SUMMARY SO FAR

1. We saw $D = \{x \mid \{x\}_1(x) \text{ does not accept}\}$

   is not r.e. by diagonalization

2. Using reductions we have:

   - K, $Halt$ are not recursive (but both are r.e.)

   - $\overline{K}$, $\overline{HALT}$ are not r.e.

$K = \{x \mid \{x\}_1 \text{ halts on input } x\}$

$D = \overline{K}$

Another example: $L = \{ x \mid \{x\} \text{ accepts at least one input} \}$

L is r.e. but not recursive.

$\boxed{L \text{ is r.e.:}}$ Enumerate all strings in $\{0,1\}^*$

$\{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ldots$

$w_1 \quad w_2 \quad w_3$

Dovetail Procedure for L on input x:

For $i = 1, 2, 3, \ldots$

For $j = 1, \ldots, i$

Simulate $\{x\}$ on w_j for i steps

If any of the simulations accepts, HALT & accept

L = {x | {x} accepts at least one input}

Want to show if L is recursive then so is K.

Let M be a TM always halts & accepts L.

Using M, construct M' that always halts & accepts K.

K = {y | {y} on y halts}

M' on input y:

Run M on <M''>

If M accepts → accept

ow → reject

Intermediate Machine M'' on z

ignore z

simulate {y} on y

If simulation halts accept

either: ① y halts any → halts & accepts all inputs

② y doesn't halt any → Not halt on any input

Another example: $L = \{ x \mid \{x\} \text{ accepts at least one input} \}$

$\boxed{L \text{ is not recursive:}}$

$L_1 = K = \{ y \mid \{y\}(y) \text{ halts} \}$

Assume $L_2 = L$ is recursive & let M_2 be TM $\mathcal{L}(M_2) = L$
and M_2 always halts

M_1 on input y:

Construct encoding z of TM $\{z\}$ where
$\{z\}$ on input x: Ignores x & runs $\{y\}$ on y
and accepts x if $\{y\}(y)$ halts

Run M_2 on z and accept y iff $M_2(z)$ accepts

Claim $\mathcal{L}(M_1) = K$ and M_1 always halts

$y \in K \Rightarrow \{y\}(y)$ halts $\Rightarrow \{z\}$ accepts all inputs $\Rightarrow M_2(z) = 1 \Rightarrow M_1(y) = 1$

$y \notin K \Rightarrow \{y\}(y)$ doesn't halt $\Rightarrow \{z\}$ accepts No input $\Rightarrow M_2(z) \neq 1 \Rightarrow M_1(y) \neq 1$

$L = \{ x \mid$ TM encoded by x halts on > 2 inputs in $\{0,1\}^* \}$

r.e.

$\overline{L} = \{ x \mid$ TM x halts on ≤ 2 inputs $\}$

not r.e.

---

if L is re and $\overline{L}$ is re $\longrightarrow$ then both are recursive

on input x : (want to decide if $x \in L$) :

For $i = 1, 2, \ldots\ldots$

Run x on TM for L for i steps
if accepts $\to$ halt & accept

Run x on TM for $\overline{L}$ for i steps
accepts $\to$ halt & reject

$L = \{ x \mid$ TM encoded by x halts on > 2 inputs in $\{0,1\}^* \}$

r.e.
Not recursive

$\overline{L} = \{ x \mid$ TM x halts on ≤ 2 inputs $\}$

Not r.e.

1. $L_1 = \{ x \mid$ TM encoded by x never moves head left on any input $\}$

2. $L_2 = \{ \langle x, y \rangle \mid$ TM x on input y never moves head left $\}$

$\overline{L_2} = \{ \langle x, y \rangle \mid x$ on input y moves head left at some point $\}$

~~ass~~: halli

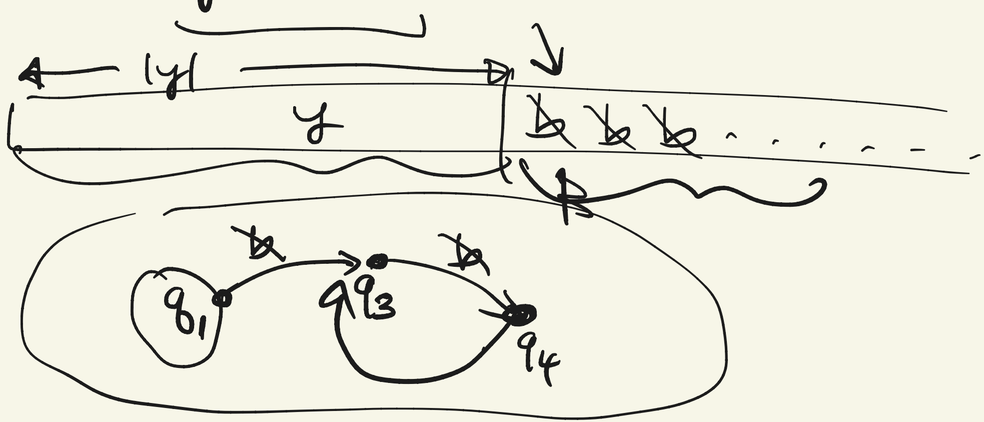
Harder case (of deciding L_2)

State transition table does have some transitions that move head to left

Machine x. Assume states of x are $q_0\, q_1\, q_2\, q_3\, q_4$

assume input/tape alphabet $= \{0, 1, \not{b}\}$

Let $y \in \{0,1\}^*$



$L = \{ x \mid$ TM encoded by x halts on > 2 inputs in $\{0,1\}^* \}$

$\overline{L} = \{ x \mid$ TM x halts on ≤ 2 inputs $\}$

$L' = \{ x \mid$ TM encoded by x halts on exactly 2 inputs in $\{0,1\}^* \}$

Not r.e.

$D = \{ x \mid \{x\}$ doesnt halt on $x \}$

assume M' accepts L' (but M' doesnt necessarily always halts)

Want to construct a TM accepts D.

Assume M' accepts L'

$L' = \{x \mid x$ halts on exactly 2 inputs$\}$

$D = \{z \mid \{z\}$ doesn't halt on $z\}$

M on z:

1. construct intermed TM N where N on y

   N on input y:

   If $y = 0$ then halt & accept

   $y = 1$ halt & accept

   If $y = 00$ simulate $\{z\}$ on z if halts then halt

   ow (for all other y)

   go into an infinite loop

2. Run M' on $\langle N \rangle$. accept iff M' accepts

Completeness

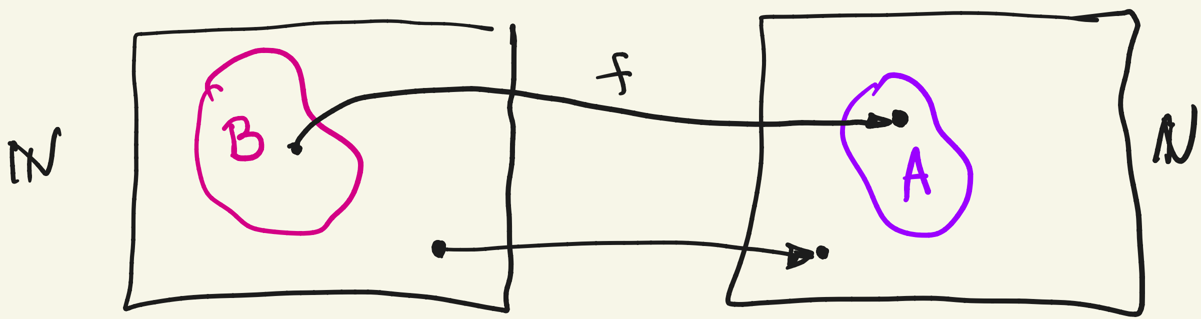
A set $A \subseteq \mathbb{N}$ is r.e.-complete if

(1) A is r.e.

(2) $\forall B \subseteq \mathbb{N}$, if B is r.e. then $B \leq_m A$

$\exists$ computable function $f : \mathbb{N} \Rightarrow \mathbb{N}$ such that

$\forall x \quad f(x) \in A \iff x \in B$



Completeness

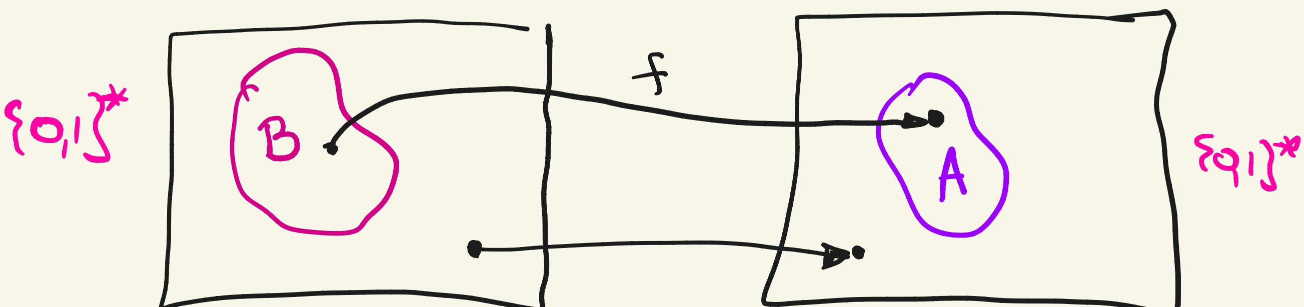
Similarly a language $A \subseteq \{0,1\}^*$ is r.e.-complete if

(1) A is r.e.

(2) $\forall B \subseteq \{0,1\}^*$, if B is r.e. then $B \le_m A$

$\exists$ computable function $f : \mathbb{N} \Rightarrow \mathbb{N}$ such that

$\forall x \quad f(x) \in A \iff x \in B$



Hilbert's 10^{th} Problem (1900)

A diophantine equation is of the form $p(\vec{x}) = 0$ where p is a polynomial over variables $x_1, \dots, x_n$ with integer coefficients

Ex $\quad 3x_1^5 x_2^3 + (x_1 + 1)^8 - x_7^{10} = 0$

$$\mathcal{L}_{DIOPH} = \{ \langle p \rangle \mid p \text{ has a solution over } \mathbb{N} \}$$

Theorem

$\mathcal{L}_{DIOPH}$ is r.e.-complete

An Equivalent Characterization of RE Sets

Let $f : \mathbb{N} \to \mathbb{N}$

Then $R_f \subseteq \mathbb{N} \times \mathbb{N}$

is the set of all pairs (x, y) such that $\underbrace{f(x) = y}$

* **Theorem** f computable if and only if R_f is r.e.

Proof $\Rightarrow$: Suppose f computable.

TM for R_f on input (x,y):

Run TM computing f on x.

If it halts and outputs y then accept (x,y)

Otherwise reject (x,y)

An Equivalent Characterization of RE Sets

Let $f : \mathbb{N} \to \mathbb{N}$

Then $R_f \subseteq \mathbb{N} \times \mathbb{N}$

is the set of all pairs (x, y) such that $f(x) = y$

* **Theorem** f computable if and only if R_f is r.e.

Proof $\Leftarrow$: Let R_f be r.e. with TM M

On x: Enumerate all $\mathbb{N}$: $y_1, y_2, \ldots$

For $i = 1, 2, \ldots$

For all $j \leq i$: simulate M on (x, y_j) for i steps

If simulation accepts (x, y_j),

halt & output y_j

A second Characterization of RE Sets

***Theorem** A relation $A \subseteq \mathbb{N}^{\cancel{k}}$ is r.e. if and only if there is a recursive relation $R \subseteq \mathbb{N}^{k+1}$ such that

$\mathbb{N}^2$

$$\vec{x} \in A \iff \exists y\, R(\vec{x}, y) \quad \forall \vec{x} \in \mathbb{N}^n$$

Note We defined A to be r.e. iff there is a TM M such that $\forall \vec{x} \in \mathbb{N}^n \; \big(M(\langle x \rangle) \text{ accepts} \iff \vec{x} \in A \big)$

A language $L \subseteq \{0,1\}^*$ is r.e.

iff there exists a ~~[illegible]~~ relation $R \subseteq \{0,1\}^* \times \{0,1\}^*$

s.t. $\forall x \in \{0,1\}^*$

$x \in L$ iff $\exists z \in \{0,1\}^*\ R(x,z)$

where R is recursive

Ex. Let $L = \text{Halt} = \{\langle x,y\rangle \mid$ TM encoded by x halts on input $y\}$

Let $R(\langle x,y\rangle, z) = \begin{cases} 1/\text{accept} & \text{if } x \text{ halts on } y \text{ in exactly } z \text{ steps} \\ 0 & \text{ow.} \end{cases}$

# of steps

A Second Characterization of RE Sets

*** Theorem** A relation $A \subseteq \mathbb{N}^k$ is r.e. if and only if there is a recursive relation $R \subseteq \mathbb{N}^{k+1}$ such that

$$\vec{x} \in A \iff \exists y\, R(\vec{x}, y) \quad \forall \vec{x} \in \mathbb{N}^n$$

Proof sketch

$\Rightarrow$: Let A be r.e., $\mathcal{L}(M) = A$

$R(\vec{x}, y)$: view y as encoding of an $m \times m$ tableaux for some $m \in \mathbb{N}$

$(\vec{x}, y) \in R \iff M(\vec{x})$ halts in m steps and accepts and y is the $m \times m$ tableaux of $M(\vec{x})$

A second Characterization of RE Sets

*** Theorem** A relation $A \subseteq \mathbb{N}^k$ is r.e. if and only if there is a recursive relation $R \subseteq \mathbb{N}^{k+1}$ such that

$$\vec{x} \in A \iff \exists y\, R(\vec{x}, y) \quad \forall \vec{x} \in \mathbb{N}^n$$

Proof sketch

$\Leftarrow$ Let $R \subseteq \mathbb{N}^{k+1}$ be recursive relation such that $\vec{x} \in A \iff \exists y\, R(\vec{x}, y)$, + Let $\mathcal{L}(M) = R$

on input $\vec{x}$:

For $i = 1, 2, \ldots$

For $j = 1$ to i

Run M on $(\vec{x}, y_j)$

halt + accept if $M(\vec{x}, y_j)$ accepts

Review of Definitions

$\mathcal{L}_A = \{0, s, +, \cdot\ ; =\}$ Language of arithmetic

$\Phi_0 =$ all $\mathcal{L}_A$- sentences

$TA = \{A \in \Phi_0 \mid \mathbb{N} \models A\}$ True Arithmetic

A theory Σ is a set of sentences (over $\mathcal{L}_A$) closed under logical consequence

- We can specify a theory by a subset of sentences that logically implies all sentences in Σ

Σ is consistent iff $\Phi_0 \neq \Sigma$ (iff $\forall A \in \Phi_0$, either A or $\neg A$ Not in Σ)

Σ is complete iff Σ is consistent and $\forall A$ either A or $\neg A$ is in Σ

Σ is <u>sound</u> iff $\Sigma \subseteq TA$

Let $\mathfrak{M}$ be a model/structure over $\mathcal{L}_A$

$Th(\mathfrak{M}) = \{ A \in \Phi_0 \mid \mathfrak{M} \models A \}$

$Th(\mathfrak{M})$ is <u>complete</u> (for all structures $\mathfrak{M}$)

<u>Note</u> $TA = Th(\mathbb{N})$ is complete, consistent, & sound

$VALID = \{ A \in \Phi_0 \mid \ \models A \}$ $\longleftarrow$ smallest theory

Let Σ be a theory

Σ is <u>axiomatizable</u> if there exists a set $\Gamma \subseteq \Sigma$

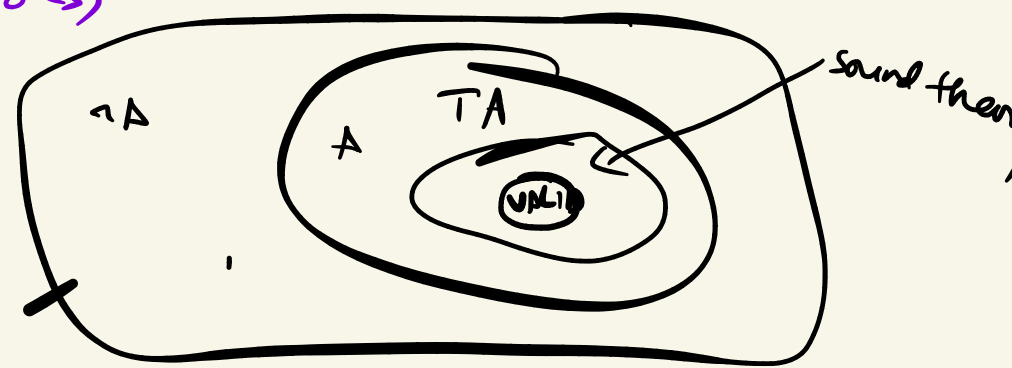
such that ① Γ is recursive

② $\Sigma = \{ A \in \Phi_0 \mid \Gamma \models A \}$

Theorem Σ is axiomatizable iff Σ is r.e.

(P. 76 of Notes)

all sentences



Let Σ be a theory

Σ is <u>axiomatizable</u> if there exists a set $\Gamma \subseteq \Sigma$ such that ① Γ is recursive
② $\Sigma = \{ A \in \Phi_0 \mid \Gamma \models A \}$

<u>Theorem</u> Σ is axiomatizable $\Leftrightarrow$ Σ is r.e.

<u>Proof</u> $\Rightarrow$. Suppose Σ is axiomatizable, Γ recursive

Define $R(x,y) =$ true iff y encodes a Γ-LK proof of (the formula encoded by) x

R is recursive, so by previous *Theorem, Σ is r.e.

Let Σ be a theory

Σ is <u>axiomatizable</u> if there exists a set $\Gamma \subseteq \Sigma$ such that
1. Γ is recursive
2. $\Sigma = \{A \in \Phi_0 \mid \Gamma \models A\}$

<u>Theorem</u> Σ is axiomatizable iff Σ is r.e.

<u>Proof</u> $\Rightarrow$. Suppose Σ is axiomatizable, Γ recursive

Define $R(x,y)$ = true iff y encodes a Γ-LK proof of (the formula encoded by) x

R is recursive, so by previous *Theorem, Σ is r.e.

$\Leftarrow$ By *Theorem, Σ = range of total computable function f

$\therefore \quad \Sigma = \{f(0), f(1), f(2), \dots\}$

in other words, there is an effective enumeration of Σ

$\Sigma = \{A_0, A_1, A_2, A_3, \dots\}$

$\Sigma = \{A_0, A_1, A_2, \dots\}$

where $f(0) = A_0$, $f(1) = A_1, \dots \}$

where f is computable.

Let $\Gamma = \{B_0, B_1, \dots\}$ where $B_i = A_0 \wedge A_1 \wedge \dots A_i$

<u>Klaim</u> Γ is recursive and $\Sigma = \{A \mid \Gamma \models A\}$

- Γ is recursive:

  given sentence F, check if $F = F_1 \wedge F_2 \wedge \dots \wedge F_m$

  then check if $\forall i \in [m]$, $F_i = A_i$

  (which can be done since f is computable)

- $\Sigma = \{A \mid \Gamma \models A\}$: $\forall A_i \in \Sigma$, $B_i = A_1 \wedge \dots \wedge A_i \models A_i$