Contextual Modal Type Theory — Brigitte Pientka

Lecture 4 - *June 26, 2025*

1 Notations

As discussed in the previous lecture, the following notations are used:

- $\Box A$: A is necessarily true / A is valid
- Δ : context of 'global' valid assumptions, 'live forever', $\{A_1 : \mathsf{valid}, \ldots, A_n : \mathsf{valid}\}$
- Γ : context of 'local' true assumptions, 'live here and now', $\{A_1 : \mathsf{true}, \ldots, A_n : \mathsf{true}\}$

2 Contextual Types

2.1 Intro to contextual types

- We'll examine a common example from natural deduction:

$$x : A \supset B \supset C, y : A \wedge B \vdash \boxed{} : B \supset C$$

- Usually, we would like to fill the hole by applying $\mathsf{proj}_1\, B$ to A to get $B \supset C$
- However, $\boxed{}$ here is not necessarily closed
- The idea of contextual type is to pair the context (here is $x : A \supset B \supset C, y : A \wedge B$) with the conclusion (here is $B \supset C$)[1]
- Think about when we type check the following:

$$\frac{x : \mathsf{int} \vdash \boxed{} + 1 : \mathsf{int}}{\cdot \vdash \lambda x.\boxed{} + 1 : \mathsf{int}} \tag{1}$$

- We know that $\boxed{}$ stands for int in the context of $x : \mathsf{int}$

Types/Props $A := \ldots \mid \Box(\Psi \Vdash A)$
We can read $\Psi \Vdash A$ as "pair the context Ψ with the conclusion A".

Terms $M := \ldots \mid \mathsf{box}\ (\Psi, M)$

Some possible inhabitants of $\boxed{}$ given example 1:

- 0
  - A note: the term $\mathsf{box}\ (x : \mathsf{int}, 0)$ has type $\Box(x : \mathsf{int} \Vdash \mathsf{int})$
- x (because we already have that $x : \mathsf{int}$)
- $x * x$

Takeaway: we have a typing derivation that we haven't yet finished, represented by our $\boxed{}$ s. Thus we refer to the conclusion to infer the contextual type.

2.2 Contextual rule rewrites

$$\frac{\Delta; \Psi \vdash M : A}{\Delta; \Gamma \vdash \mathsf{box}\ (\Psi, M) : \Box(\Psi \Vdash A)} \quad (\Box I) \qquad \frac{\Delta; \Gamma \vdash M : \Box(\Psi \Vdash A) \quad (\Delta, u : (\Psi \Vdash A)); \Gamma \vdash N}{\Delta; \Gamma \vdash \mathsf{let\ box}\ u = M\ \mathsf{in}\ N : C} \quad (\Box E)$$

$$\frac{x : A\mathsf{true} \in \Gamma}{\Delta; \Gamma \vdash x : A} \qquad \frac{u : (\Psi \Vdash A) \in \Delta \quad \Delta; \Gamma \vdash \sigma : \Psi}{\Delta; \Gamma \vdash \mathsf{clo}\ (u, \sigma) : A}$$

We define the relation between Ψ and Γ by providing witnesses:
in equation $\Delta; \Gamma \vdash \sigma : \Psi$,

- $\sigma = M_1/x_1, \ldots, M_n/x_n$
- $\Psi = x_1 : A_1\mathsf{true}, \ldots, x_n : A_n\mathsf{true}$

where $\Delta; \Gamma \vdash M_i : A_i$.

In more formal terms,

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

In essence, Ψ is the domain, Γ is the range, and σ is a mapping from $\Psi \longrightarrow \Gamma$. We provide instantiations for all $x \in \Psi$, then make sure that all of these instantiations make sense in Γ.

Intuitively, doesn't it make sense that $\Box(x : A, y : B \Vdash C)$ is "equivalent" to $\Box(A \supset B \supset C)$? A small derivation:

$$\dfrac{\dfrac{\dfrac{\Delta; x : A, y : B \vdash \ldots C}{\Delta; x : A \vdash \ldots B \supset C}}{\Delta; \cdot \vdash \ldots : A \supset B \supset C}}{\Delta; \Gamma \vdash \ldots : \Box A \supset B \supset C}$$

3 Comparing contextual and non-contextual types

3.1 Example 1: implication chains

Type: $\Box(C \supset A) \supset \Box(C \supset D \supset A)$
Term: $\lambda x : \Box(C \supset A).\mathsf{let\ box}\ u = x\ \mathsf{in\ box}\ (\lambda y : C.\lambda x : D.\ u\ y)$ (where $u = C \supset A$ valid)

We can see that our box holds instructions followed by a function application: u applied to y.

Now using contextual types:
Type: $\Box(x' : C \Vdash A) \supset \Box(y : C, x : D \Vdash A)$
Term:
$\lambda x : \Box(x' : C \Vdash A).\ \mathsf{let\ box}\ u = x\ \mathsf{in\ box}\ (y : C, x : D.\ \mathsf{clo}\ (u, y/x'))$ (where $u = (x' : C \Vdash A)$)

In this example, we turn x's into ys using a closure (clo) instead of a function application. Closures fire much sooner than function applications, which we'll explore more in the following example.

3.2 Example 2: nth function

Rewriting our nth function from the previous lecture, we have type signature

$$\mathsf{nth} : \mathsf{int} \longrightarrow \Box(v : \mathsf{bool\_vec} \Vdash \mathsf{bool})$$

where

$$\begin{aligned} \mathsf{nth}\ 0 &= \mathsf{box}\ (v : \mathsf{bool\_vec}, \mathsf{hd}\ v) \\ \mathsf{nth}\ (\mathsf{suc}\ n) &= \mathsf{let\ box}\ u = \mathsf{nth}\ n\ \mathsf{in\ box}\ (v : \mathsf{bool\_vec}, \mathsf{clo}\ (u, \mathsf{tl}\ v)). \end{aligned}$$

Let's evaluate an example, nth 2:

$$\begin{aligned} \mathsf{nth}\ 2 &\longrightarrow \mathsf{let\ box}\ u = \mathsf{nth}\ 1\ \mathsf{in\ box}\ (v : \mathit{bool\_vec}, \mathsf{clo}\ (u, \mathsf{tl}\ v/v_1)) \\ \mathsf{nth}\ 1 &\longrightarrow \mathsf{let\ box}\ u = \mathsf{nth}\ 0\ \mathsf{in\ box}\ (v : \mathit{bool\_vec}, \mathsf{clo}\ (u, \mathsf{tl}\ v/v_0)) \\ &\longrightarrow \mathsf{let\ box}\ u = \mathsf{box}\ (v_0, \mathsf{hd}\ v_0)\ \mathsf{in\ box}\ (v : \mathit{bool\_vec}, \mathsf{clo}\ (u, \mathsf{tl}\ v/v_0)) \\ &\longrightarrow \mathsf{box}\ (v, \mathsf{clo}\ (\mathsf{hd}\ v_0, \mathsf{tl}\ v/v_0)) \\ &\longrightarrow \mathsf{box}\ (v.\mathsf{hd}(\mathsf{tl}\ v)) \end{aligned}$$

We can see that our contextual-typed version is eager, thus giving us the expected answer that our lazy function-evaluated nth function failed to show in the previous lecture. Functions evaluate once they have all of the information; contextual types treat arguments like syntax and splice until they get the most simplified result.

- Substitution for "local" variables: $[M/x]x = M$
- Modal substitution for global variables: $[\![(\Psi.M)/u]\!]\mathsf{clo}\ (u, \sigma) = [\sigma]M$

One example of where this is used is in simplifying expressions like box (fn $x \to x + (x + 0)$). We want our program to continue to analyze this expression, resulting in the simpler box (fn $x \to x+x$). However, the program can't *evaluate* because we don't have a value of x provided. Thus, we use syntax manipulation - $(x + 0)$ is itself a contextual type.

4 Another view: "quote and splice"

In the Kripke-like "quote and splice" view, we have expressions like

$$\Gamma_1; \ldots ; \Gamma_n \vdash M : A$$

and rules like

$$\frac{\overrightarrow{\Gamma}; \Gamma; \cdot \vdash M : A}{\overrightarrow{\Gamma}; \Gamma \vdash \Box M : \Box A} \qquad \frac{\overrightarrow{\Gamma}; \Gamma \vdash M : \Box A}{\overrightarrow{\Gamma}; \Gamma_i; \ldots ; \Gamma_n \vdash \mathsf{unbox}_n M : A}$$

The two views are proven to be equivalent. The Kripke view is more akin to code, but the contextually-typed view is more well-behaved.

References

[1] Aleksandar Nanevski, Frank Pfenning, and Brigitte Pientka. "Contextual modal type theory". In: *ACM Trans. Comput. Logic* 9.3 (June 2008). ISSN: 1529-3785. DOI: 10.1145/1352582.1352591. URL: https://doi.org/10.1145/1352582.1352591.