Type Theory Study Group

meeting 3

Hypothetical Judgements & Statics

Speakers:

- Danny Gratzer
- Mark Farrel

Notes by:

Ilan Godik

③ ① Type theory meeting

Hypothetical Judgements

Rules that state that a judgement holds when other judgements hold.

Derivability:

$\boxed{\Gamma \vdash \varphi}$

φ is derivable from the rules R together with the judgements from Γ (as axioms!)

$\Gamma = \{J_1, J_2, \ldots, J_n\}$

$$\cfrac{\cfrac{}{J_3} \qquad \begin{array}{c}\cfrac{\cfrac{}{J_1} \qquad \cfrac{}{J_2}}{}\,R_1 \\ \vdots \\ J_{n+1}\end{array}}{\varphi}\,R_2$$

③ ② Type Theory meeting

Structural Properties of the Derivability Judgement

1. Reflexivity: $\Gamma, J \vdash J$

(can derive any judgement from itself)

$$\overline{J}$$

2. Weakening: Can add judgements.

$$\frac{\Gamma \vdash J}{\Gamma, J' \vdash J}$$

(under Γ)

$$\begin{array}{c}\Gamma\\ \vdots \\ \hline J\end{array}$$

(under Γ, ignore J')

$$\begin{array}{c}\Gamma\\ \vdots \\ \hline J\end{array}$$

3. Substitution

* Stability under extension

(If the moon is blue, then I have cofee on my desk)

⇒ Can't derive this judgement.

* Can't look inside the judement.

3 3 Type Theory meeting

Addmisibility

* Can look into the judgements, and manipulate the derivation of the jugements.
* Isn't closed under extension
* Can exhaustively examine the rules and derive facts from that.

$\underline{\Gamma \vDash J}$

Given a derivation of Γ, (from R) we can construct a derivation of J

Relates to computational type theory, based on admisibility.

Gives more power to prove things.

Theorems we prove with derivability will usually be less interesting.

We'll skip General Hypothetical Judgements.
(Too many meta-theory can get a little boring)

③ ④ Type Theory meeting

Statics

* 2 main stages for programing languages:
  - Statics, analyse the program
  - Dynamics, running the programs.

In dependantly typed languages, it's hard to seperate the stages.

The book gives a simple example of a language.

* We add things to the context Γ only in let bindings:
  we only consider the free variable's names and their types;
  the expression under let will stay defined under any other expression with the same type bound the same name.

Uniqueness of types:

- Doesn't hold under subtyping for example.

③⑤ Type Theory meeting

Inversion for Typing

if $\Gamma \vdash e : \tau$, and we examine
(Admissibility) 'e' to see $e = plus(e_1, e_2)$
then the following must be true:

- $e : num$
- $e_1 : num$
- $e_2 : num$

Twelf

Nice for formalizing inductive systems
like we have here.

- Has weakening by default.
- An implementation of LF - The Edinburgh Logical Framework.
- Doesn't have admisibility

③⑥ Type Theory meeting

Exercise 4.1

$e \downarrow \tau$ Analytic mode: Checking if a judgement is derivable.
$e \uparrow \tau$ Synthetic mode: Find a unique appropriate type τ.

We've been developing the mathematical tools
we will use in the rest of the book,
haven't gotten to the meat yet.

Next time:
Chapters 5&6.
Dynamics & the coherence of the statics & the dynamics.