5CCS2FC2: Foundations of Computing II

(Beyond) The Limits of Computation

Week 11

Dr Christopher Hampson

Department of Informatics

King's College London

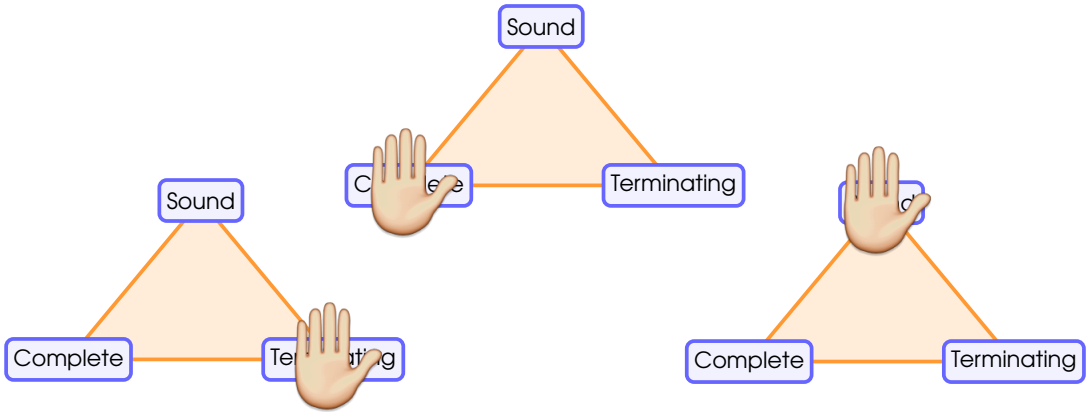
Computable Enumerability

Computable Enumerability

Computable Enumerability

- **Computable (Recursive) Enumerability**
  - A language L is said to be **computably enumerable (C.E.)** if there is *some* machine M that is:
    - **Sound** — If $M(w) = 1$ then $w \in L$
    - **Complete** — If $w \in L$ then $M(w) = 1$

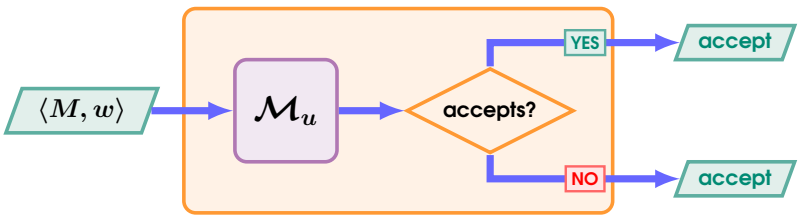
Computable Enumerability



Theorem The Halting problem $\mathbf{HALT}_{TM}$ is computably enumerable.

Proof:

Step 1) Construct the following **sound** and **complete** machine $\mathbf{halt}^{-}$:



Q.E.D.

Proof:

Step 2) Note the following relationship

$$\textbf{halt}^{-}(\langle \textbf{code}(M), w\rangle) = 1 \iff \begin{array}{r} \mathcal{M}_u(\langle \textbf{code}(M), w\rangle) = 1 \\ \text{or} \quad \mathcal{M}_u(\langle \textbf{code}(M), w\rangle) = 0 \end{array}$$

Step 3) By the definition of our **Universal Turing Machine** we have

$$\textbf{halt}^{-}(\langle \textbf{code}(M), w\rangle) = 1 \iff M(w) = 1 \quad \text{or} \quad M(w) = 0$$

Proof:

Step 4) By **definition** of $\mathbf{HALT}_{TM}$ we have that

$$\mathbf{halt}^{-}(\langle \mathbf{code}(M), w \rangle) = 1 \iff \langle \mathbf{code}(M), w \rangle \in \mathbf{HALT}_{TM}$$

- The **right-to-left** direction is **soundness**
- The **left-to-right** direction is **completeness**

Computable Enumerability

Theorem The *non*-Emptiness problem $\overline{\mathbf{E}_{TM}}$ is C.E.

Step 1) Let

$$w_0, \quad w_1, \quad w_2, \quad w_3, \quad w_4, \quad \ldots$$

be an **enumeration** of all possible input words from Σ^*.

Step 2) Construct the following algorithm $\mathbf{empty}^{-}$

$\mathbf{empty}^{-}(\mathbf{code}(M))$:

1. **For each** $i = 0, 1, 2, 3 \ldots$:
   2. Run $\mathcal{M}_u$ on $\langle \mathbf{code}(M), w_i \rangle$.
   3. **If** $\mathcal{M}_u$ accepts: **return** True

Sound ✓ **Complete** ✗

Step 2) Construct the following algorithm $\mathbf{empty}^{-}$

$\mathbf{empty}^{-}(\mathbf{code}(M))$ **:**

1. **For each** $k = 0, 1, 2, 3, \ldots$:
   2. **For each** $i = 0, 1, 2, \ldots, k$:
      3. Run $\mathcal{M}_u$ on $\langle \mathbf{code}(M), w_i \rangle$ for k steps.
      4. **If** $\mathcal{M}_u$ accepts: **Return** True

Sound ✓ **Complete** ✓

Proof:

Step 3) Note the following relationship

$$\mathbf{empty}^{-}(\mathbf{code}(M)) = 1 \iff \begin{array}{c}\text{there is some } \boldsymbol{w_i} \in \boldsymbol{\Sigma}^* \text{ such that} \\ \boldsymbol{\mathcal{M}_u}(\langle \mathbf{code}(M), \boldsymbol{w_i} \rangle) = 1 \text{ within } \boldsymbol{k} \text{ steps}\end{array}$$

Proof:

Step 4) By the definition of our **Universal Turing Machine** we have

$$\textbf{empty}^{-}(\textbf{code}(M)) = 1 \iff \text{there is some } w_i \in \Sigma^* \text{ such that } M(w_i) = 1 \text{ within } k \text{ steps}$$

Step 4) By the definition of our **Universal Turing Machine** we have

$$\textbf{empty}^{-}(\textbf{code}(M)) = 1 \iff \textbf{Language}\,(M) \neq \emptyset$$

Proof:

Step 5) By **definition** of $\overline{\mathbf{E}_{TM}}$ we have that

$$\mathbf{empty}^{-}(\mathbf{code}(M)) = 1 \iff \mathbf{code}(M) \in \overline{\mathbf{E}_{TM}}$$

- The **right-to-left** direction is **soundness**
- The **left-to-right** direction is **completeness**

Co-computable Enumerability

- **Co-computable Enumerability**
  - A language L is said to be **co-computably enumerable (co-C.E.)** if it's **complement** is computably enumerable

Co-computable Enumerability

Theorem If L and $\overline{L}$ are both C.E., then L is decidable.

Proof:

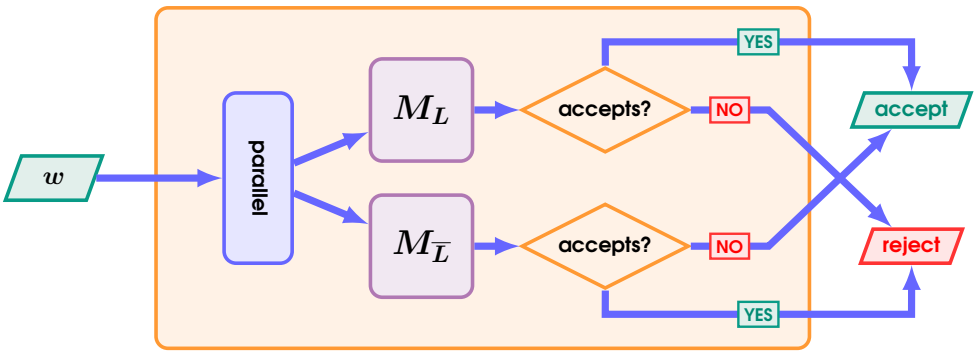
Step 1) Suppose that:

(i) L is C.E. and **(ii)** $\overline{L}$ is C.E.

Step 2) By **(i)** let M_L be a machine that is sound and complete w.r.t. L, and by **(ii)**, let $M_{\overline{L}}$ be a machine that is sound and complete w.r.t. $\overline{L}$.

Co-computable Enumerability

Step 3) Construct a new machine M' which runs M_L and $M_{\overline{L}}$ in **parallel**



Q.E.D.

Co-computable Enumerability

Theorem The *non*-Halting problem $\overline{\textbf{HALT}_{TM}}$ is **not** C.E.

Proof:

Step 1) We have that

Q.E.D.

Co-computable Enumerability

Theorem The *non*-Halting problem $\overline{\mathbf{HALT}_{TM}}$ is **not** C.E.

Proof:

Step 2) However, we have that $\mathbf{HALT}_{TM}$ is **undecidable** but **C.E.**.

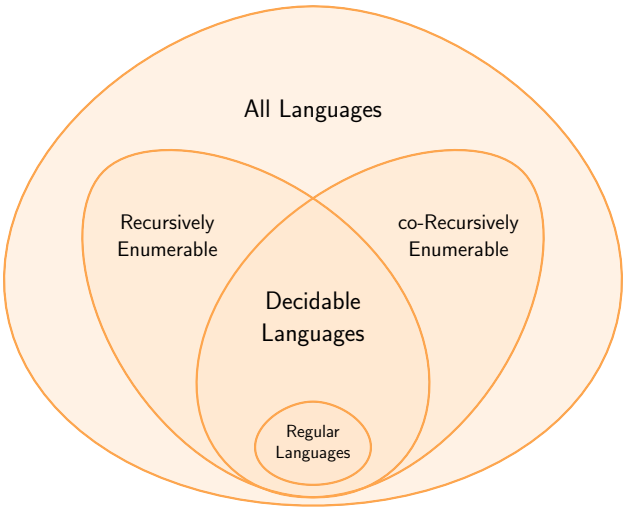
Step 3) Therefore, $\overline{\mathbf{HALT}_{TM}}$ must not be **C.E.**

Q.E.D.

Co-computable Enumerability

Theorem The Emptiness problem $\mathbf{E}_{TM}$ is **not** C.E.

Complexity Hierarchy



End of Slides!