SEQUENTIALITY
&
COMPLEXITY

Jim Royer
Syracuse University

OUTLINE

I. Some History and Background and Background

- Sequential vs. Parallel
- PCF and parallel junk
- "No junk" models of PCF
- Sequentiality beyond PCF

II. The Sequentially Realizable Functionals (SR)

III. Longley's H Functional

IV. Implementations and Costs

V. Open Problems

BACKGROUND

Sequential $\approx$ at most one thing happens at a time
Parallel $\approx$ several things can thing happen at the same time

- Almost all programming languages are intuitively sequential.
- But, nailing down the nature of sequentiality has been surprisingly hard.

History, Part I

Scott, late 1960s, early 1970s

- Invented PCF (originally PCF), a toy language for his semantics work — ancestor of ML, Haskell, . . .
- PCF is "sequential."
- Scott constructed a number of nice semantic models for PCF.
- But he noted that all of these models contained "parallel junk."

PARALLEL OR

- Here is an example of the parallel junk that bothered Scott.

  For $x, \; y \in \{\,\text{true}, \text{false}, \text{diverges}\,\}$,

$$\text{OR}_{||}(x, y) \;=\; \begin{cases} \text{true}, & \text{if } x{\downarrow}\text{=true or } y{\downarrow}\text{=true;} \\ \text{false}, & \text{if } x{\downarrow} = y{\downarrow} = \text{false;} \\ \text{diverges}, & \text{otherwise.} \end{cases}$$

- View the above as a specification, not an algorithm.

- In implementing this:
  - You can't (sequentially) eval x and then go on, if need be, to eval y.
    Why? What if $x{\uparrow}$ and $y{\downarrow} = \text{true}$?
  - You can't (sequentially) eval y and then go on, if need be, to eval x.
    Why? What if $y{\uparrow}$ and $x{\downarrow} = \text{true}$?
  - So, you have to evaluate x and y in "parallel." (Time slicing is OK.)

HISTORY, PART 2

- The lack of a "no junk" model of PCF seemed to indicate we did not understand sequentiality.

- Searching for a "no junk" model of PCF occupied quite a few people for around 20 years. Milner, Plotkin, . . . , Mulry, . . .

- "No junk" models for PCF were found in the early 1990s. Abramsky, Jagadeesan & Malacaria; Hyland & Ong; Nickau; and O'Hearn & Rieck.

- However, these were not grand revelations about sequentiality.
  Why? Because PCF $\not\equiv$ sequentiality.

RECENT HISTORY

- In the late 1990s, van Oosten and Longley discovered the sequentially realizable functionals, a class of sequential functionals that have
  - a fairly simple definition,
  - pretty mathematical properties,
  - things beyond PCF computable functionals, and
  - some fishy complexity theoretic properties.

OUTLINE

I. Some History and Background

II. The Sequentially Realizable Functionals (SR)

- Definition for type level 2, through simple dialog games
- Irredundancy and semi-irredundancy
- Definition for type level 3.
- A key example

III. Longley's H Functional

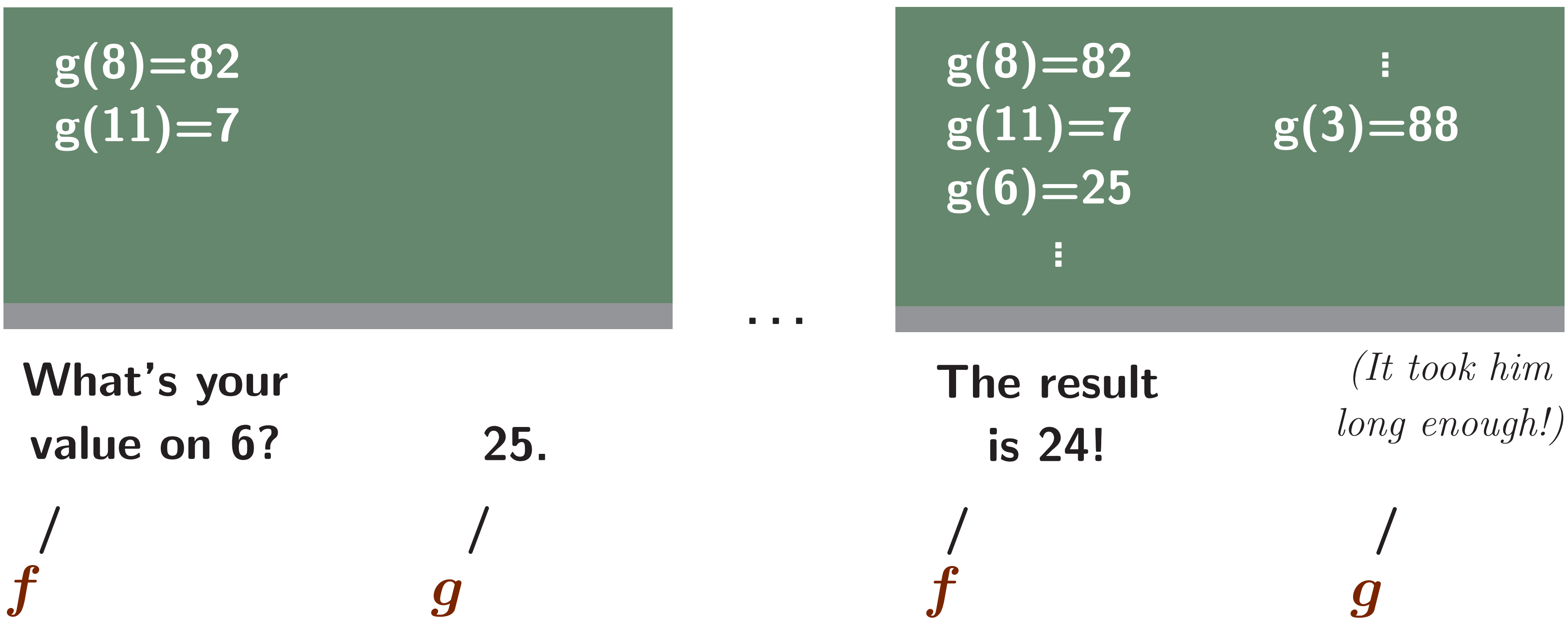
IV. Implementations and Costs

V. Open Problems

A SIMPLE DIALOG GAME

Suppose $r, g \colon \mathrm{N} \rightharpoonup \mathrm{N}$. Consider a game between r and g in which

- r asks g about certain of its values and
- based on these answers, r may eventually announce an N-value as the result of the game.



FORMALIZING THESE DIALOG GAMES

THE RESULT OF A PLAY.

Suppose α is the list of g's previous answers. Then

$$\text{play}(r,g,\alpha) \quad = \quad \begin{cases} \text{play}(\text{r}, \text{g}, (\alpha; \text{y})), & \textbf{if } r\langle\alpha\rangle\!\downarrow = 2x \text{ \& } g(x)\!\downarrow = y; \\ \text{x}, & \textbf{if } r\langle\alpha\rangle\!\downarrow = 2x+1; \\ \uparrow, & \textbf{otherwise.} \end{cases}$$

where

- $2x$ codes "the question x."
- $2x+1$ codes "the answer x".
- $\alpha \approx$ the greenboard.

TWO SPECIAL SORTS OF PLAYS.

- $r \mid g = \text{play}(r, g, [\,])$ — the result of the play of r against g
- $r \bullet g = \lambda x\,.\,\text{play}(r, g, [x])$ — the result of a parameterized play

TOO MANY DEFINITIONS

- α, β, γ — finite sequences over N
- $[x_0, \ldots, x_{j-1}]$ — the sequence $x_0, \ldots, x_{j-1}$
- $[\,]$ $=$ the empty sequence
- $[x_0, \ldots, x_{j-1}]; w \;=\; [x_0, \ldots, x_{j-1}, w]$
- $[x_0, \ldots, x_{j-1}]^{<i} \;=\; [x_0, \ldots, x_{i-1}]$, where $i \leq j$
- $\langle \alpha \rangle$ $=$an encoding of α as an element of N
  $\langle - \rangle$: sequences $\longrightarrow \mathrm{N}$ is "polynomial-time."
- $?x \;=\; 2x$. $\quad !x \;=\; 2x + 1$.
- $$\mathrm{play}(r, g, \alpha) \;=\; \begin{cases} \mathrm{play(r, g, (\alpha; y))}, & \text{if } r\langle \alpha \rangle{\downarrow} = ?x \;\&\; g(x){\downarrow} = y; \\ \mathrm{x}, & \text{if } r\langle \alpha \rangle{\downarrow} = !x; \\ \uparrow, & \text{otherwise.} \end{cases}$$
- $r \mid g \;=\; \mathrm{play}(r, g, [\,])$.
- $r \bullet g \;=\; \lambda x\,.\,\mathrm{play}(r, g, [x])$.

OBSERVATIONS

- Let $\mathcal{B}$ denote $\mathrm{N} \rightharpoonup \mathrm{N}$ and let $\mathcal{PR}$ denote the partial rec. fnctns over N.
- Then, $(\mathcal{B}, \bullet)$ and $(\mathcal{PR}, \bullet)$ are combinatory algebras. (Van Oosten, Longley)
- Given total $g_1,\ g_2 \colon \mathrm{N} \to \mathrm{N}$,

  $g_1 \leq_{\mathrm{T}} g_2$ **iff** $g_1 = r \bullet g_2$, for some $r \in \mathcal{PR}$.

- The SR functionals of type $(\mathrm{N} \rightharpoonup \mathrm{N}) \rightharpoonup \mathrm{N}$ are those of the form:

  $g \;\mapsto\; r \mid g$

  r is called a realizer of the functional $g \mapsto r \mid g$.

- For functionals above type-level 2, we
  - parameterize (as in definition of $\bullet$), and
  - impose extensionality constrains.

$r \mid g = \mathrm{play}(r, g, [\,])$. $\qquad\qquad r \bullet g = \lambda x\,.\,\mathrm{play}(r, g, [x])$.

DIGRESSION: IRREDUNDANCY

It is helpful to have a well-behaved realizer, r, in $g \mapsto r \mid g$

DEFINITION An $r \in \mathcal{PR}$ is irredundant iff
for all $\alpha = [x_0, \ldots, x_{k-1}]$ with $r\langle\alpha\rangle\!\downarrow$,

(i) for each $i < k$, $r\langle\alpha^{<i}\rangle\!\downarrow$,

(ii) in $r\langle\alpha^{<0}\rangle, \ldots, r\langle\alpha^{<k}\rangle$ no question can appear more than once and an answer may only appear as the last element.

LEMMA (Longley) There is an $\mathrm{irr} \in \mathcal{PR}$ such that for all $r \in \mathcal{PR}$,

(a) for each $g \in \mathcal{PR}$, $(\mathrm{irr} \bullet r) \mid g = r \mid g$.

I.e., $\mathrm{irr} \bullet r$ and r determine the same functional.

(b) $\mathrm{irr} \bullet r$ is irredundant, and

(c) $\mathrm{irr} \bullet r = r$ when r is irredundant.

So, irr is a retraction of $\mathcal{PR}$ onto $\{\, r \in \mathcal{PR} \mid r \text{ is irredundant} \,\}$.

SEMI-DIGRESSION: SEMI-IRREDUNDANCY

DEFINITION An $r \in \mathcal{PR}$ is semi-irredundant iff
for all $\alpha = \langle x_0, \ldots, x_{k-1} \rangle$ with $r\langle\alpha\rangle\downarrow$,
(i) for each $i < k$, $r\langle\alpha^{<i}\rangle\downarrow$, and
(ii) the questions in $r\langle\alpha^{<0}\rangle, \ldots, r\langle\alpha^{<k}\rangle$ are pointwise distinct.

These guys are not that interesting, except that they help Longley solve a certain problem — so, they figure in the definition of SR.

$\mathcal{PR}_{s.i.}$ = the semi-irredundant elements of $\mathcal{PR}$.

INTERPRETING THE TYPES $\overline{0}$, $\overline{1}$ and $\overline{2}$

The Pure Types: $\overline{0} \cong \mathrm{N}$ and $\overline{n+1} \cong \overline{n} \rightharpoonup \mathrm{N}$.

$[\![\, \overline{n} \,]\!]$ = the SR objects of type $\overline{n}$.

Type Levels 0 and 1

We take $[\![\, \overline{0} \,]\!] = \mathrm{N}$ and $[\![\, \overline{1} \,]\!] = \mathcal{PR}$.

Type Level 2

We take $[\![\, \overline{2} \,]\!]$ to be the collection of all $F \colon [\![\, \overline{1} \,]\!] \rightharpoonup \mathrm{N} \ni$

for some $r \in \mathcal{PR}_{s.i.}$,

$$F(g) \;=\; r \mid g, \text{ for each } g \in \mathcal{PR} \qquad (*)$$

- $\|F\|_{\overline{2}}$ = the collection of $r \in \mathcal{PR}_{s.i.}$ such that $(*)$ holds
  = the set of realizers for F
- $r \sim_2 r'$ **iff** $r, r' \in \|F\|_{\overline{2}}$ for some F.

INTERPRETING THE TYPE $\overline{3}$

Type Level 2, Again

We take $[\![\,\overline{2}\,]\!]$ to be the collection of all $F\colon [\![\,\overline{1}\,]\!] \rightharpoonup \mathrm{N} \ni$
for some $r \in \mathcal{PR}_{s.i.}$,

$$F(g) \;=\; r \mid g_F, \text{ for each } g_F \in \mathcal{PR} \qquad (*)$$

- $\|F\|_{\overline{2}}$ $=$ the collection of $r \in \mathcal{PR}_{s.i.}$ such that $(*)$ holds
  $=$ the set of realizers for F
- $r \sim_2 r'$ **iff** $r, r' \in \|F\|_{\overline{2}}$ for some F.

Type Level 3

We take $[\![\,\overline{3}\,]\!]$ to be the collection of all $\Psi\colon [\![\,\overline{2}\,]\!] \rightharpoonup \mathrm{N} \ni$
for some $r \in \mathcal{PR}_{s.i.}$,

$$\Psi(F) \;=\; r \mid g_F, \text{ for all } g_F \in \|F\|_{\overline{2}} \qquad (**)$$

- $\|\Psi\|_{\overline{3}}$ $=$ the coll. of $r \in \mathcal{PR}_{s.i.}$ such that $(**)$ holds
  $=$ the set of realizers for Ψ
- $r \sim_3 r'$ **iff** $r, r' \in \|\Psi\|_{\overline{3}}$ for some Ψ.

MORE ON TYPE $\overline{3}$

Type Level 3, Again

We take $[\![\, \overline{3} \,]\!]$ to be the collection of all $\Psi : [\![\, \overline{2} \,]\!] \rightharpoonup \mathrm{N} \ni$

for some $r \in \mathcal{PR}_{s.i.}$,

$$\Psi(F) \;=\; r \mid g, \text{ for all } g \in \|F\|_{\overline{2}} \qquad (**)$$

- $\|\Psi\|_{\overline{3}} \;=$ the coll. of $r \in \mathcal{PR}_{s.i.}$ such that $(**)$ holds
- $r \sim_3 r'$ **iff** $r, r' \in \|\Psi\|_{\overline{3}}$ for some Ψ.

Note: $r \in \|\Psi\|_{\overline{3}}$ **iff** $\underbrace{\text{for all } \mathrm{g}, \mathrm{g}' \text{ with } \mathrm{g} \sim_2 \mathrm{g}',\ \ \mathrm{r} \mid \mathrm{g} = \mathrm{r} \mid \mathrm{g}'.}_{\sim_2-\textbf{extensionality}}$

Extensionality is a root of the forthcoming troubles

$\|\overline{2}\| \;=\; \bigcup\{\, \|F\| \;\vdots\; F \in [\![\, \overline{2} \,]\!] \,\} \;=\; \mathcal{PR}_{s.i.} =$ the type-2 realizers.

$\|\overline{3}\| \;=\; \bigcup\{\, \|\Psi\| \;\vdots\; \Psi \in [\![\, \overline{3} \,]\!] \,\} \;=$ the $\sim_2$-extensional r = the type-3 realizers.

Note: $\|\overline{3}\| \subset \|\overline{2}\|$.

A KEY EXAMPLE, PART 1

- Fix an $n \in \mathrm{N}$.
- Define:

$$G^n(f) = \begin{cases} 0, & \text{if } f(0) = f(1) = \cdots = f(n-1) = 0; \\ \uparrow, & \text{otherwise.} \end{cases}$$

- $\zeta_i =$ a length-i sequence of 0's.
- π ranges over permutations of $\{\, 0, \ldots, n-1 \,\}$.
- Define $g^n_\pi \colon \mathrm{N} \rightharpoonup \mathrm{N}$ by:

$$g^n_\pi\langle\alpha\rangle = \begin{cases} ?\pi(i), & \text{if } \alpha = \zeta_i \text{ for } i < n; \\ !0, & \text{if } \alpha = \zeta_n; \\ \uparrow, & \text{otherwise.} \end{cases}$$

- Each g^n_π is an irredundant realizer for G^n.

A KEY EXAMPLE, PART 2

- Let $r_n \in \mathcal{PR}$ be given by:

$$r_n\langle\alpha\rangle \;=\; \begin{cases} ?\langle\zeta_i\rangle, & \text{if } \alpha = [?x_0,\ldots,?x_k] \text{ where } k < n \text{ and} \\ & \quad x_0,\ldots,x_k \text{ are } < n \text{ and pairwise distinct;} \\ !0, & \text{if } \alpha = [?x_0,\ldots,?x_k,!0] \text{ where} \\ & \quad x_0,\ldots,x_k \text{ are } < n \text{ and pairwise distinct;} \\ \uparrow, & \text{otherwise.} \end{cases}$$

- r_n is $\sim_2$-extensional and realizes $\Psi^n \in [\![\,\overline{3}\,]\!]$ where

$$\Psi^n(F) \;=\; \begin{cases} 0, & \text{if } G^n \sqsubseteq_{\text{tr}} F; \\ \uparrow, & \text{otherwise.} \end{cases}$$

Ψ^n is not Scott-continuous, and hence, not PCF computable.

OUTLINE

I. Some History and Background

II. The Sequentially Realizable Functionals (SR)

III. Longley's H Functional

- PCF $+ H =$ SR
- Longley's construction for H
- Revisiting the key example.

IV. Implementations and Costs

V. Open Problems

LONGLEY'S H

DEFINITION We say $t \in \mathcal{PR}$ realizes a retraction iff for all $r \in \mathcal{PR}_{s.i.}$:

(i) $t \bullet r$ is $\sim_2$-extensional and irredundant.
(Roughly, $\mathrm{t}\colon \|\overline{2}\| \to \|\overline{3}\|$.)

(ii) $t \bullet r \sim_3 r$ when r is $\sim_2$-extensional.
(Roughly, t is the identity on $\|\overline{3}\|$ mod $\sim_3$.)

THEOREM (Longley) There is an h that realizes a retraction.

- This h realizes Longley's H.
- Using h, one can build an interpreter for $[\![\,\overline{3}\,]\!]$.
- Through H, there are analogous retracts for each other simple type.
— So these types have interpreters too.
- PCF $+\ H$ determines SR because
h provides names for the elements of SR.

TOO MANY DEFINITIONS, TWO

- ν — finite elm of $\mathrm{N} \rightharpoonup \mathrm{N}$
- G — an element of $[\![\,\overline{2}\,]\!]$.
- $f \subset g$ iff . . . the usual thing . . .

- ν is G-minimal iff $G(\nu)\!\downarrow$ and, for each $\nu' \subset \nu$, $G(\nu')\!\uparrow$. (ν unique)

- The trace of G: $\mathrm{tr}(G) = \{\, (\nu, G(\nu)) \;\vdots\; \nu$ is G-minimal $\}$.
  Each G is completely determined by its trace.

- $G \sqsubseteq_{\mathrm{tr}} G'$ iff $\mathrm{tr}(G) \subseteq \mathrm{tr}(G')$.

- G is finite iff $\mathrm{tr}(G)$ is finite.

- Given (the graph of) a ν,
  one can eff. decide whether $\nu \in \|G\|_2$ for some G
  and if so, one can construct $\mathrm{tr}(G)$ from ν.

TWO LEMMAS OF LONGLEY

Let $\|G\|^{\mathcal{I}}$ be the collection of irredundant realizers for G.

LEMMA 1 (Finite G's have nice, finite representations.)
Suppose $G \in [\![\, \overline{2} \,]\!]$ is finite. Then:

(a) $\|G\|^{\mathcal{I}}$ is finite.

(b) Each element of $\|G\|^{\mathcal{I}}$ has finite domain.

(c) Each $r \in \|G\|$ extends exactly one element of $\|G\|^{\mathcal{I}}$.

(d) From $\mathrm{tr}(G)$ one can effectively compute
a finite list of finite graphs
making up the elms of $\|G\|^{\mathcal{I}}$.

LEMMA 2 (Computations have finite support)
Suppose $\Psi \in [\![\, \overline{3} \,]\!]$, $G \in [\![\, \overline{2} \,]\!]$, and $z \in \mathrm{N}$.
Then $\Psi(G)\!\downarrow = z$ iff for each $r \in \|\Psi\|$,
there is a finite G_0 with $G_0 \sqsubseteq_{\mathrm{tr}} G \ni$

(i) for each $\nu \in \|G_0\|^{\mathcal{I}}$, $(r \mid \nu)\!\downarrow = z$, and

(ii) each $g \in \|G\|$ extends exactly one elm of $\|G_0\|^{\mathcal{I}}$.

LONGLEY'S CONSTRUCTION

Sketch of the comp. of the play of $(h \bullet r)$ against g.

Play out "$r \mid g$" until, if ever, an result z is produced.

$\nu_0 := $ the finite part of g used in the play of "$r \mid g$."

If ν_0 realizes some finite member of $[\![\, \overline{2} \,]\!]$,
  then let G_0 denote this finite member,
  else diverge. (* Since by Lemma 2, $r \notin \|\overline{3}\|$. *)
(* By Lemma 1, $\|G_0\|^{\mathcal{I}}$ is finite. *)

For each $\nu \in \|G_0\|^{\mathcal{I}}$ **do**: (* *)
  (* Check for a failure of extensionality. *)
  Play out "$r \mid \nu$" until, if ever, a result z' is produced.
  If $z \neq z'$, **then** diverge.
End for

Output $!z$.

End sketch

Why the ?

REVISITING THE KEY EXAMPLE, PART 1

- Fix an $n \in \mathrm{N}$.
- $\zeta_i =$ a length-i sequence of 0's.
- π ranges over perms of $\{ 0, \ldots, n-1 \}$.
- Define $g^n_\pi \colon \mathrm{N} \rightharpoonup \mathrm{N}$, $G^n \in [\![\overline{2}]\!]$, and $\Psi^n \in [\![\overline{3}]\!]$ by:

$$G^n(f) = \begin{cases} 0, & \text{if } f(0) = \cdots = f(n-1) = 0; \\ \uparrow, & \text{otherwise.} \end{cases}$$

$$g^n_\pi\langle\alpha\rangle = \begin{cases} ?\pi(i), & \text{if } \alpha = \zeta_i \text{ for } i < n; \\ !0, & \text{if } \alpha = \zeta_n; \\ \uparrow, & \text{otherwise.} \end{cases}$$

$$\Psi^n(F) = \begin{cases} 0, & \text{if } G^n \sqsubseteq_{\mathrm{tr}} F; \\ \uparrow, & \text{otherwise.} \end{cases}$$

- $\|G^n\|^{\mathcal{I}} = \{ g^n_\pi \mid \pi \text{ is as above} \}$.

REVISITING THE KEY EXAMPLE, PART 2

- Let $r_n \in \mathcal{PR}$ be given by:

$$r_n\langle\alpha\rangle = \begin{cases} ?\langle\zeta_i\rangle, & \text{if } \alpha = [?x_0, \ldots, ?x_k] \text{ where } k < n \text{ and} \\ & \quad x_0, \ldots, x_k \text{ are } < n \text{ and pairwise distinct;} \\ !0, & \text{if } \alpha = [?x_0, \ldots, ?x_k, !0] \text{ where} \\ & \quad x_0, \ldots, x_k \text{ are } < n \text{ and pairwise distinct;} \\ \uparrow, & \text{otherwise.} \end{cases}$$

- r_n is $\sim_2$-extensional and realizes $\Psi^n \in [\![\,\overline{3}\,]\!]$.
- So, $h \bullet r_n \;\sim_3\; r_n$.
- The interactions in both "$(h \bullet r_n) \mid g^n_\pi$" and "$r_n \mid g^n_\pi$" are identical.
- But, at the end of the play of "$(h \bullet r_n) \mid g^n_\pi$,"
  $(h \bullet r_n)$ internally checks that $(r_n \mid g^n_{\pi'})\!\downarrow = 0$
  for each of the n! many $g^n_{\pi'}$.
- Intuitively, cost("$(h \bullet r_n) \mid g^n_\pi$") $>$ n! $\cdot$ cost("$r_n \mid g^n_\pi$").

OUTLINE

I. Some History and Background

II. The Sequentially Realizable Functionals (SR)

III. Longley's H Functional

IV. Implementations and Costs

- The cost of a play of a game
- Poly-overhead instantiations of functionals
- THM: $\mathrm{P} \neq \mathrm{NP} \implies H$ has no poly-overhead inst.

V. Open Problems

THE COST OF A PLAY

Conventions

- $\langle \varphi_i \rangle_{i \in \mathrm{N}}$ — an acceptable indexing based on RAMs
- We use the logarithmic cost model for RAMs.
- $\Phi_i(x)$ = the cost of running program i on input x

DEFINITION For each $i \in \mathrm{N}$ and $g \in \mathcal{PR}$, let $C(i,g) =$

$$\begin{cases} \sum_{j \leq k} \Phi_i \langle \alpha^{<j} \rangle, & \text{if (i) } (\varphi_i \mid g)\downarrow\text{, where } \alpha \text{ is the final sequence in the play of ``} \varphi_i \mid g\text{,'' and } k \text{ is the length of } \alpha; \\ \infty, & \text{if (ii) the play of ``} \varphi_i \mid g\text{'' is infinite or else } \varphi_i \text{ diverges at some point in this play;} \\ \text{undefined}, & \text{(iii) otherwise.} \end{cases}$$

We call $C(i,g)$ the cost of playing program i against g.

Deciding whether $C(i,g) \leq t$ is "partial basic feasible" in $|g|$, $|i|$, and $|0^t|$.

IMPLEMENTATING A REALIZER OF A RETRACTION

Suppose t realizes a retraction.

DEFINITION. $inst \in \mathcal{PR}$ is an instantiation of t iff for all i with $\varphi_i \in \mathcal{PR}_{s.i.}$, we have $inst(i)\downarrow$ and

$$\varphi_{inst(i)} = t \bullet \varphi_i.$$

Note:

- $inst$ is a mapping over programs.
- So long as $inst$ satisfies

$$\varphi_{inst(i)} = h \bullet \varphi_i$$

  $inst$ and L's construction may work very differently.

IMPLEMENTATING A REALIZER OF A RETRACTION, FEASIBLY

Suppose t realizes a retraction and $inst$ is an instantiation of t.

DEFINITION. $inst$ has polynomial overhead iff

(i) $inst$ is polynomially time computable and

(ii) there is a polynomial p $\ni$
for all $i \in \mathrm{N}$ such that φ_i is $\sim_2$-extensional and
for all $g \in \mathcal{PR}_{s.i.}$, we have

$$C(inst(i), g) \;\leq\; p\left(\max\left\{ C(i, \widehat{g}) \;:\; \begin{array}{l} \widehat{g} = g \text{ or } \widehat{g} \text{ is irre-} \\ \text{dundant and } \widehat{g} \sim_2 g \end{array} \right\}\right).$$

Notes:

- By Lemma 2, the above maximization is over finitely many things provided $C(i, g) < \infty$.
- Roughly, the LHS of the above inequality is

$$poly\left(\begin{array}{c} \text{the worse cost of } i\text{'s play against} \\ g \text{ and its irr. } \sim_2\text{-equivalents} \end{array} \right)$$

THE BAD NEWS

THEOREM 1

If $\mathrm{P} \neq \mathrm{NP}$, then h has no poly overhead instantiations.

IDEA OF THE PROOF

- **We use the G^n's and g^n_π's from before.**
- **Suppose $inst$ a poly overhead inst. of h.**

  If **for all π we have** $(\varphi_i \mid g^n_\pi)\downarrow = a$ and $C(i, g^n_\pi) \leq b$,

  then $C(inst(i), g^n_\pi) \leq p(b)$.
- **By Longley's construction:**

  if **for some π we have** $(r \mid g^n_\pi)\uparrow$,

  then **for all π', we have** $((h \bullet r) \mid g^n_{\pi'})\uparrow$.

More. . .

THE BAD NEWS, CONTINUED

- We construct a poly time function $f \ni$ for all x

  if x codes an instance of Vertex Cover on n vertices

  then for each π:

  If π codes a solution, then $(\varphi_{f(x)} \mid g^n_\pi)\uparrow$.

  If π fails to code a solution, then $C(f(x), g^n_\pi) \leq p_f(|x|)$.

- If h does have a polynomial overhead instantiation,
  then we can build a poly-time Vertex Cover tester from the above.

- But Vertex Cover is NP-complete!

THEOREM 2 Suppose t realizes a retraction.
If $\mathrm{P} \neq \mathrm{NP}$, then t has no poly overhead instantiations.

IDEA OF THE PROOF By a bit of analysis, we find
if for some π we have $(r \mid g^n_\pi)\uparrow$,
then for all π', we have $((t \bullet r) \mid g^n_{\pi'})\uparrow$.
So, we can adapt the proof of Theorem 1 to this case.

OPEN PROBLEMS

- Does SR have an efficient programming language?
  (How do you state this question more precisely?)

- Is there a reasonable "feasible" subclass of SR?

- There are longstanding complaints against the continuous functionals of finite type. (E.g., They are hard to implement, hard to program in.)
  Is there some theory that can either back up or refute this whining?
  (I like the continuous functionals.)

- GAMES + REALIZERS + COMPLEXITY
  This seems like a very interesting mix to explore.
  Our results show that when you pose the question the right way, answering it may not be that hard.