SRL transformations can grow as fast as any primitive recursive function

May 2022

Luca Roversi
Luca Paolini
Armando Matos

Abstract

SRL computations can grow as fast as any primitive recursive function in the sense that

For any positive integer k there are positive SRL programs with outputs larger than $2 \overset{k}{\uparrow} n$.

The proof is constructive: the corresponding SRL programs are described.

Note. See Knuth's notation [Knu76].

Note. The Ackermann function $a(m, n) = [2 \overset{m-2}{\uparrow} (n + 3) - 3]$, see [MP80, MP95], is not primitive recursive and thus can not be the output of a SRL computation.

(Abstract: primitive recursive as large as SRL)

The “other direction” of the inequality,

$$\boxed{\text{Primitive recursive functions can grow as fast as any SRL transformation}}$$

is simpler to prove and is not discussed here.

For that purpose a small overhead simulation technique, for instance represent $x \in \mathbb{Z}$ by a pair of non-negative integers can be used.

Index

A blank slide marks the end of the section

▷ Goals and general observations

Goal. Prove that the outputs of SRL transformations may grow as fast as any primitive recursive function. The converse also holds.

▷ Goals and general observations

Goal. Prove that the outputs of SRL transformations may grow as fast as any primitive recursive function. The converse also holds.

- For any positive integer k, there are SRL programs P with outputs that grow faster than $2 \overset{k}{\uparrow} n$, where the input values of $P(\overline{x})$ are either 0 or n, $n \in \mathbb{N}$. Thus $|\overline{x}| \leq n$.
- The programs P are explicitly described, as a function of k.

▷ Goals and general observations

Goal. Prove that the outputs of SRL transformations may grow as fast as any primitive recursive function. The converse also holds.

- For any positive integer k, there are SRL programs P with outputs that grow faster than $2 \overset{k}{\uparrow} n$, where the input values of $P(\overline{x})$ are either 0 or n, $n \in \mathbb{N}$. Thus $|\overline{x}| \leq n$.
- The programs P are explicitly described, as a function of k.

▷ Goals and general observations

The SRL language is a very simple, but non-trivial, reversible total language, whose programs define bijections $\mathbb{Z}^k \to \mathbb{Z}^k$ for some positive integer k.

See [Mat03, MRP18, PPR16, Per14].

▷ Comparing different things

SRL computation $P : \mathbb{Z}^n \to \mathbb{Z}^n$, bijection

PR computation $Q : \mathbb{N}^n \to \mathbb{N}$, function

▷ Comparing different things

“For any PR Q there is a SRL program P that grows as fast as Q” It is enough to consider a part of the SRL computations:

▷ Comparing different things

“For any PR Q there is a SRL program P that grows as fast as Q” It is enough to consider a part of the SRL computations:

1. SRL programs without the “dec” instruction.

▷ Comparing different things

"For any PR Q there is a SRL program P that grows as fast as Q" It is enough to consider a part of the SRL computations:

1. SRL programs without the "dec" instruction.
2. Non-negative SRL inputs.

▷ Comparing different things

“For any PR Q there is a SRL program P that grows as fast as Q” It is enough to consider a part of the SRL computations:

1. SRL programs without the “dec” instruction.
2. Non-negative SRL inputs.
3. A particular output of the SRL computation is selected.

▷ Comparing different things

"For any PR Q there is a SRL program P that grows as fast as Q" It is enough to consider a part of the SRL computations:

1. SRL programs without the "dec" instruction.
2. Non-negative SRL inputs.
3. A particular output of the SRL computation is selected.

Property. The contents of any SRL register never decreases — and thus is never negative.

▷ A notation used in this work

“$2 \overset{k}{\square} n$” notation / Knuth’s hyperpower notation / Fibonacci sequences.

A notation used in this report:

$$2 \overset{k}{\square} n \qquad = \qquad 2^{2^{\cdot^{\cdot^{\cdot^{2^{n}}}}}}$$ where the number of 2’s is k.

▷ A notation used in this work

index

“$2 \overset{k}{\square} n$” notation / Knuth’s hyperpower notation / Fibonacci sequences.

A notation used in this report:

$$2 \overset{k}{\square} n \qquad = \qquad 2^{2^{\cdot^{\cdot^{\cdot^{2^n}}}}}$$ where the number of 2’s is k.

For instance, $2 \overset{1}{\square} n = 2^n$.

▷ A notation used in this work

index

“$2 \overset{k}{\square} n$” notation / Knuth’s hyperpower notation / Fibonacci sequences.

A notation used in this report:

$$2 \overset{k}{\square} n \qquad = \qquad 2^{2^{\cdot^{\cdot^{\cdot^{2^n}}}}} \text{ where the number of 2's is } k.$$

For instance, $2 \overset{1}{\square} n = 2^n$.

Right associativity of exponentiation is assumed.

“Hyperpower” Knuth’s notation

index

Knuth “hyperpower” notation [Knu76]

$$a \overset{1}{\uparrow} n \quad = \quad a \uparrow n \; = \; a^n$$

“Hyperpower” Knuth’s notation

Knuth “hyperpower” notation [Knu76]

$$\begin{aligned} a \overset{1}{\uparrow} n &= a \uparrow n = a^n \\ a \overset{m}{\uparrow} n &= \underbrace{a \overset{m-1}{\uparrow} a \overset{m-1}{\uparrow} \cdots \overset{m-1}{\uparrow} a}_{n\ a\text{'s}} \qquad \text{for } m \geq 2. \end{aligned}$$

“Hyperpower” Knuth’s notation

Knuth “hyperpower” notation [Knu76]

$$
\begin{aligned}
a \overset{1}{\uparrow} n &= a \uparrow n = a^n \\
a \overset{m}{\uparrow} n &= \underbrace{a \overset{m-1}{\uparrow} a \overset{m-1}{\uparrow} \cdots \overset{m-1}{\uparrow} a}_{n\ a\text{'s}} \qquad \text{for } m \geq 2.
\end{aligned}
$$

For instance,

$$2 \overset{2}{\uparrow} n = 2 \uparrow 2 \cdots \uparrow 2 \qquad \text{number of 2's is } n$$

A simple property

Property. For a, $m \geq 1$, $n \geq 2$:

$$a \overset{m}{\uparrow} n = a \overset{m-1}{\uparrow} [a \overset{m}{\uparrow} (n-1)]$$

(Assuming right associativity of exponentiation, the square brackets may be removed.)

A simple property

Property. For a, $m \geq 1$, $n \geq 2$:

$$a \overset{m}{\uparrow} n = a \overset{m-1}{\uparrow} [a \overset{m}{\uparrow} (n-1)]$$

(Assuming right associativity of exponentiation, the square brackets may be removed.)

In the sequel: $a = 2$, and we rename m and n as k and m.
For instance

$$2 \overset{k}{\uparrow} m = 2 \overset{k-1}{\uparrow} [2 \overset{k}{\uparrow} (m-1)]$$

Ackermann function: recursive and closed-form

index

A recursive definition

$$a(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ a(m-1,1) & \text{if } m \geq 1 \text{ and } n = 0 \\ a(m-1, a(m, n-1)) & \text{if } m \geq 1 \text{ and } n \geq 1 \end{cases}$$

Ackermann function: recursive and closed-form

index

A recursive definition

$$a(m,n) = \begin{cases} n+1 & \text{if } m = 0 \\ a(m-1,1) & \text{if } m \geq 1 \text{ and } n = 0 \\ a(m-1, a(m, n-1)) & \text{if } m \geq 1 \text{ and } n \geq 1 \end{cases}$$

A closed-form expression, see [MP95]

$$a(m,n) = 2 \overset{m-2}{\uparrow} (n+3) - 3$$

Ackermann function: not primitive recursive

Theorem. The Ackermann function $a(m, n)$ is not primitive recursive. □

Ackermann function: not primitive recursive

Theorem. The Ackermann function $a(m, n)$ is not primitive recursive. $\square$

Theorem. The diagonal Ackermann function $d(m) = a(m, m)$ is not primitive recursive. $\square$

Fibonacci sequences

Fibonacci sequences

Definition.

$$\begin{cases} F_0(x,y) & = & x \\ F_1(x,y) & = & y \\ F_n(x,y) & = & F_{n-1}(x,y) + F_{n-2}(x,y) \end{cases}$$

Fibonacci sequences

Fibonacci sequences

Definition.

$$\begin{cases} F_0(x,y) & = & x \\ F_1(x,y) & = & y \\ F_n(x,y) & = & F_{n-1}(x,y) + F_{n-2}(x,y) \end{cases}$$

Note that $F_{n-2}(x,y) = F_n(x,y) - F_{n-1}(x,y)$. Thus $F_n(x,y)$ is defined for every $n \in \mathbb{Z}$ (given x and y).

Fibonacci sequences

Fibonacci sequences

Definition.

$$\begin{cases} F_0(x,y) & = & x \\ F_1(x,y) & = & y \\ F_n(x,y) & = & F_{n-1}(x,y) + F_{n-2}(x,y) \end{cases}$$

Note that $F_{n-2}(x,y) = F_n(x,y) - F_{n-1}(x,y)$. Thus $F_n(x,y)$ is defined for every $n \in \mathbb{Z}$ (given x and y).

For $x = 0$, $y = 1$:

n :	…	-3	-2	-1	1	0	1	2	3	4	5	6	7	8	…
$F_n(0,1)$:	…	-3	2	-1	1	0	1	1	2	3	5	8	13	21	…

Closed-form expression of $F_n(0,1)$

$$\begin{aligned} F_n(0,1) &= \frac{1}{\sqrt{5}}(\phi^n - \hat{\phi}^n) \\ &= \text{round}(\phi^n/(\sqrt{5})) \qquad \text{for } n \geq 0 \end{aligned}$$

Closed-form expression of $F_n(0,1)$

index

$$\begin{aligned} F_n(0,1) &= \frac{1}{\sqrt{5}}(\phi^n - \hat{\phi}^n) \\ &= \text{round}(\phi^n/(\sqrt{5})) \qquad \text{for } n \geq 0 \end{aligned}$$

where

- $\phi = (1+\sqrt{5})/2$,
- $\hat{\phi} = (1-\sqrt{5})/2$,
- $\text{round}(x) = \lfloor x + 0.5 \rfloor$.

▷ Programs that grow faster than $2 \overset{1}{\uparrow} n$

SRL programs: lower bound $2 \overset{1}{\uparrow} n = 2^n$

A SRL program:

$$Q(n, a, b): \quad \text{for } n(\text{for } b(\text{inc } a);\ \text{for } a(\text{inc } b)).$$

▷ Programs that grow faster than $2 \overset{1}{\uparrow} n$

SRL programs: lower bound $2 \overset{1}{\uparrow} n = 2^n$

A SRL program:

$$Q(n, a, b) : \quad \text{for } n(\text{for } b(\text{inc } a);\ \text{for } a(\text{inc } b)).$$

Initial values $a = 0$, $b = 1$. Some final values a', b':

▷ Programs that grow faster than $2 \overset{1}{\uparrow} n$

SRL programs: lower bound $2 \overset{1}{\uparrow} n = 2^n$

A SRL program:

$$Q(n, a, b): \quad \text{for } n(\text{for } b(\text{inc } a);\ \text{for } a(\text{inc } b)).$$

Initial values $a = 0$, $b = 1$. Some final values a', b':

n	a'	b'
0	0	1
1	1	2
2	3	5

$(n' = n)$

Recall Fibonacci...

index

m :	...	-3	-2	-1	1	0	1	2	3	4	5	6	7	...
$F_m(0,1)$:	...	-3	2	-1	1	0	1	1	2	3	5	8	13	...

Recall Fibonacci...

m :	...	-3	-2	-1	1	0	1	2	3	4	5	6	7	...
$F_m(0,1)$:	...	-3	2	-1	1	0	1	1	2	3	5	8	13	...

$$\left\{ \begin{array}{lcl} a' & = & F_{2n}(0,1) \\ b' & = & F_{2n+1}(0,1) \\ n' & = & n \end{array} \right. \qquad (\S)$$

(Proof ahead...)

Recall Fibonacci...

m :	...	-3	-2	-1	1	0	1	2	3	4	5	6	7	...
$F_m(0,1)$:	...	-3	2	-1	1	0	1	1	2	3	5	8	13	...

$$\left\{ \begin{array}{lcl} a' & = & F_{2n}(0,1) \\ b' & = & F_{2n+1}(0,1) \\ n' & = & n \end{array} \right. \qquad (\S)$$

(Proof ahead...)

Example $n = 100$.

$$\left\{ \begin{array}{lcl} a' & = & 280571172992510140037611932413038677189525 \\ b' & = & 453973694165307953197296969697410619233826 \\ n' & = & 100 \end{array} \right.$$

A note on the proof of (§)

Proof of (§) may be based on the following observations

A note on the proof of (§)

Proof of (§) may be based on the following observations

- Base case, $a = 0$, $b = 1$: trivial

A note on the proof of (§)

Proof of (§) may be based on the following observations

- Base case, $a = 0$, $b = 1$: trivial
- Step of the Fibonacci sequence:

$$\begin{cases} a' = b \\ b' = a + b \end{cases} \qquad \text{matrix } \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

A note on the proof of (§)

Proof of (§) may be based on the following observations

- Base case, $a = 0$, $b = 1$: trivial
- Step of the Fibonacci sequence:

$$\begin{cases} a' = b \\ b' = a + b \end{cases} \qquad \text{matrix} \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

- Loop body of program "for n(for b(inc a); for a(inc b))":

$$\begin{cases} a'' = a + b \\ b'' = a + 2b \end{cases} \qquad \text{matrix} \begin{bmatrix} 1 & 1 \\ 1 & 2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \times \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

A lower bound

Theorem 1.
Let $R(n, a, b) = \text{ inc } b;\ \text{for } n(\text{for } b(\text{inc } a);\ \text{for } a(\text{inc } b)).$
After the computation $R(n, 0, 0)$ *the final contents of a and b satisfy*

$$\begin{array}{ll} a'(n) > 2^n & \text{for } n \geq 4 \\ b'(n) > 2^n & \text{for } n \geq 3. \end{array}$$

▷ The theorem. . .

The main result of this report: for every positive integer k there are SRL programs that grow faster than lower bound $2 \overset{k}{\uparrow} n$

Theorem 2.
For every $k \geq 1$ there is a SRL program using $k+2$ registers such that, if all the registers are initialized with $n \geq 2$, then all the registers have a final contents of at least $2 \overset{k}{\uparrow} n$. □

(Thus the registers a and b are also initialized with n.)

Proof: a note. . .

Note. A condition like “n $\geq \ldots$” — usually not mentioned — may be a consequence of inequalities like

Proof: a note...

Note. A condition like “n $\geq$ ...” — usually not mentioned — may be a consequence of inequalities like

$$
\begin{array}{rcll}
2 \overset{k}{\square} n & = & 2^{2^{\cdot^{\cdot^{\cdot^{2^n}}}}} & \#(2\text{'s})= k \\
& = & 2 \uparrow 2 \ldots \uparrow 2 \uparrow n & \#(2\text{'s})= k \\
& \geq & 2 \uparrow 2 \ldots \uparrow 2 \uparrow 2 & \text{for } n \geq 2,\ \#(2\text{'s})= k+1 \\
& > & 2 \uparrow 2 \ldots \uparrow 2 \uparrow 2 & \text{for } n \geq 2,\ \#(2\text{'s})= k \\
& = & 2 \overset{2}{\uparrow} k. &
\end{array}
$$

where #(2’s) denotes the number of 2’s.

Proof

The proof is by induction on k

Proof

The proof is by induction on *k*

Statement of the theorem

$$\boxed{\begin{array}{c}\forall k \in \mathbb{N}^+,\ \exists(\text{SRL program } P : \mathbb{Z}^{k+2} \to \mathbb{Z}^{k+2}) : \\ \forall \mathrm{n} \in \mathbb{N}^+, n \geq 2 :\ P(\overline{\mathrm{n}})|_{\text{all}} \geq 2 \overset{k}{\uparrow} \mathrm{n}\end{array}}$$

Proof

The proof is by induction on k

Statement of the theorem

$$\boxed{\begin{array}{c}\forall k \in \mathbb{N}^+,\ \exists(\text{SRL program } P : \mathbb{Z}^{k+2} \to \mathbb{Z}^{k+2}) : \\ \forall \mathrm{n} \in \mathbb{N}^+, n \geq 2 :\ P(\overline{\mathrm{n}})|_{\mathrm{all}} \geq 2 \overset{k}{\uparrow} \mathrm{n}\end{array}}$$

$P(\overline{\mathrm{n}})|_{\mathrm{all}}$: the final contents of all the registers, when all the initial contents are n.

Proof

The proof is by induction on k

Statement of the theorem

$$\forall k \in \mathbb{N}^+,\ \exists(\text{SRL program } P : \mathbb{Z}^{k+2} \to \mathbb{Z}^{k+2}) :$$
$$\forall \mathrm{n} \in \mathbb{N}^+, n \geq 2 :\ P(\overline{\mathrm{n}})|_{\mathrm{all}} \geq 2 \overset{k}{\uparrow} \mathrm{n}$$

$P(\overline{\mathrm{n}})|_{\mathrm{all}}$: the final contents of all the registers, when all the initial contents are n.

Recall the "hyperpower" symbols... $\overset{k}{\square}$ $\overset{k}{\uparrow}$

Proof, case $k = 1$

Recall that
$T(n, a, b) = \;\; \mathrm{inc}\, b;\ \mathrm{inc}\, n;\ \mathrm{inc}\, n;\ \mathrm{inc}\, n;\ \mathrm{inc}\, n;\ Q(n, a, b); Q(a, b, n)$

Proof, case $k = 1$

index

Recall that
$T(n, a, b) = \;\; \text{inc}\, b;\ \text{inc}\, n;\ \text{inc}\, n;\ \text{inc}\, n;\ \text{inc}\, n;\ Q(n, a, b);\ Q(a, b, n)$

$\boxed{k = 1}$
Let $n = \mathrm{n}$, $a = 0$, $b = 0$ be the initial values.
We have seen that

see here and here

$$T(\mathrm{n}, \mathrm{n}, \mathrm{n}) \geq T(\mathrm{n}, 0, 0) \geq 2^{\mathrm{n}} = 2 \overset{1}{\square} \mathrm{n} \geq 2 \overset{1}{\uparrow} \mathrm{n}$$

for every output of T.

Proof, case $k = 1$, continuation

$\boxed{k = 1}$

Proof, case $k = 1$, continuation

$\boxed{k = 1}$

Summary of the proof

Proof, case $k = 1$, continuation

index

$\boxed{k = 1}$

Summary of the proof

- Let $Q(n, a, b)$=“for n(for b(inc a); for a(inc b))”

→

Proof, case $k = 1$, continuation

$\boxed{k = 1}$

Summary of the proof

- Let $Q(n, a, b)$="for n(for b(inc a); for a(inc b))" →
- $Q(\mathrm{n}, 0, 1)$: $a' = F_{2\mathrm{n}}(0, 1)$, $b' = F_{2\mathrm{n}+1}(0, 1)$, $\mathrm{n}' = \mathrm{n}$. →

Proof, case $k = 1$, continuation

index

$\boxed{k = 1}$

Summary of the proof

- Let $Q(n, a, b)$=“for n(for b(inc a); for a(inc b))” →
- $Q(\mathrm{n}, 0, 1)$: $a' = F_{2\mathrm{n}}(0, 1)$, $b' = F_{2\mathrm{n}+1}(0, 1)$, $\mathrm{n}' = \mathrm{n}$. →
- $Q(\mathrm{n}, 0, 1)|_{a,b} \geq \frac{1}{\sqrt{5}}(\phi^{2\mathrm{n}} - \hat{\phi}^{2\mathrm{n}}) \geq 2^{\mathrm{n}}$ →

Proof, case $k = 1$, continuation

$\boxed{k = 1}$

Summary of the proof

- Let $Q(n, a, b)$=“for n(for b(inc a); for a(inc b))” $\rightarrow$
- $Q(\mathrm{n}, 0, 1)$: $a' = F_{2\mathrm{n}}(0, 1)$, $b' = F_{2\mathrm{n}+1}(0, 1)$, $\mathrm{n}' = \mathrm{n}$. $\rightarrow$
- $Q(\mathrm{n}, 0, 1)|_{a,b} \geq \frac{1}{\sqrt{5}}(\phi^{2\mathrm{n}} - \hat{\phi}^{2\mathrm{n}}) \geq 2^{\mathrm{n}}$ $\rightarrow$
- $T(n, a, b)$=“inc b; inc n; inc n; inc n; inc n;
  $Q(n, a, b)$; $Q(a, b, n)$”,
  $T(\mathrm{n}, 0, 0)|_{\mathrm{all}} \geq 2^{\mathrm{n}}$ $\rightarrow$

Proof, case $k = 1$, continuation

index

$\boxed{k = 1}$

Summary of the proof

- Let $Q(n, a, b)$=“for n(for b(inc a); for a(inc b))” →
- $Q(\mathrm{n}, 0, 1)$: $a' = F_{2\mathrm{n}}(0, 1)$, $b' = F_{2\mathrm{n}+1}(0, 1)$, $\mathrm{n}' = \mathrm{n}$. →
- $Q(\mathrm{n}, 0, 1)|_{a,b} \geq \frac{1}{\sqrt{5}}(\phi^{2\mathrm{n}} - \hat{\phi}^{2\mathrm{n}}) \geq 2^{\mathrm{n}}$ →
- $T(n, a, b)$=“inc b; inc n; inc n; inc n; inc n;
  $Q(n, a, b)$; $Q(a, b, n)$”,
  $T(\mathrm{n}, 0, 0)|_{\mathrm{all}} \geq 2^{\mathrm{n}}$ →
- $T(\mathrm{n}, \mathrm{n}, \mathrm{n})|_{\mathrm{all}} \geq 2^{\mathrm{n}} = 2 \overset{1}{\uparrow} \mathrm{n}$

Proof, case $k = 1$, continuation

index

$\boxed{k = 1}$

Summary of the proof

- Let $Q(n, a, b)$="for n(for b(inc a); for a(inc b))" →
- $Q(\mathrm{n}, 0, 1)$: $a' = F_{2\mathrm{n}}(0, 1)$, $b' = F_{2\mathrm{n}+1}(0, 1)$, $\mathrm{n}' = \mathrm{n}$. →
- $Q(\mathrm{n}, 0, 1)|_{a,b} \geq \frac{1}{\sqrt{5}}(\phi^{2\mathrm{n}} - \hat{\phi}^{2\mathrm{n}}) \geq 2^{\mathrm{n}}$ →
- $T(n, a, b)$="inc b; inc n; inc n; inc n; inc n;
  $Q(n, a, b)$; $Q(a, b, n)$",
  $T(\mathrm{n}, 0, 0)|_{\mathrm{all}} \geq 2^{\mathrm{n}}$ →
- $T(\mathrm{n}, \mathrm{n}, \mathrm{n})|_{\mathrm{all}} \geq 2^{\mathrm{n}} = 2 \overset{1}{\uparrow} \mathrm{n}$

Thus the program P of the statement may be $T(n, a, b)$. QED

Proof, induction step

$\boxed{k \Rightarrow k+1}$

Assume that all the $k+2$ registers $\overline{\mathrm{x}}$ have the initial contents n.

IH, induction hypothesis: $P(\overline{\mathrm{x}})|_{\mathrm{all}} \geq 2 \overset{k}{\uparrow} \mathrm{n}$.

Proof, induction step

$\boxed{k \Rightarrow k+1}$

Assume that all the $k+2$ registers $\bar{\mathrm{x}}$ have the initial contents n.

IH, induction hypothesis: $P(\bar{\mathrm{x}})|_{\mathrm{all}} \geq 2 \overset{k}{\uparrow} \mathrm{n}$.

Consider the sequence $U(m, \overline{x}) =$“for $m(P(\overline{x}))$”, which, with the initial contents of the new register m also equal to n is (semantically)

$$U(\mathrm{n}, \bar{\mathrm{x}}) \quad = \quad \overbrace{\underbrace{P;\ \ldots\ P}_{\mathrm{n}-1};\ P}^{\mathrm{n}}(\bar{\mathrm{x}}).$$

Compare with

$$2 \overset{k+1}{\uparrow} n \quad = \quad \underbrace{2 \overset{k}{\uparrow} 2 \overset{k}{\uparrow} \cdots 2 \overset{k}{\uparrow} 2}_{n\ 2\text{'s}}$$

Usual convention:

U is executed from left to tight,

$2 \overset{k+1}{\uparrow} n$ is interpreted from right to left.

Proof, induction step, continuation

(Recall: the initial contents of every element of $\overline{x}$ is n.)

Leftmost $P(\overline{x})$ is executed first and (by the IH): $P(\overline{x}) \geq 2 \overset{k}{\uparrow} n$.

Proof, induction step, continuation

index

(Recall: the initial contents of every element of $\overline{x}$ is n.)

Leftmost $P(\overline{x})$ is executed first and (by the IH): $P(\overline{x}) \geq 2 \overset{k}{\uparrow} n$.

$P(\overline{x}) \geq 2 \overset{k}{\uparrow} n \geq 2 \overset{k}{\uparrow} 2$ (for $n \geq 2$) and

Proof, induction step, continuation

(Recall: the initial contents of every element of $\overline{\mathrm{x}}$ is n.)

Leftmost $P(\overline{\mathrm{x}})$ is executed first and (by the IH): $P(\overline{\mathrm{x}}) \geq 2 \overset{k}{\uparrow} \mathrm{n}$.

$P(\overline{\mathrm{x}}) \geq 2 \overset{k}{\uparrow} \mathrm{n} \geq 2 \overset{k}{\uparrow} 2$ (for $\mathrm{n} \geq 2$) and

"$2 \overset{k}{\uparrow} 2$" is also at the right of the expression $2 \overset{k+1}{\uparrow} \mathrm{n}$.

Proof, induction step, continuation

index

(Recall: the initial contents of every element of $\overline{x}$ is n.)

Leftmost $P(\overline{x})$ is executed first and (by the IH): $P(\overline{x}) \geq 2 \overset{k}{\uparrow} n$.

$P(\overline{x}) \geq 2 \overset{k}{\uparrow} n \geq 2 \overset{k}{\uparrow} 2$ (for $n \geq 2$) and

"$2 \overset{k}{\uparrow} 2$" is also at the right of the expression $2 \overset{k+1}{\uparrow} n$.

The leftmost sequence "$(P;\ P)(\overline{x})$" satisfies

$(P;\ P)(\overline{x}) \geq P(2 \overset{k}{\uparrow} 2)$ (IH)

(The second P receives all inputs $\geq 2 \overset{k}{\uparrow} 2$)

Proof, induction step, continuation

(Recall: the initial contents of every element of $\overline{x}$ is n.)

Leftmost $P(\overline{x})$ is executed first and (by the IH): $P(\overline{x}) \geq 2 \overset{k}{\uparrow} n$.

$P(\overline{x}) \geq 2 \overset{k}{\uparrow} n \geq 2 \overset{k}{\uparrow} 2$ (for $n \geq 2$) and
"$2 \overset{k}{\uparrow} 2$" is also at the right of the expression $2 \overset{k+1}{\uparrow} n$.

The leftmost sequence "$(P;\ P)(\overline{x})$" satisfies

$(P;\ P)(\overline{x}) \geq P(2 \overset{k}{\uparrow} 2)$ (IH)

(The second P receives all inputs $\geq 2 \overset{k}{\uparrow} 2$)

…Thus $(P;\ P)(\overline{x}) \geq 2 \overset{k}{\uparrow} (2 \overset{k}{\uparrow} 2)$ (IH+monotonicity)

Proof, induction step, continuation

(Recall: the initial contents of every element of $\overline{x}$ is n.)

Leftmost $P(\overline{x})$ is executed first and (by the IH): $P(\overline{x}) \geq 2 \overset{k}{\uparrow} n$.

$P(\overline{x}) \geq 2 \overset{k}{\uparrow} n \geq 2 \overset{k}{\uparrow} 2$ (for $n \geq 2$) and

"$2 \overset{k}{\uparrow} 2$" is also at the right of the expression $2 \overset{k+1}{\uparrow} n$.

The leftmost sequence "$(P;\ P)(\overline{x})$" satisfies

$$(P;\ P)(\overline{x}) \geq P(2 \overset{k}{\uparrow} 2) \qquad \text{(IH)}$$

(The second P receives all inputs $\geq 2 \overset{k}{\uparrow} 2$)

$$\ldots \text{Thus } (P;\ P)(\overline{x}) \geq 2 \overset{k}{\uparrow} (2 \overset{k}{\uparrow} 2) \qquad \text{(IH+monotonicity)}$$

…and this is exactly the rightmost sequence with three 2's of

$$2 \overset{k+1}{\uparrow} n = \underbrace{2 \overset{k}{\uparrow} 2 \overset{k}{\uparrow} \cdots \overbrace{2 \overset{k}{\uparrow} 2 \overset{k}{\uparrow} 2}^{(P;P)(\overline{x})\geq}}_{n\ 2\text{'s}}$$

Proof, induction step, continuation

. . . and so on. . .

More formally, use induction on k. We get

$$U(m, \overline{\mathrm{x}}) \geq 2 \overset{k+1}{\uparrow} \mathrm{n}, \qquad (\star)$$

assuming that the $k + 3$ parameters $(m, \overline{\mathrm{x}})$ are initialized with n

Proof, induction step, continuation

…and so on…

More formally, use induction on k. We get

$$U(m, \overline{\mathrm{x}}) \geq 2 \overset{k+1}{\uparrow} \mathrm{n}, \qquad (\star)$$

assuming that the $k+3$ parameters $(m, \overline{\mathrm{x}})$ are initialized with n

This inequality holds for all registers $\overline{\mathrm{x}}$… but the final contents of m is still n.

Proof, induction step, continuation

…and so on…
More formally, use induction on k. We get

$$U(m,\overline{\mathrm{x}}) \geq 2 \overset{k+1}{\uparrow} \mathrm{n}, \qquad (\star)$$

assuming that the $k+3$ parameters $(m,\overline{\mathrm{x}})$ are initialized with n

This inequality holds for all registers $\overline{\mathrm{x}}$… but the final contents of m is still n. Solution: the program

$U(m,x_1,\ldots,x_{k+2});\ U(x_1,m,x_1,\ldots,x_{k+2})$

satisfies $(\star)$ for the outputs of <u>all</u> the registers.

Proof, induction step, continuation

...and so on...

More formally, use induction on k. We get

$$U(m, \overline{\mathrm{x}}) \geq 2 \overset{k+1}{\uparrow} \mathrm{n}, \qquad (\star)$$

assuming that the $k+3$ parameters $(m, \overline{\mathrm{x}})$ are initialized with n

This inequality holds for all registers $\overline{\mathrm{x}}$... but the final contents of m is still n. Solution: the program

$U(m, x_1, \ldots, x_{k+2});\ U(x_1, m, x_1, \ldots, x_{k+2})$

satisfies $(\star)$ for the outputs of all the registers.

This finishes the proof by induction on k. □

index

▷ Loop and SRL: the same lower bounds

In Loop and SRL programs, the (absolute value of the) maximum contents of the registers is essentially the same.

▷ Loop and SRL: the same lower bounds

In Loop and SRL programs, the (absolute value of the) maximum contents of the registers is essentially the same.

Given that

▷ Loop and SRL: the same lower bounds

In Loop and SRL programs, the (absolute value of the) maximum contents of the registers is essentially the same.

Given that

- in SRL and in Loop: for every $k \in \mathbb{N}^+$ there are functions/transformations with lower bound $2 \overset{k}{\uparrow} n$;

▷ Loop and SRL: the same lower bounds

In Loop and SRL programs, the (absolute value of the) maximum contents of the registers is essentially the same.

Given that

- in SRL and in Loop: for every $k \in \mathbb{N}^+$ there are functions/transformations with lower bound $2 \overset{k}{\uparrow} n$;
- the Ackermann function $a(m, n) = [2 \overset{m-2}{\uparrow} (n + 3) - 3]$ grows faster than any primitive recursive function (and thus is not primitive recursive);

▷ Loop and SRL: the same lower bounds

index

In Loop and SRL programs, the (absolute value of the) maximum contents of the registers is essentially the same.

Given that

- in SRL and in Loop: for every $k \in \mathbb{N}^+$ there are functions/transformations with lower bound $2 \overset{k}{\uparrow} n$;
- the Ackermann function $a(m,n) = [2 \overset{m-2}{\uparrow} (n+3) - 3]$ grows faster than any primitive recursive function (and thus is not primitive recursive);
- SRL can be simulated in Loop without much difficulty

▷ Loop and SRL: the same lower bounds

index

In Loop and SRL programs, the (absolute value of the) maximum contents of the registers is essentially the same.

Given that

- in SRL and in Loop: for every $k \in \mathbb{N}^+$ there are functions/transformations with lower bound $2 \overset{k}{\uparrow} n$;
- the Ackermann function $a(m,n) = [2 \overset{m-2}{\uparrow} (n+3) - 3]$ grows faster than any primitive recursive function (and thus is not primitive recursive);
- SRL can be simulated in Loop without much difficulty

the lowest upper bounds of PR functions and of (the final register contents of) SRL programs are essentially the same.

The theorem. . .

The theorem. . .

Theorem 3.

Primitive recursive (PR) functions and the SRL transformations have essentially the same maximum growth rate in the sense that

- For every $k \geq 1$ there is a PR function $f(n)$ that grows faster than $2 \overset{k}{\uparrow} n$.

The theorem. . .

Theorem 3.
Primitive recursive (PR) functions and the SRL transformations have essentially the same maximum growth rate in the sense that

- For every $k \geq 1$ there is a PR function $f(n)$ that grows faster than $2 \overset{k}{\uparrow} n$.
- For every $k \geq 1$ there is a SRL program using $k + 2$ registers such that, if all the registers are initialized with $\mathrm{n} \geq 2$, then all their final contents are at least $2 \overset{k}{\uparrow} n$.

The theorem...

Theorem 3.
Primitive recursive (PR) functions and the SRL transformations have essentially the same maximum growth rate in the sense that

- For every $k \geq 1$ there is a PR function $f(n)$ that grows faster than $2 \overset{k}{\uparrow} n$.
- For every $k \geq 1$ there is a SRL program using $k + 2$ registers such that, if all the registers are initialized with $\mathsf{n} \geq 2$, then all their final contents are at least $2 \overset{k}{\uparrow} n$.
- No PR function $f(n)$ grows faster than the diagonal Ackermann function $2 \overset{n}{\uparrow} n$.

The theorem. . .

Theorem 3.
Primitive recursive (PR) functions and the SRL transformations have essentially the same maximum growth rate in the sense that

- For every $k \geq 1$ there is a PR function $f(n)$ that grows faster than $2 \overset{k}{\uparrow} n$.
- For every $k \geq 1$ there is a SRL program using $k + 2$ registers such that, if all the registers are initialized with $\mathsf{n} \geq 2$, then all their final contents are at least $2 \overset{k}{\uparrow} n$.
- No PR function $f(n)$ grows faster than the diagonal Ackermann function $2 \overset{n}{\uparrow} n$.
- No SRL transformation grows faster than $2 \overset{n}{\uparrow} n$. $\square$

The theorem. . .

Theorem 3.
Primitive recursive (PR) functions and the SRL transformations have essentially the same maximum growth rate in the sense that

- For every $k \geq 1$ there is a PR function $f(n)$ that grows faster than $2 \overset{k}{\uparrow} n$.
- For every $k \geq 1$ there is a SRL program using $k + 2$ registers such that, if all the registers are initialized with $\mathsf{n} \geq 2$, then all their final contents are at least $2 \overset{k}{\uparrow} n$.
- No PR function $f(n)$ grows faster than the diagonal Ackermann function $2 \overset{n}{\uparrow} n$.
- No SRL transformation grows faster than $2 \overset{n}{\uparrow} n$. $\square$

The end

index

▷ Bibliography

Donald Knuth.
Mathematics and computer science: coping with finiteness.
Science, 194(17), 1976.

Armando B. Matos.
Analysis of a simple reversible language.
Theoretical Computer Science, 290(3):2063–2074, 2003.

Armando B. Matos and António Porto.
Ackermann and the superpowers.
ACM SIGACT, 12 (Fall 1980), 1980.

Armando B. Matos and António Porto.
Ackermann and the superpowers.
Technical report, Faculdade de Ciências da Universidade do Porto, 1995.
(also in ACM SIGACT News, Volume 12 Issue 3).

▷ Bibliography

Armando B. Matos, Luca Roversi, and Luca Paolini.
The fixed point problem for general and for linear SRL programs is undecidable, 2018.
Submitted.

Kalyan Perumalla.
Introduction to Reversible Computing.
CRC Press, 2014.

Luca Paolini, Mauro Piccolo, and Luca Roversi.
A class of reversible primitive recursive functions.
Electronic Notes in Theoretical Computer Science, 322(18605):227–242, 2016.

index

▷ Optional section

This is optional material: Another proof of the existence of

SRL programs with lower bound $\geq 2 \overset{k}{\square} n$".

▷ Optional section

This is optional material: Another proof of the existence of SRL programs with lower bound $\geq 2 \overset{k}{\Box} n$”.

The program $\text{Hyper}_k(n, a, b)$:

Line	Instruction
	inc b;
1	$Q(n, a, b)$;
2	$Q(a, b, n)$;
3	$Q(n, a, b)$;
4	$Q(a, b, n)$;
$\dots$	$\dots$
k	$\begin{cases} k \text{ even: } Q(a, b, n) \\ k \text{ odd: } Q(n, a, b) \end{cases}$

A sequence of two SRL “Fibonacci programs”

index

The sequence:

$$\underbrace{Q(n, a, b)}_{(1)};\ \underbrace{Q(a, b, n)}_{(2)}$$

A sequence of two SRL “Fibonacci programs”

index

The sequence:

$$\underbrace{Q(n, a, b)}_{(1)};\ \underbrace{Q(a, b, n)}_{(2)}$$

Lower bounds of (1) and (2) for initial value $n \geq 4$:

$$(1)\begin{cases} n' & = & n \\ a' & > & 2^n \\ b' & > & 2^n \end{cases} \qquad (2)\begin{cases} a'' & = & a' \\ b'' & > & 2^{a'} \\ n'' & > & 2^{a'} \end{cases}$$

A sequence of two SRL "Fibonacci programs"

The sequence:

$$\underbrace{Q(n,a,b)}_{(1)};\ \underbrace{Q(a,b,n)}_{(2)}$$

Lower bounds of (1) and (2) for initial value $n \geq 4$:

$$(1)\begin{cases} n' & = & n \\ a' & > & 2^n \\ b' & > & 2^n \end{cases} \qquad (2)\begin{cases} a'' & = & a' \\ b'' & > & 2^{a'} \\ n'' & > & 2^{a'} \end{cases}$$

For b'' we get

$$b'' > 2^{a'} \Rightarrow b'' > 2^{(2^n)}$$

A sequence of two SRL "Fibonacci programs"

The sequence:

$$\underbrace{Q(n,a,b)}_{(1)};\ \underbrace{Q(a,b,n)}_{(2)}$$

Lower bounds of (1) and (2) for initial value $n \geq 4$:

$$(1)\left\{\begin{array}{lcl} n' & = & n \\ a' & > & 2^n \\ b' & > & 2^n \end{array}\right. \qquad (2)\left\{\begin{array}{lcl} a'' & = & a' \\ b'' & > & 2^{a'} \\ n'' & > & 2^{a'} \end{array}\right.$$

For b'' we get

$$b'' > 2^{a'} \Rightarrow b'' > 2^{(2^n)}$$

Similarly: $a'' > 2^n$, $n'' > 2^{(2^n)}$.

Recalling the notation...

$$
\begin{aligned}
2 \overset{k}{\square} n &= 2^{2^{\cdot^{\cdot^{\cdot^{2^n}}}}} && \text{where the number of 2's is } k \\
&= 2 \uparrow 2 \ldots \uparrow 2 \uparrow n \\
&\geq 2 \overset{2}{\uparrow} k && \text{for } n \geq 2.
\end{aligned}
$$

Recalling the notation. . .

index

$$\begin{aligned} 2 \overset{k}{\square} n &= 2^{2^{\cdot^{\cdot^{\cdot^{2^n}}}}} && \text{where the number of 2's is } k \\ &= 2 \uparrow 2 \ldots \uparrow 2 \uparrow n \\ &\geq 2 \overset{2}{\uparrow} k && \text{for } n \geq 2. \end{aligned}$$

More generally,

$$\begin{aligned} a \overset{1}{\uparrow} n &= a \uparrow n = a^n \\ a \overset{m}{\uparrow} n &= \underbrace{a \overset{m-1}{\uparrow} a \overset{m-1}{\uparrow} \cdots \overset{m-1}{\uparrow} a}_{n\ a\text{'s}} && \text{for } m \geq 2. \end{aligned}$$

SRL The theorem. . .

SRL The theorem…

Theorem 4.
For every positive integer k, there is a SRL *program* $\mathrm{Pr}(n, a, b)$ *such that, in the computation* $\mathrm{Pr}(n, 0, 0)$ *and for every $n \geq 0$, the final contents of the registers satisfy*

$$n'(n),\ a'(n),\ b'(n) > 2 \overset{k}{\uparrow} n$$

The proof, I: a sequence

The proof, I: a sequence

Lower bounds.

	Sequence of instructions	mem[n]	mem[a]	mem[b]
	inc b;	4	0	1
1	for n(for b(inc a); for a(inc b));			
	for b(inc a); for b(inc n)	$2 \uparrow^{1} n$	$2 \uparrow^{1} n$	$2 \uparrow^{1} n$
2	for a(for n(inc b); for b(inc n));			
	for b(inc a); for b(inc n)	$2 \uparrow^{2} n$	$2 \uparrow^{2} n$	$2 \uparrow^{2} n$
3	for n(for b(inc a); for a(inc b));			
	for b(inc a); for b(inc n)	$2 \uparrow^{3} n$	$2 \uparrow^{3} n$	$2 \uparrow^{3} n$
	...			

The proof, II: generalization

Proof: generalize the previous program!

Bottom lines for k odd:

		mem[n]	mem[a]	mem[b]
	...	...	...	...
k	for n(for b(inc a); for a(inc b));			
	for b(inc a); for b(inc n)	$2 \overset{k}{\uparrow} n$	$2 \overset{k}{\uparrow} n$	$2 \overset{k}{\uparrow} n$

Bottom lines for k even:

		mem[n]	mem[a]	mem[b]
	...	...	...	...
k	for a(for n(inc b); for b(inc n));			
	for b(inc a); for b(inc n)	$2 \overset{k}{\uparrow} n$	$2 \overset{k}{\uparrow} n$	$2 \overset{k}{\uparrow} n$

The end