chngcntr

1 Dovetailling enumeration and the π function

Let $(x)_i$ denothe the power of the ith prime factor in the decomposition of $x \in \omega$. Let φ_e an arbitrary partial computable function computed by program P_e. The Turing program P_e runs on discrete steps, and so we can conceive the following procedure:

1. Fix $n \leftarrow (x)_1, t \leftarrow (x)_2$.
2. If P_e halts in t steps from input n, halt and return 1.
3. Otherwise, set $x \leftarrow x + 1$ and go back to step (1).

This program halts if and only if P_e halts on some input, since eventually all possible inputs are considered. We use $\pi(x)$ to denote the partial computable function that performs the procedure above on program x. We observe that if $W_x \neq \emptyset$ then $\pi(x)$ always halts, meaning that $W_{\pi(x)} = \mathbb{N}$. On the contrary, if $W_x = \emptyset$ then $\pi(x)$ is undefined and $W_{\pi(x)} = \emptyset$.

Theorem 1. There is a partial computable function $\pi_k^c(x)$ that takes k arguments and halts with output c (independently of the arguments) if and only if $W_x \neq \emptyset$. Furthermore, if $W_x \neq \emptyset$, then $W_{\pi_k^c(x)} = \mathbb{N}$, and if $W_x = \emptyset$ then $W_{\pi_k^c(x)} = \emptyset$.

Proof. Trivial to derive from all of the above.

In general, if we use $\pi(x)$ to denote $\pi_1^1(x)$.

2 Index set theorems

Notation. Let $\mathcal{F}$ be the set of partial computable functions. We use $\perp_F$ to denote the partial computable function which is undefined on all input.

A set $A \subseteq \omega$ is an index set if for all $x, y \in \omega$,

$$x \in A \wedge \varphi_x = \varphi_y \Rightarrow y \in A$$

In other words, a set is an index set if all elements in the set index the same partial computable function.

Would it not be better to set $\varphi_x \simeq \varphi_y$? Think about this.

Trivially, ω and $\emptyset$ are index sets.

Theorem 2 (Index set theorem)**.** If A is a non-trivial index set, then either $K \leq_1 A$ or $K \leq_1 \overline{A}$. Furthermore, if $\perp_{\mathcal{F}} \in \overline{A}$ then $K \leq_1 A$, and vice-versa.

Proof. Assume the index of $\perp_{\mathcal{F}}$ is in A and take $y \in \overline{A}$. Define

$$\phi(u, v) = \begin{cases} \varphi_y(v) & u \in K \\ \perp & c.c. \end{cases}$$

The function above is computable. The S_n^m theorem ensures there is a total, one-to-one function f s.t. $\varphi_{f(u)}(v) = \phi(u, v)$.

If $u \in K$, then $\varphi_{f(u)} = \varphi_y$, meaning that $f(u) \in \overline{A}$. If $u \notin K$, then $f(u)$ is the index of $\perp_{\mathcal{F}}$, which is in A.

$\therefore K \leq_1 \overline{A}$.

Theorem 3 (Rice's theorem)**.** Let $\mathcal{C}$ any class of partial computable functions. Then $A = \{n : \varphi_n \in \mathcal{C}\}$ is not computable except in the trivial cases.

Proof. Assume the class $\mathcal{C}$ is non-trivial, meaning that $A = \{n : \varphi_n \in \mathcal{C}\}$ is neither ω nor $\emptyset$. Then $K \leq_1 A$ or $K \leq_1 \overline{A}$ by virtue of the index set theorem. Either case, A is not decidable. ■

2.1 Some interesting sets

The set of programs which halt with themselves as input (K), and the set of pairs (x, y) such that P_x halts on input y (K_0), are not index sets. Index sets of interest are:

1. K_1, the set of programs that halt on some input:

$$K_1 := \{x : W_x \neq \emptyset\} = \{x : \varphi_x = \perp_{\mathcal{F}}\}$$

2. *Fin*, the set of programs that halt for a finite number of inputs.

$$Fin := \{x : W_x \text{ is finite}\}$$

3. The complement of *Fin*, termed *Inf*.

4. $\mathcal{T}$, the set of total computable functions.

5. *Con* $\subseteq \mathcal{T}$, the set of constant and total functions.

6. *Cof*, the set of programs that halt on a cofinite number of input.

7. *Rec*, the set of computable (recursive) programs.

8. *Ext*, the set of programs that can be extended to total computable functions.

An interesting factt is that $K \equiv_1 K_0 \equiv_1 K_1$. We already know that $K \leq_1 K_1$, because K_1 is a non-trivial index set, so let us observe the remaining relations.

$(1 : K_1 \leq K)$ This is intuitively clear, since deciding whether a program halts with itself as input would suffice to decide whether it halts at all.

Recall that

$$W_{d(x)} = \begin{cases} \mathbb{N} & x \in K_1 \\ \emptyset & x \notin K_1 \end{cases}$$

This entails that if $x \in K_1$, then $\pi(x) \in W_{\pi(x)}$, which means that $\pi(x) \in K$. If $x \notin K_1$, clearly $\pi(x) \notin W_{\pi(x)}$, since said domain is the empty set, and $\pi(x) \notin K$. ■

The question becomes whether $K_1 \leq K$. This is intuitively clear, because deciding whether a function halts with itself as input would already decide whether it halts at all under some input.

Prove that $x \in K_1$ iff $f(x) \in K$ for some f.

Consider

$$\phi(x, y) = \begin{cases} 1 & x \in K_1 \\ \perp & c.c. \end{cases} = \varphi_{f(x)}(y)$$

with f one-to-one and total. If $x \in K_1$, then $\varphi_{f(x)}(y) = 1$ for all y. In particular, $\varphi_{f(x)}(x) = 1 \neq \perp$ for all input, and in particular $\varphi_{f(x)}(f(x)) = 1$. In other words, $f(x) \in K$.

If $x \notin K_1$, then $\phi(x, y)$ fails to converge and in particular $\varphi_{f(x)}(y) = \perp$ for all y. This means $f(x) \notin K$.

$\therefore K_1 \leq_1 K$.

Definition 1. A computably enumerable set A is 1-complete if $W_e \leq_1 A$ for every computably enumerable set W_e.

Problem: Is K_0 1-complete? The answer is yes. Take W_e arbitrary and c.e. Then $x \in W_e$ if and only if $(x, e) \in K_0$. Thus suffices to show $W_e \leq_1 K_0$.

Problem. Assume $A \leq_m B$ and A is 1-complete.