17 April 2025

# The Halting Problem & Friends

## Plan

* Computability Basics
* Announcements
* Computability Reductions

# Problems vs. Programs

Problems $\equiv$ Languages $\equiv$ Subsets of Strings

Programs $\equiv$ Strings

## Problems vs. Programs

Problems $\equiv$ Languages $\equiv$ Subsets of Strings

$$L \in \mathcal{P}(\Sigma^*)$$

Programs $\equiv$ Strings

$$\langle P \rangle \in \Sigma^*$$

Fact. $|\mathcal{P}(\Sigma^*)| > |\Sigma^*|$

Corollary. There are problems that are not solved by ANY program!

What does it mean for program to solve problem?

What does it mean for program to solve problem?

A Language $L$ is Recognizable (RE) if there exists a program $P$ s.t.

$$\mathcal{L}(P) = L$$

What does it mean for program to solve problem?

A Language $L$ is Recognizable (RE) if there exists a program $P$ s.t.

$$\underbrace{\mathcal{L}(P)}_{} = L$$

$$= \{ x \in \Sigma^* : P \text{ accepts } x \}$$

In other words: $x \in L \iff P \text{ accepts } x$.

What does it mean for program to <u>solve</u> problem?

A Language $L$ is Decidable (R) if there exists a program $D$ s.t.

$$\mathcal{L}(D) = L$$

and $D$ always halts.

accepts / rejects in finite time

What does it mean for program to solve problem?

A Language $L$ is Decidable (R) if there exists a program $D$ s.t.

$$\mathcal{L}(D) = L$$

and D always halts.

$$\Rightarrow \overline{L} = \{ x \in \Sigma^* : D \text{ rejects } x \}$$

What does it mean for program to solve a problem?

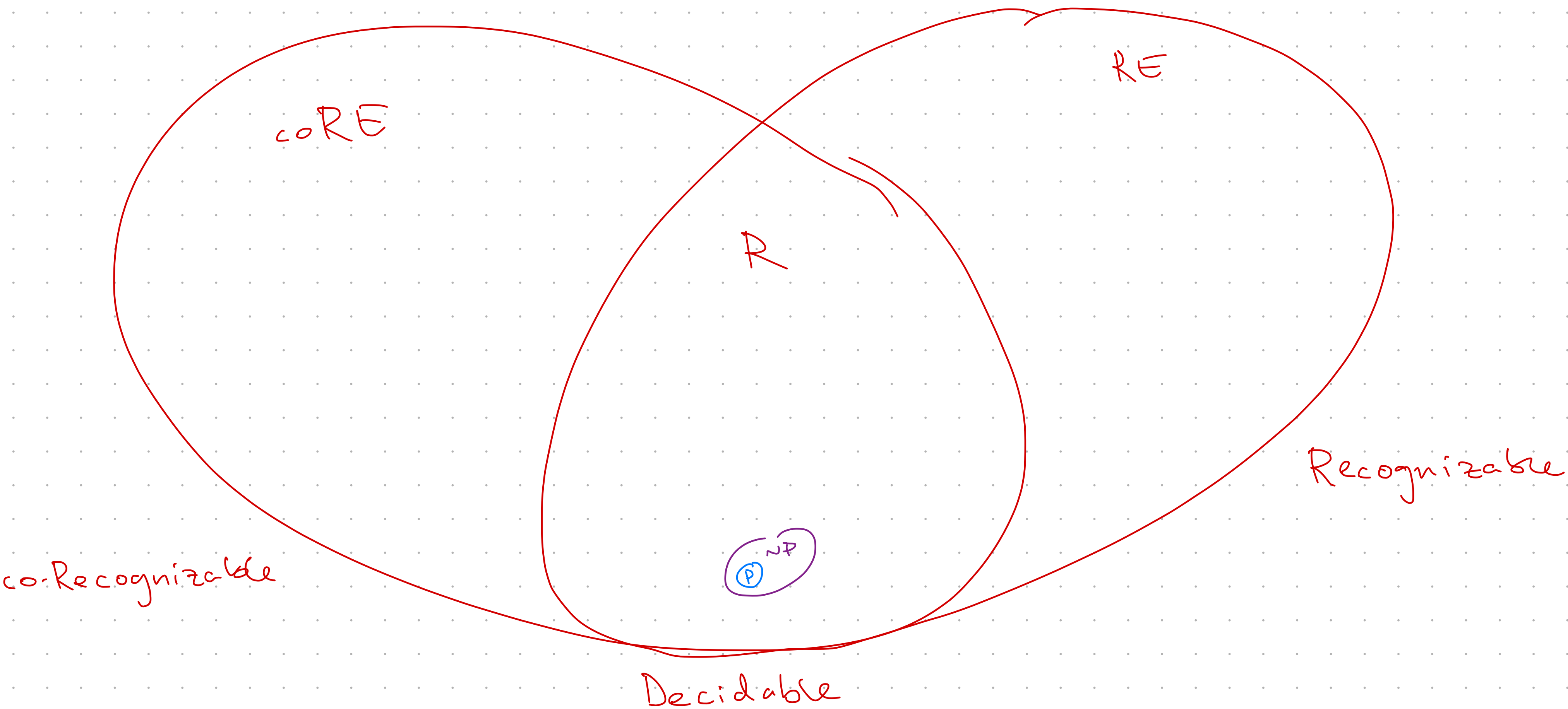
A Language $L$ is co-Recognizable (co-RE) if there exists a program $P$ s.t.

$$\mathcal{L}(P) = \overline{L}$$

$x \notin L \iff P$ accepts $x$

P recognizes the complement of $L$.

**Theorem.** A language L is Decidable iff L is Recognizable and co-Recognizable.



Theorem. A language $L$ is Decidable (R)
iff $L$ is Recognizable (RE) and co-Recognizable (coRE).

Pf $L \in R \Rightarrow L \in RE$ and $L \in coRE$.

**Theorem.** A language $L$ is Decidable (R) iff $L$ is Recognizable (RE) and co-Recognizable (coRE).

Pf $L \in R \Rightarrow L \in RE$ and $L \in coRE$.

$\Downarrow$

$\exists$ decider $D$ s.t. $\mathcal{L}(D) = L$. $\Rightarrow$ $L \in RE$

**Theorem.** A language $L$ is Decidable (R) iff $L$ is Recognizable (RE) and co-Recognizable (coRE).

**Pf** $L \in R \Rightarrow L \in RE$ and $L \in coRE$.

$\Downarrow$

$\exists$ decider $D$ s.t. $\mathcal{L}(D) = L$. $\Rightarrow$ $L \in RE$

$\Downarrow$

Consider new program $\tilde{D}$ defined as:

$\tilde{D}$ :

> On input $x$
> Run $D$ on $x$
> if $D$ rejects,
> $\boxed{\text{accept}}$

**Theorem.** A language $L$ is Decidable (R) iff $L$ is Recognizable (RE) and co-Recognizable (coRE).

**Pf** $L \in R \Rightarrow L \in RE$ and $L \in coRE$.

$\Downarrow$

$\exists$ decider $D$ s.t. $\mathcal{L}(D) = L$. $\Longrightarrow$ $L \in RE$

$\Downarrow$

Consider new program $\tilde{D}$ defined as:

$\tilde{D}$ :

| |
|---|
| On input $x$ |
| Run $D$ on $x$ |
| if $D$ rejects, $\boxed{\text{accept}}$ |

- $D$ always halts, so $D$ rejects $x$ $\Longleftrightarrow$ $x \notin L$

**Theorem.** A language $L$ is Decidable $(R)$ iff $L$ is Recognizable $(RE)$ and co-Recognizable $(coRE)$.

**Pf** $L \in R \Rightarrow L \in RE$ and $L \in coRE$.

$\Downarrow$

$\exists$ decider $D$ s.t. $\mathcal{L}(D) = L$. $\Longrightarrow$ $L \in RE$

$\Downarrow$

Consider new program $\tilde{D}$ defined as:

$\tilde{D}$ :

On input $x$
Run $D$ on $x$
if $D$ rejects, $\boxed{\text{accept}}$

- $D$ always halts, so $D$ rejects $x$ $\Longleftrightarrow$ $x \notin L$

$\Rightarrow \mathcal{L}(\tilde{D}) = \overline{\mathcal{L}(D)} = \overline{L}$

$\Longrightarrow L \in coRE$

**Theorem.** A language $L$ is Decidable ($R$) iff $L$ is Recognizable ($RE$) and co-Recognizable ($coRE$).

**Pf** $L \in R \Leftarrow L \in RE$ and $L \in coRE$.

$\Downarrow$

$\exists$ program $P$ s.t. if $x \in L$, $P$ accepts $x$

$\Downarrow$

$\exists$ program $Q$ s.t. if $x \notin L$, $Q$ accepts $x$

<u>Theorem</u>. A language $L$ is Decidable (R)

iff $L$ is Recognizable (RE) and co-Recognizable (coRE).

<u>Pf</u> $L \in R \Leftarrow L \in RE$ and $L \in coRE$.

$\Downarrow$

$\exists$ program $P$

s.t. if $x \in L$, $P$ accepts $x$

$\exists$ program $Q$

s.t. if $x \notin L$, $Q$ accepts $x$

Construct Decider for $L$ $D$.

D:

On input $x$

Run $P$ and $Q$ on $x$ in <u>parallel</u>.

if $P$ accepts, $\boxed{\text{accept}}$

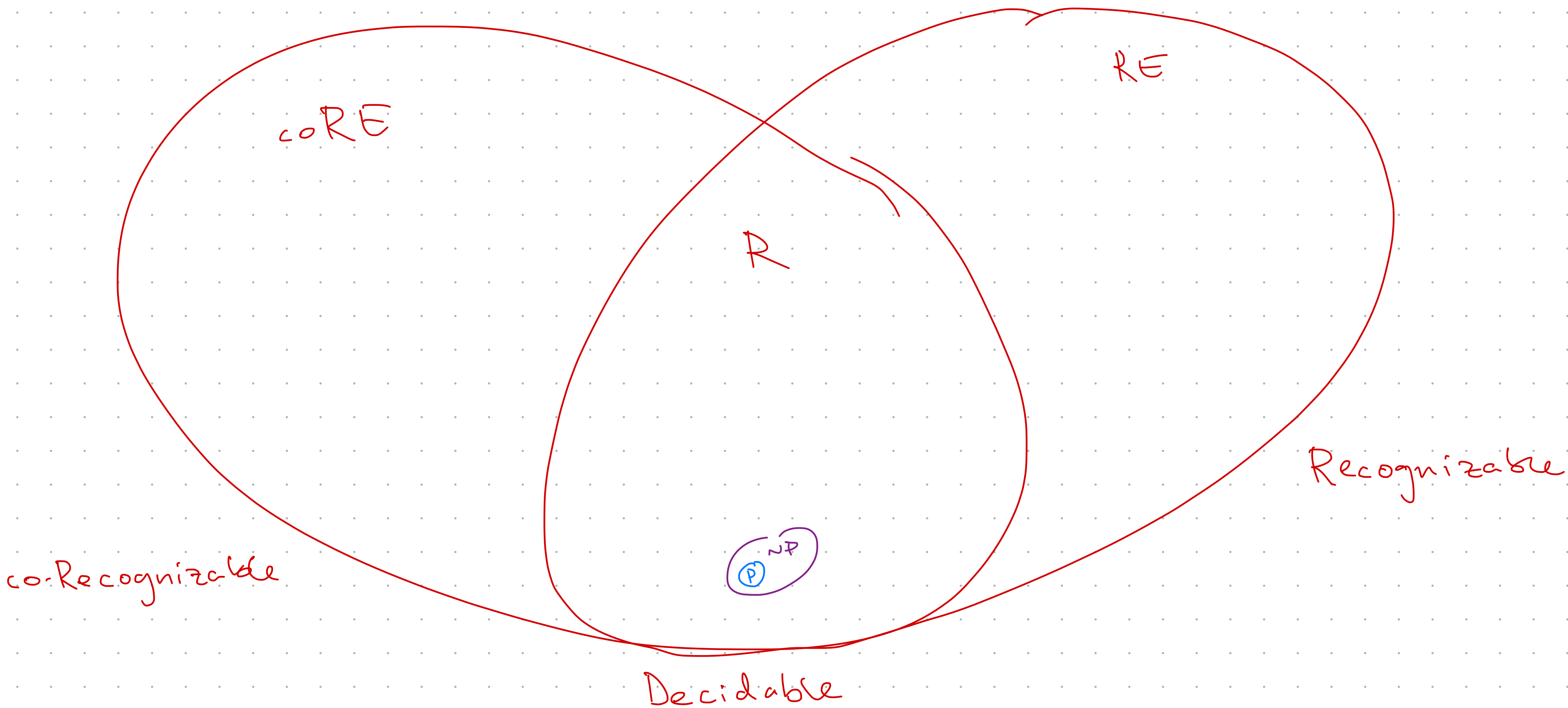
if $Q$ accepts, $\boxed{\text{reject}}$

# Announcements

* HW 7 due

* HW 8 delayed til Friday
  - Larger homework due April 29.
  - Final Required HW for class.

# Undecidable Problems $L \notin R$



$$\text{DIAGONAL} = \left\{ \langle P \rangle : P \text{ does } \underline{\text{NOT}} \text{ accept } \langle P \rangle \right\}$$

↳ The <u>set</u> of encodings of programs that do <u>NOT</u> accept their own encoding.

<u>Theorem</u> DIAGONAL is Undecidable!

$$\text{DIAGONAL} = \left\{ \langle P \rangle : P \text{ does } \underline{\text{NOT}} \text{ accept } \langle P \rangle \right\}$$

↳ The <u>set</u> of encodings of programs that do <u>NOT</u> accept their own encoding.

<u>Theorem</u> DIAGONAL is Undecidable !

→ program encodings

| | $\langle P_1 \rangle$ | $\langle P_2 \rangle$ | $\langle P_3 \rangle$ | – – – | | | |
|---|---|---|---|---|---|---|---|
| $\mathcal{L}(P_1)$ | 0 | 1 | 1 | 0 | 1 | 1 | – – – |
| $\mathcal{L}(P_2)$ | 1 | 0 | 1 | 1 | 0 | 1 | – – – |
| $\mathcal{L}(P_3)$ | 1 | 1 | 1 | 1 | 0 | 0 | – – – |
| $\vdots$ | 0 | 1 | 1 | 0 | 1 | 1 | – – – |

languages of programs

$$\text{DIAGONAL} = \left\{ \langle P \rangle : P \text{ does } \underline{\text{NOT}} \text{ accept } \langle P \rangle \right\}$$

↳ The <u>set</u> of encodings of programs that do <u>NOT</u> accept their own encoding.

**<u>Theorem</u>** DIAGONAL is Undecidable !

→ program encodings

| | $\langle P_1 \rangle$ | $\langle P_2 \rangle$ | $\langle P_3 \rangle$ | ... | | | |
|---|---|---|---|---|---|---|---|
| $\mathcal{L}(P_1)$ | 0 | 1 | 1 | 0 | 1 | 1 | ... |
| $\mathcal{L}(P_2)$ | 1 | 0 | 1 | 1 | 0 | 1 | ... |
| $\mathcal{L}(P_3)$ | 1 | 1 | 1 | 1 | 0 | 0 | ... |
| ⋮ | | | | | | | |
| | 0 | 1 | 1 | 0 | 1 | 1 | ... |

languages of programs

$$\begin{cases} 1 & \langle P_3 \rangle \in \mathcal{L}(P_4) \\ 0 & \langle P_3 \rangle \notin \mathcal{L}(P_4) \end{cases}$$

$$\text{DIAGONAL} = \{ \langle P \rangle : P \text{ does } \underline{\text{NOT}} \text{ accept } \langle P \rangle \}$$

↳ The <u>set</u> of encodings of programs that do <u>NOT</u> accept their own encoding.

<u>Theorem</u> DIAGONAL is Undecidable!

| | $\langle P_1 \rangle$ | $\langle P_2 \rangle$ | $\langle P_3 \rangle$ | ... | | | |
|---|---|---|---|---|---|---|---|
| $\mathcal{L}(P_1)$ | $\boxed{0}$ | 1 | 1 | 0 | 1 | 1 | ... |
| $\mathcal{L}(P_2)$ | 1 | $\boxed{0}$ | 1 | 1 | 0 | 1 | ... |
| $\mathcal{L}(P_3)$ | 1 | 1 | $\boxed{1}$ | 1 | 0 | 0 | ... |
| $\vdots$ | 0 | 1 | 1 | $\boxed{0}$ | 1 | 1 | ... |

$$\text{DIAGONAL} = \{ \langle P \rangle : P \text{ does } \underline{\text{NOT}} \text{ accept } \langle P \rangle \}$$

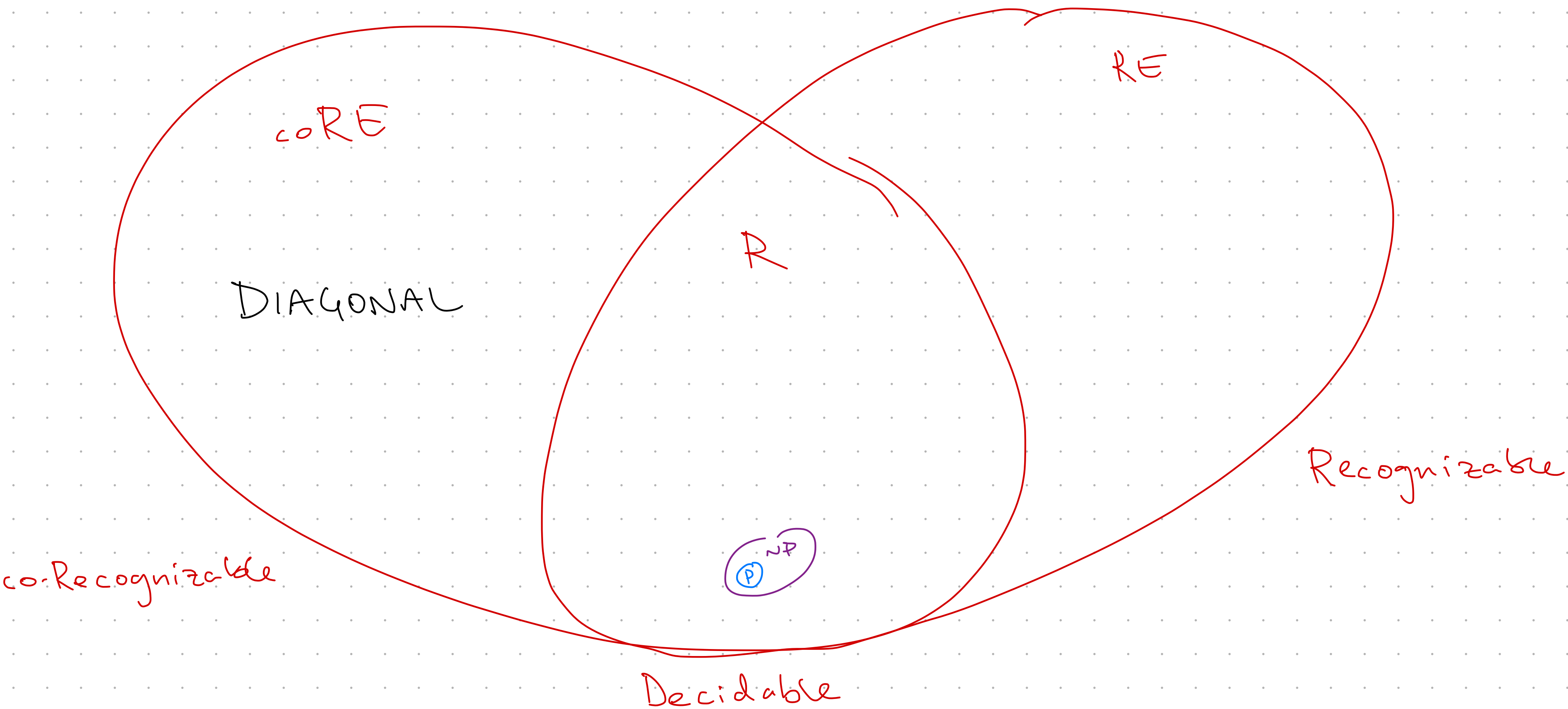
↳ The <u>set</u> of encodings of programs that do <u>NOT</u> accept their own encoding.

## <u>Theorem</u> DIAGONAL is Undecidable!

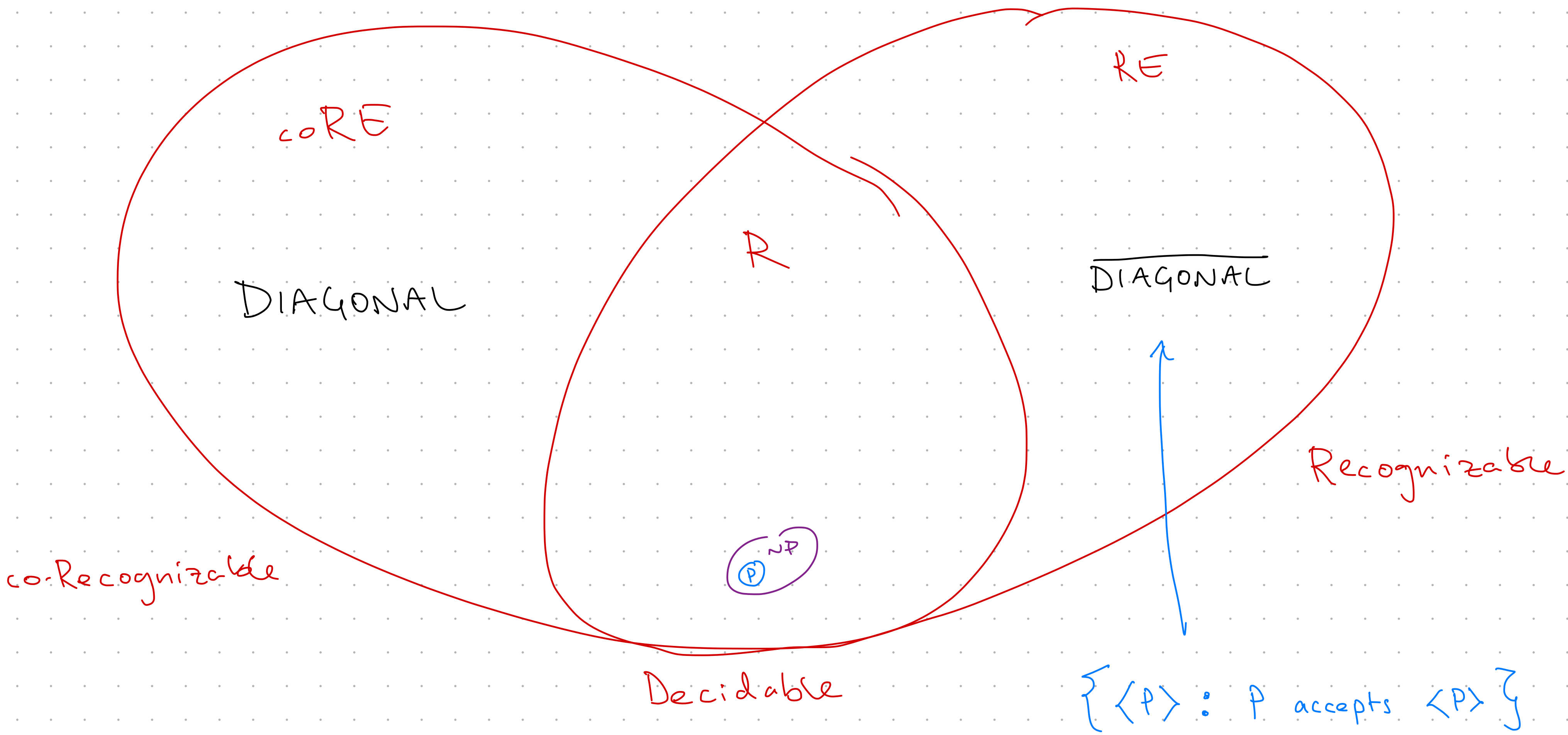
| | $\langle P_1 \rangle$ | $\langle P_2 \rangle$ | $\langle P_3 \rangle$ | --- | | | |
|---|---|---|---|---|---|---|---|
| $\mathcal{L}(P_1)$ | $\boxed{0}$ | 1 | 1 | 0 | 1 | 1 | --- |
| $\mathcal{L}(P_2)$ | 1 | $\boxed{0}$ | 1 | 1 | 0 | 1 | --- |
| $\mathcal{L}(P_3)$ | 1 | 1 | $\boxed{1}$ | 1 | 0 | 0 | --- |
| $\vdots$ | 0 | 1 | 1 | $\boxed{0}$ | 1 | 1 | --- |
| DIAGONAL → | $\boxed{1}$ | $\boxed{1}$ | $\boxed{0}$ | $\boxed{1}$ | --- | | |

$$\langle P_i \rangle \in \text{DIAGONAL} \iff \langle P_i \rangle \notin \mathcal{L}(P_i)$$

# Undecidable Problems $L \notin R$



# Undecidable Problems $L \notin R$



# So what?

* DIAGONAL is quite contrived ...

## So what?

* DIAGONAL is quite contrived ...

* Idea: Reductions!

  ↳ show that other problems capture DIAGONAL

  $\Longrightarrow$ these problems are undecidable.

$HALT = \{ \langle Q, x \rangle : \text{Program } Q \text{ halts on input } x \}$

<u>Theorem</u> $HALT \in RE \setminus R$.

- Recognizable, but Undecidable.

<u>Pf.</u> By <u>Reduction</u> from $\overline{\text{DIAGONAL}}$.

Finite running time.

$\text{HALT} = \{ \langle Q, x \rangle :$ Program $Q$ halts on input $x$ $\}$

**Theorem** $\text{HALT} \in \text{RE} \setminus \text{R}$.

– Recognizable, but Undecidable.

Pf. By Reduction from $\overline{\text{DIAGONAL}}$.

↳ Finite running time.

$$\langle P \rangle \xrightarrow[\text{reduction}]{\text{computable}} \langle Q, x \rangle$$

s.t.

$P$ accepts $\langle P \rangle \iff Q$ halts on $x$

$$\overline{\text{DIAGONAL}} = \{ \langle P \rangle : \text{Program } P \text{ accepts } \langle P \rangle \}$$

$$\text{HALT} = \{ \langle Q, x \rangle : \text{Program } Q \text{ halts on input } x \}$$

$$\langle P \rangle \xrightarrow[\text{reduction}]{\text{computable}} \langle Q, x \rangle$$

**Reduction**

on input $\langle P \rangle$ write description of $Q$ defined as

$\overline{\text{DIAGONAL}} = \{ \langle P \rangle : \text{Program } P \text{ accepts } \langle P \rangle \}$

$\text{HALT} = \{ \langle Q, x \rangle : \text{Program } Q \text{ halts on input } x \}$

$$\langle P \rangle \xrightarrow[\text{reduction}]{\text{computable}} \langle Q, x \rangle$$

Reduction

on input $\langle P \rangle$ write description of $Q$ defined as

Q:
On input $x$,
Run $P$ on $x$.
if $P$ Accepts, $\boxed{\text{accept}}$
if $P$ rejects, $\boxed{\text{Enter infinite loop}}$

Output $\langle Q, \langle P \rangle \rangle$

Consider behavior of $Q$ running on $\langle P \rangle$

$x = \langle P \rangle \longrightarrow$

On input $x$,

Run $P$ on $x$.

if $P$ Accepts, $\boxed{\text{accept}}$

if $P$ rejects,

$\boxed{\text{Enter infinite loop}}$

Consider behavior of Q running on ⟨P⟩

$x = \langle P \rangle$ →

On input ⟨P⟩
Run P on ⟨P⟩
if P Accepts, $\boxed{\text{accept}}$
if P rejects,
$\boxed{\text{Enter infinite loop}}$

if P accepts ⟨P⟩, Then Q halts

Consider behavior of $Q$ running on $\langle P \rangle$

$x = \langle P \rangle \longrightarrow$

> On input $\langle P \rangle$
> Run $P$ on $\langle P \rangle$
> if $P$ Accepts, $\boxed{\text{accept}}$
> if $P$ rejects, $\boxed{\text{Enter infinite loop}}$

if $P$ accepts $\langle P \rangle$, Then $Q$ halts

if $\underbrace{P \text{ does } \underline{\text{NOT}} \text{ accept } \langle P \rangle}$

$P$ rejects $\langle P \rangle \implies Q$ loops forever

Consider behavior of Q running on ⟨P⟩

x = ⟨P⟩ →

On input ⟨P⟩
Run P on ⟨P⟩
if P Accepts, $\boxed{\text{accept}}$
if P rejects, $\boxed{\text{Enter infinite loop}}$

if P accepts ⟨P⟩, Then Q halts

if $\underbrace{\text{P does NOT accept } \langle P \rangle}$

P rejects ⟨P⟩ ⟹ Q loops forever

P does not halt ⟹ $\underbrace{\text{Q does not halt}}$

Q does not halt.

# Undecidable Problems

$L \notin R$