# Universal TMs and the acceptance problem

Peter Mayr

Computability Theory, September 18, 2023

# Notation

Just like computing functions can be reduced to a membership question for languages, so can the checking of properties:

## Example

**Decision problem** $P$:

- **Input:** $w \in \{1\}^*$
- **Question:** Is $|w|$ a square?

Identitfy $P$ with the set of its “yes”-instances,

$$P = \{w \in \{1\}^* \ : \ |w| \text{ is a square}\}.$$

We identify

- decision problem = language
- decidable = computable

# Encoding DTMs

To present mathematical objects (tuples, graphs, TMs, ...) as input to TMs we encode them as strings, wlog over $\Sigma = \{0, 1\}$.

## Definition

Let $M = (Q, \{0, 1\}, \Gamma, s, t, r, \delta)$ be a DTM with

- $n$ states $Q = \{1, 11, \ldots, 1^n\}$,
- tape alphabet $\Gamma = \{0, 1, \sqcup, \gamma_4, \ldots, \gamma_m\}$ with letters represented in unary, $[\gamma_i] := 1^i$, square brackets denote encoding
- directions $+1$ and $-1$ represented by $[+1] := 1$ and $[-1] := 11$, resp.

Then our **encoding** $[M]$ of $M$ begins with

$$1^{|Q|}01^{|\Gamma|}0s0t0r0$$

0 as separator

followed by the encoding of all transitions $\delta(q, a) = (p, b, d)$ as

$$q0[a]0p0[b]0[d]0$$

### Note

The encoding $[M]$ essentially IS (the transition function of) $M$.

### Theorem

The language $\{[M] \ : \ M \text{ is a DTM}\}$ is computable.

### Proof.

There is a halting DTM to check for given $w \in \{0,1\}^*$ whether $w$ is the code of a DTM. □

# Encoding pairs

### Definition

Let $\Sigma := \{0, 1\}$, and $x = x_1 \dots x_k, y = y_1 \dots y_\ell \in \Sigma^*$.
Then $(x, y)$ can be encoded as the string of length $2(k + \ell + 1)$,

$$[x, y] := x_1x_1 \dots x_kx_k\underbrace{01}_{\text{separator}}y_1y_1 \dots y_\ell y_\ell$$

### Lemma

1. The language $\{[x, y] \ : \ x, y \in \Sigma^*\}$ is computable.
2. There exist computable partial functions $p_1, p_2 \colon \Sigma^* \to_p \Sigma^*$ such that $p_1([x, y]) = x$ and $p_2([x, y]) = y$.

This extends to encoding $n$-tuples via
$(a_1, \dots, a_n) = (\dots((a_1, a_2), a_3) \dots a_n)$.

# Universal Turing machines

### Question

Instead of devising a specific DTM $M$ for every task, is there a single DTM $U$ that can simulate any other?

More precisely:

### Definition

A DTM $U$ is **universal** if on input $([M], x)$ for a DTM $M$

- $U$ accepts if $M$ accepts $x$,
- $U$ rejects if $M$ rejects $x$,
- $U$ loops if $M$ loops on $x$.

Here $[M]$ is like a program that $U$ runs on $x$.

## Theorem

Universal DTMs exist.

## Proof.

Sketch universal $U$ as multitape TM with $\Sigma = \{0, 1\}, \Gamma = \{0, 1, \text{␣}\}$:

- Tape 1 holds input $([M], x)$.

configuration of

- Tape 2 holds the state of $M$ (as $[q_i] = 1^i$).
- Tape 3 simulates the tape of $M$ (encoding $M$'s tape alphabet $[\gamma_i] = 1^i0$.
- Tape 4 holds the position of $M$'s head.

Scratch tape

On input $(c, x)$:

1. $U$ checks that $c$ is proper TM-code $c = [M]$ (else rejects).
2. $U$ writes $s, x, 0$ on tapes 2,3,4, respectively.
3. To simulate a step of $M$, search for appropriate transition on tape 1 and update tapes 2,3,4 accordingly.
4. $U$ accepts/rejects/loops if $M$ accepts/rejects/loops on $x$.

# Acceptance Problem

### Definition

The language of a universal TM is the **acceptance problem**,

$$\mathrm{AP} := \{([M], x) \ : \ M \text{ is a DTM that accepts } x\}.$$

### Theorem

AP is computably enumerable.

# A non-computably enumerable language

**Idea:** For $L$ to be non-c.e. we need for every DTM $M$ some $x \in \Sigma^*$ such that

$$x \in L \text{ iff } M \text{ does not accept } x.$$

What if we simply choose $x := [M]$ above?

### Definition

The **self acceptance problem** is

$$\mathrm{SAP} := \{[M] \ : \ M \text{ is a DTM that accepts } [M]\}.$$

### Theorem

The complement $\overline{\mathrm{SAP}}$ is not c.e. (and hence not computable).

### Proof.

Seeking a contradiction, suppose $M$ is a DTM with $L(M) = \overline{\mathrm{SAP}}$. Consider 2 cases:

- If $M$ accepts $[M]$, then $[M] \in L(M) = \overline{\mathrm{SAP}}$, i.e. $M$ does not accept $[M]$.
- If $M$ does not accept $[M]$, then $[M] \in \overline{L(M)} = \mathrm{SAP}$, i.e. $M$ accepts $[M]$.

In each case we obtain a contradiction. Hence such an $M$ cannot exist. □

### Note

The proof above is by **diagonalization**: For an enumeration of all DTMs $M_1, M_2, \ldots,$ $\overline{\mathrm{SAP}}$ is different

- from $L(M_1)$ as witnessed by $[M_1]$,
- from $L(M_2)$ as witnessed by $[M_2]$,
- ...

# Reductions

Using that SAP is not computable, we can show that AP is neither.

### Theorem

AP is not computable.

### Proof.

Seeking a contradiction, suppose $U$ is a halting DTM with $L(U) = \mathrm{AP}$.

Then $\mathrm{SAP}$ is computable by the following DTM $N$:

- On input $[M]$ run $U$ on $([M], [M])$.
- If $U$ accepts $([M], [M])$, then $N$ accepts.
- If $U$ rejects $([M], [M])$, then $N$ rejects.

Then $L(N) = \mathrm{SAP}$.

Since $U$ is halting, so is $N$. Contradiction. □