6. Gödel numberings

In this section we discuss ***effective codings*** or ***Gödel numberings*** based on PR functions, and use them to code $\mathcal{G}$-programs as numbers so that they can serve as inputs to other programs—or to themselves!

Theorem 6.1 (Fundamental Theorem of Arithmetic).
Every number > 1 can be represented uniquely (apart from order) as a product of primes.

Hence for $x > 1$, we can write

$$x = p_1^{e_1} p_2^{e_2} \cdots p_k^{e_k} \tag{1}$$

for *unique* $k > 0$, $e_1, \ldots, e_k$, where $p_i = i$th prime ($p_1 = 2$), $e_i \geq 0$ for $1 \leq i \leq k$, and $e_k > 0$.

Lemma 6.2.

(a) *For $a \geq 2$, $n < a^n$.*

(b) $n \leq p_n$.

Proof: By induction on n. $\square$

Hence in (1):

$$\left.\begin{array}{l} e_i < p_i^{e_i} \leq x \quad (1 \leq i \leq k) \\ \;k \leq p_k \leq x \end{array}\right\} \tag{2}$$

6.1 PR coding of pairs of numbers

We define

$$\boldsymbol{pair}(x, y) = \langle x, y \rangle = 2^x(2y + 1) \dot{-} 1,$$

which is clearly PR.

Lemma 6.3.

$$\forall z \exists ! x, y (\langle x, y \rangle = z) \tag{3}$$

Proof: We want $z = \langle x, y \rangle$ i.e.,

$$z + 1 = 2^x(2y + 1).$$

By the Fundamental Theorem of Arithmetic (Thm 6.1),

$$z + 1 = 2^x 3^{a_2} 5^{a_3} \cdots = 2^x u$$

for *unique* x and u, where u is *odd* (possibly 1). Put $u = 2y + 1$.
So y is also uniquely determined (possibly 0). □

NOTE: Lemma 6.3 determines two ***inverse functions*** satisfying (3), i.e. the functions ***left inverse*** $\boldsymbol{\ell}(z)$ and ***right inverse*** $\boldsymbol{r}(z)$, which satisfy

$$\begin{aligned} \boldsymbol{\ell}(\langle x, y \rangle) &= x, \\ \boldsymbol{r}(\langle x, y \rangle) &= y, \\ \text{and} \qquad \langle \boldsymbol{\ell}(z), \boldsymbol{r}(z) \rangle &= z. \end{aligned}$$

Lemma 6.4. $x, y \le \boldsymbol{pair}(x, y)$.

Proof: In (3),

$$\begin{aligned} x < 2^x \le 2^x(2y+1) = z + 1, \text{ and} \\ y < 2y + 1 \le 2^x(2y+1) = z + 1. \end{aligned}$$

So $x, y \le z$. □

Lemma 6.5. $\boldsymbol{\ell}, \boldsymbol{r} \in PR$.

Proof:

$$\begin{aligned} \boldsymbol{\ell}(z) &= (\mu x \le z)(\exists y \le z)(z = \langle x, y \rangle), \text{ and} \\ \boldsymbol{r}(z) &= (\mu y \le z)(\exists x \le z)(z = \langle x, y \rangle). \quad \square \end{aligned}$$

Theorem 6.6 (Simultaneous or mutual primitive recursion).
Let

$$\begin{cases} f_1(x,0) &= g_1(x) \\ f_2(x,0) &= g_2(x) \\ f_1(x,t+1) &= h_1(x,t,f_1(x,t),f_2(x,t)) \\ f_2(x,t+1) &= h_2(x,t,f_1(x,t),f_2(x,t)). \end{cases}$$

Then $f_1, f_2 \in \mathrm{PR}(g_1, g_2, h_1, h_2)$.
Hence if $g_1, g_2, h_1, h_2 \in \mathrm{PR}$, *then so are* f_1, f_2.

Proof: We put $f(x,t) = \langle f_1(x,t), f_2(x,t)\rangle$ and show that $f \in \mathrm{PR}(g_1, g_2, h_1, h_2)$. Let

$$f(x,0) = \langle g_1(x), g_2(x)\rangle = g(x) \quad \text{(say)}$$

and

$$\begin{aligned} f(x,t+1) &= \langle h_1(x,t,f_1(x,t),f_2(x,t)), \\ &\qquad h_2(x,t,f_1(x,t),f_2(x,t))\rangle \\ &= \langle h_1(x,t,\boldsymbol{\ell}(f(x,t)),\boldsymbol{r}(f(x,t))), \\ &\qquad h_2(x,t,\boldsymbol{\ell}(f(x,t)),\boldsymbol{r}(f(x,t)))\rangle \\ &= h(x,t,f(x,t))) \quad \text{(say)} \end{aligned}$$

where $h(x,t,z) =_{df} \langle h_1(x,t,\boldsymbol{\ell}(z),\boldsymbol{r}(z)), h_2(x,t,\boldsymbol{\ell}(z),\boldsymbol{r}(z))\rangle$. So

$$\begin{aligned} & f \in \mathrm{PR}(g,h), \\ & g \in \mathrm{PR}(g_1,g_2) \qquad \text{by expl. def.,} \\ & h \in \mathrm{PR}(h_1,h_2) \qquad \text{by expl. def.} \end{aligned}$$

Therefore, by transitivity, $f \in \mathrm{PR}(g_1, g_2, h_1, h_2)$. Also

$$\begin{aligned} & f_1(x,t) = \boldsymbol{\ell}(f(x,t)) \text{ and} \\ & f_2(x,t) = \boldsymbol{r}(f(x,t)). \end{aligned}$$

So $f_1, f_2 \in \mathrm{PR}(f)$. Therefore, by transitivity again,

$$f_1,\ f_2 \in \mathrm{PR}(g_1, g_2, h_1, h_2). \qquad \square$$

6.2 PR coding of finite sequences of numbers

We define the ***code*** or ***Gödel number*** (***gn***) of a sequence $a_1, \ldots, a_n$ $(n \geq 0)$ as the number

$$[a_1, \ldots, a_n] = \prod_{i=1}^{n} p_i^{a_i}.$$

- The function $[\ \ ]\colon \mathbb{N}^* \to \mathbb{N}$ is a ***coding*** or ***Gödel numbering*** (***GN***) of $\mathbb{N}^*$.

Lemma 6.7. *For fixed n,*

$$\lambda x_1, \ldots, x_n \cdot [x_1, \ldots, x_n] \in \mathrm{PR}.$$

Proof: Clear. $\square$

Theorem 6.8 (Uniqueness of components).

$$[a_1, \ldots, a_n] = [b_1, \ldots, b_n] \Rightarrow a_i = b_i\ (i = 1, \ldots, n).$$

Proof: By the fundamental theorem of arithmetic. $\square$

NOTES:

1. $[a_1, \ldots, a_n, 0] = [a_1, \ldots, a_n]$, so trailing 0's make no difference.

2. $[0] = [0, 0] = [0, 0, 0] = \cdots = 2^0 3^0 5^0 \cdots = 1$, so 1 codes any sequence of 0's. We also assume that 1 codes the *empty sequence* [ ].

The following two functions are, in a sense, *inverses* of the GN function. Let $x = [a_1, \ldots, a_n]$ (note that $x > 0$). Define

$$(x)_i = \begin{cases} a_i & \text{if } 1 \leq i \leq n \\ 0 & \text{otherwise} \end{cases}$$

$$\begin{aligned} \boldsymbol{Lt}(x) &= \textit{length} \text{ of the sequence represented by } x \\ &= k \text{ when } x = [a_1, \ldots, a_k] \text{ with } a_k \neq 0 \\ \text{and} \qquad \boldsymbol{Lt}(0) &= 0. \end{aligned}$$

Note that $(x)_i$ is *well-defined*, since, e.g., if $x = [a_1, a_2] = [a_1, a_2, 0, 0]$, then $(x)_4 = 0$ under either interpretation.

Lemma 6.9.

(a) $([a_1, \ldots, a_n])_i = \begin{cases} a_i & \textit{if } 1 \le i \le n \\ 0 & \textit{otherwise,} \end{cases}$

(b) $[(x)_1, \ldots, (x)_n] = x$ *if* $n \ge \boldsymbol{Lt}(x) \qquad (x \ne 0)$.

Proof: From the definitions. $\square$

Theorem 6.10. $\lambda x, i \cdot (x)_i,\ \boldsymbol{Lt} \in \mathrm{PR}$.

Proof:

$$(x)_i = (\mu y < x)\neg(p_i^{y+1} | x),$$
$$\boldsymbol{Lt}(x) = \mu k[(\forall j > k)((x)_j = 0)].$$

But to apply the results of Section 5, we need bounds for k and j. So from (2) (p. 6-1),

$$\boldsymbol{Lt}(x) = (\mu k \le x)[(\forall j \le x)(k < j \Rightarrow (x)_j = 0)]. \quad \square$$

NOTE 3: For later use we define

$$\boldsymbol{concat}(x, y) = x^\frown y = \text{ concatenation of } x \text{ and } y,$$

where x and y are viewed as gn's of finite sequences.

Lemma 6.11. $\boldsymbol{concat} \in \mathrm{PR}$.

Proof: Suppose that

$$x = p_1^{a_1} \cdots p_k^{a_k}, \quad k = \boldsymbol{Lt}(x), \quad a_i = (x)_i, \quad a_k \ne 0;$$
$$y = p_1^{b_1} \cdots p_\ell^{b_\ell}, \quad \ell = \boldsymbol{Lt}(y), \quad b_i = (y)_i, \quad b_\ell \ne 0.$$

So

$$\begin{aligned} x^\frown y &= p_1^{a_1} \cdots p_k^{a_k} \cdot p_{k+1}^{b_1} \cdots p_{k+\ell}^{b_\ell} \\ &= x * \prod_{i=1}^{\boldsymbol{Lt}(y)} p_{\boldsymbol{Lt}(x)+i}^{(y)_i}. \quad \square \end{aligned}$$

Exercises:

1. Show that ***div*** and ***mod*** (p. 5-11) are PR, without using "bounded μ" or bounded quantification. Hint:
   (a) Define ***mod*** by primitive recursion (not using ***div***).
   (b) Define ***div*** by primitive recursion (using ***mod***, if you wish).

2. **(CV recursion.)** For any function f, write

$$\tilde{f}(n) \ =_{df} \ [f(0), \ldots, f(n-1)].$$

   Note: $\tilde{f}(0) = [\ ] = 1$.
   Now, given a function g, suppose f is defined by $f(n) = g(\tilde{f}(n))$.
   (The point is that the value of f at n depends explicitly on the values of f at i *for all* $i < n$, not just on $f(n-1)$, as with def. by prim. rec.)
   Show that $f \in \mathrm{PR}(g)$. (Hence if $g \in \mathrm{PR}$, then so is f.)
   In other words:

   "*CV recursion can be reduced to PR.*"

3. **(Fibonacci sequence)**
   Let $F(0) = 0$, $F(1) = 1$, $F(n+2) = F(n) + F(n+1)$. Show $F \in \mathrm{PR}$.

4. Let $f(x)$ = "no. of 1's in binary rep. of x". Show f is PR.

6.3 Gödel numbering of the $\mathcal{G}$ programming language

Let S be a set. (1) A ***Gödel numbering*** (GN) or ***effective coding*** of S is a 1-1 map $\# : S \hookrightarrow \mathbb{N}$ such that

- for all $x \in S$, we can effectively (or algorithmically) find $\#(x) \in \mathbb{N}$, and
- for all $n \in \mathbb{N}$, we can effectively determine whether $n \in \boldsymbol{ran}(\#)$, and if so, effectively find the $x \in S$ such that $\#(x) = n$.

(2) A ***listing*** or ***enumeration*** of S is a function $\ell\colon \mathbb{N} \twoheadrightarrow S$. (See p. 1-10.) If ℓ is 1-1 (hence bijective), it is called a ***listing without repetitions***.

We will be interested in ***effective listings***.

NOTE 1: If S has a ***GN or listing***, then S is ***countable*** (by Thm 1.11).

Theorem 6.12.

(*a*) *A **GN** $\#$ of S has a **left inverse** ℓ, which is an **eff. listing** of S. Further, if $\#$ is **onto** $\mathbb{N}$ (hence bij), then ℓ is **w/o reps** (hence bij).*

(*b*) *Assuming S has decidable equality:*
*An **eff. listing** ℓ of S has a **right inverse** $\#$, which is a **GN** of S. Further, if ℓ is **w/o reps** (hence bij), then $\#$ is **onto** $\mathbb{N}$ (hence bij).*

Proof: EXERCISE. $\square$

NOTE 2: This theorem gives an ***effective version*** of Thm 1.4 and Cor. 1.5 (p. 1-7), with $A = S$, $B = \mathbb{N}$, $f = \#$, $g = \ell$.
(See also p. 1-10: Cor. 1.12 and Note (*a*).)

NOTE 3: It is often convenient to make the GN ***surjective***, i.e., ***onto*** $\mathbb{N}$, by Thm 6.12(*a*).

EXAMPLE: The GN's of $\mathbb{N}^2$ and $\mathbb{N}^*$ defined above are (essentially) onto $\mathbb{N}$. (***Caution***! The "GN" of $\mathbb{N}^*$ is not 1-1, hence not strictly a GN, by our def.)

- Now we are ready to code $\mathcal{G}$-programs as numbers!

- **Effective listing of all variables** (see p. 2-1)

$$\begin{array}{cccccccc} Y, & X_1, & Z_1, & X_2, & Z_2, & X_3, & Z_3, & \ldots \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & \ldots \end{array}$$

For example, $\#(X_2) = 4$.

- **Effective listing of all labels**

$$\begin{array}{cccccccc} A_1, & B_1, & C_1, & D_1, & E_1, & A_2, & B_2, & \ldots \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 & \ldots \end{array}$$

- **Gödel numbering of all instructions**

For convenience we *replace* 'skip' by '$V \leftarrow V$' for any variable V.
Then the Gödel number of instruction I is $\#(I) = \langle a, \langle b, c\rangle\rangle$, where

— $a = \begin{cases} 0 & \text{if } I \text{ is unlabelled,} \\ \#(L) & \text{if } I \text{ has label } L; \end{cases}$

— $b = \begin{cases} 0 & \text{if } I \text{ is } \ V \leftarrow V \\ 1 & \text{if } I \text{ is } \ V{+}{+} \\ 2 & \text{if } I \text{ is } \ V{-}{-} \\ \#(L') + 2 & \text{if } I \text{ is } \ \mathsf{if}\ V \neq 0\ \mathsf{goto}\ L' \end{cases}$

— $c = \#(V) \dot{-} 1$ if the variable in I is V.

The associated ***effective listing*** of all instructions is obtained thus:
Given $q \in \mathbb{N}$, we let $a = \boldsymbol{\ell}(q)$, $b = \boldsymbol{\ell}(\boldsymbol{r}(q))$, $c = \boldsymbol{r}(\boldsymbol{r}(q))$.
Then the statement

— is *unlabelled* if $a = 0$, and it has the label with number a if $a \neq 0$.

— is $\begin{cases} V \leftarrow V & \text{if } b = 0 \\ V{+}{+} & \text{if } b = 1 \\ V{-}{-} & \text{if } b = 2 \\ \mathsf{if}\ V \neq 0\ \mathsf{goto}\ L & \text{if } b > 2 \end{cases}$

where the label L is such that $\#(L) = b - 2$.

— uses variable V with $\#(V) = c + 1$.

- **Gödel numbering of programs**

  Let $\mathcal{P} = (I_1, \ldots, I_k)$ be a program. Define

  $$\#(\mathcal{P}) = [\#(I_1), \ldots, \#(I_k)] - 1.$$

  This is ***onto*** $\mathbb{N}$. However it is ***not 1-1***, since the unlabelled statement '$Y \leftarrow Y$' has Gödel number 0, and hence we can form *many* programs $\mathcal{P}$ with the same $\#(\mathcal{P})$ by simply adding any number of unlabelled statements '$Y \leftarrow Y$'.

  To prevent this, we *stipulate* that a program may not end with an unlabelled statement of the form '$Y \leftarrow Y$'.

  Denote by $\mathcal{G}$-PROG the set of all such programs. Then

  $$\#\colon \mathcal{G}\text{-PROG} \approx \mathbb{N}$$

  is ***bijective***. So by Thm 6.12(a):

  the ***inverse*** of $\#$ is an ***effective listing w/o reps*** of $\mathcal{G}$-PROG:

  $$\mathcal{P}_0, \mathcal{P}_1, \mathcal{P}_2, \ldots.$$

  where $\mathcal{P}_n$ is the program $\mathcal{P}$ with $\#(\mathcal{P}) = n$.

EXERCISES:

1. Let $\mathcal{P}$ be the program

   $$\boxed{\begin{array}{l}\text{if } X \neq 0 \text{ goto } E \\ Y{+}{+}\end{array}}$$

   which computes the ***zero*** function. What is $\#(\mathcal{P})$?

2. What is $\mathcal{P}_0$? What is $\mathcal{P}_{99}$?

3. Show that every $\mathcal{G}$-computable function has *infinitely many* gn's, i.e.: $\forall a\ \exists$ *infinitely many* b : $\psi_{\mathcal{P}_a} = \psi_{\mathcal{P}_b}$.