Revision

The big picture and key concepts

What a computer can/can’t solve

- Church-Turing thesis: Every problem for which you can describe a solution by a step-by-step process (with no magic) >>> that problem is solvable by a machine

- Starting tools: zero function (delete/nothingness), successor (+1/make something), projection (pick a cell/choose something)

- Building tools: Recursion (loops), Composition, Minimalization(search)

Inputs/Outputs

- For a machine to perform instructions, it needs a way to refer to inputs/objects by names the machine can understand/save/process
- We can capture every finite object with a finite string of symbols from a fixed finite alphabet Σ
- We denote the set of strings by Σ^*
- One can also think of every finite string as a natural number (e.g. Godel Numbering)
- Machines themselves can be given names/numbers and so can be thought of as strings on their own, and can be referred to

Problems

- Functional: compute/build something (add, multiply, etc.)
- Decision: output Yes/No
- Every problem which we can describe in machine terms can be regarded as a decision problem
- Decision problems are enough to manifest what computers can/can't do, and what computers can do efficiently/not efficiently
- Decision problems capture the theory of computation well

Languages

- A language is a set of finite strings (a subset of Σ^*)
- Every decision problem is a membership question about some language

The halting problem (many versions, same idea)

- Remember: machines can be thought of as strings/numbers
- Consider the following language HP:

$$\{(M, w): M\ halts\ on\ input\ w\}$$

- This language cannot be decided by a computer (undecidable). First example we found of **not computable**.
- Why? A diagonalization style argument. Assume it is decidable by some machine *D* and then a contradiction can be found

Inspired by the Halting Problem

- Consider the following language $\overline{HP}$:
$$\{(M, w) : M\ does\ not\ halt\ on\ input\ w\}$$

(This is the complement of the set on the previous page if we assume that every natural number corresponds to a TM. Remember Q4A4?)

- $\overline{HP}$ is also not decidable

HP vs $\overline{HP}$

- Both are not decidable but they are very different in computationally

- HP is recognizable but $\overline{HP}$ isn’t (fist example of not recognizable)

- Recognizable is the same as Computably Enumerable

- $\overline{HP}$ is an example of what we call co-c.e.

So, The non-computable isn’t all one thing

- We just stepped into the non-computable and once we did we found two computationally different examples
- Is there a room to step further? To rocket higher in that sky?
- Yes, Aliens (non-computable oracles) can pull us higher into the world of the non-computable

Who’s the first alien we met?

- HP
- For a nice story, let A be an alien that knows how to decide HP
- A has a secret processor which they can integrate with our machines to compute and make decisions our machines can’t do on their own
- BUT, we are intelligent humans who knew how to challenge A

Hey A, try this

- Bring your enhanced machines
  $\{M^A\colon M$ a machine of ours which is designed to allow using help$\}$

- Consider this language:
  $$\{(M^A, w)\colon M^A \text{ halts on input } w\}$$

- We call this language the (first) jump of HP.

- The alien A fails to decide this language with all their fancy alien computers

A bigger alien was just found

- The alien A' which can decide
$$HP^A = \{(M^A, w) : M^A \text{ halts on input } w\}$$

- Note that we also have
$$\overline{HP}^A = \{(M^A, w) : M^A \text{ doesn't halts on input } w\}$$

- With our rich imagination as humans we can keep going higher and higher in that sky, discovering pairs of computationally different sets made by aliens (each alien level needs a jump)

Logic and Aliens

- Try to describe what is happening in every alien set we explored so far
- Example: describe "$M\ halts\ on\ input\ w$“
- You will need $\exists$
- And if you try to describe "$M\ does\ not\ halt\ on\ input\ w$“ you will need $\forall$

$\exists, \forall$

- To describe "M^A $halts\ on\ input\ w$" we will say:
  $\exists s\ M^A(w)$ halts within s steps
- The part " $M^A(w)$ halts" involves decisions about *HP* depending on the design of *M*
- Those internal decisions are either:
  $\exists t$ within t steps of enumerating HP we know something is in HP
  $\forall t,$ after t steps of enuemerating HP something is not enumerated in *HP*
- So eventually, "M^A $halts\ on\ input\ w$" can be described as
  $\exists s \forall t\ computable\ decision\ invloving\ s, t$
- This is how we got that hierarchy of Pi's and Sigmas

Related observation

- A language is decidable by alien A if and only if its complement problem is also decidable by the same alien A
- Special case: a set is computable if and only if its complement is computable
- Being computable can be considered as having a useless (computable) alien.
- Every alien level is identified by a language

Reducibilities

- Those are the communication tools languages whether alien (non-computable) or earthy (computable)

- Weak kind: Turing Reducibility
- Strong kind: m-reducibility
- Stronger kind: 1-reducibility

Let’s focus on the earth level (computable world)

- Let our concern be decidable (computably) languages
- These are some which we know we can decide quickly, and others we can decide but in longer time
- In this computable world, the reducibilities on the previous page can be slow communication or fast communication.
- Fast communication is the interesting one because it does not affect the speed with which a problem is decided (in case it uses the help of another problem)

You know the rest

That was the big picture

Techincalities

When reducing a problem to another

1. Know your super languages:

Example: Yousef’s Q4A4

$LOOP = \{(M, w_1, w_2, w_3): M\ is\ a\ TM\ that\ doesn't\ halt\ on\ at\ least\ 2\ of\ the\ w_i's\}$

$\overline{HP} = \{(M, w): M\ does\ not\ halt\ on\ input\ w\}$

We have:

$LOOP \subseteq \{(M, w_1, w_2, w_3): M \text{ is a TM and the } w_i's \text{ are strings}\}$ (call it S_{Loop})
$\overline{HP} \subseteq \{(M, w): M \text{ a TM}, w\ s\text{tring in the language } M \text{ deals with}\}$ (call it $S_{\overline{HP}}$)

Suppose we are looking at $LOOP \leq_m \overline{HP}$

- The reduction is a TM/computable function $f: S_{Loop} \rightarrow S_{\overline{HP}}$
- $(\forall s \in S_{Loop})(s \in LOOP \iff f(x) \in \overline{HP})$
- Add the requirement that f is polytime if looking at $\leq_p$
- Or, add the requirement that f is injective if looking at $\leq_1$

Suppose we are looking at $LOOP \leq_T \overline{HP}$

- Here we describe a machine M_{LOOP} that has domain S_{Loop}
- M_{LOOP} is supposed to decide $LOOP$ **using the help** of another machine $M_{\overline{HP}}$
- $M_{\overline{HP}}$ has domain $S_{\overline{HP}}$ and it decides $\overline{HP}$
- If we are looking into $\leq_T^p$, then we require in addition that M_{LOOP} communicates with $M_{\overline{HP}}$ in polynomial time