}$$

Modal Logic S4

Truth is living in the moment - here and now.

Validity is living forever and everywhere.

— Brigitte Pientka

“The world is full of possibilities, but not today.”

Necessity Modality $\Box A$

> Note what I originally wrote as $\Gamma \vdash A$ is now $\Gamma \vdash A$ true to make a clear distinction between **validity** and **true**.

Validity

- If $\varepsilon \vdash A$ true then A valid
- If A valid then $\Gamma \vdash A$ true

Notice how the second rule allows weakening while the first rule does not. This means that **validity does not depend on any local assumptions**.

Judgment

$\Delta; \Gamma \vdash A$ true

- Δ - the **global** context. It contains valid assumptions that holds **forever**.
- Γ - the **local** context. It contains assumptions that hold **here and now**.

So, the definition of validity can be written as:

$$\frac{y : A \text{ true} \in \Gamma}{\Delta; \Gamma \vdash A \text{ true}} \quad \frac{x : A \text{ valid} \in \Delta}{\Delta; \Gamma \vdash A \text{ true}}$$

Now we start to introduce the modality \Box :

$$\frac{\Delta; \cdot \vdash A \text{ true}}{\Delta; \Gamma \vdash \Box A \text{ true}} \Box I$$

Just as \rightarrow is internalizing reasoning on implication, \Box is internalizing reasoning on validity.

T law (*reflexivity*).

$$\frac{\cdot; x : \Box A \vdash A \text{ true}}{\cdot; \cdot \vdash \Box A \rightarrow A \text{ true}} \rightarrow I^x$$

i.e. If it's true everywhere and forever, then it's also true here and now.

A detour

One may be tempted to define \Box_E as such

$$\frac{\Delta; \Gamma \vdash \Box A}{\Delta; \Gamma \vdash A}$$

Let's see if it's locally complete as a sanity check.

$$\overline{\Delta; \Gamma \vdash \Box A \text{ true}}$$

should be able to expand to

$$\frac{\Delta; \Gamma \vdash \Box A \text{ true}}{\frac{\frac{\dots}{\Delta; \cdot \vdash A \text{ true}} \text{ ???}}{\Delta; \Gamma \vdash \Box A \text{ true}} \Box I}$$

Boom!

The correct solution

A let-style definition!

$$\frac{\Delta; \Gamma \vdash \Box A \text{ true} \quad \Delta, u : A \text{ valid}; \Gamma \vdash C \text{ true}}{\Delta; \Gamma \vdash C \text{ true}} \Box E \qquad \frac{\Delta; \cdot \vdash A \text{ true}}{\Delta; \Gamma \vdash \Box A \text{ true}} \Box I$$

Let's see if it's locally sound.

$$\frac{\frac{\Delta; \cdot \vdash A \text{ true}}{\Delta; \Gamma \vdash \Box A \text{ true}} \Box I \quad \Delta, u : A \text{ valid}; \Gamma \vdash C \text{ true}}{\Delta; \Gamma \vdash C \text{ true}} \Box E^u$$

($\Delta; \cdot \vdash A \text{ true}$ goes to $\Delta, u : A \text{ valid}; \Gamma \vdash C \text{ true}$ via *modal substitution*)

collapses to

$$\Delta; \Gamma \vdash C \text{ true}$$

Lemma.

1. (*Substitution*) If $\Delta; \Gamma, x : A \text{ true} \vdash C \text{ true}$ and $\Delta; \Gamma \vdash A \text{ true}$, then $\Delta; \Gamma \vdash C \text{ true}$.
2. (*Modal Substitution*) If $\Delta, y : A \text{ valid}; \Gamma \vdash C \text{ true}$ and $\Delta; \cdot \vdash A \text{ true}$, then $\Delta; \Gamma \vdash C \text{ true}$.

Thm (T). \Box satisfies *reflexivity*.

-

Thm (Distributivity). \Box satisfies *distributivity*.

$$\Box(A \rightarrow B) \rightarrow \Box A \rightarrow \Box B$$

Thm (4). \Box satisfies *transitivity*.

$$\Box A \rightarrow \Box \Box A$$

To introduce a \Box , one may attempt to use \Box_I , but this would leave us with an empty Γ which is not good. So, we need to first preserve x forever, which leads us to the use of \Box_E - saving it to Δ .

NOTE We usually put something to eternality via \Box_E rule. Note that we are gaining information via \Box_E while we give up information via \Box_I .

$$\frac{\frac{\frac{\frac{}{u}}{u : A \text{ valid}; \cdot \vdash A \text{ true}}{\Box I} \quad u : A \text{ valid}; \cdot \vdash \Box A \text{ true}}{\Box I} \quad \frac{\cdot; x : \Box A \text{ true} \vdash \Box A \text{ true} \quad u : A \text{ valid}; x : \Box A \text{ true} \vdash \Box \Box A \text{ true}}{\Box E^u}}{\cdot; x : \Box A \text{ true} \vdash \Box \Box A \text{ true}} \rightarrow I^x \quad \cdot; \cdot \vdash \Box A \rightarrow \Box \Box A \text{ true}$$

Syntax

Terms $M ::= x \mid \lambda x : A. M \mid M N \mid \langle M, N \rangle \mid M.1 \mid M.2 \mid u \mid \text{box } M \mid$

NOTE u for valid assumptions.

Now let's add proof terms to logic rules.

$$\frac{\Delta; \cdot \vdash M : A \text{ true}}{\Delta; \Gamma \vdash \text{box } M : \Box A \text{ true}} \Box I \text{ (} M \text{ is a closed term w.r.t. local (runtime) assumptions)}$$

$$\frac{\Delta; \Gamma \vdash M : \Box A \text{ true} \quad \Delta, u : A \text{ valid}; \Gamma \vdash N : C \text{ true}}{\Delta; \Gamma \vdash \text{let box } u := M \text{ in } N : C \text{ true}} \Box E \quad \frac{x : A \text{ true} \in \Gamma}{\Delta; \Gamma \vdash x : A \text{ true}}$$

$$\frac{u : A \text{ valid} \in \Delta}{\Delta; \Gamma \vdash u : A \text{ true}}$$

And now we have

1. (Substitution) If $\Delta; \Gamma, x : A \text{ true} \vdash C \text{ true}$ and $\Delta; \Gamma \vdash A \text{ true}$, then $\Delta; \Gamma \vdash C \text{ true}$. e.g.

$$(\text{box } N)[M/x] = \text{box } N$$

2. (Modal Substitution) If $\Delta, u : A \text{ valid}; \Gamma \vdash C \text{ true}$ and $\Delta; \cdot \vdash A \text{ true}$, then $\Delta; \Gamma \vdash C \text{ true}$. e.g.

$$(\text{box } N)[[M/u]] = \text{box } (N[[M/u]])$$

Now let's try to rephrase *locally soundness*

$$\frac{\frac{\Delta; \cdot \vdash M : A \text{ true}}{\Delta; \Gamma \vdash \text{box } M : \Box A \text{ true}} \Box I \quad \Delta, u : A \text{ valid}; \Gamma \vdash N : C \text{ true}}{\Delta; \Gamma \vdash \text{let box } u := \text{box } M \text{ in } N : C \text{ true}} \Box E^u$$

collapses to

$$\Delta; \Gamma \vdash N[[M/u]] : C$$

Real Programming Example

```
1 nth: int -> (bool_vec -> bool)
```

[Haskell](#)

This function does not do anything if you only pass it an integer. It just sits there, not producing anything meaningful.

How to avoid this situation and force it generate a real function?

```
1 nth: int -> □(bool_vec -> bool)
2 nth 0 = box (fun v -> hd v)
3 nth (S n) =
4   let box r = nth n in box (fun v -> r (tl v))
```

[Haskell](#)

In this case, box makes sure the function generated does not depend on int.

Compare

```
1 nth 1
2 = fun v -> tl (nth 0 v)
3 = fun v -> tl (hd v)
```

[Haskell](#)

with

```
1 nth 1
2 = let box r = nth 0 in box (fun v -> r (tl v))
3 = let box r = box (fun v -> hd v) in (fun v -> r (tl v))
```

[Haskell](#)

```
4 = box (fun v -> (fun v0 -> hd v0) (tl v))
```

Notice how the returned function is not a closure over n .

However, if you compare these two functions you still find the latter one not satisfying cuz it's returning a redex and it's in a box so it get stuck.

Contextual types to the rescue!

Contextual types

Previously, we wrote $\Box A$ to mean A starts with an empty context, which is not sufficient in many cases. So instead, let's allow specifying a context Γ for A .

Examples

Cooking metaphor

1. Add eggs, flour, sugar
2. Add \Box (a liquid)

To type \Box , it's eggs, flour, sugar \vdash liquid

Theorem prover

Holes in programs:

$\text{fun } x \rightarrow \Box +_{\text{int}} x$, you can see the hole here accepts an $x : \text{int} \vdash \text{int}$

Or $\lambda x. \lambda y. \Box y. 2 : (A \rightarrow B \rightarrow C) \rightarrow (A \times B) \rightarrow C$, where the hole accepts an $x : A \rightarrow B \rightarrow C, y : A \times B \vdash B \rightarrow C$

Syntax

Types $A ::= \dots \mid \Box(\psi \vdash A)$
 Terms $M ::= \dots \mid \text{box } (\psi.M)$
 Contexts $\Gamma, \psi ::= \dots$

E.g. $\text{box } (x : \text{int}. x + x) : \Box(x : \text{int} \vdash \text{int})$

NOTE But how to keep this thing stable under renaming?

$$\frac{\Delta; \psi \vdash M : A \text{ true}}{\Delta; \Gamma \vdash \text{box } (\psi.M) : \Box(\psi \vdash A) \text{ true}} \Box I$$

$$\frac{\Delta; \Gamma \vdash M : \Box(\psi \vdash A) \text{ true} \quad \Delta, u : A \text{ valid}; \Gamma \vdash N : C \text{ true}}{\Delta; \Gamma \vdash \text{let box } u := M \text{ in } N : C \text{ true}} \Box E \quad \frac{x : A \text{ true} \in \Gamma}{\Delta; \Gamma \vdash x : A \text{ true}}$$

$$\frac{u : \psi \vdash A \text{ valid} \in \Delta \quad \Delta, \Gamma \vdash \sigma : \psi}{\Delta; \Gamma \vdash \text{clo}(u, \sigma) : A \text{ true}}$$

Notes

- σ - substitution from ψ to Δ, Γ i.e.

$$\frac{\Delta; \Gamma \vdash \sigma : \psi \quad \Delta; \Gamma \vdash M : A}{\Delta; \Gamma \vdash (\sigma, M/x) : \psi, x : A}$$

- $\text{clo}(u, \sigma)$ - delayed substitution σ that can be applied once u is available.

NOTE Computation rules for `clo`

Recall how we have

$$(\text{box } N)[M/x] = \text{box } N$$

$$(\text{box } N)\llbracket M/u \rrbracket = \text{box } (N\llbracket M/u \rrbracket)$$

Now also,

$$\text{clo}(u, \sigma)\llbracket \psi.M/u \rrbracket = M[\sigma]$$

Beware that $M[\sigma]$ is a **local** substitution.


E.g.

$$\lambda x. \text{let box } u := x \text{ in box } (\lambda y. \lambda z. u \ y) : \Box(C \rightarrow A) \rightarrow \Box(C \rightarrow D \rightarrow A)$$

$$\lambda x. \text{let box } u := x \text{ in box}(y : C, z : D. \text{clo}(u, y/x')) : \Box(x' : C \vdash A) \rightarrow \Box(y : C, z : D \vdash A)$$


With this, we can revise our `nth` example

```
1 nth: int -> □(bool_vec -> bool)
2 nth 0 = box (fun v -> hd v)
3 nth (S n) =
4   let box r =
5     nth n in box (fun v -> r (tl v))
```

 Haskell


into this

```
1 nth: int -> □(v: bool_vec ⊢ int)
2 nth 0 = box (v: bool_vec. hd v)
3 nth (S n) =
4   let box u = nth n in
5     box (v: bool_vec. clo(u, (tl v)/v))
```

 Haskell


then we make

```
1 nth 1
2 = let box r = nth 0 in box (fun v -> r (tl v))
3 = let box r = box (fun v -> hd v) in (fun v -> r (tl v))
4 = box (fun v -> (fun v0 -> hd v0) (tl v))
```

 Haskell

into this

```
1 nth 1
2 = let box u = nth 0 in
3   box (v: bool_vec. clo(u, (tl v)/v))
4 = let box u = box (v: bool_vec. hd v) in
5   box (v: bool_vec. clo(u, (tl v)/v))
6 = box (v: bool_vec. clo(hd v0, (tl v)/v0))
7 = box (v: bool_vec. hd (tl v))
```

 Haskell

Notice how the nested evaluation is eager.

TODO What's the difference between functions and `clo`?