How much do System T recursors lift to dependent types?

Hugo Herbelin

TYPES 2024

10 June 2024

The “traditional” recursor of System T

Assuming natural numbers:

$$\frac{}{\Gamma \vdash \mathbb{N} : \mathsf{U}} \qquad \frac{}{\Gamma \vdash \mathsf{zero} : \mathbb{N}} \qquad \frac{\Gamma \vdash t : \mathbb{N}}{\Gamma \vdash \mathsf{succ}\, t : \mathbb{N}}$$

the typing of the recursor of System T is:

$$\frac{\begin{array}{ll} \Gamma \vdash A : \mathsf{U} & \Gamma \vdash u : A \\ \Gamma \vdash t : \mathbb{N} & \Gamma, n : \mathbb{N}, a : A \vdash v : A \end{array}}{\Gamma \vdash (\mathsf{match}\ t\ \mathsf{with}\ [\mathsf{zero} \to u \,|\, \mathsf{succ}\, n \to_a v]) : A}$$

and its dependently-typed variant is:

$$\frac{\begin{array}{ll} \Gamma, m : \mathbb{N} \vdash A : \mathsf{U} & \Gamma \vdash u : A[m := \mathsf{zero}] \\ \Gamma \vdash t : \mathbb{N} & \Gamma, n : \mathbb{N}, a : A[m := n] \vdash v : A[n := \mathsf{succ}\, n] \end{array}}{\Gamma \vdash (\mathsf{match}\ t\ \mathsf{with}\ [\mathsf{zero} \to u \,|\, \mathsf{succ}\, n \to_a v]) : A[m := t]}$$

Its usual extension to inductive families, taking parametricity on $\mathbb{N}$ as an example

$$\frac{\Gamma \vdash t : \mathbb{N}}{\Gamma \vdash \mathsf{is}\mathbb{N}\, t : \mathsf{U}} \qquad \frac{}{\Gamma \vdash \mathsf{iszero} : \mathsf{is}\mathbb{N}\,\mathsf{zero}} \qquad \frac{\Gamma \vdash p : \mathsf{is}\mathbb{N}\, t}{\Gamma \vdash \mathsf{issucc}\, t\, p : \mathsf{is}\mathbb{N}\,(\mathsf{succ}\, t)}$$

$$\frac{\begin{array}{ll} \Gamma, n : \mathbb{N}, y : \mathsf{is}\mathbb{N}\, n \vdash A : \mathsf{U} & \Gamma \vdash u : A[n := \mathsf{zero}][y := \mathsf{iszero}] \\ \Gamma \vdash p : \mathsf{is}\mathbb{N}\, t & \Gamma, n : \mathbb{N}, x : \mathsf{is}\mathbb{N}\, n, a : A[n := n][y := x] \vdash v : A[n := \mathsf{succ}\, n][y := \mathsf{issucc}\, n\, x] \end{array}}{\Gamma \vdash (\mathsf{match}\; p\; \mathsf{with}\; [\mathsf{iszero} \to u \mid \mathsf{issucc}\, n\, x \to_a v]) : A[n := t][y := p]}$$

We advocate for a refinement of this recursor with an as clause

$$\frac{\begin{array}{ll}\Gamma, n:\mathbb{N}, y:\mathsf{is}\mathbb{N}\,n \vdash A:\mathsf{U} & \Gamma, y:\mathsf{is}\mathbb{N}\,\mathsf{zero} \vdash u : A[n := \mathsf{zero}][y := \mathsf{iszero}] \\ \Gamma \vdash p : \mathsf{is}\mathbb{N}\,t & \Gamma, n:\mathbb{N}, x:\mathsf{is}\mathbb{N}\,n, a:A[y:=x], y:\mathsf{is}\mathbb{N}\,(\mathsf{succ}\,n) \vdash v : A[n := \mathsf{succ}\,n][y := \mathsf{issucc}\,n\,x]\end{array}}{\Gamma \vdash \left(\mathsf{match}\ t\ \mathsf{as}\ y\ \mathsf{with}\ \begin{bmatrix}\mathsf{iszero} & \to u \\ \mathsf{issucc}\,n\,x & \to_a v\end{bmatrix}\right) : A[n := t][y := p]}$$

with reduction rules:

$$\begin{array}{ll}\mathsf{match}\ \mathsf{iszero}\ \mathsf{as}\ y\ \mathsf{with}\ \begin{bmatrix}\mathsf{iszero} & \to u \\ \mathsf{issucc}\,n & \to_a v\end{bmatrix} & \to u[y := \mathsf{iszero}] \\ \mathsf{match}\ (\mathsf{issucc}\,t\,p)\ \mathsf{as}\ y\ \mathsf{with}\ \begin{bmatrix}\mathsf{iszero} & \to u \\ \mathsf{issucc}\,n\,x & \to_a v\end{bmatrix} & \to v[n := t][x := p][y := \mathsf{issucc}\,t\,p][a := \mathsf{match}\ p\ \mathsf{as}\ y\ \mathsf{with}\ \begin{bmatrix}\mathsf{iszero} & \to u \\ \mathsf{issucc}\,n\,x & \to_a v\end{bmatrix}]\end{array}$$

We retrospectively advocate this refinement also in the absence of indices

$$\frac{\begin{array}{ll}\Gamma, m : \mathbb{N} \vdash A : \mathsf{U} & \Gamma, m : \mathbb{N} \vdash u : A[m := \mathsf{zero}] \\ \Gamma \vdash t : \mathbb{N} & \Gamma, n : \mathbb{N}, a : A[m := n], m : \mathbb{N} \vdash v : A[m := \mathsf{succ}\, n]\end{array}}{\Gamma \vdash \left(\mathsf{match}\ t\ \mathsf{as}\ m\ \mathsf{with} \begin{bmatrix}\mathsf{zero} & \to u \\ \mathsf{succ}\, n & \to_a v\end{bmatrix}\right) : A[m := t]}$$

with reduction rules:

$$\begin{array}{lll}\mathsf{match}\ \mathsf{zero}\ \mathsf{as}\ m\ \mathsf{with} \begin{bmatrix}\mathsf{zero} & \to u \\ \mathsf{succ}\, n & \to_a v\end{bmatrix} & \to & u[m := \mathsf{zero}] \\ \mathsf{match}\ (\mathsf{succ}\, t)\ \mathsf{as}\ m\ \mathsf{with} \begin{bmatrix}\mathsf{zero} & \to u \\ \mathsf{succ}\, n & \to_a v\end{bmatrix} & \to & v[n := t][m := \mathsf{succ}\, t][a := \mathsf{match}\ t\ \mathsf{as}\ m\ \mathsf{with} \begin{bmatrix}\mathsf{zero} & \to u \\ \mathsf{succ}\, n & \to_a v\end{bmatrix}]\end{array}$$

Motivations

- as-patterns (written @ in Haskell or Agda) are common in pattern-matching problems: are there just syntactic sugar or have a deeper role?

- as-patterns cause typing problems with inductive families (*):

1. should they refer to the scrutinee and be typed with the type of the scrutinee?

2. or should they refer to the constructor being matched and be typed with the type of the constructor?

- some translations, e.g. parametricity, turn data types into inductive families

(*) Agda's dependent pattern-matching solves this for indices that are variables by propagating definitional equalities on indices in the branches.

The issue with parametricity

Consider the following (trivial) program:

$$m : \mathbb{N} \vdash (\mathsf{match}\ m\ \mathsf{with}\ [\mathsf{zero} \to m \mid \mathsf{succ}\, n \to_a m]) : \mathbb{N}$$

It happens that the canonically derived equivalent program in $\mathsf{is}\mathbb{N}$ is ill-typed:

$$m : \mathbb{N}, y : \mathsf{is}\mathbb{N}\, m \not\vdash (\mathsf{match}\ y\ \mathsf{with}\ [\mathsf{iszero} \to y \mid \mathsf{issucc}\, n\, x \to_a y]) : \mathsf{is}\mathbb{N}\, m$$

Assuming we would have written:

$$m : \mathbb{N} \vdash (\mathsf{match}\ m\ \mathsf{as}\ m'\ \mathsf{with}\ \big[\mathsf{zero}\ \to m' \mid \mathsf{succ}\, n\ \to_a m'\big]) : \mathbb{N}$$

The canonically derived term would now be well-typed:

$$m : \mathbb{N}, y : \mathsf{is}\mathbb{N}\, m \vdash (\mathsf{match}\ y\ \mathsf{as}\ y'\ \mathsf{with}\ \big[\mathsf{iszero}\ \to y' \mid \mathsf{issucc}\, n\, x\ \to_a y'\big]) : \mathsf{is}\mathbb{N}\, m$$

Comparison with emulation 1: expanding the dependent instances

The first emulation expands the occurrences of the term being matched to change its type:

$$m : \mathbb{N}, p : \mathsf{is}\mathbb{N}\, m \vdash (\mathsf{match}\ p\ \mathsf{with}\ [\mathsf{iszero} \to \mathsf{iszero} \,|\, \mathsf{issucc}\, n\, x \to_a \mathsf{issucc}\, n\, x]) : \mathsf{is}\mathbb{N}\, m$$

It has the expected semantics (same equational theory) but the behaviour is different in terms of sharing the representation of the underlying constructors: if the 3 occurrences of p had their representation shared (as in call-by-value), then the sharing in memory is lost.

Comparison with emulation 2: generalising the term being matched

The second emulation generalises the term being matched to change its type:

$$m : \mathbb{N}, y : \mathsf{is}\mathbb{N}\, m \vdash (\mathsf{match}\ y\ \mathsf{with}\ [\mathsf{iszero} \to \lambda y^{\mathsf{is}\mathbb{N}\,\mathsf{zero}}.y \mid \mathsf{issucc}\, n\, x \to_a \lambda y^{\mathsf{is}\mathbb{N}\,(\mathsf{succ}\, n)}.y])\, y : \mathsf{is}\mathbb{N}\, m$$

which has also the expected semantics (same equational theory), respects sharing in call-by-value reduction, but it (superficially though unexpectedly) goes out of the syntax of (0-order) primitive recursion.

Summary

Ill-typed:

$$m : \mathbb{N}, y : \mathsf{is}\mathbb{N}\, m \nvdash (\mathsf{match}\ y\ \mathsf{with}\ [\mathsf{iszero} \to y \mid \mathsf{issucc}\, n\, x \to_a y]) : \mathsf{is}\mathbb{N}\, m$$

Typed with an expansion:

$$m : \mathbb{N}, p : \mathsf{is}\mathbb{N}\, m \vdash (\mathsf{match}\ p\ \mathsf{with}\ [\mathsf{iszero} \to \mathsf{iszero} \mid \mathsf{issucc}\, n\, x \to_a \mathsf{issucc}\, n\, x]) : \mathsf{is}\mathbb{N}\, m$$

Typed with a generalisation:

$$m : \mathbb{N}, y : \mathsf{is}\mathbb{N}\, m \vdash (\mathsf{match}\ y\ \mathsf{with}\ [\mathsf{iszero} \to \lambda y^{\mathsf{is}\mathbb{N}\,\mathsf{zero}}.y \mid \mathsf{issucc}\, n\, x \to_a \lambda y^{\mathsf{is}\mathbb{N}\,(\mathsf{succ}\, n)}.y])\, y : \mathsf{is}\mathbb{N}\, m$$

Typed with an as-pattern:

$$m : \mathbb{N}, y : \mathsf{is}\mathbb{N}\, m \vdash (\mathsf{match}\ y\ \mathsf{as}\ y'\ \mathsf{with}\ [\mathsf{iszero} \to y' \mid \mathsf{issucc}\, n\, x \to_a y']) : \mathsf{is}\mathbb{N}\, m$$