A Tour of Polyadic Approximations

Damiano Mazza
CNRS, LIPN, Université Paris 13

Rencontre Chocola
Lyon, 12 October 2017

Appetizer

- How are the following things related?
  - polyadic approximations;
  - intersection type systems;
  - Boolean circuits;
  - the Cook-Levin theorem.

Appetizer

- How are the following things related?
  - polyadic approximations;
  - intersection type systems;
  - Boolean circuits;
  - the Cook-Levin theorem.

- There is an intersection type system for “Turing machines” whose derivations are Boolean circuits, which approximate the machine they type. This may be used to give a type-theoretic proof of the Cook-Levin theorem.

Girard's Approximation Theorem (1987)

[T]he approximation theorem [...] is just the mathematical contents of our slogan: *usual logic is obtained from linear logic* (without modalities) *by a passage to the limit.*

5.1. Definition (*approximants*). The connectives ! and ? are approximated by the connectives $!_n$ and $?_n$ ($n \neq 0$):

$$!_n A = (1 \mathbin{\&} A) \otimes \cdots \otimes (1 \mathbin{\&} A) \qquad (n \text{ times})$$

$$?_n A = (\perp \oplus A) \parr \cdots \parr (\perp \oplus A) \qquad (n \text{ times})$$

5.2. Theorem (*Approximation Theorem*). *Let A be a theorem of linear logic; with each occurrence of $!$ in A, assign an integer $\neq 0$; then it is possible to assign integers $\neq 0$ to all occurrences of $?$ in such a way that if B denotes the result of replacing each occurrence of $!$ (respectively $?$)* by $!_n$ *(respectively $?_n$) where n is the integer assigned to it, then B is still a theorem of linear logic.*

The dimensional ladder

0. formulas/types
1. proofs/programs
2. cut-elimination/execution
3. standardization
4. residual equivalence
5. . . .

The λ-calculus, 2-dimensionally

$M, N ::= x \mid \lambda x.M \mid MN$	terms
$\beta ::= M(\!|x \leftarrow N|\!)$	basic steps
$\rho, \tau ::= \mathsf{C}\{\beta_1, \ldots, \beta_n\} \mid \rho; \tau$	reduction terms

C is a multi-hole context, n may be 0.

$$\frac{}{M(\!|x \leftarrow N|\!) : (\lambda x.M)N \to^* M\{N/x\}}\ \beta$$

$$\frac{\beta_1 : M_1 \to^* M_1' \quad \ldots \quad \beta_n : M_n \to^* M_n'}{\mathsf{C}\{\beta_1, \ldots, \beta_n\} : \mathsf{C}\{M_1, \ldots, M_n\} \to^* \mathsf{C}\{M_1', \ldots, M_n'\}}\ \mathsf{ctxt}$$

$$\frac{\rho : M \to^* P \qquad \tau : P \to^* N}{\rho; \tau : M \to^* N}\ \mathsf{comp}$$

Basic equivalences

Structural equivalence:

$$\frac{\rho : M \to^* N \quad \tau : N \to^* P \quad \varphi : P \to^* Q}{(\rho;\tau);\varphi \equiv \rho;(\tau;\varphi)}\ \textsf{assoc}$$

$$\frac{\rho : M \to^* M'}{M;\rho \equiv \rho}\ \textsf{lunit} \qquad \frac{\rho : M' \to^* M}{\rho;M \equiv \rho}\ \textsf{runit}$$

Permutation equivalence:

$$\frac{\alpha : M \to^* M' \qquad \beta : N \to^* N'}{\mathsf{C}\{M,\beta\};\mathsf{C}\{\alpha,N'\} \sim \mathsf{C}\{\alpha,\beta\} \sim \mathsf{C}\{\alpha,N\};\mathsf{C}\{M',\beta\}}\ \textsf{par}$$

$$\frac{\alpha : \mathsf{C}\{x\} \to^* M' \qquad \beta : N \to^* N'}{\mathsf{C}\{\beta\};\alpha\{N'/x\} \sim \alpha\{N/x\};M'\{\beta/x\}}\ \textsf{unnest}$$

An operadic take on syntax

- The λ-calculus with β-reduction may be presented as a 2-operad Λ:

  0. one color;
  1. multimorphisms of $\Lambda(n)$: terms M with $\mathrm{fv}(M) \subseteq \{x_1, \ldots, x_n\}$;
  2. 2-arrows $M \Rightarrow N$: reduction terms $\rho : M \to^* N$ modulo $\sim$;

  operadic composition is substitution.

- Church-style (simple) types $=$ more than one color.

- Curry-style types $=$ 2-operad morphism $\mathcal{E} \to \Lambda$ (Melliès-Zeilberger).

- This generalizes to linear logic terms $\Lambda_!$, polyadic affine terms $\Lambda_{\mathrm{a}}^{\mathrm{p}}$. . .

Approximation order

- Polyadic affine reduction terms are endowed with an *approximation order*.
- The definition is inductive; the key case is

$$\frac{\rho_1 \sqsubseteq \rho'_1 \quad \ldots \quad \rho_n \sqsubseteq \rho'_n \quad n \leq m}{\langle \rho_1, \ldots, \rho_n \rangle \sqsubseteq \langle \rho'_1, \ldots, \rho'_m \rangle}\ \mathsf{box}$$

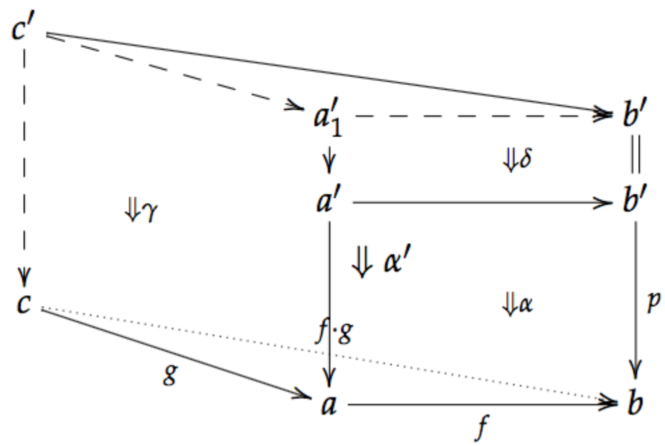
- This defines a *posetal double category*:

$$\begin{array}{ccc} t & \xrightarrow{\ \rho\ }{}^{*} & u \\ \rotatebox{-90}{$\sqsubseteq$} & \rotatebox{-90}{$\sqsubseteq$} & \rotatebox{-90}{$\sqsubseteq$} \\ t' & \xrightarrow[\ \rho'\ ]{}{}^{*} & u' \end{array}$$

Ideal completion

- The forgetful functor $\mathbf{Cat}(\mathbf{Dcpo}) \to \mathbf{Cat}(\mathbf{Pos})$ has no left adjoint.

- *Cartesian liftings* in double categories:



- A posetal dbl. cat. $\mathcal{D}$ is monotonic if $\mathcal{D}^{\mathrm{tcoop}}$ has cartesian liftings.

- The forgetful functor $\mathbf{MntDblDcpo} \to \mathbf{MntDblPos}$ has a left adjoint.

Recovering linear logic

- Let $\Lambda_{\mathsf{a}}^{\infty} := \mathrm{Hor}(\widehat{\Lambda_{\mathsf{a}}^{\sqsubseteq}})$.

Theorem (Girard's approximation theorem, computational version)
There is an isomorphism of 2-operads

$$\Lambda_{\mathsf{a}}^{\infty\mathsf{fu}}/{\approx} \quad \cong \quad \Lambda_{!},$$

where $\Lambda_{\mathsf{a}}^{\infty\mathsf{fu}}/{\approx}$ is a suitable quotient of the finitary, uniform sub-operad of $\Lambda_{\mathsf{a}}^{\infty}$.

- This yields (affine) polyadic approximations.

Modulus of continuity

- Reduction enjoys a continuity property: if $M \to^* N$, then

$$\forall\ u \sqsubset N,\ \exists\ t \sqsubset N \ \text{ such that } t \to^* u.$$

- What's the dependency of $|t|$ on the length of the reduction?
- We informally call this the "modulus of continuity".
- Bad news: the modulus of continuity is exponential!

A polynomial modulus of continuity?

- Why is exponential bad?

1. It is morally wrong (cf. abstract machines).
2. We would like to be able to use the equation

$$\frac{\text{affine } \lambda\text{-terms}}{\lambda\text{-terms}} = \frac{\text{Boolean circuits}}{\text{Turing machines}}$$

> **Theorem.** *Polytime TMs induce polysize Boolean circuits.*

3. No λ-calculus approach to non-uniform computation.

Parsimonious logic

- A logic/calculus with a polynomial modulus of continuity.
- Categorically:
  - a SMCC $\mathcal{C}$ with terminal unit;
  - a lax monoidal functor $! : \mathcal{C} \to \mathcal{C}$;
  - a natural isomorphism $!A \cong A \otimes !A$ (*Milner's law*).
- Good complexity properties (alternative to "light" logics):
  - simple types = L (deterministic logspace);
  - linear polymorphism = P (deterministic polytime).
- Extends to non-uniformity (L/poly, P/poly).

The Cook-Levin Theorem

Theorem. SAT *is* NP-*complete.*

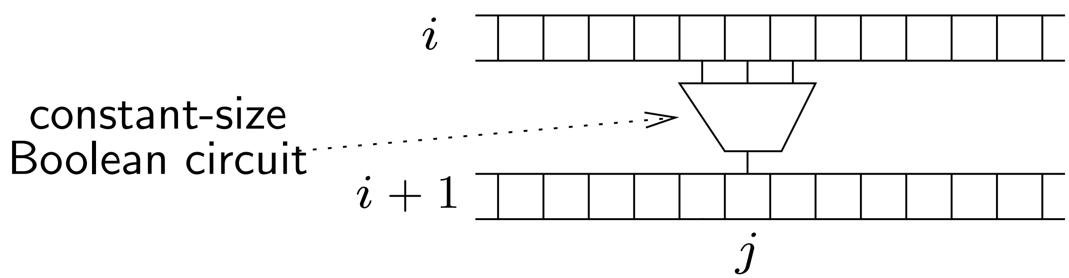
Implied by

Theorem. CIRCUIT SAT *is* NP-*complete.*

Essentially implied by

Theorem. *Polytime TMs induce polysize Boolean circuits.*

Key to the proof: "computation is local".



A more essential proof?

- Everyone believes that a circuit can encode the transition of a TM, but nobody wants to actually do it: textbooks sketch the idea, no-one wants to see the full details (they are irrelevant for the essence of the proof).
- Maybe there is a proof in which those details are simply not needed?
- Also, maybe the essence of the Cook-Levin theorem (computation is local) can take a more precise mathematical form?
- Can we prove the Cook-Levin theorem in "theory B" style?

A first-order while language (Mowl)

- Types: Bool, Str.

- Programs:

$$
\begin{array}{rll}
M, N, P, Q ::= & x \mid M[x \leftarrow N] & \text{vars and let} \\
\mid & 0 \mid 1 \mid \mathsf{if}\ M\ \mathsf{then}\ N\ \mathsf{else}\ P & \text{booleans} \\
\mid & \varepsilon \mid 0M \mid 1M \mid \mathsf{case}\ M\ \mathsf{of}\ \varepsilon.N \mid 0x.P \mid 1x.Q & \text{binary strings} \\
\mid & \mathsf{while}\ M\ \mathsf{do}\ N\ \mathsf{to}\ x := P & \text{while loop}
\end{array}
$$

- Typing and evaluation rules are as expected.

The class NP

Definition/Proposition. NP is the class of decision problems $L \subseteq \{0,1\}^*$ such that there exists a Mowl program

$$x : \mathsf{Str}, y : \mathsf{Str} \vdash M : \mathsf{Bool}$$

and two polynomials p, q such that, for all $w, w' \in \{0,1\}^*$

$$M[x \leftarrow \underline{w}][y \leftarrow \underline{w'}] \rightarrow^{l(w,w')} \mathsf{b}_{w,w'}$$

with $l(w,w') \leq p(|w| + |w'|)$ and, moreover, there exists $m \leq q(|w|)$ and $w' \in \{0,1\}^m$ such that $\mathsf{b}_{w,w'} = 1$ iff $w \in L$.

Boolean circuits

- Types: Bool.

- Programs:

$$
\begin{array}{rll}
t, u, v ::= x \;\big|\; t[x \leftarrow u] \;\big|\; \bullet & & \text{vars, let, undef} \\
\big|\; 0 \;\big|\; 1 \;\big|\; \text{if } M \text{ then } N \text{ else } P & & \text{booleans} \\
\big|\; \langle t_1, \ldots, t_n \rangle \;\big|\; t[\langle x_1, \ldots, x_n \rangle := u] & & \text{tuples}
\end{array}
$$

- Typing and evaluation rules are as expected.

Approximation order for Boolean circuits

- The key rules:

$$\frac{}{\bullet \sqsubseteq t}\ \text{undef} \qquad\qquad \frac{t_1 \sqsubseteq t'_1 \quad \dots \quad t_n \sqsubseteq t'_n}{\langle t_1, \dots, t_n \rangle \sqsubseteq \langle t'_1, \dots, t'_m \rangle}\ \text{box}\ m \geq n$$

- The same methodology used for affine polyadic terms applies here:
  - Boolean circuits form a monotonic posetal double category $\mathcal{C}$;
  - Mowl programs embed in the ideal completion of $\mathcal{C}$;
  - hence we may approximate Mowl programs by Boolean circuits;
  - hence we have an approximation presheaf $\mathrm{Mowl} \to \mathfrak{Rel}$;
  - by the Grothendieck construction, this is a type system for Mowl.

Intersection types for Mowl

See my HdR thesis.

Key properties: monotonicity

Lemma. If $t \to u$ and $t \sqsubseteq t'$, then $t' \to u'$ such that $u \sqsubseteq u'$.

Key properties: quantitative subject expansion

Lemma. Let δ be an intersection types derivation of $\Gamma \vdash M : A$ and let

$$M' \to M.$$

Then, there exists a derivation δ' of $\Gamma \vdash M' : A$ such that

$$(\delta')^- \to^* \delta^-.$$

Moreover, $\mathrm{rk}(\delta') \leq \mathrm{rk}(\delta) + 1$ and $\mathrm{tw}(\delta') \leq \mathrm{tw}(\delta) + 1$.

Key properties: uniform typings

There is a notion of *uniform typing* $\lfloor M \rfloor^{\Gamma}_{k,m}$ such that

Lemma. For a fixed Mowl program M, the Boolean circuit $(\lfloor M \rfloor^{\Gamma}_{k,m})^{-}$ may be computed in polynomial time in k and m.

Lemma. Let δ be an intersection types derivation of the judgment $\Gamma \vdash M : A$, with M containing c binary successors. Then, for all $k \geq \mathrm{rk}(\delta)$ and $m \geq \mathrm{tw}(\delta) + c$, we have

$$\delta^{-} \sqsubseteq (\lfloor M \rfloor^{\Gamma'}_{k,m})^{-},$$

where Γ' is Γ in which every type is replaced by Str_m.