COMPUTABILITY (10/11/2025)

* Parametrisation Theorem

program $e \in \mathbb{N}$ $\quad P_e = \gamma^{-1}(e)$ and $\varphi_e^{(2)}(x,y)$

for any fixed $x \in \mathbb{N}$ one obtains a function of one argument y

$x = 0 \quad y \mapsto \varphi_e^{(2)}(0,y)$

$x = 1 \quad y \mapsto \varphi_e^{(2)}(1,y)$

$\vdots \qquad \vdots$

smn - theorem : for each fixed $x \in \mathbb{N}$, the program computing the function of y defined as above $y \mapsto \varphi_e^{(2)}(x,y)$ can be constructed algorithmically using P_e and x

$P_e(x,y)$:

x

y

return ...

$x = x_0$

$P_e(\cancel{x}, y)$:

x_0

x

y

return ...

more generally

$$\varphi_e^{(m+n)}(\vec{x},\vec{y}) = \varphi_{\underbrace{s_{m,n}(e,\vec{x})}_{\text{computable}}}(\vec{y})$$

Theorem (smn Theorem)

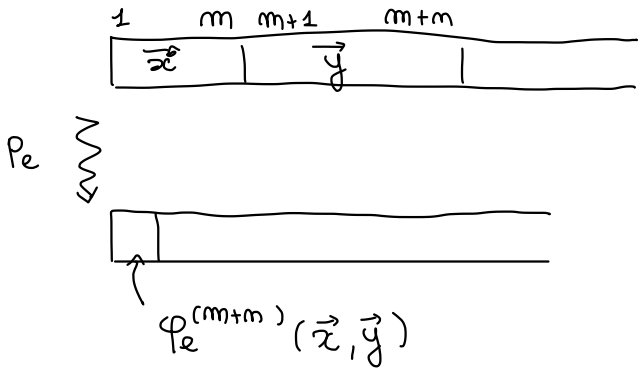
Given $m, n \geq 1$ there is a total computable function $s_{m,n} : \mathbb{N}^{m+1} \to \mathbb{N}$ such that for all $\vec{x} \in \mathbb{N}^m$, $\vec{y} \in \mathbb{N}^n$, $e \in \mathbb{N}$

$$\varphi_e^{(m+n)}(\vec{x},\vec{y}) = \varphi^{(n)}_{s_{m,n}(e,\vec{x})}(\vec{y})$$

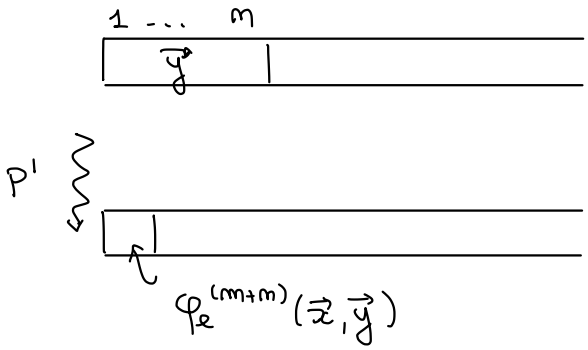
proof

intuitively $e \in \mathbb{N}$ $\quad P_e = \gamma^{-1}(e)$

starting from



one wants, given each $\vec{x} \in \mathbb{N}$, a program P' ↖ depends on e, $\vec{x}$



the program P' should

→ move $\vec{y}$ to $m+1 \dots m+m$
→ write $\vec{x}$ on $1 \dots m$
→ execute P_e

P'

$T(m, m+m)$ // move y_m to R_{m+m}
⋮
$T(1, m+1)$ // " y_1 to R_m

$Z(1)$
$S(1)$
⋮ } x_1 times
$S(1)$ // write x_1 in R_1

⋮

$Z(m)$
$S(m)$
⋮ } x_m times
$S(m)$ // write x_m to R_m

$P_e = \gamma^{-1}(e)$

$$S_{m,m}(e, \vec{x}) = \gamma(P')$$

① sequential composition of programs $\quad (e_1, e_2 \rightsquigarrow \gamma\begin{pmatrix} P_{e_1} \\ P_{e_2} \end{pmatrix}$

(1.a) $\quad upd : \mathbb{N}^2 \to \mathbb{N}$

$$upd(e,h) = \gamma\left(\begin{array}{l}\text{program obtained from } P_e = \gamma^{-1}(e) \text{ updating each} \\ \text{jump instruction } J(m_1,m_1,t) \rightsquigarrow J(m_1,m_1,t+h)\end{array}\right)$$

$$\widetilde{upd}(i,h) = \beta\left(\begin{array}{l}\text{instruction obtained from } \beta^{-1}(i), \text{ updating the} \\ \text{target if it is a jump}\end{array}\right)$$

NOTE: $\quad \beta(J(m_1,m_1,t)) = \nu(m-1, m-1, t-1) * 4 + 3$

$$= \begin{cases} i & \text{if } rm(4,i) \neq 3 \\ \nu(\nu_1(q), \nu_2(q), \nu_3(q)+h) * 4 + 3 & \text{if } rm(4,i) = 3 \\ & q = qt(4,i) \end{cases}$$

$$= i * sg(|rm(4,i) - 3|) + (\nu(\nu_1(q), \nu_2(q), \nu_3(q)+h) * 4 + 3) * \overline{sg}(|rm(4,i) - 3|)$$

now

$$upd(e,h) = \tau\left(\widetilde{upd}(a(e,1),h), \widetilde{upd}(a(e,2),h), \dots \widetilde{upd}(a(e,\ell(e)),h)\right)$$

$$= \left(\prod_{i=1}^{\ell(e)-1} p_i^{\widetilde{upd}(a(e,i),h)}\right) p_{\ell(e)}^{\widetilde{upd}(a(e,\ell(e)),\ell(e))+1} \dot{-} 2$$

$$\tau(y_1 \dots y_m) = \left(\prod_{i=1}^{m-1} p_i^{y_i}\right) * p_m^{y_{m+1}} \dot{-} 2$$

$\ell(e)$ = length of sequence encoded by e

$a(e,i)$ = component i of the sequence $\quad 1 \le i \le \ell(e)$

- $c : \mathbb{N}^2 \to \mathbb{N}$

$$c(e_1,e_2) = \text{code of concatenation of } \tau^{-1}(e_1) \text{ and } \tau^{-1}(e_2)$$

$$= \tau(a(e_1,1) \dots a(e_1,\ell(e_1)) \; a(e_2,1) \dots a(e_2,\ell(e_2)))$$

$$= \dots$$

- $seq : \mathbb{N}^2 \to \mathbb{N}$

$$seq(e_1,e_2) = \gamma\begin{pmatrix} P_{e_1} \\ P_{e_2} \end{pmatrix} = c(e_1, upd(e_2, \ell(e_1)))$$

② transf : $\mathbb{N}^2 \to \mathbb{N}$

$$transf(m,m) = \gamma \begin{pmatrix} T(m, m+m) \\ \vdots \\ T(1, m+1) \end{pmatrix}$$

$$= \dots$$

$$\begin{cases} T(m, m+m) \\ \vdots \\ T(1, m+1) \end{cases} P'$$

$$\begin{cases} z(1) \\ s(1) \\ \vdots \\ s(1) \end{cases} x_1 \text{ times}$$

$$\vdots$$

$$\begin{cases} z(m) \\ s(m) \\ \vdots \\ s(m) \end{cases} x_m \text{ times}$$

$$P_e = \gamma^{-1}(e)$$

③ set : $\mathbb{N}^2 \to \mathbb{N}$

$$set(i, x) = \gamma \begin{pmatrix} z(i) \\ s(i) \\ \vdots \\ s(i) \end{pmatrix} = \dots$$

④ finally

$$S_{m,m}(e, \vec{x}) =$$

$$seq(transf(m,m),$$

$$seq(set(1, x_1),$$

$$seq(set(2, x_2), \dots$$

$$\ddots$$

$$seq(set(m, x_m), e) \dots)$$

computable (primitive recursive) : composition of primitive recursive functions. □

Corollary Let $f : \mathbb{N}^{m+m} \to \mathbb{N}$ be a computable function.

Then there is a total computable function $s : \mathbb{N}^m \to \mathbb{N}$

s.t. $\forall \vec{x} \in \mathbb{N}^m$, $\vec{y} \in \mathbb{N}^m$

$$f(\vec{x}, \vec{y}) = \varphi^{(m)}_{s(\vec{x})}(\vec{y})$$

proof

since f is computable, there is $e \in \mathbb{N}$ s.t. $f = \varphi_e^{(m+m)}$

hence $f(\vec{x}, \vec{y}) = \varphi_e^{(m+m)}(\vec{x}, \vec{y}) = \varphi^{(m)}_{S_{m,m}(e, \vec{x})}(\vec{y})$

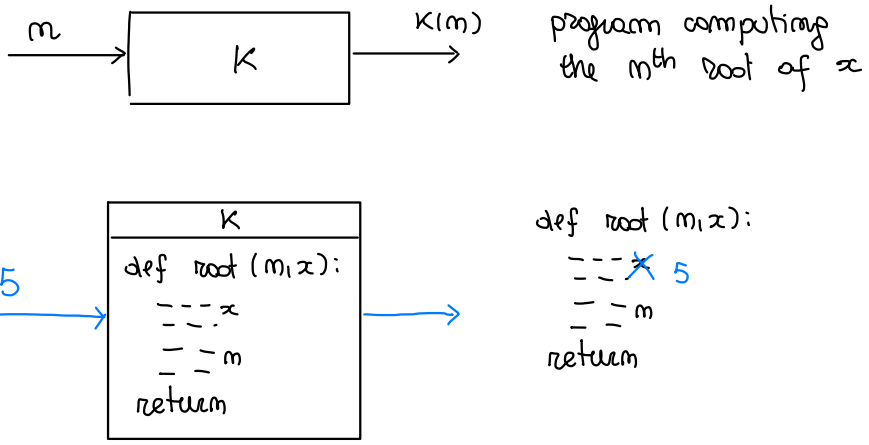
we conclude by setting $s(\vec{x}) = S_{m,m}(e, \vec{x})$ □

EXAMPLE

Prove that there exists a total computable function $K : \mathbb{N} \to \mathbb{N}$ such that

$\forall m \in \mathbb{N} \quad \forall x \in \mathbb{N}$

$$\varphi_{K(m)}(x) = \lfloor \sqrt[m]{x} \rfloor$$



the function

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

$$f(m,x) = \lfloor \sqrt[m]{x} \rfloor$$

$$= \max z \;.\; z^m \leq x$$

$$= \min z \;.\; (z+1)^m > x$$

$$= \mu z \leq x \;.\; \overline{sg}\left((z+1)^m \dot{-} x\right) \qquad \text{computable}$$

$\rightsquigarrow$ by (corollary) of smn-theorem there is $K : \mathbb{N} \to \mathbb{N}$ total computable s.t.

$$\varphi_{K(m)}(x) = f(m,x) = \lfloor \sqrt[m]{x} \rfloor$$

EXAMPLE : There is a total computable function $K : \mathbb{N} \to \mathbb{N}$ such that

$\forall m$ $\varphi_{K(m)}$ is defined only on m^{th} powers

↓ numbers of the kind y^m for some $y \in \mathbb{N}$

$$W_{K(m)} = \{ x \mid \exists y \;.\; y^m = x \}$$

$\varphi_{K(m)}(x) = - - -$

define

$$f(m,x) = \begin{cases} \sqrt[m]{x} & \text{if } x = y^m \text{ for some } y \\ \uparrow & \text{otherwise} \end{cases}$$

$$= \mu y. \quad (\!(\, y^m = x \,)\!)$$

$$= \mu y. \quad |y^m - x| \qquad \text{computable}$$

By the (corollary to) smm-theorem there is $k : \mathbb{N} \to \mathbb{N}$ total and computable s.t. $\forall m, x$

$$\varphi_{k(m)}(x) = f(m,x) = \begin{cases} \sqrt[m]{x} & \text{if } x \text{ is an } m^{th} \text{ power} \\ \uparrow & \text{otherwise} \end{cases}$$

Observe that

$$W_{k(m)} = \{ x \mid \exists y. \; x = y^m \}$$

In fact

$x \in W_{k(m)}$ iff $\varphi_{k(m)}(x) \downarrow$ iff $x = y^m$ for some $y \in \mathbb{N}$

$\varphi_{k(m)}(x) = f(m,x) = \begin{cases} \sqrt[m]{x} & \text{if } x \text{ is an } m^{th} \text{ power} \\ \uparrow & \text{otherwise} \end{cases}$

EXAMPLE : show that there is a total computable function $s : \mathbb{N} \to \mathbb{N}$ s.t.

$$W^{(k)}_{s(x)} = \{ (y_1, \dots, y_k) \mid \sum_{i=1}^{k} y_i = x \}$$

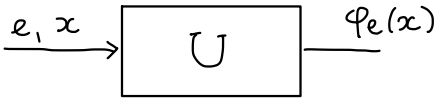
[HOME EXERCISE]

UNIVERSAL FUNCTION

Let $\psi_U : \mathbb{N}^2 \to \mathbb{N}$

$\psi_U(e, x) = \varphi_e(x)$ well-defined

Is it computable ?

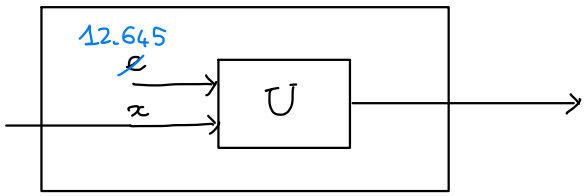


(executes $P_e = \gamma^{-1}(e)$ over x)

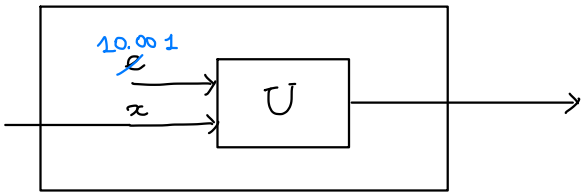
when e varies on natural numbers

$\psi(0, \_) = \varphi_0$ $\quad$ $\psi(1, \_) = \varphi_1$ $\quad$ $\psi(2, \_) = \varphi_2$ $\quad$ - - -

Turing comp.



$ 1.000.000



$ 1.000.000

Theorem (Universal Program)

Let $k \geq 1$ Then the universal function

$\psi_U : \mathbb{N}^{k+1} \to \mathbb{N}$

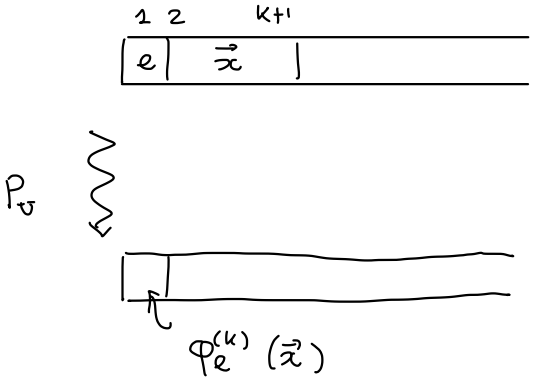
$\psi_U(e, \vec{x}) = \varphi_e^{(k)}(\vec{x})$

is computable

proof

fix $k \geq 1$

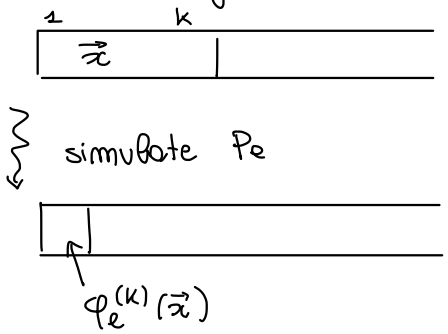
given e, $\vec{x}$



how can P_U work :

→ determine $P_e = \gamma^{-1}(e)$

→ set up memory



by Church - Turing thesis

this is computable

unsatisfactory

(better argument in the next lesson)