

12 1. Choosing from among **(REC) recursive**, **(RE) re non-recursive**, **(coRE) co-re non-recursive**, **(NRNC) non-re/non-co-re**, categorize each of the sets in a) through d). Justify your answer by showing some minimal quantification of some known recursive predicate.

a.) $\{ f \mid \text{there is a constant } C \text{ such that, for every } x, f(x) \leq C, \}$

NRNC

Justification: $\exists C \forall x \exists t [STP(f,x,t) \& \& VALUE(f,x,t) \leq C]$

If interpret as not requiring convergence then

$\exists C \forall \langle x,t \rangle [STP(f,x,t) \Rightarrow VALUE(f,x,t) \leq C]$

b.) $\{ \langle f,x,c \rangle \mid f(x) \text{ halts in no fewer than } c \cdot x + 1 \text{ steps} \}$

REC

Justification: $\sim STP(f,x,c \cdot x)$

c.) $\{ f \mid \text{range}(f) \subseteq \{0,1\} \}$ // This means range can be $\{\}$, $\{0\}$, $\{1\}$ or $\{0,1\}$ }

co-RE

Justification: $\forall \langle x,t \rangle [STP(f,x,t) \Rightarrow VALUE(f,x,t) \leq 1]$

d.) $\{ f \mid \text{range}(f) \text{ contains at least two elements} \}$

RE

Justification: $\exists \langle x,y,t \rangle [STP(f,x,t) \& STP(f,y,t) \& (VALUE(f,x,t) \neq VALUE(f,y,t))]$

6 2. Let set **A** be recursive, **B** be re non-recursive and **C** be non-re. Choosing from among **(REC) recursive**, **(RE) re non-recursive**, **(NR) non-re**, categorize the set **D** in each of a) through d) by listing **all** possible categories. No justification is required.

a.) $D = \sim A$

REC

b.) $D = B \cap \sim A$

REC, RE

c.) $D \subseteq C$

REC, RE, NR

d.) $D = A - C$

REC, RE, NR

6 3. Prove that the **Uniform Halting Problem** (the set **TOTAL**) is not recursive enumerable within any formal model of computation. (Hint: A diagonalization proof is required here.)

Assume TOTAL is re. As it is a non-empty set, e.g., the index of the function C_0 is in TOTAL, then there must exist a total recursive function that enumerates TOTAL. Call this function A . Thus, the algorithms have indices $A(0), A(1), \dots$. We also know that any complete model of computation must have a universal function. Call it φ . Here $\varphi(f,x)$ is just $\varphi_f(x)$, that is, the function whose index is f evaluated at x .

Define a new function $D(x) = \varphi(A(x),x) + 1$. Clearly D is an algorithm as it just evaluates the x -th algorithm on the input x and then adds 1. As D is an algorithm, its index is in TOTAL and so is the d -th algorithm enumerated by A , for some natural number d .

Consider $D(d)$. $D(d) = \varphi(A(d),d) + 1$ by D 's definition. But, then $D(d) = \varphi(A(d),d) + 1 = \varphi_{A(d)}(d) + 1 = D(d) + 1$

The above is clearly a contradiction since D is an algorithm. This means that TOTAL cannot be re.

- 5 4. Using many-one reduction from the known non-recursive set **HasADouble**, where $\text{HasADouble} = \{ f \mid \exists x \varphi_f(x) = 2^*x \}$, show that **IsDouble** is non-recursive, where $\text{IsDouble} = \{ f \mid \forall x \varphi_f(x) = 2^*x \}$. Just giving a construction is not sufficient; you must also explain why it satisfies the desired properties of the reduction.

Let f be an arbitrary natural number.

*Define $G_{f,x}(y) = 2^*y \cdot \exists \langle x,t \rangle [STP(f,x,t) \& VALUE(f,x,t) = 2^*x]$*

*Thus, $f \in \text{HasADouble}$ iff $\exists x \varphi_f(x) = 2^*x$ iff $\exists \langle x,t \rangle [STP(f,x,t) \& VALUE(f,x,t) = 2^*x]$ iff $\forall y G_{f,x}(y) = 2^*y$ iff $G_{f,x} \in \text{IsDouble}$.*

*This shows $\text{HasADouble} \leq_m \text{IsDouble}$ and so a solution to **IsDouble** implies one to **HasADouble**, which is known to be non-recursive. Thus, **IsDouble** must also be non-recursive.*

5. Define **NullDomain** as $\text{ND} = \{ f \mid \text{for all } x \varphi_f(x) \uparrow \}$.
- 3 a.) Show some minimal quantification of some known recursive predicate that provides an upper bound for the complexity of this set. (Hint: Look at **c.**) and **d.**) to get a clue as to what this must be.)

*$\forall \langle x,t \rangle [\sim STP(f,x,t)]$ shows **ND** is at worst **co-RE***

- 5 b.) Use Rice's Theorem to prove that **ND** is undecidable.

*Clearly $\uparrow \in \text{ND}$, where \uparrow diverges for all input. However, $C_0 \notin \text{ND}$ as it converges everything so has a non-empty domain. Thus, **ND** is non-trivial.*

Let f, g be arbitrary unique natural numbers such that $\text{dom}(f) = \text{dom}(g)$.

$f \in \text{ND} \Leftrightarrow \text{dom}(f) = \emptyset \Leftrightarrow \text{dom}(g) = \emptyset$ iff $g \in \text{ND}$

*This satisfies the second property of one of the weak forms of Rice's Theorem and so **ND** is undecidable.*

- 5 c.) Show that $\text{NotHalt} \leq_m \text{ND}$, where $\text{NotHalt} = \{ \langle f,x \rangle \mid \varphi_f(x) \uparrow \}$. Justify your construction.

Let f, x be an arbitrary pair of natural numbers. Define $G_{f,x}(y) = f(x)$.

We can see that $\text{Dom}(G_{f,x}) = \emptyset$ iff $\varphi_f(x) \uparrow$

Thus, $\langle f,x \rangle \in \text{NotHalt}$ iff $\varphi_f(x) \uparrow$ iff $\forall y G_{f,x}(y) \uparrow$ iff $G_{f,x} \in \text{ND}$

- 5 d.) Show that $\text{ND} \leq_m \text{NotHalt}$. Justify your construction.

Let f be an arbitrary natural number. Define $G_f(y) = \exists \langle x,t \rangle [STP(f,x,t)]$.

$f \in \text{ND}$ iff $\forall \langle x,t \rangle [\sim STP(f,x,t)]$ iff $\sim \exists \langle x,t \rangle [STP(f,x,t)]$ iff $\forall y G_f(y) \uparrow$ iff $\langle G_f, 0 \rangle \in \text{NotHalt}$

Here, we note that G_f is either the constant 1 (converges everywhere) or diverges everywhere.

*Hence we can choose any natural number as the argument to G_f when checking membership in **NotHalt**.*

- 3 e.) From **(a.)** through **(d.)** what can you conclude about the complexity of **ND** (choose from **REC**, **RE**, **RE-MANY-ONE-COMPLETE**, **CO-RE**, **CO-RE-MANY-ONE-COMPLETE**, **NON-RE/NON-CO-RE**)? Briefly justify your conclusion, stating what each of **(a)**, **(b)**, **(c)** and **(d)** show.

ND** is **co-RE Many-One Complete

*a) Shows **ND** is **co-RE**; b) Shows **ND** is non-recursive; c) Shows **ND** is at least as hard as **NotHalt**, which is the complement of **Halt**, a Complete RE m-1 problem; d) adds nothing.*

6. Rice's Theorem has a strong and two weak forms. Given the problem of determining membership in $IsDouble = \{ f \mid \forall x \varphi_f(x) = 2 * x \}$

Show how the strong form can prove this undecidable, but the weak forms cannot. Be sure to cover all conditions that must apply, indicating what is common between the three forms and what is not.

Clearly, $d(x) = 2x \in IsDouble$ However, $C_0 \notin IsDouble$. Thus, $IsDouble$ is non-trivial.

Strong Form:

Let f, g be arbitrary unique natural numbers such that $\forall x \varphi_f(x) = \varphi_g(x)$

*$f \in IsDouble \Leftrightarrow \forall x \varphi_f(x) = 2 * x \Leftrightarrow \forall x \varphi_g(x) = 2 * x$ iff $g \in IsDouble$*

This satisfies the second property of one of the strong forms of Rice's Theorem and so $IsDouble$ is undecidable.

Weak Form #1 (Domains)

Let f, g be arbitrary unique natural numbers such that $dom(\varphi_f) = dom(\varphi_g)$

Consider $\varphi_f(x) = 2x$ and $\varphi_g(x) = 2x-2$. Both have domains of all natural numbers and yet one is in $IsDouble$ and the other is not, so this form does not give us the desired result.

Weak Form #2 (Ranges)

Let f, g be arbitrary unique natural numbers such that $range(\varphi_f) = range(\varphi_g)$

Consider $\varphi_f(x) = 2x$ and $\varphi_g(x) = 2x-2$. Both have ranges of all even natural numbers and yet one is in $IsDouble$ and the other is not, so this form does not give us the desired result.

- 6 7. Let S be an arbitrary infinite re set. Furthermore, let S be the range of some total recursive function f_s . Show that S has an infinite recursive subset enumerated by some monotonically increasing total recursive function g_s . You must give an explicit definition of g_s that you form from f_s . Justify that your g_s enumerates a subset of S and that it is monotonically increasing. The fact that g_s is a monotonically increasing function is sufficient to show the subset it enumerates is infinite and recursive, so you do not have to show those properties.

Define g_s using primitive recursion as follows:

$$g_s(0) = f_s(0)$$

$$g_s(x+1) = f_s(\mu z [f_s(z) > g_s(x)])$$

First, since S is infinite, it has a first element $f_s(0)$ and it has no largest number.

The first property gives us the basis for g_s and the second guarantees that the search, $\mu z [f_s(z) > g_s(x)]$ will always succeed in finding a value in the range of f_s that is greater than all previous values enumerated by g_s .

Clearly the range of g_s is a subset of that of f_s and g_s is monotonically increasing. Thus, the range of g_s is an infinite recursive subset of S .