<u>Lessons touched by this meeting according to schedule:</u>

- 10. 11/11/2024
  - The diagonal method [§4.3]
- 11. 12/11/2024
  - The smn theorem (or parametrization theorem) [§4.4]
  - Universal function (some ideas)

Every argument is an *argument* **for** something. The Cantor diagonal argument is an argument to prove that set of real numbers is uncountable.

What is a countable set? Let's say a set is countable if we can start ordering the elements of a set like the first, the second and so on. Formally we have to find a bijection with natural numbers.

To prove that reals are uncountable we first assume the contrary, namely that set of reals is countable. Then we have to find a contradiction, rendering the assumption false. To do that we find a real number which is not counted. Cantor diagonal argument construct such a real number which is not counted.

So here are the steps:

Goal: Set of real numbers is uncountable.

Step 1: Assume that the set is countable. This means that the set of real numbers can be written as a set with first element, second element and so on, which is $\{r_1, r_2, r_3, \ldots\}$.

Step 2: One way to show that the assumption of step 2 is not possible is to find a real number which is not counted there. How? By Cantor diagonal argument.

Suppose that we are going to consider only numbers between 0 and 1. Then the new number is such that it is different from the first number at the first digit, from the second element at the second digit and so on. For instance look at the following:

$$
\begin{array}{cccccc}
0. & 9 & 7 & 0 & 6 & \ldots \\
0. & 8 & 2 & 4 & 3 & \ldots \\
0. & 4 & 5 & 2 & 8 & \ldots \\
0. & 1 & 2 & 5 & 3 & \ldots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ldots \\
\hline
\\
0. & 8 & 1 & 1 & 4 & \ldots
\end{array}
$$

<u>idea</u>: $x_i \quad i \in I$

$x_0 \quad x_1 \quad x_2 \quad x_3 \quad \ldots$

$\hookrightarrow x \neq x_i \quad \forall i \in I$ $\quad$ x differs from x_i "at position i"

Corollary 10.3. *The set $\bar{\mathcal{C}} = \{f : \mathbb{N} \to \mathbb{N} \mid f$ not computable$\}$ is not countable.*

Proof. We know that $|\mathcal{C}| = |\mathbb{N}|$. If $\bar{\mathcal{C}}$ were countable, then $\mathbb{N} \to \mathbb{N} = \mathcal{C} \cup \bar{\mathcal{C}}$ would be countable, which is absurd for the previous corollary. $\square$

Observation 10.4. There exists a total non-computable function $f : \mathbb{N} \to \mathbb{N}$ defined by

$$f(n) = \begin{cases} \varphi_n(n) + 1 & \text{if } \varphi_n(n) \downarrow \\ 0 & \text{if } \varphi_n(n) \uparrow \end{cases}$$

f is not computable because it differs from all computable functions. In fact

- if $\varphi_n(n) \downarrow$, then $f(n) = \varphi_n(n) + 1 \neq \varphi_n(n)$
- if $\varphi_n(n) \uparrow$, then $f(n) = 0 \neq \varphi_n(n)$

so

$$\forall n \; f \neq \varphi_n$$

Hilbert hotel parallel: <u>here</u> – (useful)

Exercise 6.6(p). Say if there can be a non-computable function $f : \mathbb{N} \to \mathbb{N}$ such that for any other non-computable function $g : \mathbb{N} \to \mathbb{N}$ the function $f + g$ defined by $(f + g)(x) = f(x) + g(x)$ is computable. Justify your answer (providing an example of such f, if it exists, or proving that cannot exist).

Suppose there exists a non-computable function f : N → N with the property that for any non-computable function g : N → N, the function f + g defined by (f + g)(x) = f(x) + g(x) is computable.

Since the quantification over g is universal, this property must also hold when g = f. That is, taking g = f, we would have that f + f = 2f is computable.

However, if 2f is computable, then f itself would be computable since:

f(x) = (2f(x))/2

This is a computable operation (division by 2) applied to a computable function (2f), which would make f computable.

This contradicts our initial assumption that f is non-computable.

Therefore, we conclude that no such non-computable function f can exist. □

The key insight here is that if f has the property that its sum with any non-computable function is computable, then in particular it must have this property when added to itself. But this leads to a contradiction since it would make f computable.

Exercise 6.20. Consider the function $f : \mathbb{N} \to \mathbb{N}$ defined by

$$f(x) = \begin{cases} x + 2 & \text{if } \varphi_x(x) \downarrow \\ x \dot{-} 1 & \text{otherwise} \end{cases}$$

Is it computable? Justify your answer.

The function f is not computable. We can prove this by showing that if f were computable, then K = {x | ϕx(x)↓} would be recursive.

Suppose f is computable. Then we can write: χK(x) = sg(f(x) - (x+2))

This is because:

- If x ∈ K, then ϕx(x)↓, so f(x) = x+2, thus χK(x) = sg(0) = 0
- If x ∉ K, then ϕx(x)↑, so f(x) = x-1, thus χK(x) = sg(-(3)) = 1

Since this would express χK as a composition of computable functions (sg and f), it would make K recursive. However, we know that K is not recursive.

Therefore, f cannot be computable. □

Kleene's smn-theorem

Given $m, n \geq 1$ there is a total computable function $s_{m,n}: \mathbb{N}^{m+1} \to \mathbb{N}$ such that $\forall \vec{x} \in \mathbb{N}^m, \forall \vec{y} \in \mathbb{N}^n, \forall e \in \mathbb{N}$

$$\phi_e^{(m+n)}(\vec{x}, \vec{y}) = \phi_{s_{m,n}(e,\vec{x})}^{(n)}(\overrightarrow{y})$$

The SMN theorem states that given a function g(x, y) which is computable, there exists a total and computable function s such that phi_s(x) (y) = g(x, y), basically "fixing" the first argument of g. It's like partially applying an argument to a function.

Usually you use it to create a reduction function by first finding an appropriate g(x, y) and then using the SMN theorem to say that there exists the previously cited function s which is also the reduction function. The difficult part is finding the appropriate function g(x, y), then the application of the SMN theorem is always the same.

In computability theory the S_n^m **theorem**, written also as "**smn-theorem**" or "**s-m-n theorem**" (also called the **translation lemma**, **parameter theorem**, and the **parameterization theorem**) is a basic result about programming languages (and, more generally, Gödel numberings of the computable functions) (Soare 1987, Rogers 1967). It was first proved by Stephen Cole Kleene (1943). The name S_n^m comes from the occurrence of an *S* with subscript *n* and superscript *m* in the original formulation of the theorem (see below).

The proof works by showing how to construct a new URM program P' from an original program Pe that computes ϕe(m+n).

```
Original Program Pe:
- Takes inputs in registers R1,...,Rm+n
- Produces output in R1

New Program P':
- Takes only n inputs
- Has m fixed values hardcoded
- Produces same output in R1
```

```
P' performs these steps:
1. Move n inputs forward m positions
2. Load the m fixed values into first m registers
3. Execute the original program Pe
```

```
s(m,n)(e,x̄) = γ(P')
where:
- γ is the program encoding function
- P' is the constructed program
```

Corollary 11.4 (Simplified *smn* theorem). *Let* $f : \mathbb{N}^{m+n} \to \mathbb{N}$ *be a computable function. There exists a total computable function* $s : \mathbb{N}^m \to \mathbb{N}$ *such that*

$$f(\vec{x}, \vec{y}) = \varphi_{s(\vec{x})}^{(n)}(\vec{y}) \qquad \forall x \in \mathbb{N}^m \quad \forall y \in \mathbb{N}^n$$

Exercise 3.3. State the smn theorem and use it to prove that there exists a total computable function $s : \mathbb{N}^2 \to \mathbb{N}$ such that $W_{s(x,y)} = \{z : x * z = y\}$

1) First, let's state the smn theorem:

For any m,n ≥ 1, there exists a total computable function s(m,n): Nm+1 → N such that for all e ∈ N, x̄ ∈ Nm, ȳ ∈ Nn:

$\phi(m+n)e(\bar{x},\bar{y}) = \phi(n)s(m,n)(e,\bar{x})(\bar{y})$

2) To solve our problem, let's first define a helper function f: N3 → N:

```
f(x,y,z) = { 0 if x * z = y
           ↑ otherwise }
```

or equivalently:

```
f(x,y,z) = μw.|x*z - y|
```

3) This function f is computable since:

- Multiplication is computable
- Absolute value is computable
- Bounded minimization of computable functions is computable

4) By the smn theorem (using m=2, n=1), there exists a total computable function s: N2 → N such that: For all x,y,z ∈ N: ϕs(x,y)(z) = f(x,y,z)

5) Let's verify that Ws(x,y) = {z : x * z = y}:

- z ∈ Ws(x,y) ⟺ ϕs(x,y)(z)↓ ⟺ f(x,y,z)↓ ⟺ x * z = y

Therefore, s is the required function. □

Exercise (2019-11-18-solved)

State the smn-theorem and use it to show there exists a total computable function $k: \mathbb{N} \to \mathbb{N}\ s.t. \forall n \in \mathbb{N}$, $\phi_{k(n)}$ is total and $E_{k(n)}$ is the set of integer divisors of n.

Solution

The smn-theorem states that, given $m, n \geq 1$ there is a computable total function $s_{m,n}: \mathbb{N}^{m+1} \to \mathbb{N}$ $s.t. \forall e \in \mathbb{N}, \vec{x} \in \mathbb{N}, \vec{y} \in \mathbb{N}^n$

$$\phi_e^{m+n}(\vec{x},\vec{y}) = \phi_{s_{m,n}(e,\vec{x})}^{(n)}(\vec{y})$$

We define a function $g: \mathbb{N} \to \mathbb{N}$ in which we can define:

$$g(n,x) = \begin{cases} x * n, & x\ is\ a\ divisor\ of\ n \\ 1, & otherwise \end{cases}$$

This is computable, given:

$$g(n,x) = (x * n) * \overline{sg}(rm(x,n)) + sg(rm(x,n))$$

For the smn-theorem, there exists a function $k: \mathbb{N} \to \mathbb{N}$ s.t. $\phi_{k(n)}(x) = f(n,x)\ \forall n, x \in \mathbb{N}$. So, as desired:

- $W_{k(n)} = \mathbb{N}\ (total)$
- $E_{k(n)} = \{x \mid rm(x,n) = 0\} \cup \{1\}$, set of divisors and 1 which is always a divisor for n

Inside of the tutoring, the notation with $\varphi_{s(x)}(y)$ was used, hence $g(x,y)$, which is the usual notation we have.

Why this choice of the helper function?

- When x divides n, we put x*n in the output set
- When x doesn't divide n, we put 1 (ensuring function is total)

Exercise (2017-11-20)

State the smn-theorem. Use it for proving there exists $k: \mathbb{N} \to \mathbb{N}$ total and computable s.t. $\forall n \in \mathbb{N}$ we have $W_{k(n)} = \{x \in \mathbb{N} \mid x \geq n\}$ and $E_{k(n)} = \{y \in \mathbb{N} \mid y\ even\}$ for all $n \in \mathbb{N}$.

Solution

The smn-theorem states that, given $m, n \geq 1$ there is a computable total function $s_{m,n}: \mathbb{N}^{m+1} \to \mathbb{N}$ $s.t. \forall e \in \mathbb{N}, \vec{x} \in \mathbb{N}^m, \vec{y} \in \mathbb{N}^n$

$$\phi_e^{m+n}(\vec{x}, \vec{y}) = \phi_{s_{m,n}(e,\vec{x})}^{(n)}(\vec{y})$$

Define:

$$f(n,x) = \begin{cases} 2 * (x \dot{-} n) & \text{se } x \geqslant n \\ \uparrow & \text{altrimenti} \end{cases} = 2 * (x \dot{-} n) + \mu z.(n \dot{-} x)$$

By the smn-theorem, there exists a total and computable function $k: \mathbb{N} \to \mathbb{N}$ s.t. $\phi_{k(n)}(x) = f(n,x)\ \forall n, x \in \mathbb{N}$. So, as desired:

- $W_{k(n)} = \{x \mid f(n,x) \downarrow\} = \{x \mid x \geqslant n\};$
- $E_{k(n)} = \{f(n,x) \mid x \in \mathbb{N}\} = \{2(x \dot{-} n) \mid x \geqslant n\} = \{2(n+z) \dot{-} n) \mid z \geqslant 0\} = \{2z \mid z \in \mathbb{N}\}.$

Why this design of the helper function?

- When x ≥ n: outputs 2*(x-n), which:
  - Controls domain through x ≥ n condition
  - Ensures even outputs by multiplying by 2
- When x < n: undefined (↑)

We then make the function computable - this works because:

- μz.(n-x) will be defined only when x ≥ n
- 2*(x-n) ensures outputs are even

Concrete example on how we use the smn-theorem (this we'll be explored in detail when it will be relevant):

Exercise 8.17. Study the recursion of the set $A = \{x \in \mathbb{N} : x \in W_x \wedge \varphi_x(x) = x^2\}$, i.e., establish if A and $\bar{A}$ are recursive/recursive enumerable.

Solution: We show that $K \leqslant A$, and thus A is not recursive. Define

$$g(x,y) = \begin{cases} y^2 & \text{if } x \in K \\ \uparrow & \text{otherwise} \end{cases}$$

The function $g(x,y)$ is computable, since

$$g(x,y) = y^2 \cdot sc_K(x)$$

Thus by the smn theorem, there exists a total computable function $s : \mathbb{N} \to \mathbb{N}$ such that for each $x, y \in \mathbb{N}$

$$\varphi_{s(x)}(y) = g(x,y)$$

The function s is a reduction function of K to A. Indeed

- if $x \in K$ then $\varphi_{s(x)}(y) = g(x,y) = y^2$ for each $y \in \mathbb{N}$. Therefore $s(x) \in W_{s(x)} = \mathbb{N}$ and $\varphi_{s(x)}(s(x)) = s(x)^2$. Thus $s(x) \in A$.
- if $x \notin K$ then $\varphi_{s(x)}(y) = g(x,y) \uparrow$ for each $y \in \mathbb{N}$. Therefore $s(x) \notin W_{s(x)} = \varnothing$. Thus $s(x) \notin A$.

Furthermore, A is r.e., since its semi-characteristic function

$$sc_A(x) = \mathbf{1}(\mu w.\ |x^2 - \varphi_x(x)|) = \mathbf{1}(\mu w.\ |x^2 - \Psi_U(x,x)))|$$

is computable. Therefore $\bar{A}$ not r.e. and thus it is not recursive. □

2.1.15 What is the universal function and how to use it?

Consider this is <u>always computable</u> and one can say ϕ_x is computable for program e: $\Psi_U(e, \vec{x}) = \phi_e^k(\vec{x})$

From what I got looking at the exercises solutions, it is a very limited case:

- basically, when you write semicharacteristic functions, so you are doing r.e./not r.e. exercises considering exercises with $\phi_x(x)$ inside the exercise definition, there is:
  - $sc_A(x) = (\ldots - \ \phi_x(x))$
- you simply replace ϕ_x with $\Psi_U(x,x)$
  - $sc_A(x) = (\ldots - \ \Psi_U(x,x))$

Consider also (to make you understand how to write this):

- $\phi_x(z) = \Psi_U(x,z)$
- $\phi_y(z) = \Psi_U(y,z)$