Computability (04/10/2021)

* Model of computation ?

- → Turing machine
- → λ-calculus (Church)
- → partial recursive functions (Gödel & Kleene)
- → deduction systems (Post)
- → URM (Unlimited register machine)

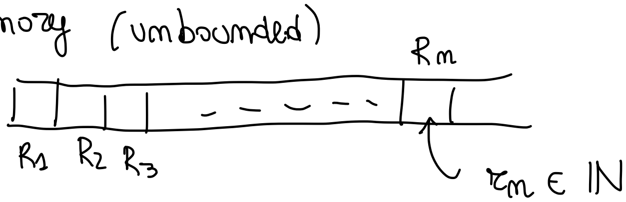
⋮

Church Turing thesis

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

* Unlimited Register Machine (URM)

→ memory (unbounded)



→ executes a program : finite list of instructions

I_1
I_2
⋮
I_s

→ Arithmetic instructions

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

→ Jump

$J(m, n, t)$ $\qquad r_m = r_n$? ↗ yes: jump to I_t ; ↘ no: move to next instruction

→ computation

→ starts from I_1 , + initial config. of registers

→ terminates if the instruction to be executed does not exist

→ last instruction

→ jump to non existing instruction

Example

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

$I_1 \quad J(1,1,1)$

Notation : Given $a_1, a_2, a_3 \dots \in \mathbb{N}$ and program P

$P(a_1, a_2, \dots)$ computation of P starting $a_1 a_2 \dots$

$P(a_1 a_2 a_3 \dots)\downarrow$ eventually terminates

$P(a_1 a_2 a_3 \dots)\uparrow$ diverges

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

$P(a_1 \dots a_k)$ for $P(a_1 \dots a_k\ 0\ 0 \dots)$

$P(a_1 \dots a_k)\downarrow a$ for $P(a_1 \dots a_k)\downarrow$ and at the end $r_1 = a$

* URM-computable function

Given a function $f : \mathbb{N}^K \to \mathbb{N}$ (possibly partial) is URM-computable if there exists a program P such that $\forall (a_1, \dots, a_K) \in \mathbb{N}^K \ \forall a \in \mathbb{N}$

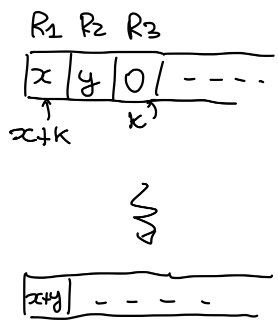
$$P(a_1 \dots a_K) \downarrow a \quad \text{iff} \quad (a_1, \dots, a_K) \in dom(f)$$
$$f(a_1 \dots a_K) = a$$

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

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

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

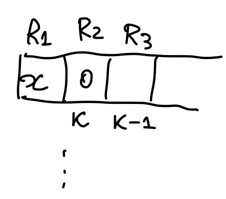
$f(x, y) = x + y$




```
LOOP:  J(2,3, STOP)
       S(1)
       S(3)
       J(1,1,LOOP)
STOP :
```

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

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




```
       J(1,2, END)
       S(2)
LOOP:  J(1,2, RES)
       S(2)
       S(3)
       J(1,1,LOOP)
RES :  T(3,1)
END :
```

$K = 1$

$K = x$?

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

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

R_1	R_2	R_3
x	0	0
	$2K$	K

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

* function computed by a program

Given P program and $k \geq 1$

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

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

Given a function $f: \mathbb{N}^k \to \mathbb{N}$ how many programs compute f?

$\nearrow \infty$

$\searrow 0$

<u>Exercise</u>: Consider URM$^-$ machine without $T(m,n)$ instruction

$\downarrow$

$\mathcal{C}^-$ functions computable in URM$^-$

$$\mathcal{C}^- \subseteq \mathcal{C}$$

$\supseteq$

$T(m,n)$

```
      Z(n)
LOOP: J(m,n,END)
      S(n)
      J(1,1,LOOP)
```

proof

($\mathcal{C}^- \subseteq \mathcal{C}$) Let $f: \mathbb{N}^K \to \mathbb{N}$ $f \in \mathcal{C}^-$ hence there is a URM$^-$ program P such that $f = f_P^{(K)}$ and observe that P is also a URM program $\Rightarrow$ $f = f_P^{(K)} \in \mathcal{C}$.

($\mathcal{C} \subseteq \mathcal{C}^-$) Let $f: \mathbb{N}^K \to \mathbb{N}$ $f \in \mathcal{C}$ hence there is P URM-program such that $f_P^{(K)} = f$

$$P \quad \begin{array}{ll} I_1 & \\ I_2 & \\ \vdots & \\ I_t & T(m,n) \\ \vdots & \\ I_s & \end{array}$$

We prove that P can be transformed in a URM$^-$ program P' such that $f_P^{(K)} = f_{P'}^{(K)}$ by induction on $h = \#\left(\begin{array}{c} T(m,n) \text{ instr.} \\ \text{inp} \end{array}\right)$

($h = 0$) P is a URM$^-$ program, hence we can take $P = P'$

($h \to h+1$) P is of the kind (assume P is well-formed)

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

OBSERVATION Given a program P there always exists P' such that $\forall k$ $f_P^{(k)} = f_{P'}^{(k)}$ and P' is "well-formed" i.e. it will always ends at its last instruction $+1$

if P $\begin{cases} I_1 \\ \vdots \\ I_s \end{cases}$ then take any jump $J(m, n, t)$ $t > s$

$\Downarrow$

$J(m, n, s+1)$

Now $\boxed{f_P^{(k)} = f_{P''}^{(k)}}$ and P'' has h $T(m,n)$ instructions thus by inductive hyp. there exists $\boxed{P' \text{ URM-program}}$ such that $\boxed{f_{P'}^{(k)} = f_{P''}^{(k)}}$

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

* Exercise :

Variant URM^s of URM-machine

$\cancel{T(m,n)}$ $\quad T_s(m,n)$ $\quad r_n \leftrightarrow r_m$

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

* Exercise : Consider $URM^=$ URM without jump instruction

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