COMPUTABILITY (07/10/2025)

Models of computation

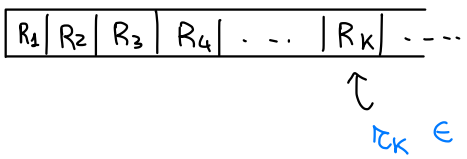
- Turing machines
- Lambda calculus (Church)
- Partial recursive functions (Gödel - Kleene)
- Canonical deduction systems (Post)
- URM (Unlimited Register machine)
- ---
- ---

Church Turing thesis

A function is computable by an effective procedure **iff** it is computable by a Turing machine

UNLIMITED REGISTER MACHINE

- memory (unbounded)



- executes a program : finite list of instruction
- instruction set

Arithmetic

- zero $Z(m)$ $r_m \leftarrow 0$
- successor $S(m)$ $r_m \leftarrow r_m + 1$
- transfer $T(m,n)$ $r_n \leftarrow r_m$

Jump

- $J(m,n,t)$ compare $r_m = r_n$? yes → jump to instr. t / no → continue with next

Program

I_1
I_2
⋮
I_s

its execution starts from

- initial configuration of registers
- instruction I_1

and it <u>terminates</u> (possibly) if the next instruction to be executed does not exist

→ last instruction and go to next

→ jump out of the program

<u>Example</u>

		R_1	R_2	R_3	
I_1	$J(2,3,5)$	1	2	0	0 ...
→ I_2	$S(1)$	2	2	0	...
I_3	$S(3)$	2	2	1	
I_4	$J(1,1,1)$	3	2	2	

* a computation can diverge (not terminating)

I_1 $J(1,1,1)$

* <u>Notation</u> Given $a_1, a_2, a_3, \ldots \in \mathbb{N}$ and a program P

$P(a_1 a_2 a_3 \ldots)$ indicates the computation of P from $a_1 a_2 a_3 \ldots$

$P(a_1 a_2 \ldots) \downarrow$ eventually terminates

$P(a_1 a_2 \ldots) \uparrow$ diverges

Given $a_1, a_2 \dots a_k \in \mathbb{N}$

$P(a_1 a_2 \dots a_k)$ to indicate $P(a_1 \dots a_k\, 0\, 0 \dots)$

$P(a_1 \dots a_k) \downarrow a$ to indicate $P(a_1 \dots a_k)$

and in the final configuration $r_1 = a$

URM-computable function

Given $f: \mathbb{N}^k \to \mathbb{N}$ (partial) is URM-computable if there is a URM-program P such that $\forall (a_1 \dots a_k) \in \mathbb{N}^k \quad \forall a \in \mathbb{N}$

$P(a_1 \dots a_k) \downarrow a$ iff $(a_1 \dots a_k) \in dom(f)$ and

$f(a_1, \dots, a_k) = a$

Example

$J(1,1,1)$ computes $f: \mathbb{N} \to \mathbb{N}$ $\quad f(a) \uparrow \quad \forall a \in \mathbb{N}$

Class of URM-computable functions

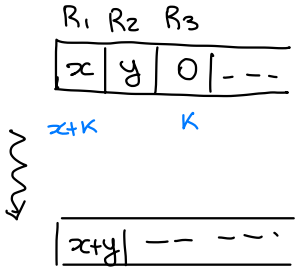
$$\mathcal{C}^{(k)} = \{ f \mid f: \mathbb{N}^k \to \mathbb{N} \text{ URM-computable} \}$$

$$\mathcal{C} = \bigcup_{k \geq 1} \mathcal{C}^{(k)}$$

Example:

* $f: \mathbb{N}^2 \to \mathbb{N}$

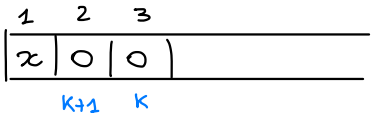
$f(x,y) = x + y$




```
LOOP: J(2,3,STOP)      // K = y?
      S(1)
      S(3)
      J(1,1,LOOP)
STOP :
```

* $g : \mathbb{N} \to \mathbb{N}$

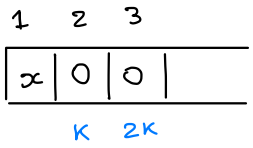
$$g(x) = x \dot{-} 1 = \begin{cases} 0 & \text{if } x = 0 \\ x - 1 & \text{if } x > 0 \end{cases}$$




```
    J(1,2,END)      // x = 0 ?
    S(2)
LOOP: J(1,2,RES)    // x = K+1 ?
    S(2)
    S(3)
    J(1,1,LOOP)
RES: T(3,1)
END:
```

* $h : \mathbb{N} \to \mathbb{N}$

$$h(x) = \begin{cases} x/2 & \text{if } x \text{ is even} \\ \uparrow & \text{otherwise} \end{cases}$$




```
LOOP: J(1,3,RES)    // x = 2K ?
    S(2)
    S(3)
    S(3)
    J(1,1,LOOP)
RES: T(2,1)
```

* Function computed by a program ?

Let P be a URM-program and $K \geq 1$

$$f_P^{(K)} : \mathbb{N}^K \to \mathbb{N}$$

$$f_P^{(K)}(a_1, \dots, a_K) = \begin{cases} a & \text{if } P(a_1, \dots, a_K) \downarrow a \\ \uparrow & \text{if } P(a_1, \dots, a_K) \uparrow \end{cases}$$

Question : given function $f : \mathbb{N}^K \to \mathbb{N}$ how many programs for f ?

0 or infinitely many

EXERCISE: Consider URM⁻ machine without $T(m,n)$

$\mathcal{C}^-$ = class of URM⁻ computable functions

$$\mathcal{C} \overset{?}{=} \mathcal{C}^-$$

$\supseteq$

$\subseteq$ ← $T(m,n)$ can be encoded

$Z(m)$
LOOP: $J(m,m,\text{END})$
$S(m)$
$J(1,1,\text{LOOP})$

proof

$(\mathcal{C}^- \subseteq \mathcal{C})$ Let $f: \mathbb{N}^k \to \mathbb{N}$ URM⁻ computable $f \in \mathcal{C}^-$

i.e. there is P URM⁻ program such that

$$f_P^{(k)} = f$$

but P is also a URM-program, hence $f \in \mathcal{C}$.

$(\mathcal{C} \subseteq \mathcal{C}^-)$ Let $f: \mathbb{N}^k \to \mathbb{N}$ $f \in \mathcal{C}$ i.e. there is P URM-program such that

$$f = f_P^{(k)}$$

I assume, without loss of generality, that P is well-formed, i.e. if it terminates it does by landing at instruction $S+1$

We show that there is P' URM⁻-program such that

$$f_{P'}^{(k)} = f_P^{(k)} = f$$

by induction on h = number of T instructions in P

$(h=0)$ P has no T instructions, thus P' can be P

$(h \to h+1)$ Let P be URM program with

$h+1$ transfer instructions

P $\begin{cases} I_1 \\ I_2 \\ \vdots \\ I_t \quad T(m,n) \\ \\ I_s \end{cases}$ $\rightsquigarrow$ P'' $\begin{cases} I_1 \\ I_2 \\ \\ I_t \quad J(1,1,SUB) \\ \\ I_s \\ I_{s+1}: J(1,1,END) \\ SUB: Z(m) \\ LOOP: J(m,m,t+1) \\ \quad\quad S(m) \\ \quad\quad J(1,1,LOOP) \\ \\ END: \end{cases}$

now P'' is a URM program such that

$\rightarrow$ $\boxed{f_{P''}^{(k)} = f_P^{(k)}}$ and it has h transfer instructions

by inductive hyp. there is P' URM-program such

$\boxed{f_{P'}^{(k)} = f_{P''}^{(k)}}$

Summming up, we have P' URM-program such that

$$f_{P'}^{(k)} = f_{P''}^{(k)} = f_P^{(k)}$$

□

OBSERVATION: Given a URM-program P there is a program P' well-formed and such that $f_P^{(k)} = f_{P'}^{(k)} \quad \forall k \in \mathbb{N}$

P $\begin{cases} I_1 \\ \vdots \\ J(m,n,t) \\ \\ I_s \end{cases}$ $t > s+1$ replace it by $\rightsquigarrow$ $J(m,n,s+1)$

Exercise : Variant of URM - machine

URM^s $\quad T(m, n)$ $\quad$ swop $T^s(m_1, n)$

$r_m \leftrightarrow r_n$

$\mathcal{C}^s = \mathcal{C}$

EXERCISE : URM$^=$ without jump

$\mathcal{C}^= = \mathcal{C}$

$\subsetneq$

↑

characterise functions in $\mathcal{C}^=$