Finite semantics of linear polymorphism

Nguyễn Lê Thành Dũng (a.k.a. Tito) — nltd@nguyentito.eu
LIPN, Université Paris 13
Joint work with T. Seiller, P. Pistone and L. Tortora de Falco

RIMS Computer Science seminar, Kyoto, March 28th, 2019

Theorem

Second-order Multiplicative-Additive Linear Logic (MALL2) *admits a non-trivial finite denotational semantics.*

This talk:

1. Motivations and applications
   1.1 Finite semantics for monomorphic type systems
   1.2 Implicit complexity in 2nd-order Elementary Linear Logic
2. One such model (which is also effective):
   coherence spaces + normal functors

An example in the simply typed λ-calculus (STλ)

Q: what languages are recognized by simply typed λ-terms?

Church encodings:

- For $w \in \{0,1\}^*$, $w : \mathsf{Str}[A]$ for any simple type A (meta-$\forall$)
  - $\mathsf{Str}[A] = (A \to A) \to (A \to A) \to (A \to A)$
  - $\overline{w} = \lambda f_0.\, \lambda f_1.\, \lambda x.\, f_{w[0]}\, (\ldots (f_{w[n-1]}\, x) \ldots)$
- $\mathsf{Bool} = o \to o \to o$ (o base type)

Choose a simple type A, and a term $t : \mathsf{Str}[A] \to \mathsf{Bool}$
$\longrightarrow$ defines language $\mathcal{L}(t) = \{w \in \{0,1\}^* \mid t\, \overline{w} \to^*_\beta \mathtt{true}\}$.

An example in the simply typed λ-calculus (STλ)

Q: what languages are recognized by simply typed λ-terms?

Church encodings:

- For $w \in \{0,1\}^*$, $w : \mathsf{Str}[A]$ for any simple type A (meta-$\forall$)
  - $\mathsf{Str}[A] = (A \to A) \to (A \to A) \to (A \to A)$
  - $\overline{w} = \lambda f_0.\, \lambda f_1.\, \lambda x.\, f_{w[0]}\, (\ldots (f_{w[n-1]}\, x) \ldots)$
- $\mathsf{Bool} = o \to o \to o$ (o base type)

Choose a simple type A, and a term $t : \mathsf{Str}[A] \to \mathsf{Bool}$
$\longrightarrow$ defines language $\mathcal{L}(t) = \{w \in \{0,1\}^* \mid t\,\overline{w} \to^*_\beta \mathtt{true}\}$.

Theorem (Hillebrand & Kanellakis, LICS'96)

The languages decided by STλ*-terms of type* $\mathsf{Str}[A] \to \mathsf{Bool}$ *are exactly the* regular *languages.*

Regular languages in $\mathrm{ST}\lambda$

Theorem (Hillebrand & Kanellakis, LICS'96)

For any type A and any $\mathrm{ST}\lambda$*-term* $t : \mathsf{Str}[A] \to \mathsf{Bool}$*, the language* $\mathcal{L}(t) = \{w \in \{0,1\}^* \mid t\,\overline{w} \to_\beta^* \mathtt{true}\}$ *is* regular.

Part 1 of proof.

Fix type A. Any *denotational semantics* $[\![-]\!]$ quotients words:

$$w \in \{0,1\}^* \rightsquigarrow \overline{w} : \mathsf{Str}[A] \rightsquigarrow [\![\overline{w}]\!]_{\mathsf{Str}[A]} \in [\![\mathsf{Str}[A]]\!]$$

When $[\![-]\!]$ *non-trivial* ($[\![\mathtt{true}]\!] \neq [\![\mathtt{false}]\!]$), $[\![\overline{w}]\!]_{\mathsf{Str}[A]}$ determines behavior of w w.r.t. all $\mathsf{Str}[A] \to \mathsf{Bool}$ terms:

$$w \in \mathcal{L}(t) \iff t\,\overline{w} \to_\beta^* \mathtt{true} \iff [\![t\,\overline{w}]\!] = [\![t]\!]([\![\overline{w}]\!]) = [\![\mathtt{true}]\!]$$

Goal: to decide $\mathcal{L}(t)$, compute $w \mapsto [\![\overline{w}]\!]$ in some model of $\mathrm{ST}\lambda$.

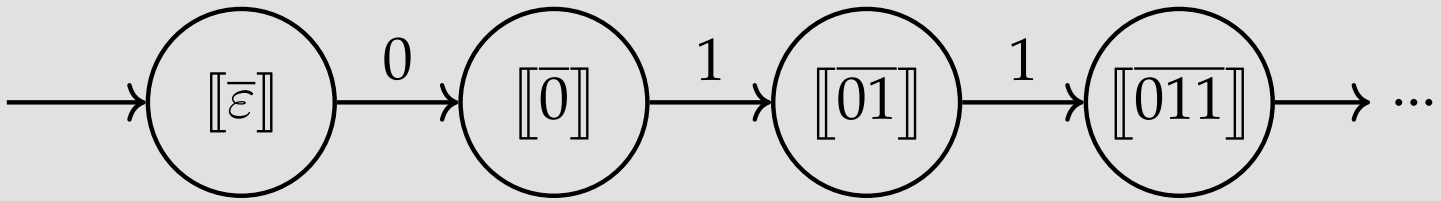
Regular languages in $\mathrm{ST}\lambda$

Theorem (Hillebrand & Kanellakis, LICS'96)

For any type A and any $\mathrm{ST}\lambda$*-term* $t : \mathsf{Str}[A] \to \mathsf{Bool}$*, the language* $\mathcal{L}(t) = \{w \in \{0,1\}^* \mid t\,\overline{w} \to_\beta^* \mathtt{true}\}$ *is* regular.

Part 2 of proof.

We use $[\![-]\!] : \mathrm{ST}\lambda \to \mathsf{FinSet}$ to build a DFA with states $Q = [\![\mathsf{Str}[A]]\!]$, acceptation as $[\![t]\!](-) = [\![\mathtt{true}]\!]$.



$$w \in \mathcal{L}(t) \iff [\![t]\!]\left([\![\overline{w}]\!]_{\mathsf{Str}[A]}\right) = [\![\mathtt{true}]\!] \iff w \text{ accepted}$$

$\longrightarrow$ *semantic evaluation* argument. □

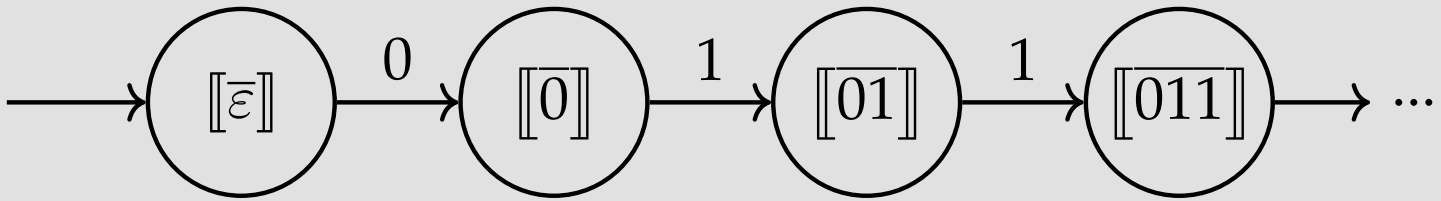
Regular languages in $\mathrm{ST}\lambda$

Theorem (Hillebrand & Kanellakis, LICS'96)

For any type A and any $\mathrm{ST}\lambda$*-term* $t : \mathsf{Str}[A] \to \mathsf{Bool}$*, the language* $\mathcal{L}(t) = \{w \in \{0,1\}^* \mid t\,\overline{w} \to_\beta^* \mathtt{true}\}$ *is* regular.

Part 2 of proof.

We use $[\![-]\!] : \mathrm{ST}\lambda \to \mathsf{FinSet}$ to build a DFA with states $Q = [\![\mathsf{Str}[A]]\!]$, acceptation as $[\![t]\!](-) = [\![\mathtt{true}]\!]$.
($|Q| < \infty$, e.g. $2^{2^{33}}$ when $A = \mathsf{Bool}$)



$$w \in \mathcal{L}(t) \iff [\![t]\!]\left([\![\overline{w}]\!]_{\mathsf{Str}[A]}\right) = [\![\mathtt{true}]\!] \iff w \text{ accepted}$$

$\longrightarrow$ *semantic evaluation* argument. □

Other applications / Moral of the story

Semantic evaluation technique for complexity:

- Implicit complexity results of the 90's in STλ
- Terui, RTA'12: STλ normalization at fixed order*
- Applications to System T / PCF (cf. L. Kristiansen's work)

Also, correspondence Church encodings / finite automata
$\rightarrow$ leads to semantic approach to *higher-order model checking*
(Salvati–Walukiewicz, Grellois–Melliès*, ...)

For applications of denotational semantics, finiteness is key.

(* using linear logic semantics)

Finite semantics for polymorphism

Previous examples work for *monomorphic* type systems.

This seems impossible with impredicative polymorphism.
For instance in System F:

Proposition

Any non-trivial semantics must be injective on $\forall X.\, \mathsf{Str}[X]$.

Finite semantics for polymorphism

Previous examples work for *monomorphic* type systems.

This seems impossible with impredicative polymorphism.
For instance in System F:

Proposition

Any non-trivial semantics must be injective on $\forall X.\, \mathsf{Str}[X]$.

This example relies fundamentally on *non-linearity*.

Actually, there are finite models:

- with impredicative quantification;
- with exponentials;
- but not with both!

A complexity application for MALL2

Theorem

MALL2 *has a non-trivial finite semantics.*

By adapting Hillebrand & Kanellakis's proof:

Corollary

The proofs of $!\mathsf{Str} \multimap !!\mathsf{Bool}$ *in 2nd order Elementary Linear Logic* (ELL2) *decide exactly the* regular *languages.*

(ELL2 = MALL2 + restricted exponentials;
using suitable definitions of Str and Bool)

Contrast with:

Theorem (Baillot, APLAS'11)

The proofs of $!\mathsf{Str} \multimap !!\mathsf{Bool}$ *in 2nd order Elementary Affine Logic* w/ recursive types *decide exactly* polynomial time *predicates.*

Towards a syntactic model

Remark: in propositional MALL (MALL0), each formula has finitely many cut-free proofs.
$\rightarrow$ Hope for a *syntactic model* of MALL2, injective on MALL0.

Difficulty of 2nd order case: arbitrarily large $\exists$ witnesses.

Towards a syntactic model

Remark: in propositional MALL (MALL0), each formula has finitely many cut-free proofs.
$\rightarrow$ Hope for a *syntactic model* of MALL2, injective on MALL0.

Difficulty of 2nd order case: arbitrarily large $\exists$ witnesses.
Solution: *observational quotient* of the syntax.

- Choose observations which "cannot inspect witnesses"
- Intuition from programming languages: $\exists$ = abstract types, dual to $\forall$ = generic programs

Equivalence for propositional observations

Definition

Let A be a MALL2 formula and $\pi, \pi' : A$. Define $\pi \sim_A \pi'$ as: for any *propositional* MALL formula B, for any proof ρ of $A \vdash B$, $\mathbf{cut}(\pi, \rho)$ and $\mathbf{cut}(\pi', \rho)$ have the same normal form.

- $\sim$ is a congruence: the quotient is a model of MALL2
- A existential-free $\Rightarrow \sim_A$ trivial
- Example: the proofs of $\exists X.\, X$ cannot be distinguished
  - 2nd-order encodings of units work, e.g. $\top \equiv \exists X.\, X$.

Theorem

For any MALL2 *formula A, there are* finitely many *classes for* $\sim_A$.

Proved with *proof nets* for MLL2, extended to MALL2 by a trick.

The observational quotient is non-effective

The observational quotient seems very concrete and simple.

However, given a type A, one cannot enumerate $A/\sim_A$,
even though it is finite. Indeed, one cannot check its emptiness:

Theorem (Lafont 1996)

MALL2 *is undecidable.*

More: adapting Lafont's proof gives

Proposition

Given a MALL2 *type* A *and* $\pi, \pi' : A$, $\pi \sim_A \pi'$ *is undecidable.*

(However, for a *fixed* A, $\sim_A$ is decidable.)

$\longrightarrow$ Search for *effective* finite models.
To overcome undecidability, enlarge the semantics.

Girard's historical models of linear logic

Let's come back to the origins of linear logic:

- decomposition $A \Rightarrow B \equiv !A \multimap B$ in *coherence spaces* ...

Girard's historical models of linear logic

Let's come back to the origins of linear logic:

- decomposition $A \Rightarrow B \equiv !A \multimap B$ in *coherence spaces* ...
- ...coming from "qualitative domains" for System F
  The system F of variable types, 15 years later (1986)
  represent types w/ parameters as *normal functors*

Girard's historical models of linear logic

Let's come back to the origins of linear logic:

- decomposition $A \Rightarrow B \equiv !A \multimap B$ in *coherence spaces* ...
- ...coming from "qualitative domains" for System F
  The system F of variable types, 15 years later (1986)
  represent types w/ parameters as *normal functors*
- previously: the 1st quantitative semantics of STλ
  Normal functors, power series and λ-calculus (1985)
  origin of "linear" terminology (degree 1 monomial)

Girard's historical models of linear logic

Let's come back to the origins of linear logic:

- decomposition $A \Rightarrow B \equiv !A \multimap B$ in *coherence spaces* ...
- ...coming from "qualitative domains" for System F
  The system F of variable types, 15 years later (1986)
  represent types w/ parameters as *normal functors*
- previously: the 1st quantitative semantics of STλ
  Normal functors, power series and λ-calculus (1985)
  origin of "linear" terminology (degree 1 monomial)
- previously[2]: dilators (`http://girard.perso.math.cnrs.fr/ptlc2.pdf`)

Girard's historical models of linear logic

Let's come back to the origins of linear logic:

- decomposition $A \Rightarrow B \equiv !A \multimap B$ in *coherence spaces* ...
- ...coming from "qualitative domains" for System F
  The system F of variable types, 15 years later (1986)
  represent types w/ parameters as *normal functors*
- previously: the 1st quantitative semantics of STλ
  Normal functors, power series and λ-calculus (1985)
  origin of "linear" terminology (degree 1 monomial)
- previously^2: dilators (`http://girard.perso.math.cnrs.fr/ptlc2.pdf`)

$\rightarrow$ the original semantics of 2nd-order (full) linear logic:
coherence spaces and normal functors

Claim: this semantics is finite and effective for MALL2**.**

Relational / coherent semantics: propositional case

Relational semantics:

- formula/type $\rightsquigarrow$ set
- proof/program $\rightsquigarrow$ subset
  - in case $[\![A \multimap B]\!] = [\![A]\!] \times [\![B]\!]$, subset = *relation*
- composition = relational composition

Coherence spaces refine this:

- type $\rightsquigarrow$ undirected graph (adjacency called *coherence*)
- program $\rightsquigarrow$ clique (pairwise coherent vertex subset)
  - notation: $c \sqsubset X$ means "c clique of X"
  - we still have $|\,[\![A \multimap B]\!]\,| = |\,[\![A]\!]\,| \times |\,[\![B]\!]\,|$ ($|\cdot|$: vertex set)
- composition = relational composition

Variable types: a relational example

Let $\pi : X^{\perp} \parr X$. For all sets S, $[\![\pi]\!]_{X \mapsto S} \subseteq S \times S$ in Rel.

This family should somehow be "uniform" in S, such that

uniform families $r_S \subset S \times S \cong$ subsets $r \subset \left[\!\!\left[\forall X.\, X^{\perp} \parr X \right]\!\!\right]$

to interpret the $\forall$-intro rule.

Variable types: a relational example

Let $\pi : X^{\perp} \parr X$. For all sets S, $[\![\pi]\!]_{X \mapsto S} \subseteq S \times S$ in Rel.

This family should somehow be "uniform" in S, such that

uniform families $r_S \subset S \times S \cong$ subsets $r \subset \left[\!\!\left[\forall X.\, X^{\perp} \parr X\right]\!\!\right]$

to interpret the $\forall$-intro rule.

What makes sense in all sets: $=$, $\neq$.
Represent equality by "bound variables": $\langle x, y, \ldots \vdash t \rangle$.

$$\{(s,s) \mid s \in S\} \leftrightarrow \{\langle x \vdash (x,x)\rangle\}$$
$$\{(s,s') \mid s \neq s' \in S\} \leftrightarrow \{\langle x, y \vdash (x,y)\rangle\}$$
$$S \times S \leftrightarrow \{\langle x \vdash (x,x)\rangle, \langle x, y \vdash (x,y)\rangle\}$$

Variable types: a relational example

Let $\pi : X^{\perp} \parr X$. For all sets S, $[\![\pi]\!]_{X \mapsto S} \subseteq S \times S$ in Rel.

This family should somehow be "uniform" in S, such that

$$\text{uniform families } r_S \subset S \times S \cong \text{subsets } r \subset \left[\!\!\left[\forall X.\, X^{\perp} \parr X\right]\!\!\right]$$

to interpret the $\forall$-intro rule.

What makes sense in all sets: $=$, $\neq$.
"substitution of bound variables" = functoriality for injections

$$\{(s,s) \mid s \in S\} \leftrightarrow \{\langle x \vdash (x,x)\rangle\}$$
$$\{(s,s) \mid s \in S\} = \{(\iota(x),\iota(x)) \mid \iota : \{x\} \hookrightarrow S\}$$
$$\{(s,s') \mid s \neq s' \in S\} \leftrightarrow \{\langle x,y \vdash (x,y)\rangle\}$$
$$\{(s,s') \mid s \neq s' \in S\} = \{(\iota(x),\iota(y)) \mid \iota : \{x,y\} \hookrightarrow S\}$$

$\{$expr. w/ binders$\} \cong \{$uniform families (def. next slide)$\}$

A relational counter-example...

Definition (Girard's "mutilation property")

A family $r_S \subset F(S)$ is *uniform* if for $X \subseteq Y$, $r_X = r_Y \cap F(X)$, and similarly for injections (F functorial on injections).

Entails Longo et al.'s *Axiom C*: $F(S)$ constant $\Rightarrow r_S$ constant

(*The genericity theorem and parametricity in the polymorphic λ-calculus*, 1993)

Uniformity *not closed under composition* in the relational model:

- $\langle x \vdash (*, x) \rangle \leftrightarrow \{*\} \times S \subseteq [\![1 \multimap X]\!]_{X \mapsto S}$
- $\langle x \vdash (x, *) \rangle \leftrightarrow S \times \{*\} \subseteq [\![X \multimap 1]\!]_{X \mapsto S}$
- composition: $\{(*, *)\}$ if $S \neq \emptyset$, $\emptyset$ if $S = \emptyset$

We generalize this: compact closed model $\Rightarrow \neg$(Axiom C).

...hence coherence spaces!

Uniformity *not closed under composition* in the relational model. But it works in coherence spaces; miracle due to *stability*.

Definition (Girard's "mutilation property")

A family $r_S \sqsubset F(S)$ is *uniform* if for $X \subseteq Y$, $r_X = r_Y \cap |F(X)|$, and similarly for embeddings (F functorial on embeddings).

where $X \subseteq Y$ means "X induced subgraph of Y".

Composition issue part of the folkore.

- Subtlety overlooked by Girard
- Can be proved for coherence spaces via "Moggi's trick" (mentioned in *Proofs and Types* appendix for another purpose!)
- In relational model: investigated by A. Bac, T. Ehrhard and C. Tasson

Variable types in coherence spaces

So types with n parameters $\rightsquigarrow$ functors $\mathsf{Coh}^n \to \mathsf{Coh}$ where

- objects of Coh: coherence spaces
- morphisms $X \hookrightarrow Y$ in Coh: embeddings
  (i.e. graph isomorphism between X and induced subgraph of Y)

Note that $X \subseteq Y \Rightarrow X^\perp \subseteq Y^\perp$ ($X^\perp$ = complement graph of X), hence *covariance* (cf. previous example $X^\perp \parr X$).

And proofs $\rightsquigarrow$ uniform families. To interpret $\forall$, we want

$$\text{uniform families } c_X \sqsubset F(X) \;\cong\; \text{cliques } c \sqsubset \mathrm{Tr}(F)$$

$\to$ for this, Girard requires F to be a *normal functor*
($\simeq$ categorification of stable functions)

Normal functors

Definition

A functor $F : \mathsf{CohI} \to \mathsf{CohI}$ is *normal* if it preserves
(1) pullbacks and (2) filtered colimits.

Theorem (Girard's normal form theorem)

If $x \in |F(X)|$ *then* $\exists! X_0 \subseteq X$ finite minimal *s.t.* $x \in |F(X_0)|$.

(More precisely: should replace inclusion by embedding in statement; categorically, initiality in a slice cat. of a cat. of elements)

- $\exists$ minimal X_0 ensured by (1) (cf. *polynomial functors*)
- finiteness ensured by (2) (normal f. = *finitary* polynomial f.)

Set of *normal forms* (morally, $X_0 = (\mathrm{fv}(x), \text{coherence})$):

$$\mathrm{NF}(F) = \{\langle X_0 \vdash x\rangle \mid x \in F(X_0) \text{ with } X_0 \text{ minimal}\}$$

Traces of normal functors

Representation of $\forall$:

- define a non-reflexive coherence relation on $\mathrm{NF}(F)$
- def. $|\mathrm{Tr}(F)| = \{\langle X_0 \vdash x\rangle \in \mathrm{NF}(F) \mid \langle X_0 \vdash x\rangle \text{ self-coherent}\}$
- restrict coherence on $\mathrm{NF}(F)$ to $|\mathrm{Tr}(F)|$
  $\to$ makes $\mathrm{Tr}(F)$ a coherence space
- its cliques are in bijection with uniform families for F

Revisiting previous examples:

- $[\![\forall X.\, X^\perp \parr X]\!] = \{\langle x \vdash (x,x)\rangle\} \cong [\![1]\!]$
  only 1 self-coherent point
- $[\![\forall X.\, 1 \multimap X]\!] = [\![\forall X.\, X \multimap 1]\!] = \emptyset$
  thus no problem with composition here!

Normal functors of finite degree

New: for our purposes, we consider *finite* normal functors, i.e.:

- preserving finiteness: $\mathrm{Card}(X) < \infty \implies \mathrm{Card}(F(X)) < \infty$
- of *finite degree*: $\deg(F) = \sup_{\langle X_0 \vdash x \rangle \in \mathrm{NF}(F)} \mathrm{Card}(X_0) < \infty$

(Mutatis mutandis for open types with n variables.)

Intuition: $\deg(F)$ = max number of bound variables used

$$\deg(F^\perp) = \deg(F) \quad \deg(X \mapsto \mathrm{Tr}(F(X, -))) \leq \deg(F)$$

$$\deg(F \otimes G) = \deg(F) + \deg(G) \quad \deg(F \oplus G) = \max(\deg(F), \deg(G))$$

Also, $\mathrm{Card}(\mathrm{Tr}(F(X, -))$ bounded using $\deg(F)$.

Theorem

Coherence spaces and finite normal functors provide a finite semantics of MALL2.

Bounds and effectivity

Theorem (finite degree = polynomial growth)

Let $F : \mathsf{Coh} \to \mathsf{Coh}$ *be a normal functor. There exists* $d \in \mathbb{N}$ *s.t.* $\mathrm{Card}(|F(X)|) = O(\mathrm{Card}(|X|^d)$ *if and only if* F *is a finite n.f.*
In that case, $\deg F$ *is the smallest such* d*.*

Corollary: $!(-)$ and $?(-)$ have infinite degree.

Complexity properties:

- Given fixed F, $X \mapsto F(X)$ computable in L (logspace)
- Given fixed $c \sqsubset \mathrm{Tr}(F)$, $X \mapsto c_X \sqsubset F(X)$ computable in L

$\to$ applied to implicit complexity, j.w.w. P. Pradic
From normal functors to logarithmic space queries (submitted)

In fact the semantics is *effective*:
$A \mapsto [\![A]\!]$ and $(\pi : A) \mapsto [\![\pi]\!] \sqsubset [\![A]\!]$ are computable.

A general remark on $\exists$ and cut

In coherence spaces, $\forall$-elim easily defined by:

$$\{(\langle X \vdash x\rangle, F(\iota)(x)) \mid \langle X \vdash x\rangle \in \mathrm{Tr}(F),\ \iota : X \hookrightarrow S\} \subseteq (\mathrm{Tr}(F) \multimap F(S))$$

$\exists$-intro less primitive, obtained by transposing the above.
Non-trivial computational contents, *erasing witnesses*:
e.g. subsumes cut, since $[\![\exists X.\, X \otimes X^\perp]\!] = [\![\bot]\!]$

$$\frac{\vdash \Gamma, A \quad \vdash \Delta, A^\perp}{\vdash \Gamma, \Delta} \quad \sim \quad \frac{\vdash \Gamma, \Delta, A \otimes A^\perp}{\vdash \Gamma, \Delta, \exists X.\, X \otimes X^\perp}$$

(as already remarked by Girard in 1986)

In general, $\exists$-intro precomputes reaction to all possible $\forall$s
$\rightarrow$ allows compression of information, thus finiteness.

In summary

- A combinatorial analysis of the historical coherence space model of polymorphism
  - categorical formalism $\longrightarrow$ syntactic presentation
- Finiteness and effectiveness results for MALL2
- Applications to implicit complexity
  - Inspired by regular languages in simply typed λ-calculus

In summary

- A combinatorial analysis of the historical coherence space model of polymorphism
  - categorical formalism $\longrightarrow$ syntactic presentation
- Finiteness and effectiveness results for MALL2
- Applications to implicit complexity
  - Inspired by regular languages in simply typed λ-calculus

Future work: alternative finite semantics

- wave-style GoI for MLL2?
- 2nd-order hypercoherences for sub-polynomial complexity

A question: does there exist a non-trivial *finite* and *compact closed* model of MALL2? (Recall that Axiom C cannot hold)