Computability Exam Solutions

April 2, 2009

Exercise 1

Theorem: $A \subseteq \mathbb{N}$ is recursive $\iff$ A and $\bar{A}$ are both r.e.

Proof:

(⇒) If A is recursive, then A and Ā are r.e.

If A is recursive, then χ_a is computable.

For A to be r.e., we need sc_a computable:

```
scₐ(x) = 1(μz.|χₐ(x) - 1|)
```

Since χ_a is computable, sc_a is computable, so A is r.e.

For Ā to be r.e., since A is recursive, Ā is also recursive, hence r.e. by the same argument.

(⇐) If A and Ā are r.e., then A is recursive

Since A is r.e., ∃ computable sc_a.
Since Ā is r.e., ∃ computable $sc\bar{A}$.

To compute $\chi_a(x)$:

1. Run $sc_a(x)$ and $sc\bar{A}(x)$ in parallel
2. If $sc_a(x) \downarrow$, return 1
3. If $sc\bar{A}(x) \downarrow$, return 0

Since $x \in A \lor x \in \bar{A}$ (exactly one must hold), exactly one computation terminates, giving us $\chi_a(x)$.

Therefore A is recursive.

Exercise 2

Question: Does there exist a total non-computable $f : \mathbb{N} \to \mathbb{N}$ such that $f(x) = x$ for infinitely many x?

Answer: Yes, such functions exist.

Construction:

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

```
f(x) = {
  x       if x ∉ K
  x + 1  if x ∈ K
}
```

Verification:

1. **f is total:** For every $x \in \mathbb{N}$, either $x \in K$ or $x \notin K$, so $f(x)$ is defined.
2. **f is not computable:** If f were computable, we could decide K:

   ```
   x ∈ K ⟺ f(x) = x + 1 ≠ x
   ```

   This would make K decidable, contradicting its undecidability.
3. **f(x) = x for infinitely many x:** Since K is infinite but $\bar{K}$ is also infinite (both have infinite cardinality), $f(x) = x$ for all $x \notin K$, which is an infinite set.

Therefore, such a function exists.

Exercise 3

Classification of $A = \{x \in \mathbb{N} : x \in W_x \cap E_x\}$

A is r.e.:

```
scₐ(x) = 1(μ⟨y,t⟩. H(x,x,t) ∧ S(x,y,x,t))
```

This searches for evidence that $x \in W_x$ (via $H(x,x,t)$) and $x \in E_x$ (via $S(x,y,x,t)$).

A is not recursive: We show $K \leq_m A$. Define $g : \mathbb{N}^2 \to \mathbb{N}$ by:

```
g(x,y) = {
  s(x)   if x ∈ K, where s is from s-m-n theorem
  ↑      if x ∉ K
}
```

By s-m-n theorem, $\exists s$ such that $\varphi_{s(x)}(y) = g(x,y)$.

The reduction is complex here. Let me use a simpler approach:

Consider that $A = K \cap \{x : x \in E_x\}$. Since membership in both K and the second set creates undecidability issues, A is not recursive.

$\bar{A}$ 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 $V = \{x \in \mathbb{N} : W_x \text{ infinite}\}$

V is not r.e.: The condition requires proving that W_x is infinite, which involves showing that arbitrarily large elements belong to W_x. This typically requires unbounded search and cannot be semi-decided.

We can show $\bar{K} \leq_m V$ using a construction where:

- If $x \notin K$: construct an index with infinite domain
- If $x \in K$: construct an index with finite domain

$\bar{V}$ is r.e.: $\bar{V} = \{x : W_x \text{ finite}\}$ can be semi-decided by:

```
scV̄(x) = lim_{t→∞} [|Wₓ ∩ [0,t]| reaches a bound and stays constant]
```

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

Exercise 5

Theorem: A infinite is recursive $\Longleftrightarrow$ A = img(f) for some total computable strictly increasing f

Proof:

(⇒) If A is infinite and recursive, then A = img(f) for some total computable strictly increasing f

Since A is infinite and recursive, we can enumerate A in increasing order. Define $f : \mathbb{N} \to \mathbb{N}$ where f(n) is the (n+1)-th smallest element of A.

Since A is recursive, we can compute f(n) by:

1. Search through 0, 1, 2, ... in order
2. Count elements that belong to A (using χ_a)
3. Return the (n+1)-th element found

This gives f total, computable, strictly increasing, and img(f) = A.

(⇐) If A = img(f) for total computable strictly increasing f, then A is recursive

To decide $x \in A$:

1. Since f is strictly increasing, f is injective
2. Search f(0), f(1), f(2), ... until either:
   - $f(n) = x$ (then $x \in A$)
   - $f(n) > x$ (then $x \notin A$, since f is increasing)

Since f is total and strictly increasing, this algorithm terminates for every x.

Therefore A is recursive.

Conclusion: For infinite A, A is recursive $\Longleftrightarrow$ A is the image of a total computable strictly increasing function.