Programming Language Expressiveness and Circuit Complexity

Denis Dancanet
Stephen Brookes

School of Computer Science
Carnegie Mellon University

http://www.cs.cmu.edu/~ddr

Twelfth Workshop on the Mathematical
Foundations of Programming Semantics
Boulder, CO
3–5 June 1996

Goal of this work

- Establish relative intensional expressiveness results for programming languages
- Definitions:
  - L_1 is *intensionally* more expressive than L_2 if L_1 can compute all the functions L_2 can, with at least the same asymptotic complexity.
  - L_1 is *extensionally* more expressive than L_2 if L_1 can compute all the functions L_2 can.
- Is nondeterministic parallelism more expressive than deterministic parallelism?

Outline

- Languages:
  - PCF
  - deterministic query
  - nondeterministic query
- Circuit semantics
- On the method and the metric
- Connection with boolean circuits
- Conclusions
- Related work

PCF

- Simple functional programming language
- Types:

  $\sigma ::= o \mid \iota \mid \sigma \to \sigma$

- Syntax:

  $M ::= c \mid x \mid \lambda x.\ M \mid MM$

- Constants:

  $tt, ff : o$
  $n : \iota$
  $=, <, >, \leq, \geq : \iota \to \iota \to o$
  $+, - : \iota \to \iota \to \iota$
  $\supset_\sigma : o \to \sigma \to \sigma \to \sigma$ (conditional, $\sigma \in \{o, \iota\}$)
  $\Omega^\sigma : \sigma$ (undefined elements)

- Standard denotational semantics for PCF:
  $\mathcal{D}$: Terms $\to$ Environments $\to \cup D_\sigma$

Deterministic query

- Syntax:
  query : $(\sigma_1 \times \cdots \times \sigma_n) \to \tau$
  query $(x_1^{\sigma_1}, \ldots, x_n^{\sigma_n})$ is
  $p_1 \Rightarrow M_1^\tau$
  $\ldots$
  $p_k \Rightarrow M_k^\tau$
- Example:
  $por \equiv \lambda xy.$ query (x, y) is
  $(tt, _) \Rightarrow tt$
  $\mid (_, tt) \Rightarrow tt$
  $\mid (ff, ff) \Rightarrow ff$
- Consistent inputs produce same output:
  if $p_i \Uparrow p_j$ then $\mathcal{D}[\![M_i]\!] = \mathcal{D}[\![M_j]\!]$.
- DPCF = PCF + deterministic query

Deterministic query: Semantics

$\mathcal{D}_{pat}$: Patterns $\rightarrow$ Environments $\rightarrow (D_{Bool} \times$ Environments$)$

$$\mathcal{D}[\![Q]\!]\rho = \mathrm{amb}_k(\mathcal{D}[\![\vec{x} \text{ is } p_1 \Rightarrow M_1^\tau]\!]\rho, \ldots, \mathcal{D}[\![\vec{x} \text{ is } p_k \Rightarrow M_k^\tau]\!]\rho)$$

$$\mathcal{D}[\![\vec{x} \text{ is } p \Rightarrow M^\tau]\!]\rho = \begin{cases} \mathcal{D}[\![M^\tau]\!](\rho \mid \mu), & \text{if } \mathcal{D}_{pat}[\![\vec{x} \text{ is } p]\!]\rho = (tt, \mu) \\ \bot, & \text{if } \mathcal{D}_{pat}[\![\vec{x} \text{ is } p]\!]\rho = (\mathit{ff}, \mu) \end{cases}$$

$$\mathcal{D}_{pat}[\![(x_1^{\sigma_1}, \ldots, x_n^{\sigma_n}) \text{ is } (e_1^{\sigma_1}, \ldots, e_n^{\sigma_n})]\!]\rho =$$
$$(b_1 \wedge \cdots \wedge b_n, \mu_1 \mid \cdots \mid \mu_n), \text{where } (b_i, \mu_i) = \mathcal{D}_{pat}[\![x_i^{\sigma_i} \text{ is } e_i^{\sigma_i}]\!]\rho$$

$$\mathcal{D}_{pat}[\![x^\sigma \text{ is } e^\sigma]\!]\rho = \begin{cases} (\mathcal{D}[\![x^\sigma]\!]\rho = \mathcal{D}[\![e^\sigma]\!]\rho, \bot), & \text{if } e^\sigma \text{ is closed} \\ (tt, y^\sigma \mapsto x^\sigma), & \text{if } e^\sigma \equiv y^\sigma \\ (tt, \bot), & \text{if } e^\sigma \equiv _ \end{cases}$$

McCarthy's ambiguity operator:
$amb(\bot, x) = x$,
$amb(x, \bot) = x$,
$amb(x, y) = x$ **or** y.

Nondeterministic query

- Example:
  $not_\perp \equiv \lambda x.$ query (x) is
  $$\begin{array}{l} \quad tt \Rightarrow \mathit{ff} \\ \mid \mathit{ff} \Rightarrow \mathit{ff} \\ \mid _ \Rightarrow tt \end{array}$$
- Under the semantics of deterministic query, nondeterministic query is not intensionally more expressive.

Nondeterministic query: Semantics

- Semantics:

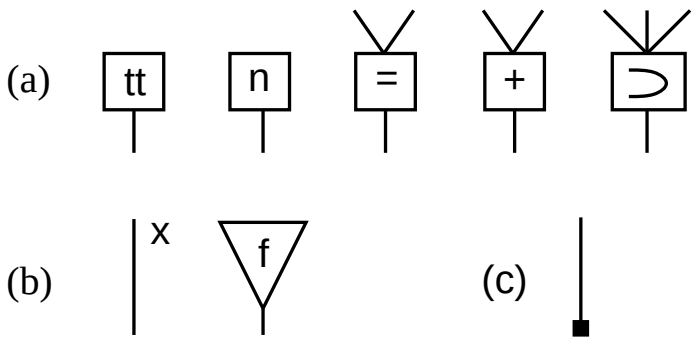
$$\mathcal{D}_{pat}[\![x^\sigma \text{ is } _{ij}]\!]\rho = \begin{cases} (\text{not } (\text{poll } x^\sigma), \bot), \\ \quad \text{if exhausted}(x^\sigma, i, j, p_1, \ldots, p_k, \rho) \\ (tt, \bot), \\ \quad \text{otherwise} \end{cases}$$

$$\text{exhausted}(x^o, i, j, p_1, \ldots, p_k, \rho) = \begin{cases} tt, & \text{if Prop}_o \\ ff, & \text{otherwise} \end{cases}$$

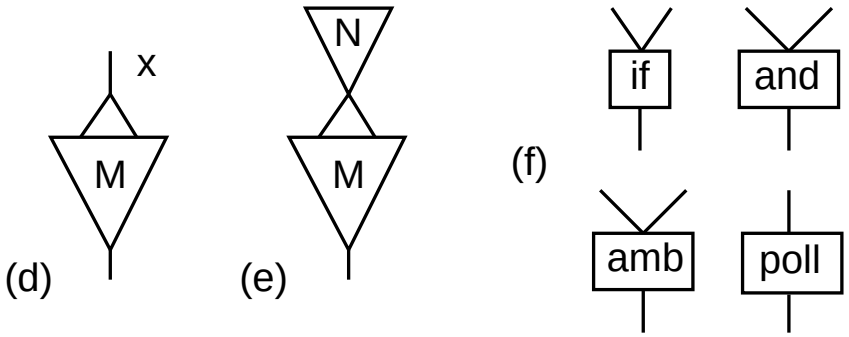
$$\text{exhausted}(x^\iota, i, j, p_1, \ldots, p_k, \rho) = \begin{cases} tt, & \text{if Prop}_\iota \\ ff, & \text{otherwise} \end{cases}$$

- $\text{Prop}_o \equiv$
  $((\exists l, m.\ \mathcal{D}[\![e_{lj}]\!]\rho = tt \wedge \mathcal{D}[\![e_{mj}]\!]\rho = ff) \vee$
  $(\exists l.\ \mathcal{D}[\![e_{lj}]\!]\rho = y^o)) \wedge (p_1 \backslash j \Uparrow \cdots \Uparrow p_k \backslash j)$
- $\text{Prop}_\iota \equiv (\exists l.\ \mathcal{D}[\![e_{lj}]\!]\rho = y^\iota) \wedge (p_1 \backslash j \Uparrow \cdots \Uparrow p_k \backslash j)$
- NPCF = PCF + nondeterministic query

Circuit semantics: Building blocks

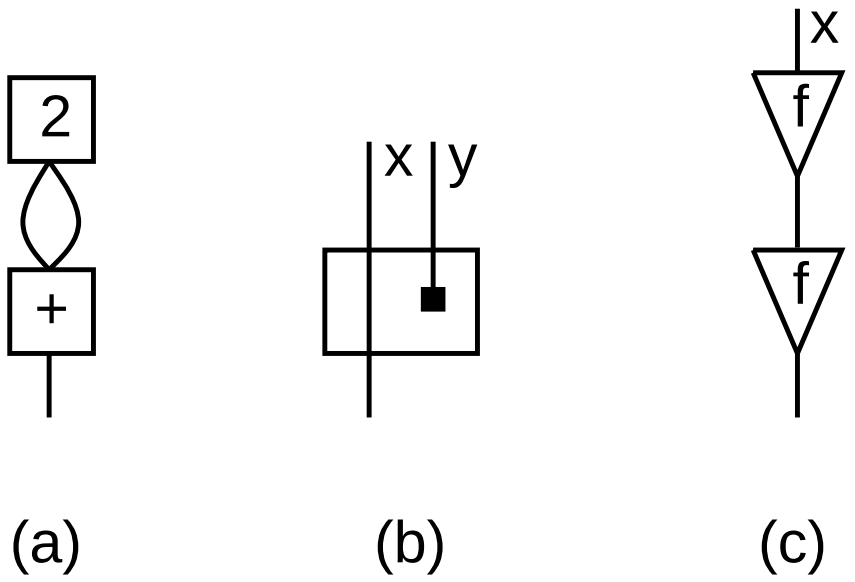


(a) constants, (b) variables, (c) ignored inputs



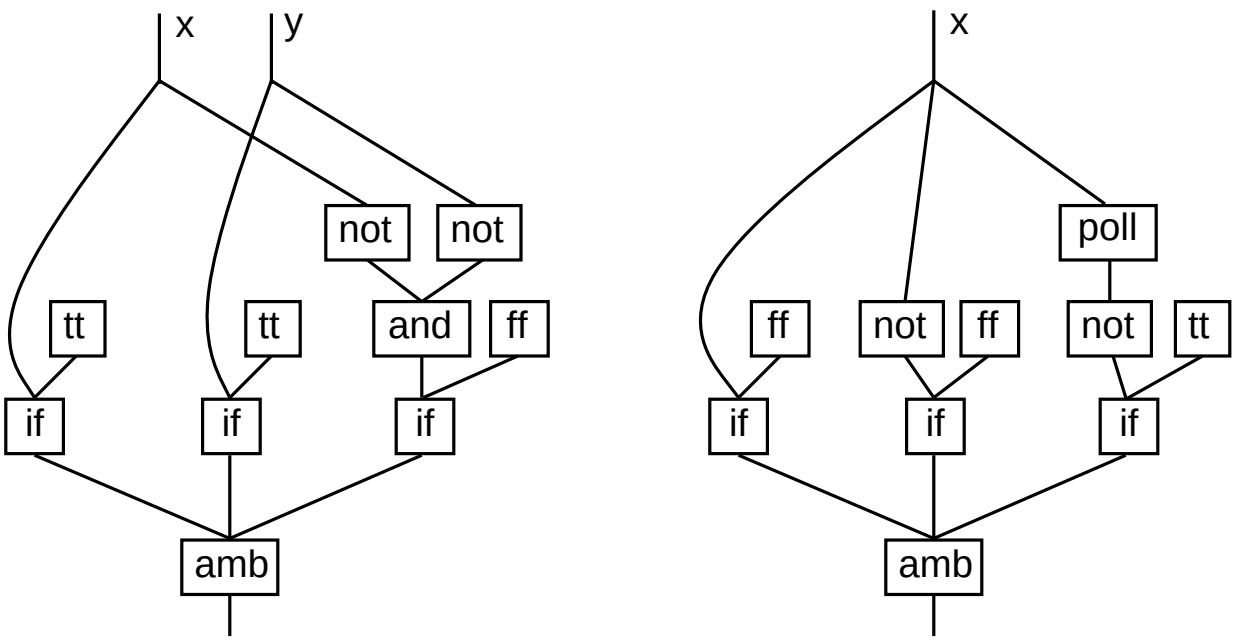
(d) abstraction, (e) application, (f) query

Circuit semantics: PCF examples



(a) $(\lambda x.\ x + x)2$, (b) $\lambda xy.\ x$, (c) $\lambda fx.\ f(fx)$

Circuit semantics: Query examples



(a) por, (b) $not_{\perp}$

Circuit semantics: Properties I

- Define extended semantics for PCF:
  $\mathcal{E}$: Terms $\rightarrow$ Environments $\rightarrow (\cup D_\sigma \times \text{S-Terms}^2)$

  $depth(M_1 + M_2) = 1 + \max(depth(M_1), depth(M_2))$
  $depth(x^\sigma) = x^\sigma$
  $depth(M_1\ M_2) = depth(M_1)\ depth(M_2)$
  $depth(\lambda x^\sigma.\ M) = \Lambda x^\sigma.\ depth(M)$

- For a PCF program M, $\mathcal{E}[\![M]\!]\perp = (v, d, s)$ if and only if the circuit representing M has depth d and size s.

Circuit semantics: Properties II

- [Blelloch & Greiner] Define call-by-speculation:
  $\rho, d \vdash M \longrightarrow_{\text{cbs}} v; d', \hat{d}', w$

$$\frac{\begin{array}{c}\rho, d \vdash M_1 \longrightarrow_{\text{cbs}} v_1; d_1, \hat{d}_1, w_1 \\ \rho, d \vdash M_2 \longrightarrow_{\text{cbs}} v_2; d_2, \hat{d}_2, w_2\end{array}}{\rho, d \vdash M_1 + M_2 \longrightarrow_{\text{cbs}} \begin{array}{l} v_1 + v_2; 1 + \max(d_1, d_2), \\ 1 + \max(\hat{d}_1, \hat{d}_2), 1 + w_1 + w_2\end{array}}$$

- If $\mathcal{E}[\![M]\!]\perp = (v, d, s)$, and
  $\perp, 0 \vdash M \longrightarrow_{\text{cbs}} v; t, \hat{t}, w$, for a program M, then:

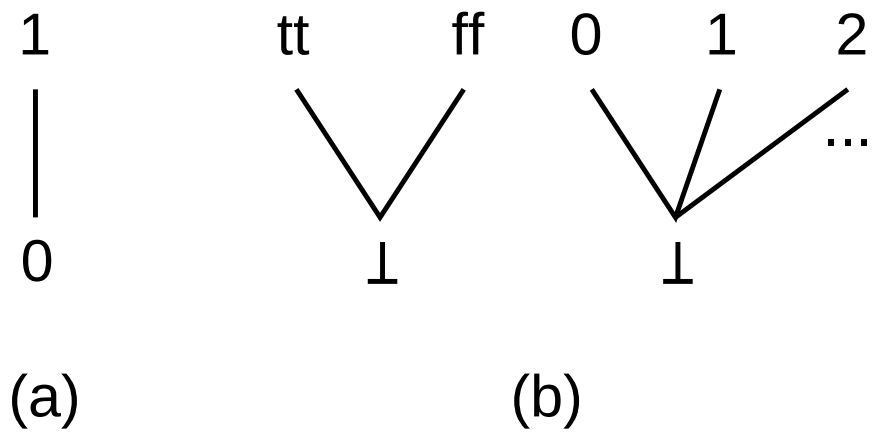
  1. $s \leq w$;
  2. $d \leq \hat{t}$;
  3. if M is conditional-free, then $d = t$.

Method and metric

- Circuit semantics defines parallel complexity: parallel time and parallel work
- Notions of expressiveness:
  - time-expressiveness
  - work-expressiveness
- Assume hardware that can detect undefined inputs
- Question: On such hardware, is NPCF intensionally more time- or work-expressive than DPCF?

Boolean circuits

- Two bases for boolean circuits:
  - monotone: AND, OR
  - De Morgan: AND, OR, NOT
- Monotonicity:



(a) circuit domain, (b) PCF domains

Connection with PCF

- Given a monotone (De Morgan) circuit computing f we can construct from it a DPCF (NPCF) program computing f_{PCF}, whose dimensions in the circuit semantics are the same.
- If DPCF can compute $f : \{\bot, tt\}^n \to tt$, then monotone circuits can compute f_{Bool} with a circuit whose dimensions are the same as the circuit semantics of the DPCF program.

Applications

- [Tardos] There exists a polynomial time computable monotone function whose monotone complexity is exponential.
- There exists a function computable by an NPCF program with polynomial work, but for which the best DPCF program does exponential work.
- [Raz & Wigderson] There is a monotone function in NC^1 that has no monotone NC circuits.
- There exists a function computable by an NPCF program in logarithmic time, and for which the best DPCF program takes more than polylogarithmic time.

Conclusions

- Nondeterministic parallelism is intensionally more expressive than deterministic parallelism:
  - exponentially more work-expressive
  - exponentially more time-expressive
  - assuming hardware which detects undefined inputs
- Established connection between semantics of programming languages and circuit complexity
- Circuit semantics is a useful tool for intensional exploration

Related work

- [Brookes & Geva] Parallel algorithms, intensional semantics
- Profiling semantics:
  - [Blelloch & Greiner] parallel call-by-value, call-by-speculation
  - [Roe] parallel lenient language
  - [Hudak & Anderson] pomsets
- [Felleisen] Theory of expressiveness
- [Sur & Böhm] Id
- [Panangaden & Shanbhogue] Indeterminate Kahn networks