An intuitionistic logic that proves Markov's principle

Hugo Herbelin
INRIA - PPS
23 avenue d'Italie
F-75214 Paris Cedex 13
Hugo.Herbelin@inria.fr

Abstract

We design an intuitionistic predicate logic that supports a limited amount of classical reasoning, just enough to prove a variant of Markov's principle suited for predicate logic (namely $\neg\neg\exists x\, A(x) \to \exists x\, A(x)$ *for* $A(x)$ *an implication-free proposition), while still satisfying the core properties of intuitionistic logic, meaning here the disjunction and existence properties.*

At the computational level, the extraction of an existential witness out of a proof of its double negation is done by using a form of statically-bound exception mechanism, what can be seen as a formulation of Friedman's A-translation in direct-style.

Introduction

In arithmetic, Markov's principle is a weak classical scheme stating $\neg\neg\exists x\, A(x) \to \exists x\, A(x)$ for every decidable formula $A(x)$. Though not derivable in Heyting Arithmetic (i.e. intuitionistic arithmetic), its formulation as a rule (Markov's rule) is admissible (see e.g. [12]).

Markov's principle is realizable in arithmetic by unbounded search (ensured terminating by classical reasoning) and this is generally the way it is implemented for program extraction.

A simple method to prove the intuitionistic admissibility of Markov's rule has been given by Friedman who introduced for this purpose a notion of A-translation [6]. In fact, Friedman's proof does not prove $\neg\neg\exists x\, A(x) \to \exists x\, A(x)$ for any propositional formula $A(x)$ but only for those $A(x)$ that do not contain the implication connective. More generally, Friedman's method can be interpreted from the proof-theoretical point of view as a proof that $\neg\neg A \to A$ is admissible in intuitionistic predicate logic whenever A is a $\forall$-$\to$-free formula, the case of $\exists x\, A(x)$ for A $\to$-free being just a prototypical instance of the scheme.

We concentrate here on predicate logic and refers to as Markov's principle for predicate logic (shortly below Markov's principle) for the scheme that asserts $\neg\neg\exists\vec{x} A(\vec{x}) \to \exists\vec{x} A(\vec{x})$ whenever $A(\vec{x})$ is an $\to$-free proposition. Especially, we show that adding classical reasoning on $\forall$-$\to$-free formulas proves Markov's principle while still preserving the intuitionistic specificity of the logic (namely the disjunction and existence properties).

At the level of computation, the added classical rules can be seen as a mechanism of (statically-bound) exception throwing similar to the one introduced by Nakano in his (intuitionistic) `catch` and `throw` calculus [9][1]

1 IQC_{MP}: An intuitionistic predicate logic that proves Markov's principle

Usual intuitionistic predicate logic (IQC) is defined from a set of function symbols $f, g, \ldots$, each of a given arity and a set of predicate symbols P, Q, R, ... each with an arity too. Functions symbols of arity 0 are called constants and predicate symbols of arity 0 are called atomic propositions.

Terms are built from a set of variables x, y, ... by

$$t, u \quad ::= \quad f(\vec{t}) \mid x$$

where f ranges over function symbols and $\vec{t}$ denotes in $f(\vec{t})$ a sequence of terms of length the arity of f. Formulas are built from the standard connectives and quantifiers by the grammar

$$\begin{array}{lll} A, B & ::= & P(\vec{t}) \mid \top \mid \bot \mid A \to B \mid A \wedge B \mid A \vee B \\ & \mid & \forall x\, A \mid \exists x\, A \end{array}$$

[1] Ordinary (dynamically bound) exceptions, as found in the ML, C++ or Java programming languages can actually be used too, this is part of ongoing work.

$$\frac{A \in \Gamma}{\Gamma \vdash_\Delta A}\ \textsc{axiom}$$

$$\frac{\Gamma \vdash_\Delta A_1 \qquad \Gamma \vdash_\Delta A_2}{\Gamma \vdash_\Delta A_1 \wedge A_2}\ \wedge_I \qquad \frac{\Gamma \vdash_\Delta A_1 \wedge A_2}{\Gamma \vdash_\Delta A_i}\ \wedge_E^i$$

$$\frac{\Gamma \vdash_\Delta A_i}{\Gamma \vdash_\Delta A_1 \vee A_2}\ \vee_I^i \qquad \frac{\Gamma \vdash_\Delta A_1 \vee A_2 \qquad \Gamma, A_1 \vdash_\Delta B \qquad \Gamma, A_2 \vdash_\Delta B}{\Gamma \vdash_\Delta B}\ \vee_E$$

$$\frac{\Gamma, A \vdash_\Delta B}{\Gamma \vdash_\Delta A \to B}\ \to_I \qquad \frac{\Gamma \vdash_\Delta A \to B \qquad \Gamma \vdash_\Delta A}{\Gamma \vdash_\Delta B}\ \to_E$$

$$\frac{\Gamma \vdash_\Delta A(x) \qquad x \text{ fresh}}{\Gamma \vdash_\Delta \forall x A(x)}\ \forall_I \qquad \frac{\Gamma \vdash_\Delta \forall x A(x)}{\Gamma \vdash_\Delta A(t)}\ \forall_E$$

$$\frac{\Gamma \vdash_\Delta A(t)}{\Gamma \vdash_\Delta \exists x A(x)}\ \exists_I \qquad \frac{\Gamma \vdash_\Delta \exists x A(x) \qquad \Gamma, A(x) \vdash_\Delta B \qquad x \text{ fresh}}{\Gamma \vdash_\Delta B}\ \exists_E$$

$$\frac{}{\Gamma \vdash_\Delta \top}\ \top_I \qquad \frac{\Gamma \vdash_\Delta \bot}{\Gamma \vdash_\Delta C}\ \bot_E$$

$$\frac{\Gamma \vdash_{T,\Delta} T}{\Gamma \vdash_\Delta T}\ \textsc{catch} \qquad \frac{\Gamma \vdash_\Delta T \qquad T \in \Delta}{\Gamma \vdash_\Delta C}\ \textsc{throw}$$

Figure 1. Inference rules of IQC_{MP}

where P ranges over predicate symbols and $\vec{t}$ is a sequence of terms whose length is the arity of P. Negation $\neg A$ is defined as $A \to \bot$.

IQC_{MP} is an extension of IQC. Its inference rules are given on Figure 1 (we use natural deduction). The subclass of $\forall$-$\to$-free formulas plays a special role and we use T, U, ... to denote such formulas:

$$T, U \quad ::= \quad P(\vec{t}) \mid \top \mid \bot \mid T \wedge U \mid T \vee U \mid \exists x T$$

Contexts of formulas, written Γ, are ordered sequences of formulas. Contexts of $\forall$-$\to$-free formulas, written Δ, are ordered sequences of $\forall$-$\to$-free formulas. By $\neg\Delta$ is meant the context obtained by distributing $\neg$ over the formulas of Δ. Note that IQC can be characterized as the subset of IQC_{MP} obtained by removing the rules catch and throw and keeping Δ empty.

The main difference between IQC_{MP} and IQC is that the former supports classical reasoning on $\forall$-$\to$-free formulas. This is implemented by the rules catch and throw which say that to prove a $\forall$-$\to$-free formula T, one is allowed to change its mind during the proof and to restart a new proof of T at any time.

The main properties of IQC_{MP} is that it proves Markov's principle while still retaining the disjunction and existence properties that are characteristic of intuitionistic logic.

Theorem 1 *In IQC_{MP}, for T $\forall$-$\to$-free, and in particular for Markov's principle, i.e. for T being $\exists\vec{x}A(\vec{x})$ with $A(\vec{x})$ $\to$-free, we have $\vdash \neg\neg T \to T$.*

Proof: One gets a proof of $T \vdash_T \bot$ by applying throw. By $\to_I$ and $\to_E$ we obtain a proof of $\neg\neg T \vdash_T \bot$. By applying $\bot_E$ followed by catch, we get a proof of $\neg\neg T \vdash T$ from which $\neg\neg T \to T$ derives (note that we freely use the lemma that $\Gamma \vdash_\Delta A$ implies $\Gamma' \vdash_{\Delta'} A$ for $\Gamma \subset \Gamma'$ and $\Delta \subset \Delta'$; this weakening lemma, as expected, indeed holds in IQC_{MP}). ■

Let us formally write MP for the scheme $\neg\neg\exists\vec{x}A(\vec{x}) \to \exists\vec{x}A(\vec{x})$ where $A(\vec{x})$ is an $\to$-free proposition. We have:

Theorem 2 $\Gamma \vdash A$ *in* IQC_{MP} *iff* $MP, \Gamma \vdash A$ *in* IQC.

Proof: Of course, MP is equivalent to the scheme $\neg\neg T \rightarrow T$ for T a $\forall$-$\rightarrow$ free formula. Then, by Theorem 1, $\Gamma, MP \vdash A$ implies $\Gamma \vdash A$ which obviously is a proof of IQC_{MP}. Conversely, we prove by induction of a derivation of $\Gamma \vdash_\Delta A$ in IQC_{MP}, that $\Gamma, \neg\Delta, MP \vdash A$ holds in IQC. All cases are direct and we use MP for interpreting the rule catch. ∎

Theorem 3 (Disjunction property) *In* IQC_{MP}, *if* $\vdash A_1 \vee A_2$ *then* $\vdash A_1$ *or* $\vdash A_2$.

Theorem 4 (Existence property) *In* IQC_{MP}, *if* $\vdash \exists x\, A(x)$ *then there exists t such that* $\vdash A(t)$.

The proof of these last two theorems is the subject of the next section.

2 The proof theory of IQC_{MP}

We show that the proofs of IQC_{MP} have a computational interpretation as programs in a λ-calculus extended with a mechanism of statically-bound exceptions implemented with operators named `catch` and `throw`.

The language of proofs is defined by the grammar

$$\begin{array}{lcl} p, q & ::= & a \mid \iota_i(p) \mid (p,q) \mid (t,p) \mid \lambda a.p \mid \lambda x.p \mid () \\ & \mid & \texttt{case}\ p\ \texttt{of}\ [a_1.p_1 \mid a_2.p_2] \\ & \mid & \pi_i(p) \mid \texttt{dest}\ p\ \texttt{as}\ (x,a)\ \texttt{in}\ q \\ & \mid & p\,q \mid p\,t \mid \texttt{efq}\ p \\ & \mid & \texttt{catch}_\alpha p \mid \texttt{throw}_\alpha p \end{array}$$

where a, b, ... range over a first set of proof variables. and α, β, ... range over another set of proof variables. The constructions $\lambda a.p$, `case` p `of` $[a_1.p_1 \mid a_2.p_2]$ and `dest` p `as` (x,a) `in` q bind a, a_1 and a_2. The constructions $\lambda x.p$ and `dest` p `as` (x,a) `in` q bind x. The construction $\texttt{catch}_\alpha p$ binds α. The binders are considered up to the actual name used to represent the binder (so-called α-conversion).

The annotation of IQC_{MP} with proof-terms is given in Figure 2 where the contexts Γ and Δ are now maps from variable names to formulas. For instance, the proof of Markov's principle in Theorem 1 is

$$\lambda a.\texttt{catch}_\alpha\ \texttt{efq}\ (a\,\lambda b.\texttt{throw}_\alpha\, b)$$

A subclass of proofs will play a particular role in extracting the intuitionistic content of weakly classical proofs of IQC_{MP}. These are the values defined by

$$V \quad ::= \quad a \mid \iota_i(V) \mid (V,V) \mid (t,V) \mid \lambda a.p \mid \lambda x.p \mid ()$$

Another class of expressions will be useful to define the reduction, it is the class of elementary evaluation contexts defined by

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

For $F[\ ]$ an elementary evaluation context and p a proof, we write $F[p]$ for the proof obtained by plugging p into the hole of $F[\ ]$.

We can now define evaluation in IQC_{MP} as the congruent closure of the following reductions:

$$\begin{array}{lcl} (\lambda a.p)\,V & \rightarrow & p[a \leftarrow V] \\ (\lambda x.p)\,t & \rightarrow & p[x \leftarrow t] \\ \texttt{case}\ \iota_i(V)\ \texttt{of}\ [a_1.p_1 \mid a_2.p_2] & \rightarrow & p_i[a_i \leftarrow V] \\ \texttt{dest}\ (t,V)\ \texttt{as}\ (x,a)\ \texttt{in}\ p & \rightarrow & p[x \leftarrow t][a \leftarrow V] \\ \pi_i(V_1,V_2) & \rightarrow & V_i \\ F[\texttt{efq}\ p] & \rightarrow & \texttt{efq}\ p \\ F[\texttt{throw}_\alpha p] & \rightarrow & \texttt{throw}_\alpha p \\ \texttt{catch}_\alpha \texttt{throw}_\alpha p & \rightarrow & \texttt{catch}_\alpha p \\ \texttt{catch}_\alpha \texttt{throw}_\beta V & \rightarrow & \texttt{throw}_\beta\, V\ \ (\alpha \neq \beta) \\ \texttt{catch}_\alpha V & \rightarrow & V \end{array}$$

where the substitutions $p[a \leftarrow V]$ and $p[x \leftarrow t]$ are capture-free with respect to the three kinds of variables (x, a and α).

Note that this is a call-by-value reduction semantics and that we do not consider commutative cuts just because we are only concerned with the normalization of closed proofs and commutative cuts are not needed for that purpose.

The operators `catch` and `throw` behave like the similarly named operators of Nakano [9] or Crolard [4]. Like in [9], but on the contrary of [4] (or of Parigot's $\lambda\mu$-calculus [10] to which the calculus of [4] is equivalent), `catch` does not capture its surrounding context (i.e. there is no rule of the form $F[\texttt{catch}_\alpha p] \rightarrow \texttt{catch}_\beta\, p[\texttt{throw}_\alpha\,[\ ] \leftarrow \texttt{throw}_\beta\, F[\ ]])$. As such, `throw` behaves as an exception raiser and `catch` as an exception handler but still not as in standard programming languages like Java or ML, since there exceptions are dynamically bound (i.e. the substitution is not capture-free) while in IQC_{MP} they are statically-bound (i.e. the substitution is capture-free)[2]. Alternatively, $\texttt{catch}_\alpha p$

[2]Compare substitution with capture
$(\lambda a.\texttt{catch}_\alpha(a, \texttt{throw}_\alpha 1))(\texttt{throw}_\alpha 2) \rightarrow \texttt{catch}_\alpha(\texttt{throw}_\alpha 2, \texttt{throw}_\alpha 1)$
to capture-free substitution
$(\lambda a.\texttt{catch}_\alpha(a, \texttt{throw}_\alpha 1))(\texttt{throw}_\alpha 2) \rightarrow \texttt{catch}_\beta(\texttt{throw}_\alpha 2, \texttt{throw}_\beta 1)$.

$$\frac{(a : A) \in \Gamma}{\Gamma \vdash_\Delta a : A}\ \textsc{axiom}$$

$$\frac{\Gamma \vdash_\Delta p_1 : A_1 \qquad \Gamma \vdash_\Delta p_2 : A_2}{\Gamma \vdash_\Delta (p_1, p_2) : A_1 \wedge A_2}\ \wedge_I \qquad \frac{\Gamma \vdash_\Delta p : A_1 \wedge A_2}{\Gamma \vdash_\Delta \pi_1 p : A_i}\ \wedge_E^o$$

$$\frac{\Gamma \vdash_\Delta p : A_i}{\Gamma \vdash_\Delta \iota_i(p) : A_1 \vee A_2}\ \vee_I^i \qquad \frac{\Gamma \vdash_\Delta p : A_1 \vee A_2 \qquad \Gamma, a_1 : A_1 \vdash_\Delta p_1 : B \qquad \Gamma, a_2 : A_2 \vdash_\Delta p_2 : B}{\Gamma \vdash_\Delta \texttt{case}\ p\ \texttt{of}\ [a_1.p_1 \mid a_2.p_2] : B}\ \vee_E$$

$$\frac{\Gamma, a : A \vdash_\Delta p : B}{\Gamma \vdash_\Delta \lambda a.p : A \to B}\ \to_I \qquad \frac{\Gamma \vdash_\Delta p : A \to B \qquad \Gamma \vdash_\Delta q : A}{\Gamma \vdash_\Delta p\, q : B}\ \to_E$$

$$\frac{\Gamma \vdash_\Delta p : A(x) \qquad x \text{ fresh}}{\Gamma \vdash_\Delta \lambda x.p : \forall x A(x)}\ \forall_I \qquad \frac{\Gamma \vdash_\Delta p : \forall x A(x)}{\Gamma \vdash_\Delta pt : A(t)}\ \forall_E$$

$$\frac{\Gamma \vdash_\Delta p : A(t)}{\Gamma \vdash_\Delta (t, p) : \exists x A(x)}\ \exists_I \qquad \frac{\Gamma \vdash_\Delta p : \exists x A(x) \qquad \Gamma, a : A(x) \vdash_\Delta q : B \qquad x \text{ fresh}}{\Gamma \vdash_\Delta \texttt{dest}\ p\ \texttt{as}\ (x, a)\ \texttt{in}\ q : B}\ \exists_E$$

$$\frac{}{\Gamma \vdash_\Delta () : \top}\ \top_I \qquad \frac{\Gamma \vdash_\Delta p : \bot}{\Gamma \vdash_\Delta \texttt{efq}\ p : C}\ \bot_E$$

$$\frac{\Gamma \vdash_{\alpha:T,\Delta} p : T}{\Gamma \vdash_\Delta \texttt{catch}_\alpha\ p : T}\ \textsc{catch} \qquad \frac{\Gamma \vdash_\Delta p : T \qquad (\alpha : T) \in \Delta}{\Gamma \vdash_\Delta \texttt{throw}_\alpha\ p : C}\ \textsc{throw}$$

Figure 2. Proof-term annotation of IQC_{MP}

can be seen as a delimited control operator (i.e. as an expression of the form $\#\,\texttt{callcc}_\alpha\, p$ where # is a delimiter that blocks the interaction of $\texttt{callcc}_\alpha\, p$ with its context and expects it first to evaluate – to a value – before being observed by its surrounding evaluation context).

We now check that the reduction system is compatible with typing.

Theorem 5 (Strengthening) *If* $\Gamma \vdash_\Delta V : T$ *then* $\Gamma \vdash_\Delta V : T$.

Proof: Obvious since the syntax of V refers to no p (and hence to no `catch` or `throw`) as soon as $\to$ and $\forall$ are excluded. ■

Theorem 6 (Subject reduction) *If* $\Gamma \vdash_\Delta p : A$ *and* $p \to q$ *then* $\Gamma \vdash_\Delta q : A$

Proof: By checking all cases, using Strengthening for the last three rules. Note that since `catch`, on the contrary of standard classical operators like `callcc`, does not capture its context, its type remains unchanged and the $\forall$-$\to$ constraint on the formulas of Δ is preserved. ■

We then characterize the set of normal forms in IQC_{MP}.

Theorem 7 (Characterization of normal forms) *The set of normal forms for $\to$ corresponds to the entry r of the following grammar:*

$$\begin{array}{rcl}
r & ::= & W \mid s \mid \texttt{efq}\ H[I[a]] \mid \texttt{throw}_\alpha\ s \\
W & ::= & a \mid \iota_i(W) \mid (W, W) \mid (t, W) \mid \lambda x.r \mid \lambda a.r \mid () \\
s & ::= & H[I[a]] \mid \iota_i(s) \mid (s, s) \mid (W, s) \mid (s, W) \mid (t, s) \\
G[\,] & ::= & \texttt{case}\ [\,]\ \texttt{of}\ [a_1.r_1 \mid a_2.r_2] \\
 & \mid & \pi_i([\,]) \mid \texttt{dest}\ [\,]\ \texttt{as}\ (x, a)\ \texttt{in}\ r \\
 & \mid & [\,]\, r \mid [\,]\, t \mid \texttt{catch}_\alpha \texttt{efq}\ [\,] \\
H[\,] & ::= & G[\,] \mid J[G[\,]] \\
I[\,] & ::= & [\,] \mid H[I[\,]] \\
J[\,] & ::= & [\,] \mid \texttt{catch}_\alpha \texttt{throw}_\beta\ J[\,] \mid \texttt{catch}_\alpha\ J[\,]
\end{array}$$

Proof: By inspection of the form of proofs that are not reducible. ■

We then check that the reduction system is not too simple and that it at least produces head-normal form on closed proofs.

Theorem 8 (Progress) *If $\vdash_\Delta p : A$ and p is not a (closed) value then p is reducible*

Proof: According to Theorem 7, closed normal forms are necessarily in the set W. But this set is a subset of the set of values. ∎

Theorem 9 (Normalization) *If $\Gamma \vdash_\Delta p : A$ then p is normalizable*

Proof: By mapping IQC_{MP} into the sequent calculus $LK_{\mu\tilde{\mu}}^{\vee_a \wedge_a \to \forall \exists \top \bot}$ [8]. The strong normalization of $LK_{\mu\tilde{\mu}}^{\vee_a \wedge_a \to \forall \exists \top \bot}$ is obtained by canonically extending the proof of strong normalization for $LK_{\mu\tilde{\mu}}^{\to}$ of Polonovoski [11] with connectives (additive) $\vee$, (additive) $\wedge$, $\forall$, $\exists$, $\top$, $\bot$ as in [1]. ∎

We are now ready to prove Theorems 3 and 4. Given a proof of $\vdash p : A_1 \vee A_2$, we know by progress and normalization that p eventually reduces to a value V which, by subject reduction, satisfies $\vdash V : A_1 \vee A_2$. By inspection of the possible forms of V, we know that we have either a proof of $\vdash A_1$ or a proof of $\vdash A_2$. Similarly, from $\vdash p : \exists x A(x)$ we know $\vdash V : \exists x A(x)$ for some V and hence $\vdash A(t)$ for some term t.

3 Discussion and relation to other works

The codereliction of differential proof nets In terms of polarity in linear logic [7], the $\forall$-$\to$-free constraint characterizes the formulas can be interpreted as purely positive formulas (positive formulas of which no subformula is negative). In the framework of polarized linear logic, Markov's principle expresses then that from a purely positive formula P possibly proved using weakening or contraction, i.e. a proof of Q differs from P by the insertion at some places of the "why not" exponential connective of linear logic, a linear proof of P can be extracted. Interestingly, this corresponds to applying the codereliction rule of differential proof nets [5].

Nakano and Crolard's `catch` and `throw` calculi Our calculus is very similar to the one proposed by Nakano [9]. However, in [9], the rule of introduction of implication requires Δ to be empty what prevents from deriving Markov's principle. Actually, as expressed in Theorem 5 of [9], the logical expressiveness Nakano's calculus is the one of LJ.

Another variant of intuitionistic logic with control operators that does not increase the logical expressiveness can also be found in Crolard [3].

Friedman's A-translation Expressed in our calculus, Friedman's A-translation [6] maps a proof of $\Gamma \vdash_\Delta A$ in IQC_{MP} to a proof of $\Gamma_\Delta \vdash A_\Delta$ in IQC, where B_Δ is obtained by replacing each atom of B by the disjunction of the formulas in Δ. Through this translation, the throw rule (assuming w.l.o.g that it is used with atomic conclusions) is interpretable as an injection. Also, for B $\forall$-$\to$-free, we have $B_\Delta \to B \vee \bigvee \Delta$ from what we see that the catch rule is interpretable (the proof of $B_\Delta \to B \vee \bigvee \Delta$ can be logically seen as the property that a purely positive formula B can be purified from all its calls to `throw` by following a call-by-value reduction strategy; especially, when, B is a conjunction, there are two asymmetrical proofs of $C_\Delta \wedge D_\Delta \to (C \wedge D) \vee \bigvee \Delta$ which match the two asymmetrical ways to evaluate a pair along call-by-value reduction). Henceforth, our calculus can be seen as a *direct-style* representation of A-translation in the same way as `callcc` provides with a direct-style representation of continuation-passing-style translation.

Independence of premises The principle of independence of premises (IP) goes from $\Gamma \vdash \neg B \to \exists x A(x)$ to $\Gamma \vdash \exists x (\neg B \to A(x))$. This principle is not admissible in IQC_{MP} because if it were, taking Γ empty and $A(x)$ an arbitrary atomic formula, one would obtain from MP that $\exists x (\neg\neg \exists y A(y) \to A(x))$, from Theorem 1 that $\exists x (\exists y A(y) \to A(x))$ and from Theorem 4 that $\exists y A(y) \to A(t)$ for some term t. However, using Theorem 7, one sees that no normal proof can have this type (the same reasoning holds in Heyting Arithmetic taking for A a formula such that neither $\vdash \neg\neg A$ nor $\vdash \neg A$ holds).

Markov's principle in arithmetic Since any decidable formula can be expressed in terms of bounded existential quantification, conjunction and disjunction over decidable atoms, and hence as a $\forall$-$\to$-free formula, we believe that by using an axiom-free presentation of Heyting Arithmetic, one could directly extend IQC_{MP} to the arithmetic case. We would then get a constructive content of $\neg\neg \exists x A(x) \to \exists x A(x)$ for $A(x)$ decidable much direct that the commonly accepted realizer that successively checks the truth of each instance of $A(n)$. Especially, our constructivization of Markov's principle not only contains its own proof of termination but it also directly evaluates to the witness of the existential quantification.

Completeness proofs Gödel and Kreisel proved that completeness for classical predicate logic implies Markov's principle. More precisely, Berardi and Valentini [2] showed that Markov's principle is necessary as soon as $\perp$ in the syntax is interpreted as $\perp$ in the model (a similar phenomenon happens for the completeness of intuitionistic logic for which the interpretation of $\perp$ has to be weakened so as to obtain intuitionistic proofs; consider e.g. Veldman [13] or Friedman proofs). We believe that both completeness for classical logic and for intuitionistic logic could be carried out in an extension of IQC_{MP} to second-order arithmetic without having to weaken the interpretation of $\perp$.

4 Conclusion

We showed that adding classical reasoning on $\forall$-$\rightarrow$-free formulas to intuitionistic logic preserves the intuitionistic character of the logic, as witnessed by the preservation of the disjunction and existence properties, while providing with an effective intuitionistic proof of Markov's principle. To compute with Markov's principle, we used a form of statically-bound exception mechanism.

5 Acknowledgments

I thank Andreas Abel, Tristan Crolard, Danko Ilik, Guillaume Munch–Maccagnoni and Noam Zeilberger for fruitful discussions on this topic.

References

[1] F. Barbanera and S. Berardi. A symmetric λ-calculus for classical program extraction. *Information and Computation*, 125(2):103–117, 1996.

[2] S. Berardi and S. Valentini. Krivine's intuitionistic proof of classical completeness (for countable languages). *Ann. Pure Appl. Logic*, 129(1-3):93–106, 2004.

[3] T. Crolard. *Extension de l'isomorphisme de Curry-Howard au traitement des exceptions*. Ph.D. thesis, University Paris 7, Dec. 1996.

[4] T. Crolard. A confluent lambda-calculus with a catch/throw mechanism. *J. Funct. Program.*, 9(6):625–647, 1999.

[5] T. Ehrhard and L. Regnier. Differential interaction nets. *Electr. Notes Theor. Comput. Sci.*, 123:35–74, 2005.

[6] H. Friedman. Classically and intuitionistically provably recursive functions. In D. S. Scott and G. H. Muller, editors, *Higher Set Theory*, volume 669 of *Lecture Notes in Mathematics*, pages 21–27. Springer, Berlin/Heidelberg, 1978.

[7] J.-Y. Girard. Linear logic. *Theor. Comput. Sci.*, 50:1–102, 1987.

[8] H. Herbelin. *C'est maintenant qu'on calcule: au cœur de la dualité*. Habilitation thesis, University Paris 11, Dec. 2005.

[9] H. Nakano. A constructive formalization of the catch and throw mechanism. In *Proceedings, Seventh Annual IEEE Symposium on Logic in Computer Science, 22-25 June 1992, Santa Cruz, California, USA*, pages 82–89. IEEE Computer Society, 1992.

[10] M. Parigot. Lambda-mu-calculus: An algorithmic interpretation of classical natural deduction. In *Logic Programming and Automated Reasoning: International Conference LPAR '92 Proceedings, St. Petersburg, Russia*, pages 190–201. Springer-Verlag, 1992.

[11] E. Polonovski. *Subsitutions explicites, logique et normalisation*. Ph.D. thesis, University Paris 7, June 2003.

[12] A. S. Troelstra. *Mathematical Investigation of Intuitionistic Arithmetic and Analysis*, volume 344 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1973.

[13] W. Veldman. An intuitionistic completeness theorem for intuitionistic predicate logic. *J. Symb. Log.*, 41(1):159–166, 1976.