Computability Exam Solutions

July 13, 2010

Exercise 1

Definition of Unbounded Minimization and Closure Properties

Definition: Given a function $f : \mathbb{N}^{k+1} \to \mathbb{N}$, the unbounded minimization operation $\mu y.f(\vec{x},y)$ produces a function $g : \mathbb{N}^k \to \mathbb{N}$ defined by:

```
g(x⃗) = μy.f(x⃗,y) = {
  the least y such that f(x⃗,y) = 0  if such y exists
  ↑                                   otherwise
}
```

Proof that computable functions are closed under unbounded minimization

Let $f : \mathbb{N}^{k+1} \to \mathbb{N}$ be computable, and define $g(\vec{x}) = \mu y.f(\vec{x},y)$.

Since f is computable, there exists a URM program P_f that computes f.

Algorithm to compute g:

```
On input x⃗:
1. Initialize y = 0
2. Loop:
   a. Compute f(x⃗,y) using program P_f
   b. If f(x⃗,y) = 0, return y and halt
   c. Otherwise, increment y and continue
```

This algorithm terminates and returns the correct value if $\exists y$ such that $f(\vec{x},y) = 0$, and diverges otherwise (which is the correct behavior for μ).

Since the algorithm uses only computable operations, g is computable.

Are total computable functions closed under minimization?

Answer: No.

Counterexample: Define $f : \mathbb{N}^2 \to \mathbb{N}$ by:

```
f(x,y) = {
  0  if x ∈ K and y = 0
  1  otherwise
}
```

where K is the halting set.

The function f is total and computable since:

- We can check if $y = 0$ (decidable)
- We can semi-decide if $x \in K$, and if it fails, return 1
- If $x \in K$ and $y = 0$, return 0

Now consider $g(x) = \mu y.f(x,y)$:

```
g(x) = {
  0  if x ∈ K
  ↑  if x ∉ K
}
```

The function g is partial (not total) since it diverges for $x \notin K$.

Therefore, applying minimization to a total computable function can yield a partial function, so total computable functions are not closed under unbounded minimization.

Exercise 2

Question: Does there exist a non-computable f such that $D = \{x \in \mathbb{N} \mid f(x) \neq \varphi_x(x)\}$ is finite?

Answer: No, such a function cannot exist.

Proof:

Suppose f is non-computable and $D = \{x : f(x) \neq \varphi_x(x)\}$ is finite.

Let $D = \{d_1, d_2, \ldots, d_k\}$ be the finite set where f differs from the diagonal.

For all $x \notin D$, we have $f(x) = \varphi_x(x)$.

Construction of computable function agreeing with f:

Define $g : \mathbb{N} \to \mathbb{N}$ by:

```
g(x) = {
  f(dᵢ)      if x = dᵢ for some i ∈ {1,...,k}
  ϕₓ(x)      if x ∉ D
}
```

Since D is finite, we can hardcode the values $f(d_1), \ldots, f(d_k)$. For $x \notin D$, we compute $\varphi_x(x)$.

Analysis:

- $g(x) = f(x)$ for all $x \in D$ (by construction)
- $g(x) = \varphi_x(x) = f(x)$ for all $x \notin D$ (since $f(x) = \varphi_x(x)$ when $x \notin D$)

Therefore $g(x) = f(x)$ for all $x \in \mathbb{N}$, so $g = f$.

Computability of g: For any input x:

1. Check if $x \in \{d_1, \ldots, d_k\}$ (decidable since D is finite)
2. If yes, return the precomputed value $f(d_i)$
3. If no, compute and return $\varphi_x(x)$

The function g is computable since:

- Membership in finite sets is decidable
- $\varphi_x(x)$ is computable (universal function)
- Finite case analysis is computable

This contradicts the assumption that f is non-computable.

Therefore, no such non-computable function f can exist.

Exercise 3

Classification of $A = \{x \in \mathbb{N} \mid \varphi_x(y) = x \cdot y$ for some $y\}$

A is r.e.:

```
sc_a(x) = 1(μ⟨y,t⟩. S(x,y,x·y,t))
```

This searches for y,t such that $\varphi_x(y) = x \cdot y$ in exactly t steps.

A is not recursive: We can show this using Rice's theorem or by reduction. The set A is saturated since it expresses a property of functions: "the function outputs $x \cdot y$ for some input y."

By Rice's theorem, since A is saturated and non-trivial ($A \neq \emptyset$ and $A \neq \mathbb{N}$), A is not recursive.

To see $A \neq \emptyset$: The function $\varphi_e(y) = e \cdot y$ (multiplication by constant e) satisfies $\varphi_e(1) = e \cdot 1 = e$, so $e \in A$.

To see $A \neq \mathbb{N}$: The everywhere undefined function $\emptyset$ cannot output anything, so its index is not in A.

$\bar{A}$ is not r.e.: Since A is r.e. but not recursive, $\bar{A}$ is not r.e.

Final classification: A is r.e. but not recursive; $\bar{A}$ is not r.e.

Exercise 4

Classification of $B = \{x \in \mathbb{N} : E_x \not\subseteq W_x\}$

This is equivalent to $B = \{x : \exists y \in E_x$ such that $y \notin W_x\}$.

B is r.e.:

```
scB(x) = 1(μ⟨y,z,t,s⟩. S(x,z,y,t) ∧ ∀u≤s ¬H(x,y,u))
```

This searches for evidence that some $y \in E_x$ is not in W_x (by finding y in the codomain but not finding it in the domain within some time bound).

Actually, this is tricky because proving $y \notin W_x$ requires showing φ_x never halts on y, which is undecidable.

B is not r.e.: The condition $E_x \nsubseteq W_x$ requires proving that some element of E_x never appears in W_x, which involves proving non-termination.

We can show $\bar{K} \leq_m \bar{B}$. If $x \notin K$, then we can construct an index where $E_x \subseteq W_x$. If $x \in K$, we can construct an index where $E_x \nsubseteq W_x$.

$\bar{B}$ is r.e.: $\bar{B} = \{x : E_x \subseteq W_x\}$ can be semi-decided by verifying that every element that appears in E_x eventually appears in W_x.

```
scB̄(x) = lim_{t→∞} [∀y(∃z,s≤t S(x,z,y,s) → ∃u≤t H(x,y,u))]
```

Final classification: B is not r.e.; $\bar{B}$ is r.e. but not recursive.

Exercise 5

Theorem: $A \neq \emptyset$ is r.e. $\iff \exists f : \mathbb{N} \to \mathbb{N}$ with dom(f) = Pr and img(f) = A

where Pr is the set of prime numbers.

Proof:

(⇒) If A is r.e. and $A \neq \emptyset$, then such f exists

Since A is r.e., there exists a computable enumeration $g : \mathbb{N} \to A$ (possibly with repetitions).

Since Pr is infinite and $A \neq \emptyset$, we can define $f : \mathbb{N} \to \mathbb{N}$ by:

```
f(x) = {
  g(π⁻¹(x))  if x ∈ Pr, where π: ℕ → Pr is the enumeration of primes
  ↑          if x ∉ Pr
}
```

More precisely:

- Let $p_0, p_1, p_2, \ldots$ be the enumeration of primes
- Let $h : \mathbb{N} \to \mathbb{N}$ be the function with $h(p_i) = i$ and h(x) undefined for non-primes
- Define f(x) = g(h(x))

Then dom(f) = Pr and img(f) = img(g) = A.

Since membership in Pr is decidable and g is computable, f is computable.

(⇐) If such f exists, then A is r.e.

Given $f : \mathbb{N} \to \mathbb{N}$ with $dom(f) = Pr$ and $img(f) = A$, we have $A = img(f)$.

Since f is computable (we can check if $x \in Pr$ and compute $f(x)$ if so), and A is the image of a computable function, A is r.e.

Conclusion: $A \neq \emptyset$ is r.e. $\iff \exists$ computable f with $dom(f) = Pr$ and $img(f) = A$.