5.8 Equivalence between rec.fn & programs

Prop5.21
Any recursive function can be computed by a while program.

Example
$\mu z.P(\vec{x}, z)$ is computed by

$z := 0;$
while$(\neg P(\vec{x}, z))\{$
    $z := z + 1;$
$\}$
return $z;$

Exercise : Simulate primitive recursion using while-programs.

The other direction is non-trivial.

Def5.22(Normalized while programs)
Those programs in the following form.

$w := f(x_1, \cdots, x_n);$
while$(Q(w) \neq 0)\{$
    $w := q(w);$
$\}$
$y = h(w);$
return $y;$

Rem5.23
-We don't bother to define the syntax of while-lang.
-On normalized ones:
    * only one working variables – (Challenge 1)
    * only one while-loop – (Challenge 2)

Prop5.24
The function computed by a normalized whileprogram is recursive.

Proof
Let the function be denoted by $\varphi : \mathbb{N}^n \dashrightarrow \mathbb{N}$
Then

$$\varphi(\vec{x}) = h\left(g^{\#}\left(f(\vec{x}), \mu z.Q\Big(g^{\#}\big(f(\vec{x}), z\big)\Big)\right)\right)$$

$$\text{Recall}: \; g^{\#}(x,y) = \underbrace{g(g(\cdots g(}_{y \; times} x))\cdots)$$

Next

any while program
$\downarrow$
normalized while program

Challenge 1:Gödel Coding
Challenge 2:Program Counter

Def5.25(Gödel Coding)
We define $G : \mathbb{N}^* \to \mathbb{N}$ by : $\left(\mathbb{N}^* = \bigcup_{n=0}^{\infty} \mathbb{N}^n\right)$

$$G(x_1, x_2, \cdots, x_n) := \prod_{i=1}^{n} (pr(i))^{x_i+1}$$

Example

$$\begin{aligned} G(1,4,0,2,0) &= 2^{1+1} \cdot 3^{4+1} \cdot 5^{0+1} \cdot 7^{2+1} \cdot 11^{0+1} \\ &= 2^2 \cdot 3^5 \cdot 5^1 \cdot 7^3 \cdot 11^1 \\ &= 18336780 \end{aligned}$$

Lem5.26
For each m,the restriction
$G|_{\mathbb{N}^m} = \mathbb{N} \to \mathbb{N}$ is PR.

Prop5.27(Decoding is also PR)
There are PR functions

$$\begin{aligned} |\_| \quad & : \mathbb{N} \to \mathbb{N} \\ & s.t. \quad |G(x_1 \cdots x_m)| = m \\ \lambda x.\lambda y.(x)_y \quad & : \mathbb{N}^2 \to \mathbb{N} \\ & s.t. \quad \Big(G(x_1 \cdots x_n)_i\Big) = x_i \qquad (i \in [1 \cdots n]) \end{aligned}$$

Proof
$|\_|$:Idea : find first i
$s.t.$ the remainder of $x \div pr(i)$ is not 0.

Notation 5.28
In that follows, $G(x_1, \cdots, x_m)$ is denoted by $< x_1, \cdots, x_m >$

Normalizing while-programs, By an example

$^{[1]}$while$(x_3 == 0)\{$
  $^{[2]}$if$(x_1 == 0)\{$
    $^{[3]}$while$(P(x_1, x_2))\{$
      $^{[4]}x_1 := f(x_3);$
    $\}^{[5]}$
  $\}$else$\{$
    $^{[6]}x_2 := g(x_2);$
  $\}^{[7]}$
$\}^{[8]}$
return x_1;

First, put markers, $[1], \cdots, [8]$
Then the given program is equivalent to

while$(pc \neq 8)\{$
  cases $pc == 1$ and $x_3 == 0 : pc = 2;$
    $pc == 1$ and $x_3 \neq 0 : pc = 8;$
    $pc == 2$ and $x_1 == 0 : pc = 3;$
    $pc == 2$ and $x_1 \neq 0 : pc = 6;$
    $pc == 3$ and $P(x_1, x_2) : pc = 4;$
    $pc == 3$ and $\neg P(x_1, x_2) : pc = 5;$
    $pc == 4 \ : x_1 := f(x_3); pc = 3;$
    $pc == 5 \ : pc = 7;$

$pc == 6 \quad : x_2 := g(x_2); pc = 7;$

$pc == 7 \quad : pc = 1;$

}

return x_1;

Finally, we encode the multiple variables into a single var. By Gödel Coding

$$\begin{aligned} w &:= < 1, x_1, x_2, x_3 > \\ \rightarrow pc &= (w)_1 \\ x_1 &= (w)_2 \\ x_2 &= (w)_3 \\ x_3 &= (w)_4 \end{aligned}$$

Thm5.27

Recursive functions are exactly those functions computed by while programs.

From the above proofs,we also obtain:

Thm5.30(Kleene's normal form)

For any rec func,$f : \mathbb{N}^n \dashrightarrow \mathbb{N}$.

there exist

a PR predicate $P \subseteq \mathbb{N}^{n+1}$

a PR function $g : \mathbb{N}^{n+1} \to \mathbb{N}$

such that

$$f(\vec{x}) = g\left(\vec{x}, \mu z.P(\vec{x}, z)\right)$$

Proof

$$\begin{aligned} rec.func &\mapsto while - program \\ &\mapsto normalized \quad while - program \\ &\mapsto rec.func(\leftarrow aboveform) \end{aligned}$$

5.9 Universal recursive function

Thm5.31
There exists a recursive function

$$comp : \mathbb{N}^2 \dashrightarrow \mathbb{N}$$

such that:
for each recursive function $f : \mathbb{N}^m \dashrightarrow \mathbb{N}$
there is a natural number p, s.t.

$$f(\vec{x}) = comp(p, < \vec{x} >)$$

(7/11 講義分終わり：Universal recursive function が何故存在すると言えるのかの証明は講義では行いませんでした。)