More decision problems regarding machine codes

$L_{halt} := \{ \langle \alpha, \beta \rangle \mid M_\alpha \text{ halts on input } \beta \}$

$L_{empty} := \{ \alpha \mid L_\alpha = \emptyset \}$ $\quad L_{total} := \{ \alpha \mid L_\alpha = \{0,1\}^* \}$ Exercise: which of these (& their complements) are in R and/or RE?

$L_{eq} := \{ \langle \alpha, \beta \rangle \mid L_\alpha = L_\beta \}$

- Return to proof of $\overline{L_u} \notin RE$ via $L_d \notin RE$: "if we find a machine accepting $\overline{L_u}$, we can use it to accept L_d ↯"
  ↳ let's generalize this.

Df: Many-to-one reduction between languages:

$K \le_m L \equiv \exists f : \{0,1\}^* \to \{0,1\}^*$ computable s.t. $\forall \alpha \in \{0,1\}^* \; \alpha \in K \Leftrightarrow f(\alpha) \in L$

↑ sometimes denoted $K \to L$

$\le_m$ is a partial quasi-order on languages

Lemma: If $K \le_m L$ and $L \in RE$, then $K \in RE$.
If $K \le_m L$ and $L \in R$, then $K \in R$.
proof: compose machines for f and L

Corollary: If $K \le_m L$ and $K \notin RE$, then $L \notin RE$. (Similarly $K \notin R \Rightarrow L \notin R$.)

- Our original proof used $L_d \le_m \overline{L_u}$ & $L_d \notin RE$ to show $\overline{L_u} \notin RE$.

Exercise: Find reductions between L_u, L_{halt}, L_{empty}, L_{eq} & their complements.

Example: $\overline{L_{halt}} \xrightarrow{\le_m} L_{empty}$: given $\langle \alpha, \beta \rangle$, construct TM M_γ which ignores its input & runs M_α on input β

↳ $L_\gamma = \emptyset$ if $M_\alpha(\beta)$ diverges
$L_\gamma = \{0,1\}^*$ otherwise

$\Rightarrow (\gamma \in L_{empty} \Leftrightarrow \langle \alpha, \beta \rangle \in \overline{L_{halt}})$

$L_{empty} \xrightarrow{\le_m} \overline{L_{halt}}$: for given α, construct M_β which ignores its input, simulates M_α on all inputs in parallel & stops if $M_\alpha(\gamma)$ stops on some γ... $\langle \beta, \varepsilon \rangle \in \overline{L_{halt}}$
$\Leftrightarrow L_\alpha = \emptyset$.
$\Leftrightarrow \alpha \in L_{empty}$.

$L \xrightarrow{\le_m} M \Leftrightarrow \overline{L} \xrightarrow{\le_m} \overline{M}$

Also, $L_{halt} \le_m L_u \le_m L_{halt}$

- Semantic properties of machines

Df: Property of languages: $P \subseteq RE$ ... P is non-trivial $\equiv P \neq \emptyset$ & $P \neq RE$.
(semantic)

$L_P := \{ \alpha \in \{0,1\}^* \mid L_\alpha \in P \}$ ... all machines whose languages have the property P

Thm (Rice's): For every non-trivial property P, the language L_P is undecidable.

Proof idea: Show that $L_{halt} \to L_P$ for every non-trivial P.

<u>Proof:</u> Assume that $L_P \in R$ for some P.
WLOG $\emptyset \notin P$ ... otherwise use $\overline{P}$ ... $L_{\overline{P}} = \overline{L_P}$, so it's also in R.
Find $L_\omega \in P$ ... exists as P is non-trivial
Reduction: if we want to answer $\langle\alpha,\beta\rangle \in L_{halt}$, i.e. if $M_\alpha(\beta)$ halts
construct M_γ which does on input δ:

this is computable

- run M_α on β (1)
- run M_ω on δ (2)

if $\langle\alpha,\beta\rangle \in L_{halt}$: (1) halts, (2) halts if $\delta \in M_\omega \Rightarrow L_\gamma = L_\omega \in P$
$\notin L_{halt}$: (1) diverges, (2) doesn't run $\Rightarrow L_\gamma = \emptyset \notin P$

So this shows $L_{halt} \le_m L_P$ ... but $L_{halt} \notin R$, so $L_P \notin R$.

What is the "hardest" language in a class?

- Let $\mathcal{C}$ be a set of languages.
- L is <u>$\mathcal{C}$-hard</u> $\equiv \forall K \in \mathcal{C}: K \le_m L$
- L is <u>$\mathcal{C}$-complete</u> $\equiv L$ is $\mathcal{C}$-hard & $L \in \mathcal{C}$

more precisely, it's $\mathcal{C}$-m-hard (complete wrt. $\le_m$)

<u>Thm:</u> L_u is RE-complete.

<u>Pf:</u> ① $L_u \in RE$
② for $K \in RE$, we find α: $L_\alpha = K$
Then f reducing K to L_u is $\beta \mapsto \langle\alpha,\beta\rangle$

"Natural" undecidable problems (not directly involving machines)

- given a set of axioms and a formula φ, is φ provable?
- given a system of multi-variate polynomial equations over $\mathbb{Z}$, does it have a solution in $\mathbb{Z}$? $\rightarrow$ Matijasević theorem

both in $RE \setminus R$ (in suitable encoding)

& many more (e.g., plane tiling)

<u>Relative computability</u>

- given any language $A \subseteq \{0,1\}^*$, we can define TM with an oracle giving access to A (see section on TM extensions)
- we can define <u>relative</u> language classes $R[A]$ and $RE[A]$
- we also have $M_\alpha[A]$, $L_\alpha[A]$, $L_u[A]$

If $A \in R$, then $R[A] = R$ and $RE[A] = RE$ (in particular for $A = \emptyset$)

Previous arguments about plain TM can be trivially relativized, so in particular: $R[A] \subsetneq RE[A] \subsetneq 2^{\{0,1\}^*}$

$R[A] = RE[A] \cap \text{co-}RE[A]$ ← co-$T = \{L \mid \overline{L} \in T\}$

and also $L_u[A]$ is $RE[A]$-complete

Df: Arithmetical hierarchy: classes $\Sigma_n, \Pi_n, \Delta_n$ for $n \in \mathbb{N}$

- $\Sigma_0 = \Pi_0 = \Delta_0 = R$
- $\Sigma_{n+1} = RE[\Sigma_n]$
- $\Pi_{n+1} = co\text{-}RE[\Pi_n]$
- $\Delta_{n+1} = R[\Sigma_n]$

this means: $RE[\mathcal{C}] = \bigcup_{L \in \mathcal{C}} RE[L]$

→ We have: $\Sigma_1 = RE$, $\Pi_1 = co\text{-}RE$, $\Delta_1 = R[R] = R$ } oracles from R do not add power to TM

$\Sigma_n \subseteq \Sigma_{n+1}$

this inclusion is strict ~~as $L_u[\Sigma_n]$ is RE~~

$RE[L_u^n] \not\subseteq \Sigma_n$, ~~otherwise we would have~~ $\Sigma_n \subseteq R[\Sigma_n] = R[L_u^n] \subsetneq RE[L_u^n] = \Sigma_{n+1}$

- $L_u^1 = L_u$

  $L_u^{n+1} = L_u[L_u^n]$ → L_u^n is $\le\Sigma_n$-complete (by induction: $RE[\Sigma_n] = RE[L_u^n]$, so $L_u[L_u^n]$ is $\le\Sigma_{n+1}$-complete)

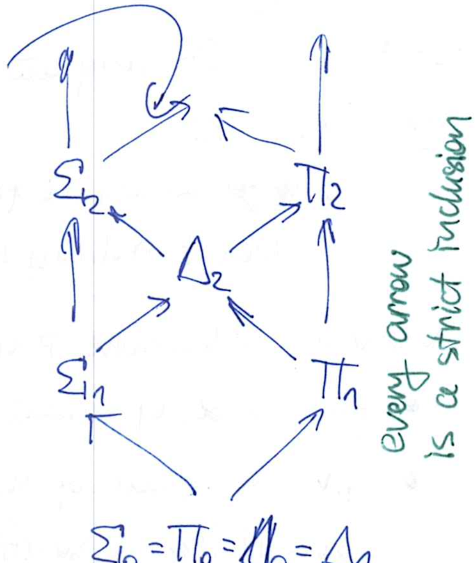
Also: $\Sigma_n \subseteq \Delta_{n+1}$ ... and this is strict as Σ_n is not closed under complement, while Δ_{n+1} is

$\Pi_n \subseteq \Delta_{n+1}$ ... we can negate oracle's answer

$\Delta_{n+1} = \Sigma_{n+1} \cap \Pi_{n+1}$ ... relative Post's thm.

$\Delta_{n+1} \subseteq \Sigma_{n+1}$ ... this is $R[L_u^n] \subsetneq RE[L_u^n]$

$\Delta_{n+1} \subseteq \Pi_{n+1}$ ... analogous for co-RE



Quantified formulas

- every language in R can be interpreted as a predicate $\varphi(\alpha)$ with string parameter φ – decidable predicates
- $\psi(\beta) \equiv \exists\alpha\, \varphi(\alpha,\beta)$ lies in RE

  ... and every $L \in RE$ can be written in this way [α = # steps after which a machine stops]
- $\psi(\beta) \equiv \forall\alpha\, \varphi(\alpha,\beta)$ ... this is co-RE ($\neg\forall\alpha\, \varphi(\alpha,\beta) \Leftrightarrow \exists\alpha \underbrace{\neg\varphi(\alpha,\beta)}_{\text{decidable}}$)
- ... $\exists\alpha_1 \exists\alpha_2\, \varphi(\alpha_1,\alpha_2,\beta)$ is again RE ... we can say $\exists\alpha$ s.t. $\alpha = \langle\alpha_1,\alpha_2\rangle$ & decode α inside φ
- $\exists\alpha_1 \forall\alpha_2\, \varphi(\alpha_1,\alpha_2,\beta)$ is Σ_2
- in general: $\exists\alpha_1 \forall\alpha_2 \exists\alpha_3 \dots Q_n\alpha_n\, \varphi(\alpha_1 - \alpha_n, \beta)$ is Σ_n

  $\forall\alpha_1 \exists\alpha_2 \forall\alpha_3 \dots Q_n\alpha_n\, \varphi(\alpha_1 - \alpha_n, \beta)$ is Π_n

$\subseteq \Sigma_2$: $\exists\alpha_1 \forall\alpha_2\, \varphi(\dots) \Leftrightarrow \exists\alpha_1 \neg\underbrace{(\exists\alpha_2\, \varphi(\dots))}_{\text{can be answered by oracle } L_u \in \Sigma_1}$ ← so this is in $RE[\Sigma_1] = \Sigma_2$

$\supseteq \Sigma_2$ consider $L \in \Sigma_2 = RE[\Sigma_1]$:

$\exists\gamma$ computation of TM $[L_u]$ $\exists\delta$ queries for L_u (γ, δ consistent, & answers for δ true) see next page

(continued) We want to show that $\forall L \in \Sigma_2$ there is an equivalent formula $\exists\forall\ldots$

- let M be a $TM[\Sigma_1]$ … that is $TM[L_u]$ accepting L
- formula: $\exists\delta$ (check that δ is consistent with input α, with oracle L_u, and with itself)

$\exists\delta$: computation of M including all oracle queries & answers

input α: decidable

itself: decidable

oracle L_u: for positive answers: $\exists\gamma_1 - \gamma_k$ $\psi(\gamma_1, \text{question 1})$ $\vdots$ $\psi(\gamma_k, \text{question } k)$ ← Σ_1-formula for L_u

for negative answers: $\forall\varepsilon_1 - \varepsilon_\ell\ \neg\psi(\varepsilon_1, \ldots)$ & …

together:

$$\exists\delta\,(\exists\gamma\ \forall\varepsilon\ \varphi(\alpha,\delta,\gamma,\varepsilon))$$

γ: code of $\gamma_1 - \gamma_k$; ε: code of $\varepsilon_1 - \varepsilon_\ell$

… but this is $\exists\langle\delta,\gamma\rangle\ \forall\varepsilon\ \varphi(\ldots)$ as we need.

COMPLEXITY

Df:

- For a multi-tape machine M, define run time $t_M(\alpha)$ for input α as # steps before computation $M(\alpha)$ halts.
  - $t_M : \underbrace{\{0,1\}^*}_{\text{generally } \Sigma^*} \to \mathbb{N}^*$, where $\mathbb{N}^* = \mathbb{N} \cup \{\infty\}$ — for divergent computations
- Time complexity of machine M: $T_M : \mathbb{N} \to \mathbb{N}^*$ s.t. $T_M(n) = \max\limits_{\alpha \in \Sigma^n} t_M(\alpha)$.

  M always halts $\Leftrightarrow$ $T_M(n)$ finite for all n.

Df: Asymptotic notation: for functions $f, g : \mathbb{N} \to \mathbb{R}$ define:

1. $f \in O(g) \equiv \exists c\ \forall^* n\ f(n) \le c \cdot g(n)$ ← asymptotic upper bound

   $\forall^*$: for all but finitely many exceptions
2. $f \in \Omega(g) \equiv \exists c\ \forall^* n\ f(n) \ge c \cdot g(n)$ ← asymp. lower bound
3. $f \in \Theta(g) \equiv f \in O(g)\ \&\ f \in \Omega(g)$ ← both at once

   … that is, $\Theta(g) = O(g) \cap \Omega(g)$
4. $f \in o(g) \equiv \lim\limits_{n\to\infty} \dfrac{f(n)}{g(n)} = 0$ ← strict upper bound
5. $f \in \omega(g) \equiv g \in o(f)$ ← strict lower bound

Examples: $f : n \mapsto 5n^3 - 7n^2 + 18$

$\in O(n^3), O(n^4), O(2^n)$ $\quad \in o(n^4)$

$\in \Omega(n^3), \Omega(n^2), \Omega(1)$ $\quad \in \omega(n^2)$

$\in \Theta(n^3)$ "drop lower-order terms & multiplicative constant"

$O(1)$ = "bounded by constant"