pty.1 Recovering Derivations from Proof Terms

int:pty:tp:sec

Now let us consider the other direction: translating terms back to natural deduction trees. We will use still use the double refutation of the excluded middle as example, and let S denote this term, i.e.,

$$\lambda y^{(\varphi\vee(\varphi\to\bot))\to\bot}.\, y(\mathrm{in}_2^{\varphi}(\lambda x^{\varphi}.\, y\mathrm{in}_1^{\varphi\to\bot}(x))) : ((\varphi \vee (\varphi \to \bot)) \to \bot) \to \bot$$

For each natural deduction rule, the term in the conclusion is always formed by wrapping some operator around the terms assigned to the premise(s). Rules correspond uniquely to such operators. For example, from the structure of the S we infer that the last rule applied must be $\to$Intro, since it is of the form $\lambda y^{\cdots}.\ \ldots$, and the λ operator corresponds to $\to$Intro. In general we can recover the skeleton of the derivation solely by the structure of the term, e.g.,

$$
{}^{2}\dfrac{\dfrac{[y :]^2 \qquad \dfrac{{}^{1}\dfrac{\dfrac{[y :]^2 \qquad \dfrac{[x]^1}{\mathrm{in}_1^{\varphi\to\bot}(x) :}\,\vee\mathrm{Intro}_1}{y(\mathrm{in}_1^{\varphi\to\bot}(x)) :}\,\to\mathrm{Elim}}{\lambda x^{\varphi}.\, y(\mathrm{in}_1^{\varphi\to\bot}(x)) :}\,\to\mathrm{Intro}}{\mathrm{in}_2^{\varphi}(\lambda x^{\varphi}.\, y\mathrm{in}_1^{\varphi\to\bot}(x)) :}\,\vee\mathrm{Intro}_2}{y(\mathrm{in}_2^{\varphi}(\lambda x^{\varphi}.\, y\mathrm{in}_1^{\varphi\to\bot}(x))) :}\,\to\mathrm{Elim}}{\lambda y^{(\varphi\vee(\varphi\to\bot))\to\bot}.\, y(\mathrm{in}_2^{\varphi}(\lambda x^{\varphi}.\, y(\mathrm{in}_1^{\varphi\to\bot}(x)))) :}\,\to\mathrm{Intro}
$$

Our next step is to recover the formulas these terms witness. We define a function $F(\Gamma, M)$ which denotes the formula witnessed by M in context Γ, by induction on M as follows:

$$
\begin{aligned}
F(\Gamma, x) &= \Gamma(x) \\
F(\Gamma, \langle N_1, N_2\rangle) &= F(\Gamma, N_1) \wedge F(\Gamma, N_2) \\
F(\Gamma, \mathrm{p}_i(N)) &= \varphi_i \text{ if } F(\Gamma, N) = \varphi_1 \wedge \varphi_2 \\
F(\Gamma, \mathrm{in}_i^{\varphi}(N)) &= \begin{cases} F(N) \vee \varphi & \text{if } i = 1 \\ \varphi \vee F(N) & \text{if } i = 2 \end{cases} \\
F(\Gamma, \mathrm{case}(M, x_1.N_1, x_2.N_2)) &= F(\Gamma \cup \{x_i : F(\Gamma, M)\}, N_i) \\
F(\Gamma, \lambda x^{\varphi}.\, N) &= \varphi \to F(\Gamma \cup \{x : \varphi\}, N) \\
F(\Gamma, NM) &= \psi \text{ if } F(\Gamma, N) = \varphi \to \psi
\end{aligned}
$$

where $\Gamma(x)$ means the formula mapped to by x in Γ and $\Gamma \cup \{x : \varphi\}$ is a context exactly as Γ except mapping x to φ, whether or not x is already in Γ.

Note there are cases where $F(\Gamma, M)$ is not defined, for example:

1. In the first line, it is possible that x is not in Γ.
2. In recursive cases, the inner invocation may be undefined, making the outer one undefined too.

3. In the third line, its only defined when $F(\Gamma, M)$ is of the form $\varphi_1 \lor \varphi_2$, and the right hand is independent on i.

As we recursively compute $F(\Gamma, M)$, we work our way up the natural deduction derivation. The every step in the computation of $F(\Gamma, M)$ corresponds to a term in the derivation to which the derivation-to-term translation assigns M, and the formula computed is the end-formula of the derivation. However, the result may not be defined for some choices of Γ. We say that such pairs $\langle \Gamma, M\rangle$ are *ill-typed*, and otherwise *well-typed*. However, if the term M results from translating a derivation, and the formulas in Γ correspond to the undischarged assumptions of the derivation, the pair $\langle \Gamma, M\rangle$ will be well-typed.

Proposition pty.1. *If D is a derivation with undischarged assumptions φ_1, …, φ_n, M is the proof term associated with D and $\Gamma = \{x_1 : \varphi_1, \dots, x_n : \varphi_n\}$, then the result of recovering derivation from M in context Γ is D.*

In the other direction, if we first translate a typing pair to natural deduction and then translate it back, we won't get the same pair back since the choice of variables for the undischarged assumptions is underdetermined. For example, consider the pair $\langle \{x : \varphi, y : \varphi \to \psi\}, yx\rangle$. The corresponding derivation is

$$\dfrac{\varphi \to \psi \qquad \varphi}{\psi} \to\text{Elim}$$

By assigning different variables to the undischarged assumptions, say, u to $\varphi \to \psi$ and v to φ, we would get the term uv rather than yx. There is a connection, though: the terms will be the same up to renaming of variables.

Now we have established the correspondence between typing pairs and natural deduction, we can prove theorems for typing pairs and transfer the result to natural deduction derivations.

Similar to what we did in the natural deduction section, we can make some observations here too. Let $\Gamma \vdash M : \varphi$ denote that there is a pair (Γ, M) witnessing the formula φ. Then always $\Gamma \vdash x : \varphi$ if $x : \varphi \in \Gamma$, and the following rules are valid:

$$\dfrac{\Gamma \vdash M_1 : \varphi_1 \qquad \Delta \vdash M_2 : \varphi_2}{\Gamma, \Delta \vdash \langle M_1, M_2\rangle : \varphi_1 \land \varphi_2} \land\text{Intro} \qquad \dfrac{\Gamma \vdash M : \varphi_1 \land \varphi_2}{\Gamma \vdash \mathrm{p}_i(M) : \varphi_i} \land\text{Elim}_i$$

$$\dfrac{\Gamma \vdash M_1 : \varphi_1}{\Gamma \vdash \mathrm{in}_1^{\varphi_2}(M) : \varphi_1 \lor \varphi_2} \lor\text{Intro}_1 \qquad \dfrac{\Gamma \vdash M_2 : \varphi_2}{\Gamma \vdash \mathrm{in}_2^{\varphi_1}(M) : \varphi_1 \lor \varphi_2} \lor\text{Intro}_2$$

$$\dfrac{\Gamma \vdash M : \varphi \lor \psi \qquad \Delta_1, x_1 : \varphi_1 \vdash N_1 : \chi \qquad \Delta_2, x_2 : \varphi_2 \vdash N_2 : \chi}{\Gamma, \Delta, \Delta' \vdash \mathrm{case}(M, x_1.N_1, x_2.N_2) : \chi} \lor\text{Elim}$$

$$\dfrac{\Gamma, x : \varphi \vdash N : \psi}{\Gamma \vdash \lambda x^{\varphi}.\, N : \varphi \to \psi} \to\text{Intro} \qquad \dfrac{\Gamma \vdash Q : \varphi \qquad \Delta \vdash P : \varphi \to \psi}{\Gamma, \Delta \vdash PQ : \psi} \to\text{Elim}$$

$$\dfrac{\Gamma \vdash M : \bot}{\Gamma \vdash \mathrm{contr}_{\varphi}(M) : \varphi} \bot\text{Elim}$$

These are the typing rules of the simply typed lambda calculus extended with product, sum and bottom.

In addition, the $F(\Gamma, M)$ is actually a type checking algorithm; it returns the type of the term with respect to the context, or is undefined if the term is ill-typed with respect to the context.

Photo Credits

Bibliography