Recursively Enumerable Sets and Degrees: A Study of Computable Functions and Computably Generated Sets

Robert I. Soare

January 7, 2020

Contents

1 Recursive Functions

1.1 Formal Definitions of Computable Functions

1.1.1 Primitive Recursive Functions

Definition 1.1. The class of primitive recursive functions is the smallest class $\mathcal{C}$ of functions closed under the following schema

1. the **successor function**, $\lambda x[x+1] \in \mathcal{C}$
2. the **constant functions**, $\lambda x_1 \ldots x_n[m] \in \mathcal{C}$, $0 \leq n, m$
3. the **identity functions**, $\lambda x_1 \ldots x_n[x_i] \in \mathcal{C}$, $1 \leq i \leq n$
4. (Composition) If $g_1, \ldots, g_m, h \in \mathcal{C}$, then

$$f(x_1, \ldots, x_n) = h(g_1(x_1, \ldots, x_n), \ldots, g_m(x_1, \ldots, x_n))$$

   is in $\mathcal{C}$ where $g_1, \ldots, g_m$ are functions of n variables and h is a function of m variables
5. (Primitive Recursion) If $g, h \in \mathcal{C}$ and $n \geq 1$ then $f \in \mathcal{C}$ where

$$f(0, x_2, \ldots, x_n) = g(x_2, \ldots, x_n)$$
$$f(x_1+1, x_2, \ldots, x_n) = h(x_1, f(x_1, \ldots, x_n), x_2, \ldots, x_n)$$

Hence a function is primitve recursive if there is a **derivation**, namely a sequence $f_1, \ldots, f_k = f$ s.t. for each $f_i, i \leq k$ is either an initial function or obtained from 4 or 5.

A predicate (relation) is **primitive recursive** if its characteristic function is.

1.1.2 Diagonalization and Partial Recursive Functions

Although the primitive recursive functions include all the usual functions from elementary number theory they fail to include **all** computable functions. Each derivation of a primitive recursive function is a finite string of symbols from a fixed finite alphabet, and thus all derivations can be effectively listed. Let f_n be the function corresponding to the nth derivation in this listing. Then the function $g(x) = f_x(x) + 1$ cannot be primitive recursive.

The same argument applies to any effective set of schemata which produces only **total** functions. Thus to obtain all computable functions we are forced to consider computable **partial** functions.

Definition 1.2 (Kleene)**.** The class of **partial recursive** (p.r.) functions is the least class obtained by closing under schemata 1 through 5 for the primitive recursive functions and the following schemata 6. A **total recursive** function (abbreviated **recursive** function) is a partial recursive function which is total.

6. (Unbounded Search) If $\theta(x_1, \ldots, x_n, y)$ is a partial recursive function of $n + 1$ variables, and

$$\psi(x_1, \ldots, x_n) = \mu y[\theta(x_1, \ldots, x_n, y) \downarrow= 0 \\ \wedge (\forall z \leq y)[\theta(x_1, \ldots, x_n, z) \downarrow]]$$

Definition 1.3. A relation $R \subseteq \omega^n, n \geq 1$ is **recursive** ({primitive recursive, has property } P) if its characteristic function χ_R is recursive (primitive recursive) where $\chi_R(x_1 \ldots, x_n) = 1$ if and only if $(x_1, \ldots, x_n) \in R$.

1.1.3 Turing Computable Functions

A **Turing machine** M includes a two-way infinite **tape** divided into **cells**, a **reading head** which scans one cell of the tape at a time, and a finite set of internal **states** $Q = \{q_0, \ldots, q_n\}, n \geq 1$. Each cell is either blank (B) or has written on it the symbol 1. In a single step the machine may simultaneously

1. change from one state to another
2. change the scanned symbol s to another symbol $s' \in S = \{1, B\}$
3. move the reading head one cell to the right (R) or left (L)

The operation of M is controlled by a partial map $\delta : Q \times S \to Q \times S \times \{R, L\}$

The map δ viewed as a finite set of quintuples is called a {Turing program}. The **input** integer x is represented by a string of $x + 1$ consecutive 1's.

1.2 The Basic Results

Church's Thesis asserts that these functions coincide with the intuitively computable functions. We shall accept Church's Thesis and from now on shall use the terms "partial recursive" "Turing computable" and "computable" interchangeably

Definition 1.4. Let P_e be the Turing program with code number (Gödel number) e (also called **index** e) in this listing and let $\varphi_e^{(n)}$ be the partial fucntions of n variables computed by P_e, where φ_e abbreviates $\varphi_e^{(1)}$

Lemma 1.5 (Padding Lemma). *Each partial recursive function φ_x has $\aleph_0$ indices, and furthermore for each x we can effectively find an infinite set A_x of indices for the same partial function*

Proof. For any program P_x mentioning internal states $\{q_0, \dots, q_n\}$ add extraneous instructions $q_{n+1}Bq_{n+1}BR, q_{n+2}Bq_{n+2}, BR, \dots$ to get new programs for the same functions □

Theorem 1.6 (Normal Form Theorem (Kleene). *There exist a predicate $T(e,x,y)$ (called the **Kleene T-predicate**) and a function $U(y)$ which are recursive (indeed primitive recursive) s.t.*

$$\varphi_e(x) = U(\mu y T(e,x,y))$$

It follows from the Normal Form Theorem that every Turing computable partial function is partial recursive.

Theorem 1.7 (Enumeration Theorem). *There is a p.r. function of 2 variables $\varphi_z^{(2)}(e,x)$ s.t. $\varphi_z^{(2)}(e,x) = \varphi_e(x)$. Indeed the Enumeration Theorem holds for p.r. functions of n variables*

Proof. Let $\varphi_z^{(2)}(e,x) = U(\mu y T(e,x,y))$. For $\varphi_z^{(n)}(e, x_1, \dots, x_{n-1})$, by s-m-n theorem,

$$\varphi_z^{(n)}(e,\bar{x}) = \varphi^{(n-1)}_{s^2_{n-1}(z,e)}(\bar{x})$$

Thus we only need to make sure that $s^2_{n-1}(z,e) \in A_e$, which can be effectively found. □

Theorem 1.8 (Parameter Theorem (s-m-n Theorem)). *For every $m,n \geq 1$ there exists a 1:1 recursive function s_n^m of $m+1$ variables s.t. for all $x, y_1, y_2, \dots, y_m$*

$$\varphi^{(n)}_{s_n^m(x,y_1,\dots,y_m)} = \lambda z_1, \dots, z_n(\varphi_x^{(m+n)}(y_1, \dots, y_m, z_1, \dots, z_n))$$

Proof. (informal). For simplicity consider the case $m = n = 1$. The program $P_{s_1^1(x,y)}$ on input z first obtains P_x and then applies P_x to input (y,z) □

Remark. Here is an interesting question in StackExchange

The s-m-n theorem asserts that y may be treated as a fixed parameter in the program $P_{s(x,y)}$ which operate on z and furthermore that the index $s(x,y)$ of this program is effective in x and y. A simple application of the \$s\$-\$m\$-n theorem is the existence of a recursive function $f(x)$ s.t. $\varphi_{f(x)} = 2\varphi_x$.

Let $\psi(x,y) = 2\varphi_x(y)$. By Church's Thesis $\psi(x,y) = \varphi_e^{(2)}(x,y)$ for some e. Let $f(x) = s_1^1(e,x)$

We let $\langle x,y\rangle$ denote the image of (x,y) under the standard pairing function $\frac{1}{2}(x^2+2xy+y^2+3x+y)$ which is a bijective recursive function from $\omega^2 \to \omega$. Let π_1 and π_2 denote the inverse functions $\pi_1(\langle x,y\rangle) = x$

Definition 1.9. We write $\varphi_{e,s}(x) = y$ if $x,y,e < s$ and y is the output $\varphi_e(x)$ in $< s$ steps of the Turing machine P_e. If such a s exists we say $\varphi_{e,s}(x)$ **converges**, which we write as $\varphi_{e,s}(x)\downarrow$, and **diverges** ($\varphi_{e,s}(x)\uparrow$). Similarly, we write $\varphi_e(x)\downarrow$ if $\varphi_{e,s}(x)\downarrow$ for some s

Theorem 1.10. *1. The set $\{\langle e,x,s\rangle : \varphi_{e,s}(x)\downarrow\}$ is recursive*

2. The set $\{\langle e,x,y,s\rangle : \varphi_{e,s}(x) = y\}$ is recursive

Proof. From Church's Thesis since they are all computable □

1.3 Recursively Enumerable Sets and Unsolvable Problems

Definition 1.11. 1. A set A is **recursively enumerable** (r.e.) if A is the domain of some p.r. function

2. let the eth r.e. set be denoted by

$$W_e = \mathrm{dom}(\varphi_e) = \{x : \varphi_e(x)\downarrow\} = \{x : (\exists y)T(e,x,y)\}$$

3. $W_{e,s} = \mathrm{dom}(\varphi_{e,s})$

Note that $\varphi_e(x) = x$ iff $(\exists s)[\varphi_{e,s} = y]$ and $x \in W_e$ iff $(\exists s)(x \in W_{e,s})$

Definition 1.12. Let $K = \{x : \varphi_x(x) \text{ converges }\} = \{x : x \in W_x\}$

Proposition 1.13. *K is r.e.*

Proof. K is the domain of the following p.r. function

$$\psi(x) = \begin{cases} x & \text{if } \varphi_x(x) \text{ converges,} \\ \text{undefined} & \text{otherwise} \end{cases}$$

Now ψ is p.r. by Church's Thesis since $\psi(x)$ can be computed by applying program P_x to input x and giving output x only if $\varphi(x)$ converges. Alternatively and more formally, $K = \mathrm{dom}(\theta)$ where $\theta(x) = \varphi_z^{(2)}(x,x)$ for $\varphi_z^{(2)}$ the p.r. function defined in the Enumeration Theorem □

Corollary 1.14. *K is not recursive*

Proof. If K had a recursive characteristic function χ_K then the following function would be recursive

$$f(x) = \begin{cases} \varphi_x(x) + 1 & \text{if } x \in K \\ 0 & \text{if } x \notin K \end{cases}$$

However f cannot be recursive since $f \neq \varphi_x$ for any x $\square$

Definition 1.15. $K_0 = \{\langle x, y\rangle : x \in W_y\}$

K_0 is p.r. but

Proposition 1.16. *K_0 is not recursive*

Proof. $x \in K$ iff $\langle x, x\rangle \in K$ $\square$

The **halting problem** is to decide for arbitrary x and y whether $\varphi_x(y)\downarrow$. Corollary 1.14 asserts the unsolvability of the halting problem.

Definition 1.17.
1. A is a **many-one reducible** (**m-reducible**) to B (written $A \leq_m B$) if there is a recursive function f s.t. $f(A) \subset B$ and $f(\bar{A}) \subseteq \bar{B}$, i.e. $x \in A$ iff $f(x) \in B$
2. A is **one-one reducible** (**1-reducible**) to B ($A \leq_1 B$) if $A \leq_m B$ by a 1:1 recursive function

The proof of corollary 1.14 established that $K \leq_1 K_0$ via the function $f(x) = \langle x, x\rangle$

Definition 1.18.
1. $A \equiv_m B$ if $A \leq_m B$ and $B \leq_m A$
2. $A \equiv_1 B$ if $A \leq_1 B$ and $B \leq_1 A$
3. $\deg_m(A) = \{B : A \equiv_m B\}$
4. $\deg_1(A) = \{B : A \equiv_1 B\}$

The equivalence classes under $\equiv_m$ and $\equiv_1$ are called the **m-degrees** and **1-degrees** respectively

Proposition 1.19. *If $A \leq_m B$ and B is recursive then A is recursive*

Proof. $\chi_A(x) = \chi_B(f(x))$ $\square$

Theorem 1.20. $K \leq_1$ *Tot* $:= \{x : \varphi_x$ *is a total function*$\}$

Proof. Define the function

$$\psi(x,y) = \begin{cases} 1 & \text{if } x \in K \\ \text{undefined} & \text{otherwise} \end{cases}$$

By s-m-n theorem, there is a 1:1 recursive function f s.t. $\varphi_{f(x)}(y) = \psi(x,y)$. Choose e s.t. $\varphi_e(x,y) = \psi(x,y)$ since ψ is p.r. and define $f(x) = s^1_1(e,x)$. Note that

$$x \in K \implies \varphi_{f(x)} = \lambda y[1] \implies \varphi_{f(x)} \text{ total} \implies f(x) \in \text{Tot}$$
$$x \notin K \implies \varphi_{f(x)} = \lambda y[\text{undefined}] \implies \varphi_{f(x)} \text{ not total} \implies f(x) \notin \text{Tot}$$

□

Definition 1.21. A set $A \subseteq \omega$ is an **index set** if for all x and y

$$(x \in A \wedge \varphi_x = \varphi_y) \implies y \in A$$

Theorem 1.22. *If A is a nontrivial index set, i.e., $A \neq \emptyset, \omega$, then either $K \leq_1 A$ or $K \leq_1 \overline{A}$*

Proof. Choose e_0 s.t. $\varphi_{e_0}(y)$ is undefined for all y. If $e_0 \in \overline{A}$, then $K \leq_1 A$ as follows. Since $A \neq \emptyset$ we can choose $e_1 \in A$. Now $\varphi_{e_1} \neq \varphi_{e_0}$ because A is an index set. By s-m-n theorem define a 1:1 recursive function f s.t.

$$\varphi_{f(x)}(y) = \begin{cases} \varphi_{e_1}(y) & x \in K \\ \text{undefined} & x \notin K \end{cases}$$

Now

$$x \in K \implies \varphi_{f(x)} = \varphi_{e_1} \implies f(x) \in A$$
$$x \notin K \implies \varphi_{f(x)} = \varphi_{e_0} \implies f(x) \in \overline{A}$$

□

It's possible that both $K \leq_1 A$ and $K \leq_1 \overline{A}$ for an index set A, for example if $A = \text{Tot}$

Corollary 1.23 (Rice's Theorem)**.** *Let $\mathcal{C}$ be any class of partial recursive functions. Then $\{n : \varphi_n \in \mathcal{C}\}$ is recursive iff $\mathcal{C} = \emptyset$ or $\mathcal{C}$ is the set of all partial recursive functions*

Proof. $\mathcal{C}$ is an index set and hence is trivial. □

Definition 1.24.

$$\begin{aligned}
&K_1 = \{x : W_x \neq \emptyset\}\\
&\text{Fin} = \{x : W_x \text{ is finite}\}\\
&\text{Inf} = \omega - \text{Fin} = \{x : W_x \text{ is infinite}\}\\
&\text{Tot} = \{x : \varphi_x \text{ is total}\} = \{x : W_x = \omega\}\\
&\text{Con} = \{x : \varphi_x \text{ is total and constant}\}\\
&\text{Cof} = \{x : W_x \text{ is cofinite}\}\\
&\text{Rec} = \{x : W_x \text{ is recursive}\}\\
&\text{Ext} = \{x : \varphi_x \text{ is extendible to a total recursive function}\}
\end{aligned}$$

Definition 1.25. An r.e. set A is **1-complete** if $W_e \leq_1 A$ for every r.e. set W_e

K_0 is 1-complete because $x \in W_e$ iff $\langle x, e\rangle \in K_0$

Definition 1.26. Let A **join** B written $A \oplus B$ be

$$\{2x : x \in A\} \cup \{2x + 1 : x \in B\}$$

Exercise 1.3.1. 1. $A \leq_m A \oplus B$ and $B \leq_M A \oplus B$
2. if $A \leq_m C$ and $B \leq_m C$ then $A \oplus B \leq_m C$

Exercise 1.3.2. $K \equiv_1 K_0 \equiv_1 K_1$

Proof. From proof of theorem 1.20, $K \leq_1 A$ for $A = K_1$, con or Inf.
$K_0 \leq K$ for the same reason.
This method only concerns with the latter item. □

Exercise 1.3.3. Prove directly (without Rice's theorem) that $K \leq_1$ Fin

Proof. Let

$$\varphi_{f(x)}(s) = \begin{cases} 0 & x \notin K_s \\ \text{undefined} & x \in K_s \end{cases}$$

where $K_s = W_{e,s}$ for some e s.t. $K = W_e$. If $x \in K$, then $\mathrm{dom}(\varphi_{f(x)})$ is finite □

Exercise 1.3.4. For any x show that $\overline{K} \leq_1 \{y : \varphi_x = \varphi_y\}$ and $\overline{K} \leq_1 \{y : W_x = W_y\}$

Proof. Use the method of exercise 1.3.3. If $x \notin W_x$, then $\mathrm{dom}(\varphi_{f(x)}) = \omega$. □

Exercise 1.3.5. Ext $\neq \omega$

Proof. Use K. If $\psi(x)$ can be extended to a recursive function, then K would be recursive. □

Exercise 1.3.6.

1. Disjoints sets A and B are **recursively inseparable** if there is no recursive set C s.t. $A \subseteq C$ and $C \cap B = \emptyset$. Show that there exists disjoint r.e. sets which are recursively inseparable.
2. Give an alternative proof that Ext $\neq \omega$
3. For A and B as in part 1, prove that $K \equiv_1 A$ and $K \equiv_1 B$

Proof.

1. Consider $A = \{x : \varphi_x(x) = 0\}$ and $B = \{x : \varphi_x(x) = 1\}$. If there is a such recursive set C and its characteristic function is φ_y, then

$$\varphi_y(x) = \begin{cases} 1 & \varphi_x(x) = 0 \\ 1 & \dots \\ 0 & \dots \\ 0 & \varphi_x(x) = 1 \end{cases}$$

hence $\varphi_y(y)$ leads to a contradiction.

2. corollary from 1.
3. The method are the same as 1.20

□

Exercise 1.3.7. A set A is **cylinder** if $(\forall B)[B \leq_m A \Longrightarrow B \leq_1 A]$

1. Show that any index set is a cylinder
2. Show that any set of the form $A \times \omega$ is a cylinder
3. Show that A is a cylinder iff $A \equiv_1 B \times \omega$ for some set B

Proof.

1. If different $x, y \in B$ and $f(x) = f(y)$, we could just add redundent computation and $\varphi_{f(x)} = \varphi_{f(y)}$
2. to make sure images are different by ω
3.

□

Exercise 1.3.8. Show that the partial recursive functions are not closed under μ, i.e., there is a p.r. function ψ s.t. $\lambda x[\mu y[\psi(x, y) = 0]]$ is not p.r.

Proof. $\psi(x, y) = 0$ if $y = 1$ or $y = 0$ and $\varphi_x(x) \downarrow$. □

Exercise 1.3.9. If A is recursive and $B, \overline{B}$ are each $\neq \emptyset$, then $A \leq_m B$

Proof. choose elements $b \in B$ and $b' \in \overline{B}$. Then

$$\psi_{f(x)}(s) = \begin{cases} b & x \in A \\ b' & x \notin A \end{cases}$$

$\square$

Exercise 1.3.10. Prove that Inf $\equiv_1$ Tot $\equiv_1$ Con

Proof. Tot $\equiv_1$ Con is obvious. For Inf $\leq_1$ Con, define

$$\psi(e, x) = \begin{cases} 0 & \text{if } (\exists y > x)[\varphi_e(y) \downarrow] \\ \uparrow & \text{otherwise} \end{cases}$$

$\square$

Exercise 1.3.11. Fin $\leq_1$ Cof

Proof.

$$\varphi_{f(e)}(s) = \begin{cases} \uparrow & \text{if } W_{e,s+1} - W_{e,s} \neq \emptyset \\ 0 & \text{otherwise} \end{cases}$$

$\square$

1.4 Recursive Permutation and Myhill's Isomorphism Theorem

Definition 1.27.

1. A **recursive permutation** is a 1:1, recursive function from ω to ω
2. A property of set is **recursively invariant** if it's invariant under all recursive permutation

Examples:

1. A is r.e.
2. A has cardinality n
3. A is recursive

Properties that not recursively invariant:

1. $2 \in A$
2. A contains the even integers
3. A is an index set

Definition 1.28. A is **recursively isomorphic** to B (written $A \equiv B$) if there is a recursive permutation p s.t. $p(A) = B$

Definition 1.29. The equivalence classes under $\equiv$ are called **recursive isomorphism types**

Theorem 1.30 (Myhill Isomorphism Theorem)**.** $A \equiv B \iff A \equiv_1 B$

Proof. ($\Longrightarrow$) trivial.

($\Longleftarrow$) Let $A \leq_1 B$ via f and $B \leq_1$ via g. We define a recursive permutation h by stages so that $h(A) = B$. We let $h = \bigcup_s h_s$, where $h_0 = \emptyset$ and h_s is that portion of h defined by the end of stage s. Assume h_s is given so that in particular we can effectively check for membership in $\mathrm{dom}(h_s)$ and $\mathrm{ran}(h_s)$ which we both assume finite

Stage $s+1 = 2x+1$. Assume that h_s is $1:1$, $\mathrm{dom}(h_s)$ is finite and $y \in A$ iff $h_s(y) \in B$ for all $y \in \mathrm{dom}(h_s)$.If $h_s(x)$ is defined, do nothing. Otherwise enumerate the set $\{f(x), f(h_s^{-1}f(x)), \ldots, f(h_s^{-1}f)^n(x), \ldots\}$ until the fist element y not yet in $\mathrm{ran}(h_s)$. Define $h_{s+1}(x) = y$. y must exist since f and h_s are $1:1$ and $x \notin \mathrm{dom}(h_s)$

Stage $s+1 = 2x+2$. Define $h^{-1}(x)$ similarly with $f, h_s, \mathrm{dom}()$ and $\mathrm{ran}()$ replaced by $g, h_s^{-1}, \mathrm{ran}(), \mathrm{dom}()$ respectively □

Definition 1.31. A function f **dominates** a function g if $f(x) \geq g(x)$ for almost every (all but finitely many) $x \in \omega$

Exercise 1.4.1 ($\times$). Prove that the primitive recursive permutations do not form a group under composition

Proof. Define $g(x) = \mu y T(e, x, y)$. g dominates all primitive recursive functions since $y \geq U(y)$ for all y. Suppose f is a primitive recursive permutation and $f(g(x)) = x$ if x is even. Note that given y we can primitively recursively compute whether there is an x s.t. $g(x) = y$ □

2 Fundamentals of Recursively Enumerable Sets and the Recursion Theorem

2.1 Equivalent Definitions of Recursively Enumerable Sets