Church $\Rightarrow$ Scott = PTIME

An application of resource sensitive realizability

Aloïs Brunel

Ecole Normale Supérieure de Lyon

(Joint work with Kazushige Terui)

Background (1/2)

Some well-known factors that can make complexity explodes :

- Non-Linearity : non-linear use of function variables can increase the complexity. We can limit the use of higher-order variables (e.g, using typing systems).

- Nested recursion : when functions are defined by multiple and nested recursions, there is a risk of complexity explosion.
  Data tiering (Leivant) : multiple copies of the binary words algebra, indiced by tiers :

$$\mathbb{W}_0, \mathbb{W}_1, ..., \mathbb{W}_n, ...$$

  the output of a function defined by recursion on a variable of tier n lives in a lower tier.

Background (2/2)

- Leivant and Marion (TLCA '93) used the concept of data tiering in a λ-calculus to characterize Ptime. One base (concrete) $\mathbb{W}$ added to λ-calculus, where recursion is not allowed, and the (logical) binary algebra of Church words. The functions from Church words to $\mathbb{W}$ are exactly the Ptime functions.

- We would like the characterization fully logical : replace $\mathbb{W}$ by a logical data structure (Scott words) defined in a linear logic based type system.

- Realizability semantics : Dal Lago & Hofmann

- Try to apply this proof technique to our system

Syntax of $\mathbf{DIAL}_{lin}$

$\mathbf{DIAL}_{lin}$ is a type system for the pure λ-calculus.

- **Terms** $t, u ::= x \mid \lambda x.t \mid tu$
- **Reduction** $(\lambda x.t)u \to_\beta t[u/x]$
- If it exists, we denote by $[\![t]\!]_\beta$ the β-normal form of t.

Linear formulas and general formulas

$$L, M ::= \alpha \mid \forall \alpha L \mid \mu \alpha L^{(*)} \mid L \multimap M$$

$$A, B ::= L \mid \forall \alpha A \mid L \multimap B \mid A \Rightarrow B.$$

$(*)$: only if α occurs only positively in L.

Thus the linear formulas are the formulas that do not contain any $\Rightarrow$.

Typing rules of $DIAL_{lin}$

typing judgement : $\underbrace{\Gamma}_{\text{non-linear variables}} ; \overbrace{\Delta}^{\text{affine variables}} \vdash t : A$

$$\frac{}{x : A; \vdash x : A}\,(ax1) \qquad \frac{}{; x : L \vdash x : L}\,(ax2)$$

$$\frac{\Gamma; \Delta \vdash t : \mu\alpha L}{\Gamma; \Delta \vdash t : L[\mu\alpha L/\alpha]}\,(\mu_e) \qquad \frac{\Gamma; \Delta \vdash t : L[\mu\alpha L/\alpha]}{\Gamma; \Delta \vdash t : \mu\alpha L}\,(\mu_i)$$

$$\frac{\Gamma; \Delta \vdash t : A \qquad \alpha \notin FV(\Gamma; \Delta)}{\Gamma; \Delta \vdash t : \forall\alpha A}\,(\forall_i) \qquad \frac{\Gamma; \Delta \vdash t : \forall\alpha A}{\Gamma; \Delta \vdash t : A[L/\alpha]}\,(\forall_e)$$

$$\frac{\Gamma_1; \Delta \vdash t : A \Rightarrow B \qquad \Gamma_2; \vdash u : A}{\Gamma_1, \Gamma_2; \Delta \vdash tu : B}\,(\Rightarrow_e) \qquad \frac{\Gamma, z : A; \Delta \vdash t : B}{\Gamma; \Delta \vdash \lambda z\, t : A \Rightarrow B}\,(\Rightarrow_i)$$

$$\frac{\Gamma_1; \Delta_1 \vdash t : L \multimap B \qquad \Gamma_2; \Delta_2 \vdash u : L}{\Gamma_1, \Gamma_2; \Delta_1, \Delta_2 \vdash tu : B}\,(\multimap_e) \qquad \frac{\Gamma; \Delta, z : L \vdash t : B}{\Gamma; \Delta \vdash \lambda z\, t : L \multimap B}\,(\multimap_i)$$

$$\frac{\Gamma, x : A, y : A; \Delta \vdash t : B}{\Gamma, z : A; \Delta \vdash t[z/x, z/y] : B}\,(Contr) \qquad \frac{\Gamma; \Delta, x : L \vdash t : B}{\Gamma, x : L; \Delta \vdash t : B}\,(Derel) \qquad \frac{\Gamma; \Delta \vdash t : B}{\Gamma, \Gamma'; \Delta, \Delta' \vdash t : B}\,(Weak)$$

Typing rules of $DIAL_{lin}$

typing judgement : $\underbrace{\Gamma}_{\text{non-linear variables}} ; \overbrace{\Delta}^{\text{affine variables}} \vdash t : A$

$$\frac{\Gamma; \Delta \vdash t : \forall \alpha A}{\Gamma; \Delta \vdash t : A[L/\alpha]} (\forall_e)$$

$$\frac{\Gamma_1; \Delta \vdash t : A \Rightarrow B \qquad \Gamma_2; \vdash u : A}{\Gamma_1, \Gamma_2; \Delta \vdash tu : B} (\Rightarrow_e)$$

$$\frac{\Gamma, x : A, y : A; \Delta \vdash t : B}{\Gamma, z : A; \Delta \vdash t[z/x, z/y] : B} (Contr)$$

Church numerals and words

- Church naturals :

$$\begin{aligned} \mathsf{N}^\bullet &\equiv \forall\alpha(\alpha \multimap \alpha) \Rightarrow (\alpha \multimap \alpha) \\ \mathsf{n}^\bullet &= \lambda fa.\underbrace{f(\ldots f}_{n \text{ times}}(a)\ldots) \\ \mathsf{mult}^\bullet &= \lambda n\lambda m\lambda f\,.n(m\,f) \\ \mathsf{mon}^\bullet_n &= \lambda x\lambda f\,.\underbrace{x(\ldots(x}_{n \text{ times}}\,f)) : \mathsf{N}^\bullet \Rightarrow \mathsf{N}^\bullet \end{aligned}$$

- Church words :

$$\begin{aligned} \mathsf{W}^\bullet &\equiv \forall\alpha(\alpha \multimap \alpha) \Rightarrow (\alpha \multimap \alpha) \Rightarrow (\alpha \multimap \alpha) \\ \mathsf{w}^\bullet &= \lambda f_0\lambda f_1\lambda a.f_{i_1}(\ldots f_{i_n}(a)\ldots) \end{aligned}$$

- Iteration : only linear functions (of type $\alpha \multimap \alpha$ where α is linear). Hence we cannot encode exponentiation which needs to iterate a function like

$$\mathit{double} = \lambda n.\lambda fa.n\,f\,(n\,f\,a) : \mathsf{N}^\bullet \Rightarrow \mathsf{N}^\bullet$$

Scott numerals

- Scott numerals are represented by the linear type $\mathsf{N}^\circ \equiv \mu\beta\forall\alpha(\beta \multimap \alpha) \multimap (\alpha \multimap \alpha)$. They have constant time successor, predecessor and discriminator, but don't support iteration.

$$
\begin{aligned}
\epsilon^\circ &= \lambda xyz.z \\
(0\mathsf{w})^\circ &= \lambda xyz.x(\mathsf{w}^\circ) \\
(1\mathsf{w})^\circ &= \lambda xyz.y(\mathsf{w}^\circ) \\
\mathsf{queue}^\circ &= \lambda w.(w(\lambda x.x)(\lambda x.x)) : \mathsf{W}^\circ \multimap \mathsf{W}^\circ \\
\\
e.g \quad (101)^\circ &= \lambda xyz.y(\lambda xyz.x(\lambda xyz.y(\lambda xyz.z)))
\end{aligned}
$$

- The only inhabitants of W° are scott words w°.

Scott words

- Scott words are represented by the linear type $\mathrm{W}^\circ \equiv \mu\beta\forall\alpha(\beta \multimap \alpha) \multimap (\beta \multimap \alpha) \multimap (\alpha \multimap \alpha)$. They have constant time successor, predecessor and discriminator, but don't support iteration.

$$
\begin{aligned}
\epsilon^\circ &= \lambda xyz.z \\
(0\mathrm{w})^\circ &= \lambda xyz.x(\mathrm{w}^\circ) \\
(1\mathrm{w})^\circ &= \lambda xyz.y(\mathrm{w}^\circ) \\
\mathrm{queue}^\circ &= \lambda w.(w(\lambda x.x)(\lambda x.x)) : \mathrm{W}^\circ \multimap \mathrm{W}^\circ \\
\\
e.g \quad (101)^\circ &= \lambda xyz.y(\lambda xyz.x(\lambda xyz.y(\lambda xyz.z)))
\end{aligned}
$$

- The only inhabitants of W° are scott words w°.

Results

- Informally, we claim that

$$\mathrm{W}^{\bullet} \Rightarrow \mathrm{W}^{\circ} = \mathbf{PTIME}$$

- $\mathrm{W}^{\bullet} \Rightarrow \mathrm{W}^{\circ}$ is expressive enough : we can type Church monomials and we can encode the one-step transition function of a Turing Machine using a linear type, we can then iterate it using a monomial.
  $\rightarrow$ **PTIME**-completeness

- $\mathrm{W}^{\bullet} \Rightarrow \mathrm{W}^{\circ}$ is not too permissive : we cannot type exponentials.
  $\rightarrow$ **PTIME**-soundness

Results

PTIME-completeness

For every polynomial time function $f : \{0,1\}^* \to \{0,1\}^*$, there exists a λ-term t_f of type $W^\bullet \Rightarrow W^\circ$ in $\mathbf{DIAL}_{lin}$ such that given $w \in \{0,1\}^*$, we have

$$[\![t_f w^\bullet]\!]_\beta = f(w)^\circ$$

$\to$ usual encoding of Turing Machines in $\mathbf{DIAL}_{lin}$

PTIME-soundness

For every λ-term t of type $W^\bullet \Rightarrow W^\circ$, the associated function $f_t : \{0,1\}^* \to \{0,1\}^*$ defined by

$$f_t(w_1) = w_2 \Leftrightarrow [\![t w_1{}^\bullet]\!]_\beta = w_2{}^\circ$$

is a polynomial time function.

Results

PTIME-completeness

For every polynomial time function $f : \{0,1\}^* \to \{0,1\}^*$, there exists a λ-term t_f of type $\mathrm{W}^\bullet \Rightarrow \mathrm{W}^\circ$ in $\mathbf{DIAL}_{lin}$ such that given $w \in \{0,1\}^*$, we have

$$[\![t_f \mathrm{w}^\bullet]\!]_\beta = \mathrm{f}(\mathrm{w})^\circ$$

$\to$ usual encoding of Turing Machines in $\mathbf{DIAL}_{lin}$

PTIME-soundness

For every λ-term t of type $\mathrm{W}^\bullet \Rightarrow \mathrm{W}^\circ$, the associated function $f_t : \{0,1\}^* \to \{0,1\}^*$ defined by

$$f_t(w_1) = w_2 \Leftrightarrow [\![t \mathrm{w}_1{}^\bullet]\!]_\beta = \mathrm{w}_2{}^\circ$$

is a polynomial time function.

Weak call-by-value and time measure (Dal Lago & Martini) (1/3)

- Terms $t, u ::= x \mid \lambda x.t \mid tu$
- Values $v ::= x \mid \lambda x.t$
- Reduction

$$\frac{}{(\lambda x.t)v \to t[v/x]} \qquad \frac{t_1 \to t_2}{t_1 u \to t_2 u} \qquad \frac{t_1 \to t_2}{u t_1 \to u t_2}$$

- Notations : We note $|t|$ the size of t. We denote by $t \Downarrow$ the fact that t normalizes for this strategy. If it exists, $[\![t]\!]_{CBV}$ is the normal form of t for this strategy (in contrast to $[\![t]\!]_{\beta}$ which is the β-normal form).

Weak call-by-value and time measure (Dal Lago & Martini) (2/3)

- Cost measure

$$\frac{}{t \overset{0}{\twoheadrightarrow} t} \qquad \frac{t \to u \quad n = max\{|u| - |t|, 1\}}{t \overset{n}{\twoheadrightarrow} u} \qquad \frac{s \overset{n}{\twoheadrightarrow} t \qquad t \overset{m}{\twoheadrightarrow} u}{s \overset{n+m}{\twoheadrightarrow} u}$$

If the variable x is affine in t (that is, x appears at most once in t), then

$$(\lambda x.t)u \to t[u/x]$$

The duplication of the argument in the following reduction is taken into account :

$$(\mathrm{n}^{\bullet} g) \to (\lambda a.(\underbrace{g \ldots (g}_{n \text{ times}} \ a) \ldots))$$

Weak call-by-value and time measure (Dal Lago & Martini) (2/3)

- Cost measure

$$\frac{}{t \overset{0}{\twoheadrightarrow} t} \qquad \frac{t \to u \quad n = max\{|u| - |t|, 1\}}{t \overset{n}{\twoheadrightarrow} u} \qquad \frac{s \overset{n}{\twoheadrightarrow} t \qquad t \overset{m}{\twoheadrightarrow} u}{s \overset{n+m}{\twoheadrightarrow} u}$$

If the variable x is affine in t (that is, x appears at most once in t), then

$$(\lambda x.t)u \overset{1}{\twoheadrightarrow} t[u/x]$$

The duplication of the argument in the following reduction is taken into account :

$$(\mathsf{n}^{\bullet} g) \to (\lambda a.(\underbrace{g \ldots (g}_{n \text{ times}} \ a) \ldots))$$

Weak call-by-value and time measure (Dal Lago & Martini) (2/3)

- Cost measure

$$\frac{}{t \overset{0}{\twoheadrightarrow} t} \qquad \frac{t \to u \quad n = max\{|u| - |t|, 1\}}{t \overset{n}{\twoheadrightarrow} u} \qquad \frac{s \overset{n}{\twoheadrightarrow} t \qquad t \overset{m}{\twoheadrightarrow} u}{s \overset{n+m}{\twoheadrightarrow} u}$$

If the variable x is affine in t (that is, x appears at most once in t), then

$$(\lambda x.t)u \overset{1}{\twoheadrightarrow} t[u/x]$$

The duplication of the argument in the following reduction is taken into account :

$$(\lambda fa.(\underbrace{f \ldots (f}_{n \text{ times}} \ a) \ldots))g \overset{(n\times|g|+2)-(n+|g|+3)}{\twoheadrightarrow} (\lambda a.(\underbrace{g \ldots (g}_{n \text{ times}} \ a) \ldots))$$

Weak call-by-value and time measure (Dal Lago & Martini) (2/3)

- Cost measure

$$\frac{}{t \overset{0}{\twoheadrightarrow} t} \qquad \frac{t \to u \quad n = max\{|u| - |t|, 1\}}{t \overset{n}{\twoheadrightarrow} u} \qquad \frac{s \overset{n}{\twoheadrightarrow} t \qquad t \overset{m}{\twoheadrightarrow} u}{s \overset{n+m}{\twoheadrightarrow} u}$$

If the variable x is affine in t (that is, x appears at most once in t), then

$$(\lambda x.t)u \overset{1}{\twoheadrightarrow} t[u/x]$$

The duplication of the argument in the following reduction is taken into account :

$$(\mathsf{n}^\bullet g) \overset{((n-1)\times|g|-|g|-1)}{\twoheadrightarrow} (\lambda a.(\underbrace{g \ldots (g}_{n \text{ times}} \ a)\ldots))$$

Weak call-by-value and time measure (Dal Lago & Martini) (3/3)

If $t \Downarrow$ then there exists a unique $n \in \mathbb{N}$ such that $t \xrightarrow{n} \!\!\!\!\rightarrow [\![t]\!]_{CBV}$. We denote it by $Time(t)$.

Theorem (2006, Dal Lago& Martini)

There exists a Turing machine M_{eval} with the following property : given a λ-term t such that $t \Downarrow$ and $TS(t) = Time(t) + |t| = n$, M_{eval} computes $[\![t]\!]_{CBV}$ in time $O(n^4)$.

$\rightarrow$ allows us to reason only on λ-calculus instead of Turing Machines.

How is it proved ?

PTIME-Soundness

For every λ-term t of type $\mathsf{W}^\bullet \Rightarrow \mathsf{W}^\circ$, the associated function $f_t : \{0,1\}^* \to \{0,1\}^*$ defined by

$$f_t(w_1) = w_2 \Leftrightarrow [\![t \mathsf{w}_1{}^\bullet]\!]_\beta = \mathsf{w}_2{}^\circ$$

is a polynomial time function.

This is proved in two steps :

1. Each bit of the result $[\![tw]\!]_\beta$ can be computed in polynomial time (using weak call-by-value strategy).
2. The length of $[\![tw]\!]_\beta$ is polynomial of $|w|$ (not proved here).

Each of these statements is proved using a variant of Dal Lago & Hofmann **realizability** technique.

How is it proved ?

PTIME-Soundness

For every λ-term t of type $\mathsf{W}^\bullet \Rightarrow \mathsf{W}^\circ$, the associated function $f_t : \{0,1\}^* \to \{0,1\}^*$ defined by

$$f_t(w_1) = w_2 \Leftrightarrow [\![t \mathsf{w}_1{}^\bullet]\!]_\beta = \mathsf{w}_2{}^\circ$$

is a polynomial time function.

This is proved in two steps :

1. Each bit of the result $[\![tw]\!]_\beta$ can be computed in polynomial time (using weak call-by-value strategy).
2. The length of $[\![tw]\!]_\beta$ is polynomial of $|w|$ (not proved here).

Each of these statements is proved using a variant of Dal Lago & Hofmann **realizability** technique.

How is it proved ?

PTIME-Soundness

For every λ-term t of type $W^\bullet \Rightarrow W^\circ$, the associated function $f_t : \{0,1\}^* \to \{0,1\}^*$ defined by

$$f_t(w_1) = w_2 \Leftrightarrow [\![t w_1{}^\bullet]\!]_\beta = w_2{}^\circ$$

is a polynomial time function.

This is proved in two steps :

1. Each bit of the result $[\![tw]\!]_\beta$ can be computed in polynomial time (using weak call-by-value strategy).
2. The length of $[\![tw]\!]_\beta$ is polynomial of $|w|$ (not proved here).

Each of these statements is proved using a variant of Dal Lago & Hofmann **realizability** technique.

The core of the realizability framework

Realizability is used to capture computational properties and to give meaning to the logic.

- A language Λ of *realizers* : the programs we want to state properties on. In Dal Lago& Hofmann, realizers are closed values for the WCBV. Here we take all the closed λ-terms.

- A relation $t \Vdash A$, where t is a realizer and A a $\mathbf{DIAL}_{lin}$ formula, defined only by the structure of A and the computational behaviour of t. This relation informally means
"*t is a program that behaves with respect to the specification A*"

- An adequacy theorem : "If $\vdash t : A$ then $t \Vdash A$".

The core of the realizability framework

- A language Λ of *realizers* : the programs we want to state properties on. In Dal Lago & Hofmann, realizers are closed values for the WCBV. Here we take all the closed λ-terms.

- A set Π of *majorizers*, used to impose resource bound on the realizers. In Dal Lago& Hofmann, Π can be any resource monoids. Here we take higher-order additive terms.

- A relation $(t, p) \Vdash A$, where t is a realizer, p a majorizer and A a $\mathbf{DIAL}_{lin}$ formula. This means
  "*t is a program whose specification is A and that uses at most p resources to run*"
- An adequacy theorem : "If $\vdash t : A$ then there exists p such that $(t, p) \Vdash A$".

Dal Lago & Hofmann's realizability

Realizers are closed values.

The set of majorizers is a resource monoid $(M, +, 0, \leq, D)$:

- $(M, +, 0, \leq)$ is a preordered commutative monoid.
- $D(.,.)$ is a kind of distance between elements of M.

Example : $(\mathbb{N}, +, 0, \leq, (x, y) \mapsto |y - x|)$.

The arrow construction : $t, p \Vdash A \multimap B$ iff for every argument $u, q \Vdash A$, we have :

- The result is bounded by some majorizer r : $[\![tu]\!], r \Vdash B$
- The time needed for the computation of this result is bounded :

$$Time(tu) \leq D(p + q, r)$$

Higher-order additive terms as resources representation

- Simply typed λ-terms with base constants :
  - Integers (base type), $n : o$.
  - Addition on integers, $+ : o \to o \to o$.

- We identify terms by $\alpha\beta\eta$-equivalence and usual arithmetic equivalences.

- Examples :
  - $\lambda n.(n + 20) : o \to o$
  - $\lambda f\, \lambda n.(\underbrace{f(n) + f(n) + ... + f(n)}_{1000\ \textit{times}}) : (o \to o) \to o \to o$

- For every higher-order additive term p, we can lower it to base type o. The lowering operator is denoted by $\downarrow p$.

- A last notation : $p + n = \lambda x_1 ... \lambda x_n.(p(x_1, ..., x_n) + n)$.

o-translation (1/2)

- Informally, $t, p \Vdash A$ we require that the higher-order skeleton of p follows the structure of A. That is, we define a traduction $o(A)$ of the formula A of $\mathbf{DIAL}_{lin}$ into the simple types.

- $o(L) = o$: we only need integers to bound linear realizers runtime.

- $o(L \multimap B) = o(B)$

- $o(A \Rightarrow B) = o(A) \to o(B)$

- $o(\forall \alpha . A) = o(A)$: the quantifier is linear

o-translation (2/2)

For example, the translation of the Scott word type (which is a linear type) is

$$o(\mathsf{W}^{\circ}) = o$$

the translation of the Church word type is

$$\begin{aligned} o(\mathsf{W}^{\bullet}) &= o(\forall\alpha.(\alpha \multimap \alpha) \Rightarrow (\alpha \multimap \alpha) \Rightarrow \alpha \multimap \alpha) \\ &= o \to o \to o \end{aligned}$$

Saturated Sets

τ-saturated set

If τ is a higher-order additive type, we say that $X \subseteq \Lambda \times \Pi$ is *saturated set of type* τ if whenever $(t, p) \in X$, p is a closed higher order additive term of type τ and the following holds :

- $TS(t) \leq \downarrow p$.
- $(t,\ p + n) \in X$ for every $n \in \mathbb{N}$.
- Others properties that mimic structural rules and identity (weakening, contraction, exchange, identity). For example, the exchange condition implies : If $(\lambda x_1 x_2 . t, p) \in X$ then $(\lambda x_2 x_1 . t, p) \in X$.

In particular, $\{\ (t, n) \mid t \Downarrow$ and $TS(t) \leq n\ \}$ is the greatest o saturated set.

Time Realizability : the construction

(Realizability)

We define the relation $t, p \Vdash_\eta A$, where $t \in \Lambda$, p is a closed higher order additive term of type $o(A)$ and η is a valuation (from atoms to o-saturated set).

The definition proceeds by induction on A.

- $t,\ n \Vdash_\eta \alpha$ iff $(t,\ n) \in \eta(\alpha)$.
- $t,\ p \Vdash_\eta L \multimap A$ iff $TS(t) \leq\!\downarrow p$ and $u, m \Vdash_\eta L$ implies $tu, p + m \Vdash_\eta A$ for every u, m.
- $t,\ p \Vdash_\eta B \Rightarrow A$ iff $TS(t) \leq\!\downarrow p$ and $u, q \Vdash_\eta B$ implies $tu, p(q) \Vdash_\eta A$ for every u, q.

Time Realizability : the construction

(Realizability)

We define the relation $t, p \Vdash_\eta A$, where $t \in \Lambda$, p is a closed higher order additive term of type $o(A)$ and η is a valuation (from atoms to o-saturated set).

The definition proceeds by induction on A.

- $t,\ n \Vdash_\eta \alpha$ iff $(t,\ n) \in \eta(\alpha)$.
- $t,\ p \Vdash_\eta L \multimap A$ iff $TS(t) \leq\!\downarrow p$ and $u, m \Vdash_\eta L$ implies $tu, p + m \Vdash_\eta A$ for every u, m.
- $t,\ p \Vdash_\eta B \Rightarrow A$ iff $TS(t) \leq\!\downarrow p$ and $u, q \Vdash_\eta B$ implies $tu, p(q) \Vdash_\eta A$ for every u, q.

Time Realizability : the construction

(Realizability)

We define the relation $t, p \Vdash_\eta A$, where $t \in \Lambda$, p is a closed higher order additive term of type $o(A)$ and η is a valuation (from atoms to o-saturated set).

The definition proceeds by induction on A.

- $t,\ n \Vdash_\eta \alpha$ iff $(t,\ n) \in \eta(\alpha)$.
- $t,\ p \Vdash_\eta L \multimap A$ iff $TS(t) \leq\!\downarrow p$ and $u, m \Vdash_\eta L$ implies $tu, p + m \Vdash_\eta A$ for every u, m.
- $t,\ p \Vdash_\eta B \Rightarrow A$ iff $TS(t) \leq\!\downarrow p$ and $u, q \Vdash_\eta B$ implies $tu, p(q) \Vdash_\eta A$ for every u, q.

Time Realizability : the construction

(Realizability)

We define the relation $t, p \Vdash_\eta A$, where $t \in \Lambda$, p is a closed higher order additive term of type $o(A)$ and η is a valuation (from atoms to o-saturated set).

The definition proceeds by induction on A.

- $t,\ n \Vdash_\eta \alpha$ iff $(t,\ n) \in \eta(\alpha)$.
- $t,\ p \Vdash_\eta L \multimap A$ iff $TS(t) \leq\downarrow p$ and $u, m \Vdash_\eta L$ implies $tu, p + m \Vdash_\eta A$ for every u, m.
- $t,\ p \Vdash_\eta B \Rightarrow A$ iff $TS(t) \leq\downarrow p$ and $u, q \Vdash_\eta B$ implies $tu, p(q) \Vdash_\eta A$ for every u, q.

Time Realizability : the remaining cases

- The universal quantifier construction is :

$$(t, p) \Vdash_\eta \forall \alpha A \text{ iff for every o-saturated set } X,\ (t, p) \Vdash_{\eta\{\alpha \leftarrow X\}} A$$

$\rightarrow$ corresponds to a linear quantifier

- We can use a well chosen Tarski least fixpoint on some operator to define the interpretation of the μ construction and to obtain that :

$\rightarrow \quad (t, p) \Vdash_\eta \mu \alpha L \Leftrightarrow (t, p) \Vdash_\eta L[\mu \alpha L / \alpha]$

Time Realizability : the remaining cases

- The universal quantifier construction is :

$$(t, p) \Vdash_\eta \forall \alpha A \text{ iff for every o-saturated set } X,\ (t, p) \Vdash_{\eta\{\alpha \leftarrow X\}} A$$

$\rightarrow$ corresponds to a linear quantifier

- We can use a well chosen Tarski least fixpoint on some operator to define the interpretation of the μ construction and to obtain that :

$\rightarrow \quad (t, p) \Vdash_\eta \mu\alpha L \Leftrightarrow (t, p) \Vdash_\eta L[\mu\alpha L/\alpha]$

Time Realizability

- We can prove that for each formula A, the set of (t, p) such that $t, p \Vdash A$ is $o(A)$-saturated.
- In particular, $t, p \Vdash_\eta A$ implies $TS(t) \leq\downarrow p$.
- For every $n \in \mathbb{N}$, we have $\mathrm{n}^\bullet, p_n \Vdash \mathrm{N}^\bullet$ with $p_n = \lambda z.n(z+3)+3 : o \to o$
- For every $w \in \{0,1\}^n$, we have $\mathrm{w}^\bullet, q_w \Vdash \mathrm{W}^\bullet$ with $q_w = \lambda z_0 z_1.|w|(z_0 + z_1 + 3) + 3 : o \to o \to o$

Time Realizability : adequacy lemma

Adequacy (very simplified version)

If $\vdash t : A$ is derivable in $\mathbf{DIAL}_{lin}$, then there exists a majorizer $p : o(A)$ such that for any valuation η we have $t, p \Vdash_\eta A$

Applying adequacy (1/2)

We have observed that every Church word $\mathrm{w}^\bullet$ is bounded by a linear majorizer $q_w = \lambda z_0 z_1 . |w|(z_0 + z_1 + 3) + 3$.

Weak soundness

Let L be a linear formula. If we have $\vdash t : \mathrm{W}^\bullet \Rightarrow L$, then there exists a polynomial P such that for every $w \in \{0,1\}^*$, $\mathit{Time}(t\mathrm{w}^\bullet) \leq P(|w|)$.

If $w \in \{0,1\}^*$, then because of adequacy, there is some $p : (o \to o \to o) \to o$ such that $t, p \Vdash \mathrm{W}^\bullet \Rightarrow L$ then $t\mathrm{w}^\bullet, p(q_w) \Vdash L$. And in particular $\mathit{TS}(t\mathrm{w}^\bullet) \leq p(q_w)$.

But we can show that $p(q_w)$ is polynomial in $|w|$.

Applying adequacy (2/2)

We define booleans : $\mathsf{B}_2^\circ = \forall\alpha.\alpha \multimap \alpha \multimap \alpha$, $\mathsf{b}_0^\circ = \lambda xy.x$ and $\mathsf{b}_1^\circ = \lambda xy.y$.

P-soundness for predicates

If $t : \mathsf{W}^\bullet \Rightarrow \mathsf{B}_2^\circ$, then the predicate $f_t : \{0,1\}^* \to \{0,1\}$ defined by $f_t(w) = 1 \Leftrightarrow [\![t\mathsf{w}^\bullet]\!]_\beta = \mathsf{b}_1{}^\circ$ is a polynomial time predicate.

This is basically because when $\lambda x.tx\mathsf{b}_0^\circ\mathsf{b}_1^\circ : \mathsf{W}^\bullet \Rightarrow \mathsf{B}_2^\circ$ and because $(\lambda x.tx\mathsf{b}_0^\circ\mathsf{b}_1^\circ)\mathsf{w}^\bullet$ reduces either to b_0° or b_1° by the weak call-by-value strategy.

- Using the same kind of trick, for each $t : \mathsf{W}^\bullet \Rightarrow \mathsf{W}^\circ$, and for each $w \in \{0,1\}^*$, we can compute in polynomial time each bit of the result $t\mathsf{w}^\bullet$.
- We can prove that the size of the result $[\![t\mathsf{w}^\bullet]\!]_\beta$, that is the number of bits of the output word, is bounded by a polynomial in the size of $|w|$ (using another realizability argument).

Conclusion and remaining questions

We have used a variant of Dal Lago & Hofmann realizability framework to prove that in $\mathbf{DIAL}_{lin}$, $Church \Rightarrow Scott = Ptime$, which recasts the original result of Leivant and Marion.
Questions :

- Can we drop the restriction on $\multimap$?
- We have to deal with a dual type system. Can we deal directly with the ! connective.
- Saturation by biorthogonality ?
- Can we find other typing systems to accomodate different complexity classes like **PSPACE** ?

Thank you !

Comparisons with Dal Lago & Hofmann realizability

In Dal Lago & Hofmann realizability : realizers are closed values.
The definition would be

- $t, p \Vdash A \Rightarrow B$ iff every time $u, q \Vdash A$ then
  - $[\![tu]\!]_{CBV}, p(q) \Vdash B$
  - $TS(tu) \leq\downarrow (p(q))$

- $t,\ p \Vdash A \Rightarrow B$ iff $TS(t) \leq\downarrow p$ and $u, q \Vdash B$ implies $tu, p(q) \Vdash B$ for every u, q.

Comparisons with Dal Lago & Hofmann realizability

In Dal Lago & Hofmann realizability : realizers are closed values.
The definition would be

- $t, p \Vdash A \Rightarrow B$ iff every time $u, q \Vdash A$ then there exists some $\bar{p} : o(B)$ such that
  - $[\![tu]\!]_{CBV}, \bar{p} \Vdash B$
  - $\bar{p} \leq p(q)$
  - $TS(tu) + \downarrow \bar{p} \leq \downarrow (p(q))$

- $t,\ p \Vdash A \Rightarrow B$ iff $TS(t) \leq \downarrow p$ and $u, q \Vdash B$ implies $tu, p(q) \Vdash B$ for every u, q.