A computational proof of dependent choice, compatible with classical logic

(partially work in progress)

Hugo Herbelin

Réalisabilité à Chambéry #5

8 June 2012

Outline

- Thanks to strong sums, Martin-Löf's intuitionistic type theory proves the axiom of choice

- Strong sums do not marry well with computational classical logic

- Countable universal quantification can be turned into infinite conjunction in PA^ω that can be evaluated lazily

- Restricting proofs of strong sums to "negative-elimination free" proofs allows to ensure these proofs to be essentially intuitionistic while keeping the rest of the logic compatible with classical reasoning

- Countable choice and dependent choice are *intuitionistically* provable in PA^ω + negative-elimination-free strong sums

- Comparison with realisability-based approaches (Krivine, Berardi-Bezem-Coquand, Escardó-Oliva)

The axiom of choice in Martin-Löf's intuitionistic type theory

Using strong sums (a.k.a. strong existential, or Σ-types)

$$\frac{\Gamma \vdash p : \exists x^T\, A(x)}{\Gamma \vdash \texttt{prf}\, p : A(\texttt{wit}\, p)}$$

the (intensional) axiom of choice is provable in Martin-Löf's intuitionistic type theory:

$$\begin{array}{rcl} AC_{A,B} & \triangleq & \lambda H.(\lambda x.\texttt{wit}\,(H\,x), \lambda x.\texttt{prf}\,(H\,x)) \\ & : & \forall x^A \exists y^B\, P(x,y) \Rightarrow \exists f^{A \Rightarrow B}\, \forall x^A\, P(x, f(x)) \end{array}$$

Strong sums are incompatible with classical logic

Consider computational classical logic:

$$\frac{\Gamma, \alpha : A^{\perp\!\!\!\perp} \vdash p : A}{\Gamma \vdash \texttt{catch}_\alpha\, p : A} \qquad \frac{\Gamma \vdash p : A \qquad (\alpha : A^{\perp\!\!\!\perp}) \in \Gamma}{\Gamma \vdash \texttt{throw}_\alpha\, p : C}$$

Example, Drinker Paradox:

$$\begin{array}{rcl} DP & \triangleq & \texttt{catch}_\alpha.(x_0, \lambda y.\lambda H_x.\texttt{catch}_\beta \texttt{throw}_\alpha(y, \lambda y'.\lambda H_y.\texttt{throw}_\beta H_y)) \\ & : & \exists x\, \forall y\, (P(x) \Rightarrow P(y)) \end{array}$$

Strong sums are incompatible with classical logic

What is $\texttt{wit}$ on a proof that might backtrack on the witnesses it provides?

Indeed, applied to $\texttt{prf}$, the standard rule for computational classical logic gives:

$$\begin{array}{l} \texttt{prf}\,(\texttt{callcc}_\alpha(t_1, ...(\texttt{throw}_\alpha(t_2, p))...)) \\ \quad \rightarrow \texttt{callcc}_\alpha\texttt{prf}\,(t_1, ...(\texttt{throw}_\alpha\texttt{prf}\,(t_2, p))...) \end{array}$$

and the right-hand side is not typable because the two occurrences of α get the a priori incompatible types $P(t_1)$ and $P(t_2)$

In particular, in the proof of choice,

$$\begin{array}{rcl} AC_{A,B} & \triangleq & \lambda H.(\lambda x.\texttt{wit}\,(H\,x), \lambda x.\texttt{prf}\,(H\,x)) \\ & : & \forall x^A \exists y^B\, P(x,y) \Rightarrow \exists f^{A \Rightarrow B}\, \forall x^A\, P(x, f(x)) \end{array}$$

if $Hx : \exists y\, P(x,y)$ is classically proved then what $\texttt{wit}\,(H\,x)$ should be is unclear, and how to keep it "synchronised" with $\texttt{prf}\,(H\,x)$ is even more unclear.

A trick to recover *countable* choice

Turn $\forall x \exists y\, P(x,y)$ into a infinite conjunction $\exists y\, P(0,y) \wedge \exists y\, P(1,y) \wedge\ \ldots$ and prove instead

$$\begin{array}{rcl} AC'_{\mathbb{N},B} & \triangleq & \lambda H.(\lambda n.\mathtt{wit}\,(\mathtt{nth}\,n\,H), \lambda n.\mathtt{prf}\,(\mathtt{nth}\,n\,H)) \\ & : & (\exists y\, P(0,y) \wedge \exists y\, P(1,y) \wedge\ \ldots) \Rightarrow \exists f^{A \Rightarrow B}\, \forall x^A\, P(x, f(x)) \end{array}$$

Now, the infinite conjunction is a "positive" object and we just have to evaluate it in (lazy) call-by-value order to ensure that at the time `wit` and `prf` are called, the underlying stream is evaluated at this position.

Classical arithmetic in finite types: PA^ω

(the language of expressions: system T)

$$
\begin{array}{lcl}
T, U & ::= & \mathbb{N} \mid T \Rightarrow U \\
t, u & ::= & x \mid 0 \mid S(t) \mid \texttt{rec}\ t\ \texttt{of}\ [t \mid (x,y).t] \mid \lambda x.t \mid t\,t
\end{array}
$$

$$
\frac{(x : T) \in \Gamma}{\Gamma \vdash x : T} \qquad \frac{\Gamma, x : U \vdash t : T}{\Gamma \vdash \lambda x.t : U \Rightarrow T} \qquad \frac{\Gamma \vdash t : U \Rightarrow T \qquad \Gamma \vdash u : U}{\Gamma \vdash t\,u : T}
$$

$$
\frac{}{\Gamma \vdash 0 : \mathbb{N}} \qquad \frac{\Gamma \vdash t : \mathbb{N}}{\Gamma \vdash S(t) : \mathbb{N}} \qquad \frac{\Gamma \vdash t : \mathbb{N} \qquad \Gamma \vdash t_0 : U \qquad \Gamma, x : \mathbb{N}, y : U \vdash t_S : U}{\Gamma \vdash \texttt{rec}\ t\ \texttt{of}\ [t_0 \mid (x,y).t_S] : U}
$$

$$
\begin{array}{lcl}
(\lambda x.t)\,u & \equiv & t[x \leftarrow u] \\
\texttt{rec}\ 0\ \texttt{of}\ [t_0 \mid (x,y).t_S] & \equiv & t_0 \\
\texttt{rec}\ S(t)\ \texttt{of}\ [t_0 \mid (x,y).t_S] & \equiv & t_S[x \leftarrow t][y \leftarrow \texttt{rec}\ t\ \texttt{of}\ [t_0 \mid (x,y).t_S]]
\end{array}
$$

Classical arithmetic in finite types: PA^ω

(formulas and equational theory)

$$A, B \ ::= \ t = u \mid \top \mid \bot \mid A \Rightarrow B \mid A \wedge B \mid A \vee B \mid \forall x^T\, A \mid \exists x^T\, A$$

$$\begin{array}{lcl}
0 = 0 & \equiv & \top \\
0 = S(u) & \equiv & \bot \\
S(t) = 0 & \equiv & \bot \\
S(t) = S(u) & \equiv & t = u
\end{array}$$

Classical arithmetic in finite types: PA^ω

(intuitionistic inference rules)

$$\frac{(a:A)\in\Gamma}{\Gamma\vdash a:A} \qquad \frac{\Gamma\vdash p:A \qquad \Gamma,a:A\vdash q:B}{\Gamma\vdash \texttt{let}\ a=p\ \texttt{in}\ q:B} \qquad \frac{\Gamma\vdash p:A \qquad A\equiv B}{\Gamma\vdash p:B} \qquad \frac{}{\Gamma\vdash ():\top} \qquad \frac{\Gamma\vdash p:\bot}{\Gamma\vdash \texttt{exfalso}\ p:C}$$

$$\frac{\Gamma\vdash p_1:A_1 \qquad \Gamma\vdash p_2:A_2}{\Gamma\vdash (p_1,p_2):A_1\wedge A_2} \qquad \frac{\Gamma\vdash p:A_1\wedge A_2 \qquad \Gamma,a_1:A_1,a_2:A_2\vdash q:B}{\Gamma\vdash \texttt{split}\ p\ \texttt{as}\ (a_1,a_2)\ \texttt{in}\ q:B}$$

$$\frac{\Gamma\vdash p:A_i}{\Gamma\vdash \iota_i(p):A_1\vee A_2} \qquad \frac{\Gamma\vdash p:A_1\vee A_2 \qquad \Gamma,a_1:A_1\vdash p_1:B \qquad \Gamma,a_2:A_2\vdash p_2:B}{\Gamma\vdash \texttt{case}\ p\ \texttt{of}\ [a_1.p_1\,|\,a_2.p_2]:B}$$

$$\frac{\Gamma,a:A\vdash p:B}{\Gamma\vdash \lambda a.p:A\Rightarrow B} \qquad \frac{\Gamma\vdash p:A\Rightarrow B \quad \Gamma\vdash q:A}{\Gamma\vdash p\,q:B} \qquad \frac{\Gamma,x:T\vdash p:A(x)}{\Gamma\vdash \lambda x.p:\forall x^T\,A(x)} \qquad \frac{\Gamma\vdash p:\forall x^T\,A(x) \qquad \Gamma\vdash t:T}{\Gamma\vdash pt:A(t)}$$

$$\frac{\Gamma\vdash p:A(t) \qquad \Gamma\vdash t:T}{\Gamma\vdash (t,p):\exists x^T\,A(x)} \qquad \frac{\Gamma\vdash p:\exists x^T\,A(x) \qquad \Gamma,x:T,a:A(x)\vdash q:B}{\Gamma\vdash \texttt{dest}\ p\ \texttt{as}\ (x,a)\ \texttt{in}\ q:B}$$

$$\frac{t\equiv u}{\Gamma\vdash \texttt{refl}:t=u} \qquad \frac{\Gamma\vdash p:t=u \qquad \Gamma\vdash q:P(t)}{\Gamma\vdash \texttt{subst}\,p\,q:P(u)} \qquad \frac{\Gamma\vdash t:\mathbb{N} \qquad \Gamma\vdash p:P(0) \qquad \Gamma,x:T,a:P(x)\vdash q:P(S(x))}{\Gamma\vdash \texttt{ind}\ t\ \texttt{of}\ [p\,|\,(x,a).q]:P(t)}$$

Classical arithmetic in finite types: PA^{ω}

(classical logic)

$$\frac{\Gamma, \alpha : A^{\perp\!\!\!\perp} \vdash p : A}{\Gamma \vdash \texttt{catch}_{\alpha}\, p : A} \qquad \frac{\Gamma \vdash p : A \qquad (\alpha : A^{\perp\!\!\!\perp}) \in \Gamma}{\Gamma \vdash \texttt{throw}_{\alpha}\, p : C}$$

Classical arithmetic in finite types: PA^ω

(call-by-value evaluation semantics, minimal part)

$$
\begin{array}{lcl}
(\lambda a.q)\, p & \to & \texttt{let } a = p \texttt{ in } q \\
(\lambda x.p)\, t & \to & p[x \leftarrow t] \\
\texttt{case } \iota_i(p) \texttt{ of } [a_1.p_1 \,|\, a_2.p_2] & \to & \texttt{let } a_i = p \texttt{ in } p_i \\
\texttt{dest } (t,p) \texttt{ as } (x,a) \texttt{ in } q & \to & \texttt{let } a = p \texttt{ in } q[x \leftarrow t] \\
\texttt{split } (p_1,p_2) \texttt{ as } (a_1,a_2) \texttt{ in } q & \to & \texttt{let } a_1 = p_1 \texttt{ in let } a_2 = p_2 \texttt{ in } q \\
\texttt{let } a = b \texttt{ in } q & \to & q[a \leftarrow b] \\
\texttt{let } a = \lambda b.q \texttt{ in } q & \to & q[a \leftarrow \lambda b.q] \\
\texttt{let } a = \lambda x.p \texttt{ in } q & \to & q[a \leftarrow \lambda x.t] \\
\texttt{let } a = () \texttt{ in } q & \to & q[a \leftarrow ()] \\
\texttt{let } a = \iota_i(p) \texttt{ in } q & \to & \texttt{let } b = p \texttt{ in } q[a \leftarrow \iota_i(b)] \\
\texttt{let } a = (t,p) \texttt{ in } q & \to & \texttt{let } b = p \texttt{ in } q[a \leftarrow (t,b)] \\
\texttt{let } a = (p_1,p_2) \texttt{ in } q & \to & \texttt{let } a_1 = p_1 \texttt{ in let } a_2 = p_2 \texttt{ in } q[a \leftarrow (a_1,a_2)] \\
\texttt{subst refl } p & \to & p \\
\texttt{ind } 0 \texttt{ of } [p \,|\, (x,a).q] & \to & p \\
\texttt{ind } S(t) \texttt{ of } [p \,|\, (x,a).q] & \to & q[x \leftarrow t][a \leftarrow \texttt{ind } t \texttt{ of } [p \,|\, (x,a).q]]
\end{array}
$$

Classical arithmetic in finite types: PA^ω

(call-by-value evaluation semantics, non minimal part)

$$
\begin{array}{lcl}
F[\texttt{exfalso } p] & \to & \texttt{exfalso } p \\
F[\texttt{throw}_\alpha p] & \to & \texttt{throw}_\alpha p \\
F[\texttt{catch}_\alpha p] & \to & \texttt{catch}_\alpha F[p[\alpha \leftarrow F]] \\
\texttt{exfalso exfalso } p & \to & \texttt{exfalso } p \\
\texttt{exfalso throw}_\beta p & \to & \texttt{throw}_\beta p \\
\texttt{exfalso catch}_\beta p & \to & \texttt{exfalso } p[\alpha \leftarrow \texttt{exfalso } [\,]] \\
\texttt{throw}_\beta \texttt{exfalso } p & \to & \texttt{exfalso } p \\
\texttt{throw}_\beta \texttt{throw}_\alpha p & \to & \texttt{throw}_\alpha p \\
\texttt{throw}_\beta \texttt{catch}_\alpha p & \to & \texttt{throw}_\beta p[\alpha \leftarrow \beta] \\
\texttt{catch}_\alpha \texttt{throw}_\alpha p & \to & \texttt{catch}_\alpha p \\
\texttt{catch}_\beta \texttt{catch}_\alpha p & \to & \texttt{catch}_\beta p[\alpha \leftarrow \beta]
\end{array}
$$

where

$$
\begin{array}{lcl}
F[\,] & ::= & \iota_i([\,]) \mid ([\,], p) \mid (V, [\,]) \mid (t, [\,]) \\
& \mid & \texttt{case } [\,] \texttt{ of } [a_1.p_1 \mid a_2.p_2] \mid \texttt{split } [\,] \texttt{ as } (a_1, a_2) \texttt{ in } q \mid \texttt{subst } [\,]\, p \\
& \mid & \texttt{dest } [\,] \texttt{ as } (x, a) \texttt{ in } p \mid [\,]\, q \mid [\,]\, t \mid \texttt{let } a = [\,] \texttt{ in } q
\end{array}
$$

Note: can be reduced to 2 rules if one decomposes $\texttt{catch}_\alpha\, p$, $\texttt{throw}_\alpha\, p$ and $\texttt{exfalso}\ p$ as $\mu\alpha.[\alpha]p$, $\mu\_.[\alpha]p$ and $\mu\_.[\mathbf{tp}_\perp]p$ respectively (for $\mathbf{tp}_\perp$ evaluation context constant witnessing $\perp$-elimination).

PA^ω has coinductive formulas

For instance, the infinite conjunction $P(0) \wedge P(1) \wedge \ldots$ can be represented by

$$\exists f^{\mathbb{N} \Rightarrow \mathbb{N}} \, (f(0) = 1 \wedge \forall n \, (f(n) = 1 \Rightarrow (P(n) \wedge f(S(n)) = 1))$$

(standard second order encoding, using quantification over functions rather than on predicates)

For convenience, add primitive cofixpoints to PA^ω

$$\frac{\Gamma, f : T \Rightarrow \mathbb{N}, x : T, b : f(x) = 1 \vdash p : A \qquad f(\_) = 1 \text{ possibly occurs in positive } A}{\Gamma \vdash \mathtt{cofix}^t_{bx} p : \nu^t_{fx} A}$$

with equation

$$\nu^t_{fx} A \;\equiv\; A[x \leftarrow t][f(y) = 1 \leftarrow \nu^y_{fx} A]$$

For instance, $\nu^3_{fx}(P(x) \wedge f(S(x)) = 1)$ represents $P(3) \wedge P(4) \wedge \ \ldots$

Extend evaluation semantics of PA^ω to cofixpoints

$$
\begin{array}{ll}
\texttt{case cofix}^t_{bx}p \texttt{ of } [a_1.p_1 \mid a_2.p_2] & \to \texttt{let } c = \texttt{cofix}^t_{bx}p \texttt{ in case } c \texttt{ of } [a_1.p_1 \mid a_2.p_2] \\
\texttt{dest cofix}^t_{bx}p \texttt{ as } (x, a) \texttt{ in } q & \to \texttt{let } c = \texttt{cofix}^t_{bx}p \texttt{ in dest } c \texttt{ as } (x, a) \texttt{ in } q \\
\texttt{split cofix}^t_{bx}p \texttt{ as } (a_1, a_2) \texttt{ in } q & \to \texttt{let } c = \texttt{cofix}^t_{bx}p \texttt{ in split } c \texttt{ as } (a_1, a_2) \texttt{ in } q \\
\texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in exfalso } q & \to \texttt{exfalso let } a = \texttt{cofix}^t_{bx}p \texttt{ in } q \\
\texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in throw}_\alpha q & \to \texttt{throw}_\alpha \texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in } q \\
\texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in catch}_\alpha q & \to \texttt{catch}_\alpha \texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in } q \\
F[\texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in } q] & \to \texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in } F[q] \\
\texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in } D[\texttt{case } a \texttt{ of } [a_1.p_1 \mid a_2.p_2]] & \to \\
\multicolumn{2}{r}{\texttt{let } a = p[b \leftarrow \lambda y.\texttt{cofix}^y_{bx}p][x \leftarrow t] \texttt{ in } D[\texttt{case } a \texttt{ of } [a_1.p_1 \mid a_2.p_2]]} \\
\texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in } D[\texttt{split } a \texttt{ as } (a_1, a_2) \texttt{ in } q] & \to \\
\multicolumn{2}{r}{\texttt{let } a = p[b \leftarrow \lambda y.\texttt{cofix}^y_{bx}p][x \leftarrow t] \texttt{ in } D[\texttt{split } a \texttt{ as } (a_1, a_2) \texttt{ in } q]} \\
\texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in } D[\texttt{dest } a \texttt{ as } (x, a') \texttt{ in } q] & \to \\
\multicolumn{2}{r}{\texttt{let } a = p[b \leftarrow \lambda y.\texttt{cofix}^y_{bx}p][x \leftarrow t] \texttt{ in } D[\texttt{dest } a \texttt{ as } (x, a') \texttt{ in } q]}
\end{array}
$$

where

$$D[\,] \ ::= \ [\,] \mid D[F[\,]] \mid \texttt{let } a = \texttt{cofix}^t_{bx}p \texttt{ in } D[\,]$$

Adding strong elimination of existential to PA^ω (first step)

Replace weak elimination of existential by

$$\frac{\Gamma \vdash V : \exists x^T A(x)}{\Gamma \vdash \mathtt{prf}\, V : A(\mathtt{wit}\, V)} \qquad \frac{\Gamma \vdash V : \exists x^T A(x)}{\Gamma \vdash \mathtt{wit}\, V : T}$$

where

$$V \ ::= \ a \mid \iota_i(V) \mid (V, V) \mid (t, V) \mid \lambda a.p \mid \lambda x.p \mid () \mid \mathtt{refl}$$

Moreover, forbid dependency in implication introduction and cut

$$\frac{\Gamma, a : A \vdash p : B \quad a \notin FV(B)}{\Gamma \vdash \lambda a.p : A \Rightarrow B} \qquad \frac{\Gamma \vdash p : A \qquad \Gamma, a : A \vdash q : B \qquad a \notin FV(B)}{\Gamma \vdash \mathtt{let}\ a = p\ \mathtt{in}\ q : B}$$

And update the equational theory, evaluation contexts and evaluation rules

$$\mathtt{wit}\,(t, p) \ \equiv \ t \qquad F[\,] \ ::= \ \ldots \mid \mathtt{prf}\,[\,] \qquad \mathtt{prf}\,(t, p) \ \rightarrow \ p$$

Claim: The resulting system is equivalent to PA^ω for judgements not mentioning `wit`: the restriction on p combined with call-by-value ensures that p is "evaluated" when substituted and that no classical reasoning occurs before taking the witness.

Adding strong elimination of existential to PA^ω (second step)

Replace weak elimination of existential by

$$\frac{\Gamma \vdash p : \exists x^T\, A(x) \qquad p \text{ is positively eliminated}}{\Gamma \vdash \mathtt{prf}\, p : A(\mathtt{wit}\, p)}$$

where

- a value is positively eliminated
- if p, q, p_1 and p_2 are positively eliminated then $\mathtt{case}\ a\ \mathtt{of}\ [a_1.p_1 \,|\, a_2.p_2]$, $\mathtt{dest}\ q\ \mathtt{as}\ (x, a)\ \mathtt{in}\ p$ and $\mathtt{split}\ q\ \mathtt{as}\ (a_1, a_2)\ \mathtt{in}\ p$ are positively eliminated

Claim: we still get a system equivalent to PA^ω.

Adding strong elimination of existential to PA^ω (third step)

Replace weak elimination of existential by

$$\frac{\Gamma \vdash p : \exists x^T A(x) \qquad p \text{ is strongly N-elimination-free}}{\Gamma \vdash \mathtt{prf}\, p : A(\mathtt{wit}\, p)}$$

where

- a value is strongly N-elimination-free
- if p, q, p_1 and p_2 are strongly N-elimination-free then $\mathtt{ind}\ t\ \mathtt{of}\ [p_1 \,|\, (x,a).p_2]$, $\mathtt{case}\ a\ \mathtt{of}\ [a_1.p_1 \,|\, a_2.p_2]$, $\mathtt{dest}\ q\ \mathtt{as}\ (x,a)\ \mathtt{in}\ p$ and $\mathtt{split}\ q\ \mathtt{as}\ (a_1,a_2)\ \mathtt{in}\ p$ are strongly N-elimination-free

Claim: we then get the strength of countable choice

The proof of countable choice

$$\begin{array}{lcl} AC_{\mathbb{N}} & \triangleq & \lambda a.\mathtt{let}\ b = \mathtt{cofix}^0_{bn}(a\,n, b(Sn))\ \mathtt{in} \\ & & (\lambda n.\mathtt{wit}\,(\mathtt{nth}_C\,n\,b), \lambda n.\mathtt{prf}\,(\mathtt{nth}_C\,n\,b)) \\ & : & \forall n \exists y\, P(n,y) \Rightarrow \exists f\, \forall n\, P(n, f(n)) \end{array}$$

where

$$\begin{array}{lcl} \mathtt{nth}_C\,n & : & R_C(0) \Rightarrow R_C(n) \\ \mathtt{nth}_C\,n & \triangleq & \lambda b.\pi_1(\mathtt{ind}\ n\ \mathtt{of}\ [b\,|\,(m,c).\pi_2(c)]) \end{array}$$

(s is the stream of type $R_C(0) \triangleq \exists y\, P(0,y) \wedge \exists y\, P(1,y) \wedge\ \ldots$ extracted from the hypothesis)

Adding strong elimination of existential to PA^ω (fourth step)

Replace weak elimination of existential by

$$\frac{\Gamma \vdash p : \exists x^T\, A(x) \qquad p \text{ is N-elimination-free}}{\Gamma \vdash \mathtt{prf}\, p : A(\mathtt{wit}\, p)}$$

where

- a value is N-elimination-free
- if p, q, p_1 and p_2 is N-elimination-free then $\mathtt{prf}\, p$, $\mathtt{ind}\ t\ \mathtt{of}\ [p_1 \,|\, (x,a).p_2]$, $\mathtt{case}\ a\ \mathtt{of}\ [a_1.p_1 \,|\, a_2.p_2]$, $\mathtt{dest}\ q\ \mathtt{as}\ (x,a)\ \mathtt{in}\ p$ and $\mathtt{split}\ q\ \mathtt{as}\ (a_1,a_2)\ \mathtt{in}\ p$ are N-elimination-free.

Claim: we then get the strength of dependent choice

The proof of dependent choice

$$
\begin{aligned}
DC \quad & \triangleq \lambda a.\lambda x_0.\texttt{let}\ b = \texttt{s}\,a\,x_0\ \texttt{in} \\
& \quad (\lambda n.\texttt{wit}\,(\texttt{nth}_D\,n\,(x_0, b)), \\
& \quad\ (\texttt{refl}, \lambda n.\pi_1(\texttt{prf}\,(\texttt{prf}\,(\texttt{nth}_D\,n\,(x_0, b)))))) \\
& : \forall x \exists y\, P(x, y) \Rightarrow \\
& \quad \forall x_0\, \exists f\, (f(0) = x_0 \wedge \forall n\, P(f(n), f(S(n))))
\end{aligned}
$$

where

$$
\begin{aligned}
\texttt{nth}_D\,n & : \exists x\, R_D(x) \Rightarrow \exists x\, R_D(x) \\
\texttt{nth}_D\,n & \triangleq \lambda b.\texttt{ind}\ n\ \texttt{of}\ [b \,|\, (m, c).\texttt{dest}\ c\ \texttt{as}\ (x, d)\ \texttt{in} \\
& \qquad\qquad\qquad (\texttt{wit}\,(\texttt{prf}\,d), \pi_2(\texttt{prf}\,(\texttt{prf}\,d)))] \\
\texttt{s}\,a\,x & : R_D(x) \\
\texttt{s}\,a\,x & \triangleq \texttt{cofix}^x_{bn}(\texttt{dest}\ a\,n\ \texttt{as}\ (y, c)\ \texttt{in}\ (y, (c, by)))
\end{aligned}
$$

(s is a stream of type $R_D(x_0) \triangleq \exists x_1\,(P(x_0, x_1) \wedge \exists x_2\,(P(x_1, x_2) \wedge\ \ldots))$ obtained by recursively applying the hypothesis)

(that exactly the strength of dependent choice is captured is still a conjecture)

Properties of the systems with N-elimination-free strong elimination of existential quantification

Subject reduction: if $\Gamma \vdash p : A$ and $p \rightarrow q$ then $\Gamma \vdash q : A$

Normalisation: if $\Gamma \vdash p : A$ then p normalises [the proof, which is still in progress, uses dependent choice at the meta-level]

Progress: if $\vdash p : A$ and p not a value then p reduces

Evaluation: $\vdash p : A$ then $\vdash V : A$ for some V s.t. $P \xrightarrow{*} V$

Conservativity over HA^ω for closed $\forall$-$\Rightarrow$-ν-$\texttt{wit}$-free and Σ^0_1-formulas: if $\vdash T$ and T $\forall$-$\Rightarrow$-ν-$\texttt{wit}$-free or Σ^0_1 then $\vdash_{HA^\omega} T$

Consistency: $\nvdash \bot$

Comparison with Krivine's realiser of the axioms of countable and dependent choice (restated as a proof in PA_2 + quote)

Krivine's "proof" only supports the existence of *relational* choice functions

It needs classical logic

It relies on a "quote" effect χ typed with

$$\frac{\Gamma \vdash p : \exists X\, P(X)}{\Gamma \vdash \chi\, p : \exists n\, P(\Phi_P(n))} \qquad \text{where } \Phi_P \text{ is a formal predicate constant}$$

$$\begin{array}{lll} AC_{\mathbb{N}} & \triangleq & \lambda a.(U_P, \\ & & \quad \lambda x.\mathtt{dest}\ \chi\,(a\,x)\ \mathtt{as}\ (n,b)\ \mathtt{in}\ \mathtt{catch}_\alpha \mathtt{wf}_x\,(\lambda n'.\lambda f.\lambda b'.\mathtt{throw}_\alpha(\uparrow^{n' f b'}_{P(x,Y)} b'))\,n\,b) \\ & : & \forall x^{\mathbb{N}} \exists Y^{\mathbb{N} \Rightarrow \star} P(x,Y) \\ & & \Rightarrow \exists U^{\mathbb{N} \Rightarrow \mathbb{N} \Rightarrow \star} \forall x^{\mathbb{N}}\, P(x, U(x)) \end{array}$$

where

$$
\begin{array}{lcl}
V(x,n) & \triangleq & \neg P(x,\Phi_P(x,n)) \\
Z(x,n) & \triangleq & \forall m < n\, V(x,m) \Rightarrow V(x,n) \\
U_P(x) & \triangleq & \forall n\, (\neg Z(x,n) \Rightarrow \Phi_P(x,n)) \qquad \text{``exists } n \text{ minimal s.t. } \Phi_P(x,n)\text{''}
\end{array}
$$

$$
\mathtt{wf}_x \quad : \quad \forall n\, Z(x,n) \Rightarrow \forall n\, V(x,n) \qquad \text{``if } P(x,\Phi_P(x,n))\text{, there is a minimal } n \text{ for it''}
$$

and for $f : \forall m < n\, V(x,m)$ and $b : P(x,\Phi_P(x,n))$

$$
\begin{array}{lcl}
\uparrow_A^{nfb} & : & A(\Phi_P(x,n)) \Rightarrow A(U_P(x)) \\
\uparrow_{Y(t)}^{nfb} c & \triangleq & \lambda n'.\lambda k.\mathtt{if}\ n = n'\ \mathtt{then}\ c\ \mathtt{else}\ k\, \lambda f'.\lambda b'.\mathtt{if}\ n' < n\ \mathtt{then}\ f\, n'\, b'\ \mathtt{else}\ f'\, n\, b \\
\uparrow_{A\wedge B}^{nfb} c & \triangleq & (\uparrow_A^{nfb} (\pi_1 c), \uparrow_B^{nfb} (\pi_2 c)) \\
\uparrow_{A\Rightarrow B}^{nfb} c & \triangleq & \lambda a.(\uparrow_B^{nfb} (c\, (\downarrow_A^{nfb} a))) \\
\ldots & &
\end{array}
$$

$$
\begin{array}{lcl}
\downarrow_A^{nfb} & : & A(U_P(x)) \Rightarrow A(\Phi_P(x,n)) \\
\downarrow_{Y(t)}^{nfb} c & \triangleq & c\, n\, \lambda k.k\, f\, b \\
\downarrow_{A\wedge B}^{nfb} c & \triangleq & (\downarrow_A^{nfb} (\pi_1 c), \downarrow_B^{nfb} (\pi_2 c)) \\
\downarrow_{A\Rightarrow B}^{nfb} c & \triangleq & \lambda a.(\downarrow_B^{nfb} (c\, (\uparrow_A^{nfb} a))) \\
\ldots & &
\end{array}
$$

How to implement quote

Krivine implements χ by quoting the top argument of the stack at runtime. It seems that an alternative implementation is possible by quoting instead the witness:

$$\begin{array}{lcl} \chi\, p & \triangleq & (\lfloor \mathtt{wit}\, p \rfloor, \mathtt{prf}\, p) \\ \Phi(n) & \triangleq & \lceil n \rceil \end{array}$$

so that the reduction rule is

$$\chi(U, p) \to (\lfloor U \rfloor, p)$$

Quoting needs its argument closed. The rule can however be used as a local rule: only the decidability of equality $\lfloor U \rfloor = \lfloor U' \rfloor$ will need U and U' to be closed so as to be evaluable.

Comparison with Coquand-Berardi-Bezem's realiser of the axioms of countable choice

As rephrased by Berger, Coquand-Berardi-Bezem's "proof" builds a choice function by *update* induction.

Initially, the choice function returns a dummy value everywhere.

Each time a proof of $P(n, f(n))$ is requested, the proof of $\exists y\, P(n, y)$ together with a continuation that updates the choice function.

If, later on, the proof of some $P(n, f(n))$ has already been asked, the former value is retrieved.

In our case, the choice function has no default value. The proofs of $\exists y\, P(i, y)$ for $i \leq n$ are executed whenever either $f(n)$ or $P(n, f(n))$ is requested (but alternative, more sophisticated, evaluation strategies for PA^ω can be imagined).

Comparison with Escardó-Oliva's realiser of the axiom of countable and dependent choice

Similar idea of evaluating a cofixpoint.

Note: Other realisation exists (e.g. Spector's functional interpretation based on bar recursion).

Summary

By adding an appropriate intuitionistically-restricted rule for strong elimination of existential to PA^ω, we computationally capture the strength of either countable choice or dependent choice.

This can be turned into a Martin-Löf-style type theory by allowing dependent products with the restriction that they are instantiated only by N-elimination-free expressions.

Provides with an intuitionistic proof of bar induction compatible with classical logic:

$$\forall f\, \exists n\, B(f_{|n}) \Rightarrow \forall g \begin{pmatrix} \forall l\, (B(l) \Rightarrow g(l) = 0) \wedge \\ \forall l\, (\forall x\, g(l \star x) = 0 \Rightarrow g(l) = 0) \end{pmatrix} \Rightarrow g(\langle\rangle) = 0$$

Our proof of choice uses a weak form of effect (lazy evaluation) but we suspect that other proofs using effects are possible...