From finite semantics to regular languages (and beyond) in second-order linear logic

NGUYỄN Lê Thanh Dũng (a.k.a. Tito) — nltd@nguyentito.eu
Laboratoire d’informatique de Paris Nord (LIPN), Université Paris 13
joint work with Thomas SEILLER (CNRS, LIPN)
ELICA project final meeting, Paris, October 11th, 2018
Curry-Howard approach to implicit complexity:

1. Define logic / programming language
2. Bound evaluation complexity (*soundness*)
3. Show language expressivity (*extensional completeness*)
4. Result: set of expressible functions = some complexity class

Step 1 requires creativity. Examples from linear logic:
LLL $\leadsto$ polytime (Girard), SBAL $\leadsto$ logspace (Schöpp)...
Implicit complexity with proofs-as-programs

Curry-Howard approach to implicit complexity:

1. Define logic / programming language
2. Bound evaluation complexity (*soundness*)
3. Show language expressivity (*extensional completeness*)
4. Result: set of expressible functions = some complexity class

Step 1 requires creativity. Examples from linear logic:
LLL $\leadsto$ polytime (Girard), SBAL $\leadsto$ logspace (Schöpp)...

This talk: instead, ask (2)–(4) for well-known systems:

- simply typed $\lambda$-calculus (ST$\lambda$)
  - recall old methods and results
- and later Elementary Linear Logic (ELL)
  - new results inspired by ST$\lambda$ techniques

…or rather, (naturally) restricted situations within ST$\lambda$/ELL.
Simply-typed \( \lambda \)-calculus and implicit complexity?

Recall \( k\text{-EXPTIME} = \text{DTIME}(\text{tower of exponentials of height } k) \),
\[ \text{ELEMENTARY} = \bigcup_{k \in \mathbb{N}} k\text{-EXPTIME}. \]

**Claim:** \( \text{ST}\lambda \) characterizes \( \text{ELEMENTARY} \).

Parameter controlling complexity: *functionality order*

\[
\text{ord}(\alpha \rightarrow \beta) = \max(\text{ord}(\alpha) + 1, \text{ord}(\beta))
\]

- **Soundness:** \( \forall k \in \mathbb{N} \exists f(k) \in \mathbb{N} \text{ s.t. normalization of } \lambda\text{-terms with order } \leq k \text{ subterms is in } f(k)\text{-EXPTIME} \)
- **Extensional completeness:** naive attempt fails
Church encodings of inputs in $\text{ST}_\lambda$

Church (or Böhm–Berarducci) encodings:

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

Choose a simple type $A$, and a term $t : \text{Str}[A] \to \text{Bool}$

$\longrightarrow$ defines language $\mathcal{L}(t) = \{w \in \{0, 1\}^* \mid t \overline{w} \xrightarrow{\beta} \text{true}\}$. Not all ELEMENTARY languages possible... but then what?
Church encodings of inputs in $\text{ST} \lambda$

Church (or Böhm–Berarducci) encodings:

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

Choose a simple type $A$, and a term $t : \text{Str}[A] \rightarrow \text{Bool}$
defines language $\mathcal{L}(t) = \{ w \in \{0, 1\}^* \mid t \overline{w} \rightarrow^* \text{true} \}$.
Not all ELEMENTARY languages possible... but then what?

**Theorem (Hillebrand & Kanellakis, LICS’96)**

*The languages decided by $\text{ST} \lambda$-terms of type $\text{Str}[A] \rightarrow \text{Bool}$ are exactly the regular languages.*
Regular languages in $\text{ST}\lambda$

Theorem (Hillebrand & Kanellakis, LICS’96)

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

Part 1 of proof.

Fix type $A$. Any denotational semantics $\llbracket - \rrbracket$ quotients words:

$$ w \in \{0, 1\}^* \leadsto \overline{w} : \text{Str}[A] \leadsto \llbracket \overline{w} \rrbracket_{\text{Str}[A]} \in \llbracket \text{Str}[A] \rrbracket $$

When $\llbracket - \rrbracket$ non-trivial ($\llbracket \text{true} \rrbracket \neq \llbracket \text{false} \rrbracket$), $\llbracket \overline{w} \rrbracket_{\text{Str}[A]}$ determines behavior of $w$ w.r.t. all $\text{Str}[A] \to \text{Bool}$ terms:

$$ w \in \mathcal{L}(t) \iff t \overline{w} \to^*_\beta \text{true} \iff \llbracket t \overline{w} \rrbracket = \llbracket t \rrbracket(\llbracket \overline{w} \rrbracket) = \llbracket \text{true} \rrbracket $$

Goal: to decide $\mathcal{L}(t)$, compute $w \mapsto \llbracket \overline{w} \rrbracket$ in some model of $\text{ST}\lambda$. 
Regular languages in \( ST\lambda \)

**Theorem (Hillebrand & Kanellakis, LICS’96)**

For any type \( A \) and any \( ST\lambda \)-term \( t : Str[A] \rightarrow \text{Bool} \), the language \( \mathcal{L}(t) = \{ w \in \{0, 1\}^* | t \bar{w} \rightarrow^\beta \text{true} \} \) is regular.

**Part 2 of proof.**

We use \([–] : ST\lambda \rightarrow \text{FinSet} \) to build a DFA with states \( Q = [\text{Str}[A]] \), acceptation as \([t](–) = [\text{true}]\).

\[
\begin{array}{c}
[\bar{0}] \quad 0 \quad [0] \quad 1 \quad [01] \quad 1 \quad [011] \quad \ldots
\end{array}
\]

\( w \in \mathcal{L}(t) \iff [t]([\bar{w} \text{Str}[A]]) = [\text{true}] \iff w \) accepted

\( \longrightarrow \) semantic evaluation argument.
Regular languages in STλ

Theorem (Hillebrand & Kanellakis, LICS’96)

For any type A and any STλ-term \( t : \text{Str}[A] \to \text{Bool} \), the language \( \mathcal{L}(t) = \{ w \in \{0, 1\}^* \mid t \bar{w} \to^*_{\beta} \text{true} \} \) is regular.

Part 2 of proof.

We use \( [-] : \text{STλ} \to \text{FinSet} \) to build a DFA with states \( Q = [\text{Str}[A]] \), acceptance as \( [t](-) = [\text{true}] \).

(\( |Q| < \infty \), e.g. \( 2^{2^{33}} \) when \( A = \text{Bool} \))

\[
\begin{array}{cccccc}
& & & & & \\
\overset{0}{\rightarrow} & \overset{1}{\rightarrow} & \overset{1}{\rightarrow} & \ldots \\
\text{[ε]} & \rightarrow & \text{[0]} & \rightarrow & \text{[01]} & \rightarrow \\
\end{array}
\]

\( w \in \mathcal{L}(t) \iff [t][[w]_{\text{Str}[A]}] = [\text{true}] \iff w \text{ accepted} \)

\( \rightarrow \) semantic evaluation argument.
Moral of the story

Finite denotational semantics have complexity consequences.

- Analogous results for tree automata, and for propositional linear logic (using your favorite finite model)
- Another application to ST\(\lambda\) at fixed order:

**Theorem (Terui, RTA’12)**

*Normalizing an ST\(\lambda\)-term of type \(\text{Bool}\) w/ order \(\leq r\) subterms is*

- \(k\)-EXPTIME-complete for \(r = 2k + 2\)
- \(k\)-EXPSPACE-complete for \(r = 2k + 3\)

**Proof of membership in \(k\)-EXPTIME / \(k\)-EXPSPACE.**

\(\beta\)-reduce to halve order, then evaluate in LL Scott model.
Need to change input representation!

Hillebrand, Kanellakis & Mairson, motivated by database queries, encode *finite relational (1st-order) structures* as inputs → completeness for **ELEMENTARY**.

- We’ll come back to this later
- “β-convertibility of STλ terms ∉ ELEMENTARY” (Statman 1979) can be recovered from this
- Refined by H&K: characterization of \( k\text{-EXPTIME} / k\text{-EXPSPACE} \) in STλ+constants+equality at fixed order
  - Also using semantic evaluation for soundness!
Regular languages in Elementary Linear Logic
Elementary Linear Logic (ELL)

ELL = Multiplicative-Additive Linear Logic (MALL) + ?/! rules:

\[
\begin{align*}
\Gamma, ?A, ?A & \vdash \Gamma, ?A \\
\Gamma & \vdash \Gamma, ?A \\
A_1, \ldots, A_n, B & \vdash ?A_1, \ldots, ?A_n, !B
\end{align*}
\]

!-intro: promotion and dereliction must come together.
→ enforces stratification by !-depth

(subproofs cannot change depth during cut-elimination)

Representable functions in 2nd-order ELL = ELEMENTARY:

- Soundness: normalization in \( f(\text{depth}) \)-EXPTIME
  - depth in ELL \( \simeq \) order in ST\(\lambda\)
- Extensional completeness: Church encoding works
  - thanks to (impredicative) polymorphism!
Data types:

- $\text{Bool} = 1 \oplus 1$
- $\text{Str} = \forall X. ! (X \rightarrow X) \rightarrow ! (X \rightarrow X) \rightarrow ! (X \rightarrow X)$

Extensional completeness: all languages $L \in \text{ELEMENTARY}$ expressible by ELL proofs of $! \text{Str} \rightarrow !^k \text{Bool}$ ($k$ depends on $L$).

Soundness (reformulated):
proofs of $!^k \text{Bool}$ can be normalized in $f(k)$-EXPTIME.

Question: what do we get for a fixed depth $k$?
Depth $k = 2$ case, in a variant of ELL:

**Theorem (Baillot, APLAS’11)**

*The proofs of* $\text{Str} \to \text{!!Bool}$ *in 2nd order elementary affine logic with recursive types decide exactly the languages in* $\text{P}$.*

Proof idea.
Adapt Hillebrand & Kanellakis’s ST proof. Requires non-trivial finite semantics for 2nd order MALL (MALL2).
Depth $k = 2$ case, in a variant of ELL:

**Theorem (Baillot, APLAS’11)**

The proofs of $\text{Str} \rightarrow \text{!!Bool}$ in 2nd order elementary affine logic with recursive types decide exactly the languages in $P$.

Recursive types are crucial for the above, as we show:

**Theorem**

The proofs of $\text{Str} \rightarrow \text{!!Bool}$ in 2nd order ELL decide exactly the regular languages.

**Proof idea.**

Adapt Hillebrand & Kanellakis’s $\text{ST}\lambda$ proof. Requires non-trivial finite semantics for 2nd order MALL (MALL2).
Finite semantics for MALL2

Choice of semantics: syntax/(observational equivalence).

**Definition (Eqv. for propositional observations)**

Let $A$ be a MALL2 formula and $\pi, \pi' : A$. Define $\pi \sim_A \pi'$ as:

$$\forall B \text{ MALL0}, \forall \rho : (A \vdash B), \text{ cut}(\pi, \rho) \equiv \text{ cut}(\pi', \rho)$$

- MALL0 = propositional MALL
- $\equiv$ is usual proof equivalence on MALL0 (think $\beta\eta$)

**Theorem**

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

**Corollary**

*There exists a non-trivial finite semantics for MALL2.*

New result of independent interest, cf. Pistone’s talk 2 days ago.
Let $\pi : !\text{Str} \to !!\text{Bool}$. There exists

$$\hat{\pi} : \text{Str}[A_1] \to \ldots \to \text{Str}[A_n] \to !!\text{Bool}$$

such that $\forall w. \pi(!\overline{w}) = !\hat{\pi}(\overline{w}[A_1], \ldots, \overline{w}[A_n])$. (Str = $\forall X. \text{Str}[X]$)

Thanks to stratification, w.l.o.g. $A_1, \ldots, A_n \in \text{MALL2}$. Using finite MALL2 semantics $[\_\_\_]$ principal, $\overline{w}[A]$ induces map

$$||w||_A : [A \to A] \times [A \to A] \to [A \to !!A]$$

such that $\overline{w}[A](!f_1, !f_2) = !g \implies ||w||_A([f_1], [f_2]) = [g]$.

- Church encoding $\implies ||w||_A$ computable by automaton
- $(||w||_{A_1}, \ldots, ||w||_{A_n})$ determine $\hat{\pi}(\overline{w}[A_1], \ldots, \overline{w}[A_n])$ and therefore $\pi(!\overline{w})$
What about higher depths?

We solved depth $k = 2$ case for ELL. (First characterization of regular languages in a type system with impredicative quantification?)

When $k > 2$:

Theorem (Baillot, APLAS’11)

*The proofs of $!\text{Str} \rightarrow !^k\text{Bool}$ in EAL+rectypes decide exactly the languages in $(k - 2)$-EXPTIME.*

- For ELL without recursive types, we get a class between $(k - 3)$-EXPTIME and $(k - 2)$-EXPTIME… which one exactly?
- Semantics probably has a role to play in the answer
Inputs as finite models: towards logarithmic space in ELL?
A bit of descriptive complexity

Data represented as (totally ordered) *finite first-order structures* (a.k.a. *finite models*), over a signature of relation symbols.

**Example**

Signature for binary strings: \( \langle \leq, S \rangle \).
Finite models are \((D, \leq^D, S^D), |D| < \infty. S^D(d) = \text{“}d\text{th bit is 1”}\).  

*Descriptive complexity*: characterize a complexity class \( C \) as set of queries written in some logic \( L_C \), i.e. “is this \( L_C \) formula true in this finite model?”. For instance:

**Theorem (Fagin 1974)**

*Queries in existential second-order logic = NP.*
Finite models in $\text{ST} \lambda$ and extensional completeness

With type $d$ of elements (equipped with $\text{Eq} : d \rightarrow d \rightarrow \text{Bool}$),

- Represent $k$-ary relations as lists of $k$-tuples
  \[ \text{Rel}_k[d, A] = (d^k \rightarrow A \rightarrow A) \rightarrow A \rightarrow A \]
  (in the spirit of database theory: relation = set of records)

- Provide list of all domain elements ($\text{List}[d, A] = \text{Rel}_{1}[d, A]$)

**Theorem (Hillebrand, Kanellakis & Mairson, LICS’93)**

Terms $t : \text{List}[d, A] \rightarrow \text{Rel}_{k_1}[d, A_1] \rightarrow \ldots \rightarrow \text{Rel}_{k_m}[d, A_m] \rightarrow \text{Bool}$

in $\text{ST} \lambda$ compute exactly $\text{ELEMENTARY}$ queries over finite models.

To feed input, instantiate $d = o^n \rightarrow o$ ($n =$ domain size).
Program has “$\forall d \exists A$”, input has “$\exists d \forall A$”. Size of semantics depends on input, breaking earlier expressivity upper bound.
We transpose this idea to second-order ELL:

- **We use** \( \text{Rel}_k[D] = D \otimes^k \leadsto \text{Bool} \),
  \( \text{List}[D] = \forall X. !(D \leadsto X \leadsto X) \leadsto !(X \leadsto X) \)**

- **Allow non-linear use of** \( D \):
  \( \text{Cont}[D] = D \leadsto D \otimes D \), \( \text{Wk}[D] = D \leadsto 1 \)

\[
\text{Inp}_r = \exists D. \forall X. \text{List}[D] \otimes \bigotimes_{i=1}^{n} \text{Rel}_{k_i}[D] \otimes \! \text{Cont}[D] \otimes \! \text{Wk}[D]
\]
(choose \( r \) to satisfy stratification constraint)

- **For size** \( n \) **domain, witness** \( D = 1 \oplus \ldots (n \text{ times}) \ldots \oplus 1 \)
  (positive, therefore duplicable)
Towards logarithmic space in ELL? (1)

**Theorem (Immerman 1983)**

Queries in first-order logic with deterministic transitive closure = logarithmic space (L) queries.

**Proposition**

All L queries on finite models for a given signature can be computed by an ELL proof of \( \text{Inp}_2 \rightarrow !!\text{Bool} \).

Proof idea: compute transitive closure of a relation \( \mathcal{R} \subseteq D^k \times D^k \) by iterating \( \varphi_{\mathcal{R}} : \mathcal{P}(D^k \times D^k) \rightarrow \mathcal{P}(D^k \times D^k) \).

Determinism of \( \mathcal{R} \) ensures linearity: \( \varphi_{\mathcal{R}} : \text{Rel}_{2k} \rightarrow \text{Rel}_{2k} \) in ELL.

This is remarkable enough to hope for:

**Conjecture**

Conversely, proofs of \( \text{Inp}_2 \rightarrow !!\text{Bool} \) only decide L queries.
Towards logarithmic space in ELL? (2)

**Conjecture**

ELL proofs of the following type only decide \( \land \) queries:

\[
\exists D. \forall i \in [n] \left( \bigotimes_{i=1}^{n} \text{Rel}_{k_i}[D] \otimes \text{Cont}[D] \otimes \text{Wk}[D] \Rightarrow \text{Bool} \right)
\]

In predicative case (\( \forall/\exists \) range over propositional formulae):

- conjecture seems very likely
- already non-trivial… (maybe Geometry of Interaction works?)
- note that ext. completeness holds w/o impredicativity

In general case, I have no intuition or methods available 😞
Conclusion and future work
Conclusion

We brought methods from the ST\(\lambda\) tradition to 2nd order ELL, showing that similar phenomena occur in both:

- *Church encodings* of inputs restrict expressivity
- *Semantic evaluation* can prove this (and lots of other stuff)
- To overcome this, one can represent inputs as *finite models*

**Lemma (or Theorem, if you care about semantics)**

The quotient of MALL2 by propositional observations is finite.

**Theorem (or Corollary)**

Proofs of \(!\text{Str} \rightarrow !!\text{Bool}\) in ELL decide regular languages.

Moral: *geometry* (e.g. stratification) and *typing* jointly control complexity; semantics reflects the latter.
Open problems / future work

Logspace conjecture: what kind of techniques can solve this??

Classes characterized by higher fixed depths?
(For both $\mathsf{Str} \rightarrow !^k\mathsf{Bool}$ and $\mathsf{Inp}_k \rightarrow !^k\mathsf{Bool}$...)

Related: complexity of normalizing a proof of $!^k\mathsf{Bool}$ in $\mathsf{ELL}$?

- $k = 0$: $\mathsf{P}$-complete
- $k = 1$: $\mathsf{PSPACE}$-hard, in $\mathsf{EXPTIME}$
- $k \geq 2$: $(k - 1)$-$\mathsf{EXPTIME}$-hard, in $k$-$\mathsf{EXPTIME}$

(For $\mathsf{EAL} + \mathsf{rectypes}$, $k$-$\mathsf{EXPTIME}$-complete by Baillot’s results.)

On $\mathsf{MALL}2$ semantics: further investigations ongoing, j.w.w. P. Pistone and L. Tortora de Falco.