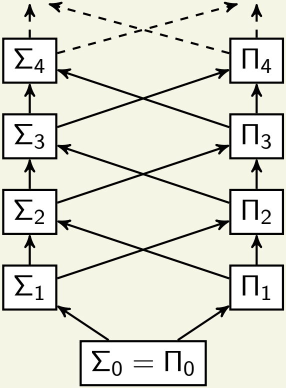
Computability and Complexity
COSC 4200

The Arithmetical Hiearchy

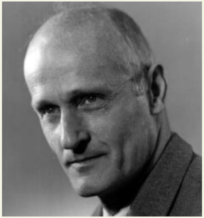
Arithmetical Hierarchy

The Arithmetical Hierarchy is a collection of classes that sit above Turing-recognizable and co-Turing-recognizable.



The Arithmetical Hierarchy

$\Pi_0 = \Sigma_0 =$ decidable
$\Sigma_1 =$ Turing-recognizable
$\Pi_1 =$ co-Turing-recognizable



Stephan Kleene
(1909-1994)

Quantifiers and Predicates

Recall that we proved another way to define Turing-recognizability is in terms of existential quantifiers and decidable predicates.

Theorem

A language A is Turing-recognizable if and only if there is a decidable language D such that for all $x \in \Sigma^*$,

$$x \in A \iff (\exists w \in \Sigma^*)\ \langle x, w \rangle \in D.$$

Quantifiers and Predicates

Theorem

A language A is Turing-recognizable if and only if there is a decidable language D such that for all $x \in \Sigma^*$,

$$x \in A \iff (\exists w \in \Sigma^*)\ \langle x, w \rangle \in D.$$

If we instead use a $\forall$ quantifier, we get another way to define co-Turing-recognizability.

Corollary

A language A is co-Turing-recognizable if and only if there is a decidable language D such that for all $x \in \Sigma^*$,

$$x \in A \iff (\forall w \in \Sigma^*)\ \langle x, w \rangle \in D.$$

Proof of Corollary. Let A be co-Turing-recognizable.

Proof of Corollary. Let A be co-Turing-recognizable. Then A^c is Turing-recognizable.

Proof of Corollary. Let A be co-Turing-recognizable. Then A^c is Turing-recognizable. By the Theorem, there is a decidable language C such that for all $x \in \Sigma^*$,

$$x \in A^c \iff (\exists w \in \Sigma^*)\ \langle x, w \rangle \in C.$$

Proof of Corollary. Let A be co-Turing-recognizable. Then A^c is Turing-recognizable. By the Theorem, there is a decidable language C such that for all $x \in \Sigma^*$,

$$x \in A^c \iff (\exists w \in \Sigma^*)\ \langle x, w \rangle \in C.$$

Let $D = C^c$.

Proof of Corollary. Let A be co-Turing-recognizable. Then A^c is Turing-recognizable. By the Theorem, there is a decidable language C such that for all $x \in \Sigma^*$,

$$x \in A^c \iff (\exists w \in \Sigma^*)\ \langle x, w \rangle \in C.$$

Let $D = C^c$. We have

$$x \in A \quad \iff \quad x \notin A^c$$

Proof of Corollary. Let A be co-Turing-recognizable. Then A^c is Turing-recognizable. By the Theorem, there is a decidable language C such that for all $x \in \Sigma^*$,

$$x \in A^c \iff (\exists w \in \Sigma^*)\ \langle x, w \rangle \in C.$$

Let $D = C^c$. We have

$$\begin{aligned} x \in A &\iff x \notin A^c \\ &\iff \neg\left[(\exists w \in \Sigma^*)\ \langle x, w \rangle \in C\right] \end{aligned}$$

Proof of Corollary. Let A be co-Turing-recognizable. Then A^c is Turing-recognizable. By the Theorem, there is a decidable language C such that for all $x \in \Sigma^*$,

$$x \in A^c \iff (\exists w \in \Sigma^*)\ \langle x, w \rangle \in C.$$

Let $D = C^c$. We have

$$\begin{aligned} x \in A &\iff x \notin A^c \\ &\iff \neg\left[(\exists w \in \Sigma^*)\ \langle x, w \rangle \in C\right] \\ &\iff (\forall w \in \Sigma^*)\ \neg\left[\langle x, w \rangle \in C\right] \end{aligned}$$

Proof of Corollary. Let A be co-Turing-recognizable. Then A^c is Turing-recognizable. By the Theorem, there is a decidable language C such that for all $x \in \Sigma^*$,

$$x \in A^c \iff (\exists w \in \Sigma^*)\ \langle x, w \rangle \in C.$$

Let $D = C^c$. We have

$$\begin{aligned} x \in A &\iff x \notin A^c \\ &\iff \neg\left[(\exists w \in \Sigma^*)\ \langle x, w \rangle \in C\right] \\ &\iff (\forall w \in \Sigma^*)\ \neg\left[\langle x, w \rangle \in C\right] \\ &\iff (\forall w \in \Sigma^*)\ \langle x, w \rangle \notin C \end{aligned}$$

Proof of Corollary. Let A be co-Turing-recognizable. Then A^c is Turing-recognizable. By the Theorem, there is a decidable language C such that for all $x \in \Sigma^*$,

$$x \in A^c \iff (\exists w \in \Sigma^*) \; \langle x, w \rangle \in C.$$

Let $D = C^c$. We have

$$\begin{aligned}
x \in A &\iff x \notin A^c \\
&\iff \neg\, [(\exists w \in \Sigma^*) \; \langle x, w \rangle \in C] \\
&\iff (\forall w \in \Sigma^*) \; \neg\, [\langle x, w \rangle \in C] \\
&\iff (\forall w \in \Sigma^*) \; \langle x, w \rangle \notin C \\
&\iff (\forall w \in \Sigma^*) \; \langle x, w \rangle \in D.
\end{aligned}$$

Proof of Corollary. Let A be co-Turing-recognizable. Then A^c is Turing-recognizable. By the Theorem, there is a decidable language C such that for all $x \in \Sigma^*$,

$$x \in A^c \iff (\exists w \in \Sigma^*)\ \langle x, w \rangle \in C.$$

Let $D = C^c$. We have

$$\begin{aligned}
x \in A &\iff x \notin A^c \\
&\iff \neg\left[(\exists w \in \Sigma^*)\ \langle x, w \rangle \in C\right] \\
&\iff (\forall w \in \Sigma^*)\ \neg\left[\langle x, w \rangle \in C\right] \\
&\iff (\forall w \in \Sigma^*)\ \langle x, w \rangle \notin C \\
&\iff (\forall w \in \Sigma^*)\ \langle x, w \rangle \in D.
\end{aligned}$$

This proves the left-to-right direction. The proof of the right-to-left direction is similar. $\square$

Defining the Arithmetial Hierarchy

The arithmetical hierarchy is defined inductively using both $\exists$ and $\forall$ quantifiers.

Defining the Arithmetial Hierarchy

The arithmetical hierarchy is defined inductively using both $\exists$ and $\forall$ quantifiers.

- Define
$$\Sigma_0 = \Pi_0 = \{A \mid A \text{ is decidable}\}.$$

Defining the Arithmetial Hierarchy

The arithmetical hierarchy is defined inductively using both $\exists$ and $\forall$ quantifiers.

- Define
$$\Sigma_0 = \Pi_0 = \{A \mid A \text{ is decidable}\}.$$
- For $k \geq 1$, the class Σ_k consists of all B such that for some $A \in \Pi_{k-1}$,
$$x \in B \iff (\exists w)\ \langle x, w \rangle \in A.$$

Defining the Arithmetial Hierarchy

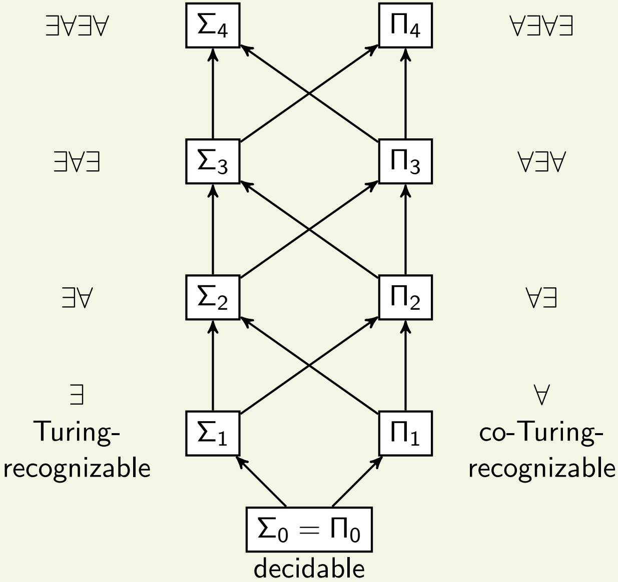
The arithmetical hierarchy is defined inductively using both $\exists$ and $\forall$ quantifiers.

- Define
$$\Sigma_0 = \Pi_0 = \{A \mid A \text{ is decidable}\}.$$
- For $k \geq 1$, the class Σ_k consists of all B such that for some $A \in \Pi_{k-1}$,
$$x \in B \iff (\exists w)\ \langle x, w \rangle \in A.$$
- For $k \geq 1$, define
$$\Pi_k = \{A \mid A^c \in \Sigma_k\}.$$

Defining the Arithmetial Hierarchy

The arithmetical hierarchy is defined inductively using both $\exists$ and $\forall$ quantifiers.

- Define
$$\Sigma_0 = \Pi_0 = \{A \mid A \text{ is decidable}\}.$$
- For $k \geq 1$, the class Σ_k consists of all B such that for some $A \in \Pi_{k-1}$,
$$x \in B \iff (\exists w)\ \langle x, w \rangle \in A.$$
- For $k \geq 1$, define
$$\Pi_k = \{A \mid A^c \in \Sigma_k\}.$$
Equivalently, Π_k consists of all B such that for some $A \in \Sigma_{k-1}$,
$$x \in B \iff (\forall w)\ \langle x, w \rangle \in A.$$



The Arithmetical Hierarchy

First Level

$A \in \Sigma_1$ if there is a decidable B such that for all $x \in \Sigma^*$,

$$x \in A \iff (\exists w)\ \langle x, w \rangle \in B.$$

$\Sigma_1 =$ Turing-recognizable

First Level

$A \in \Sigma_1$ if there is a decidable B such that for all $x \in \Sigma^*$,

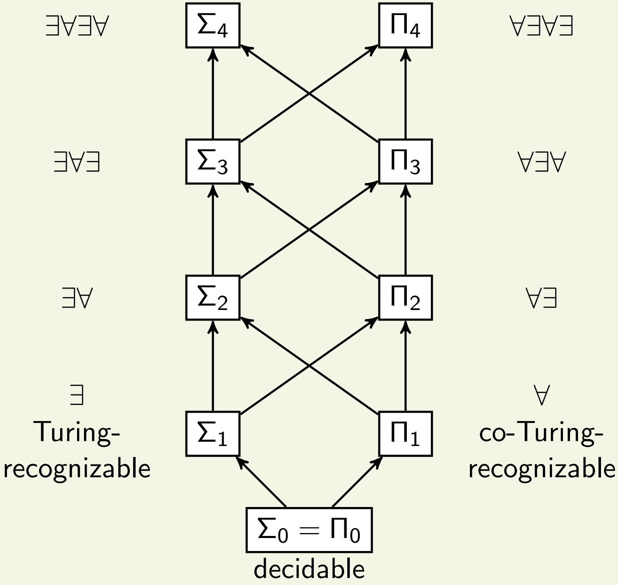
$$x \in A \iff (\exists w)\ \langle x, w \rangle \in B.$$

$\Sigma_1 =$ Turing-recognizable

$A \in \Pi_1$ if there is a decidable B such that for all $x \in \Sigma^*$,

$$x \in A \iff (\forall w)\ \langle x, w \rangle \in B.$$

$\Pi_1 =$ co-Turing-recognizable



The Arithmetical Hierarchy

Second Level

$A \in \Sigma_2$ if there is a decidable B such that for all $x \in \Sigma^*$,

$$x \in A \iff (\exists y)(\forall z)\ \langle x, y, z \rangle \in B.$$

Second Level

$A \in \Sigma_2$ if there is a decidable B such that for all $x \in \Sigma^*$,

$$x \in A \iff (\exists y)(\forall z)\ \langle x, y, z \rangle \in B.$$

$A \in \Pi_2$ if there is a decidable B such that for all $x \in \Sigma^*$,

$$x \in A \iff (\forall y)(\exists z)\ \langle x, y, z \rangle \in B.$$

Third Level

$A \in \Sigma_3$ if there is a decidable B such that for all $x \in \Sigma^*$,

$$x \in A \iff (\exists w)(\forall y)(\exists z)\ \langle x, w, y, z \rangle \in B.$$

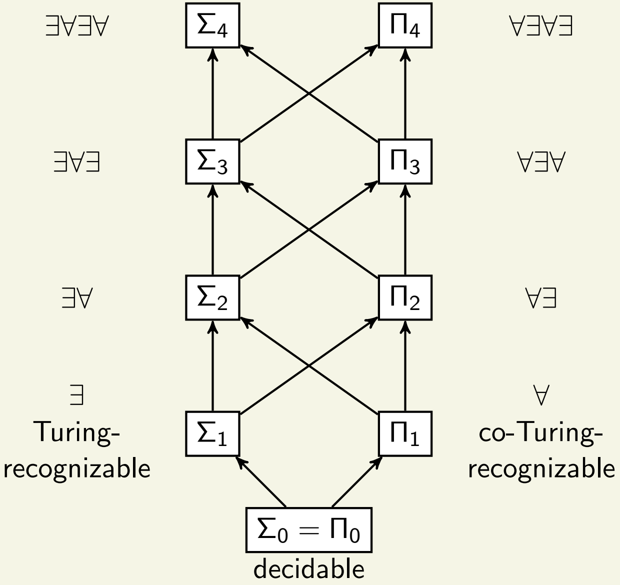
Third Level

$A \in \Sigma_3$ if there is a decidable B such that for all $x \in \Sigma^*$,

$$x \in A \iff (\exists w)(\forall y)(\exists z)\ \langle x, w, y, z \rangle \in B.$$

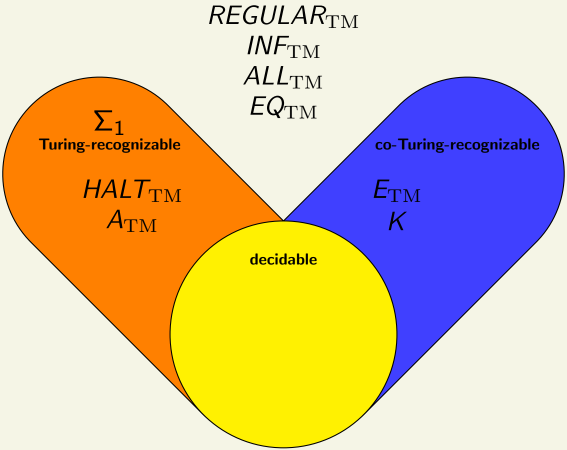
$A \in \Pi_3$ if there is a decidable B such that for all $x \in \Sigma^*$,

$$x \in A \iff (\forall w)(\exists y)(\forall z)\ \langle x, w, y, z \rangle \in B.$$



The Arithmetical Hierarchy

Decision Problems about TMs



Let's use the Arithmetical Hierarchy to classify these problems and some new ones:

$$FIN_{\mathrm{TM}} = \{\langle M\rangle \mid L(M) \text{ is finite}\}$$

$$INFCOMP_{\mathrm{TM}} = \{\langle M\rangle \mid L(M)^c \text{ is infinite}\}$$

Acceptance Problem for TMs

Recall the *acceptance problem* for TMs:

$$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input string } w\ \}.$$

We showed A_{TM} is undecidable and Turing-recognizable. Let's see how A_{TM} meets the quantifier definition for Σ_1:

Acceptance Problem for TMs

Recall the *acceptance problem* for TMs:

$$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input string } w\ \}.$$

We showed A_{TM} is undecidable and Turing-recognizable. Let's see how A_{TM} meets the quantifier definition for Σ_1:

We have

$$\begin{aligned} \langle M, w\rangle \in A_{\mathrm{TM}} \quad &\iff \quad M \text{ accepts } w \\ &\iff \quad (\exists t)\ M \text{ accepts } w \text{ in } t \text{ steps.} \end{aligned}$$

Acceptance Problem for TMs

Recall the *acceptance problem* for TMs:

$$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input string } w\ \}.$$

We showed A_{TM} is undecidable and Turing-recognizable. Let's see how A_{TM} meets the quantifier definition for Σ_1:

We have

$$\begin{aligned} \langle M, w\rangle \in A_{\mathrm{TM}} \quad &\Longleftrightarrow \quad M \text{ accepts } w \\ &\Longleftrightarrow \quad (\exists t)\ M \text{ accepts } w \text{ in } t \text{ steps.} \end{aligned}$$

Define

$$D = \{\langle M, x, t\rangle \mid \ \text{accepts } x \text{ in } t \text{ steps}\}.$$

Acceptance Problem for TMs

Recall the *acceptance problem* for TMs:

$$A_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that accepts input string } w\ \}.$$

We showed A_{TM} is undecidable and Turing-recognizable. Let's see how A_{TM} meets the quantifier definition for Σ_1:

We have

$$\begin{aligned} \langle M, w\rangle \in A_{\mathrm{TM}} \quad &\Longleftrightarrow \quad M \text{ accepts } w \\ &\Longleftrightarrow \quad (\exists t)\ M \text{ accepts } w \text{ in } t \text{ steps.} \end{aligned}$$

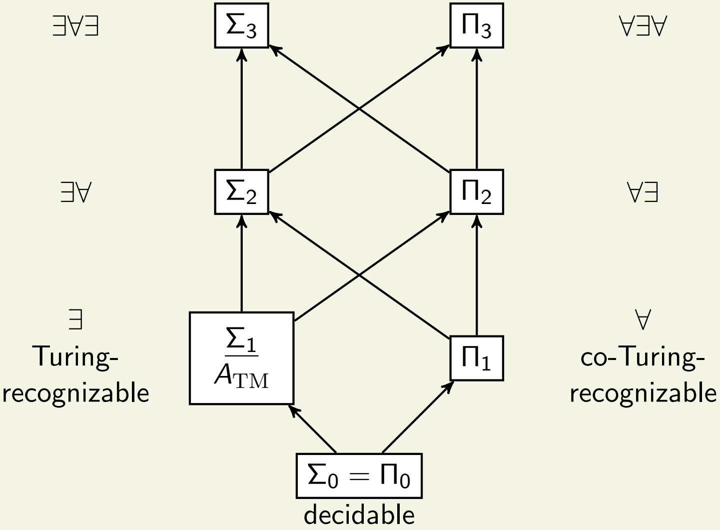
Define

$$D = \{\langle M, x, t\rangle \mid \text{ accepts } x \text{ in } t \text{ steps}\}.$$

Then

$$\langle M, w\rangle \in A_{\mathrm{TM}} \iff (\exists t)\ \langle\langle M, w\rangle, t\rangle \in D.$$

Since D is a decidable predicate and we used $\exists$, this shows $A_{\mathrm{TM}} \in \Sigma_1$.



The Arithmetical Hierarchy

Halting Problem for TMs

Recall the *halting problem* for TMs:

$HALT_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that halts on input string } w\ \}.$

We showed $HALT_{\mathrm{TM}}$ is undecidable and Turing-recognizable. Let's see how $HALT_{\mathrm{TM}}$ meets the quantifier definition for Σ_1:

Halting Problem for TMs

Recall the *halting problem* for TMs:

$HALT_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that halts on input string } w\ \}.$

We showed $HALT_{\mathrm{TM}}$ is undecidable and Turing-recognizable. Let's see how $HALT_{\mathrm{TM}}$ meets the quantifier definition for Σ_1:

We have

$$\begin{aligned}\langle M, w\rangle \in HALT_{\mathrm{TM}} \quad &\iff \quad M \text{ halts on } w \\ &\iff \quad (\exists t)\ M \text{ halts on } w \text{ in } t \text{ steps.}\end{aligned}$$

Halting Problem for TMs

Recall the *halting problem* for TMs:

$HALT_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that halts on input string } w \}.$

We showed $HALT_{\mathrm{TM}}$ is undecidable and Turing-recognizable. Let's see how $HALT_{\mathrm{TM}}$ meets the quantifier definition for Σ_1:

We have

$$
\begin{aligned}
\langle M, w\rangle \in HALT_{\mathrm{TM}} \quad &\Longleftrightarrow \quad M \text{ halts on } w \\
&\Longleftrightarrow \quad (\exists t)\ M \text{ halts on } w \text{ in } t \text{ steps.}
\end{aligned}
$$

Define

$$D = \{\langle M, x, t\rangle \mid \text{ halts on } x \text{ in } t \text{ steps}\}.$$

Halting Problem for TMs

Recall the *halting problem* for TMs:

$HALT_{\mathrm{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM that halts on input string } w\ \}$.

We showed $HALT_{\mathrm{TM}}$ is undecidable and Turing-recognizable. Let's see how $HALT_{\mathrm{TM}}$ meets the quantifier definition for Σ_1:

We have

$$\begin{aligned} \langle M, w\rangle \in HALT_{\mathrm{TM}} \quad &\iff \quad M \text{ halts on } w \\ &\iff \quad (\exists t)\ M \text{ halts on } w \text{ in } t \text{ steps.} \end{aligned}$$

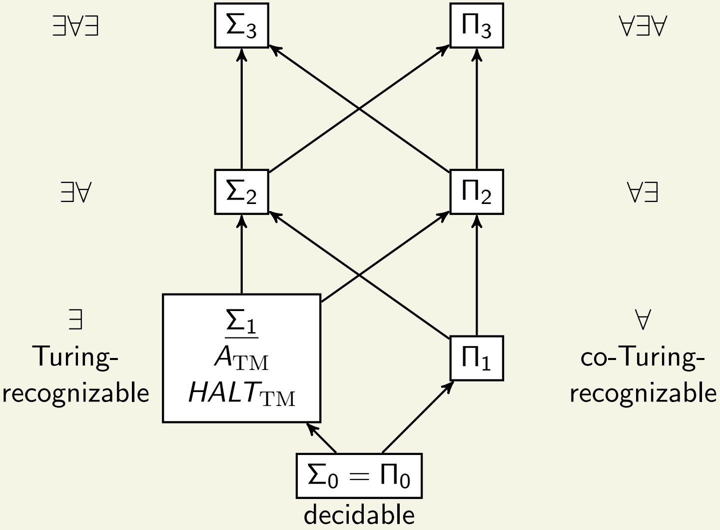
Define

$$D = \{\langle M, x, t\rangle \mid \text{ halts on } x \text{ in } t \text{ steps}\}.$$

Then

$$\langle M, w\rangle \in HALT_{\mathrm{TM}} \iff (\exists t)\ \langle\langle M, w\rangle, t\rangle \in D.$$

Since D is a decidable predicate and we used $\exists$, this shows $HALT_{\mathrm{TM}} \in \Sigma_1$.



The Arithmetical Hierarchy

Emptiness Problem for TMs

Recall the *emptiness problem* for TMs:

$$E_{\mathrm{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset \ \}.$$

We showed E_{TM} is undecidable and co-Turing-recognizable. Let's see how E_{TM} meets the quantifier definition of Π_1:

Emptiness Problem for TMs

Recall the *emptiness problem* for TMs:

$$E_{\mathrm{TM}} = \{\langle M\rangle \mid M \text{ is a TM and } L(M) = \emptyset\ \}.$$

We showed E_{TM} is undecidable and co-Turing-recognizable. Let's see how E_{TM} meets the quantifier definition of Π_1:

We have

$$\langle M\rangle \in E_{\mathrm{TM}} \quad\Longleftrightarrow\quad L(M) = \emptyset$$

Emptiness Problem for TMs

Recall the *emptiness problem* for TMs:

$$E_{\mathrm{TM}} = \{ \langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset \}.$$

We showed E_{TM} is undecidable and co-Turing-recognizable. Let's see how E_{TM} meets the quantifier definition of Π_1:

We have

$$\begin{aligned} \langle M \rangle \in E_{\mathrm{TM}} \quad &\iff \quad L(M) = \emptyset \\ &\iff \quad (\forall x)\ M \text{ does not accept } x \end{aligned}$$

Emptiness Problem for TMs

Recall the *emptiness problem* for TMs:

$$E_{\mathrm{TM}} = \{ \langle M \rangle \mid M \text{ is a TM and } L(M) = \emptyset \}.$$

We showed E_{TM} is undecidable and co-Turing-recognizable. Let's see how E_{TM} meets the quantifier definition of Π_1:

We have

$$\begin{aligned}
\langle M \rangle \in E_{\mathrm{TM}} \quad &\iff \quad L(M) = \emptyset \\
&\iff \quad (\forall x)\ M \text{ does not accept } x \\
&\iff \quad (\forall x)(\forall t)\ M \text{ does not accept } x \text{ in } t \text{ steps}
\end{aligned}$$

Emptiness Problem for TMs

Recall the *emptiness problem* for TMs:

$$E_{\mathrm{TM}} = \{\langle M\rangle \mid M \text{ is a TM and } L(M) = \emptyset\ \}.$$

We showed E_{TM} is undecidable and co-Turing-recognizable. Let's see how E_{TM} meets the quantifier definition of Π_1:

We have

$$\begin{aligned}
\langle M\rangle \in E_{\mathrm{TM}} \quad &\Longleftrightarrow \quad L(M) = \emptyset \\
&\Longleftrightarrow \quad (\forall x)\ M \text{ does not accept } x \\
&\Longleftrightarrow \quad (\forall x)(\forall t)\ M \text{ does not accept } x \text{ in } t \text{ steps} \\
&\Longleftrightarrow \quad (\forall \langle x, t\rangle)\ M \text{ does not accept } x \text{ in } t \text{ steps}
\end{aligned}$$

Emptiness Problem for TMs

Recall the *emptiness problem* for TMs:

$$E_{\mathrm{TM}} = \{\langle M\rangle \mid M \text{ is a TM and } L(M) = \emptyset\ \}.$$

We showed E_{TM} is undecidable and co-Turing-recognizable. Let's see how E_{TM} meets the quantifier definition of Π_1:

We have

$$\begin{aligned}
\langle M\rangle \in E_{\mathrm{TM}} \quad &\Longleftrightarrow \quad L(M) = \emptyset \\
&\Longleftrightarrow \quad (\forall x)\ M \text{ does not accept } x \\
&\Longleftrightarrow \quad (\forall x)(\forall t)\ M \text{ does not accept } x \text{ in } t \text{ steps} \\
&\Longleftrightarrow \quad (\forall \langle x, t\rangle)\ M \text{ does not accept } x \text{ in } t \text{ steps}
\end{aligned}$$

Define

$$D = \{\langle M, \langle x, t\rangle\rangle \mid M \text{ accepts } x \text{ in } t \text{ steps}\}.$$

Emptiness Problem for TMs

Recall the *emptiness problem* for TMs:

$$E_{\mathrm{TM}} = \{\langle M\rangle \mid M \text{ is a TM and } L(M) = \emptyset\ \}.$$

We showed E_{TM} is undecidable and co-Turing-recognizable. Let's see how E_{TM} meets the quantifier definition of Π_1:

We have

$$\begin{aligned} \langle M\rangle \in E_{\mathrm{TM}} \quad &\iff \quad L(M) = \emptyset \\ &\iff \quad (\forall x)\ M \text{ does not accept } x \\ &\iff \quad (\forall x)(\forall t)\ M \text{ does not accept } x \text{ in } t \text{ steps} \\ &\iff \quad (\forall \langle x, t\rangle)\ M \text{ does not accept } x \text{ in } t \text{ steps} \end{aligned}$$

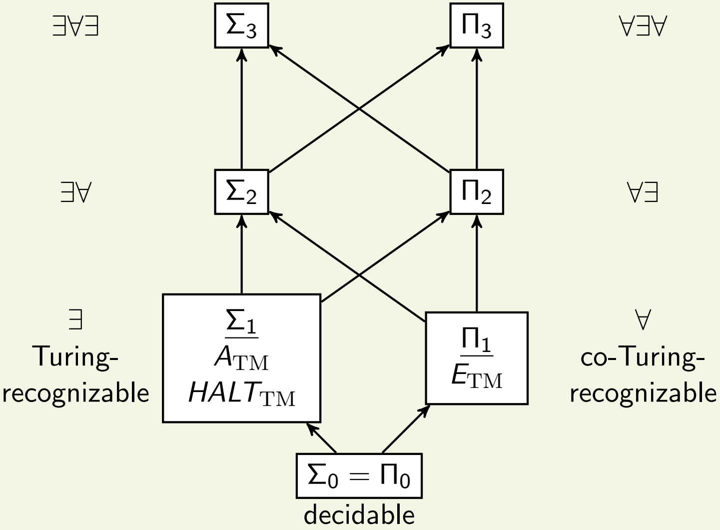
Define

$$D = \{\langle M, \langle x, t\rangle\rangle \mid M \text{ accepts } x \text{ in } t \text{ steps}\}.$$

Then

$$\langle M\rangle \in E_{\mathrm{TM}} \iff (\forall \langle x, t\rangle)\ \langle M, x\rangle \in D.$$

Since D is a decidable predicate and we used $\forall$, this shows



The Arithmetical Hierarchy

Diaganol Halting Problem

Recall the *diaganol halting problem*:

$$K = \{\langle M \rangle \mid M \text{ does not accept } \langle M \rangle\}.$$

We showed K is undecidable and co-Turing-recognizable. Let's see how K meets the quantifier definition of Π_1:

Diaganol Halting Problem

Recall the *diaganol halting problem*:

$$K = \{\langle M \rangle \mid M \text{ does not accept } \langle M \rangle\}.$$

We showed K is undecidable and co-Turing-recognizable. Let's see how K meets the quantifier definition of Π_1:

We have

$$\langle M \rangle \in K \quad \Longleftrightarrow \quad M \text{ does not accept } \langle M \rangle$$

Diaganol Halting Problem

Recall the *diaganol halting problem*:

$$K = \{\langle M \rangle \mid M \text{ does not accept } \langle M \rangle\}.$$

We showed K is undecidable and co-Turing-recognizable. Let's see how K meets the quantifier definition of Π_1:

We have

$$\begin{aligned} \langle M \rangle \in K \quad & \iff \quad M \text{ does not accept } \langle M \rangle \\ & \iff \quad (\forall t)\ M \text{ does not accept } \langle M \rangle \text{ in } t \text{ steps} \end{aligned}$$

Diaganol Halting Problem

Recall the *diaganol halting problem*:

$$K = \{\langle M \rangle \mid M \text{ does not accept } \langle M \rangle\}.$$

We showed K is undecidable and co-Turing-recognizable. Let's see how K meets the quantifier definition of Π_1:

We have

$$\begin{aligned} \langle M \rangle \in K \quad &\iff \quad M \text{ does not accept } \langle M \rangle \\ &\iff \quad (\forall t)\ M \text{ does not accept } \langle M \rangle \text{ in } t \text{ steps} \end{aligned}$$

Define

$$D = \{\langle M, t \rangle \mid M \text{ does not accept } \langle M \rangle \text{ in } t \text{ steps}\}.$$

Diaganol Halting Problem

Recall the *diaganol halting problem*:

$$K = \{\langle M \rangle \mid M \text{ does not accept } \langle M \rangle\}.$$

We showed K is undecidable and co-Turing-recognizable. Let's see how K meets the quantifier definition of Π_1:

We have

$$\begin{aligned} \langle M \rangle \in K \quad &\iff \quad M \text{ does not accept } \langle M \rangle \\ &\iff \quad (\forall t)\ M \text{ does not accept } \langle M \rangle \text{ in } t \text{ steps} \end{aligned}$$

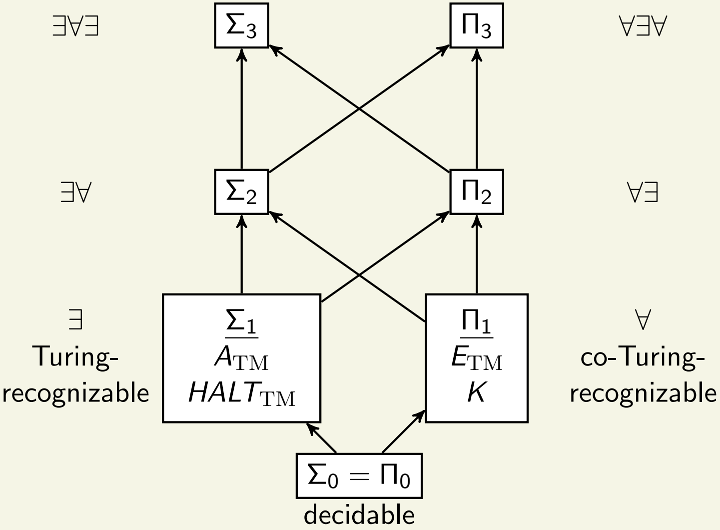
Define

$$D = \{\langle M, t \rangle \mid M \text{ does not accept } \langle M \rangle \text{ in } t \text{ steps}\}.$$

Then

$$\langle M \rangle \in K \iff (\forall t)\ \langle M, t \rangle \in D.$$

Since D is a decidable predicate and we used $\forall$, this shows $K \in \Pi_1$.



The Arithmetical Hierarchy

All Problem for TMs

Recall

$$ALL_{\mathrm{TM}} = \{\langle M \rangle L(M) = \Sigma^*\}$$

Given $\langle M \rangle$, the problem is to determine whether M accepts every string.

We showed that ALL_{TM} is neither Turing-recognizable nor co-Turing-recognizale.

Classifying ALL_{TM}

We have

$$\langle M \rangle \in ALL_{\mathrm{TM}} \iff L(M) = \Sigma^*$$

Classifying ALL_{TM}

We have

$$\begin{aligned} \langle M \rangle \in ALL_{\mathrm{TM}} \quad &\iff \quad L(M) = \Sigma^* \\ &\iff \quad (\forall x)\ M \text{ accepts } x \end{aligned}$$

Classifying ALL_{TM}

We have

$$\begin{aligned}
\langle M \rangle \in ALL_{\mathrm{TM}} \quad &\iff \quad L(M) = \Sigma^* \\
&\iff \quad (\forall x)\ M \text{ accepts } x \\
&\iff \quad (\forall x)(\exists t)\ M \text{ accepts } x \text{ in } t \text{ steps}
\end{aligned}$$

Classifying ALL_{TM}

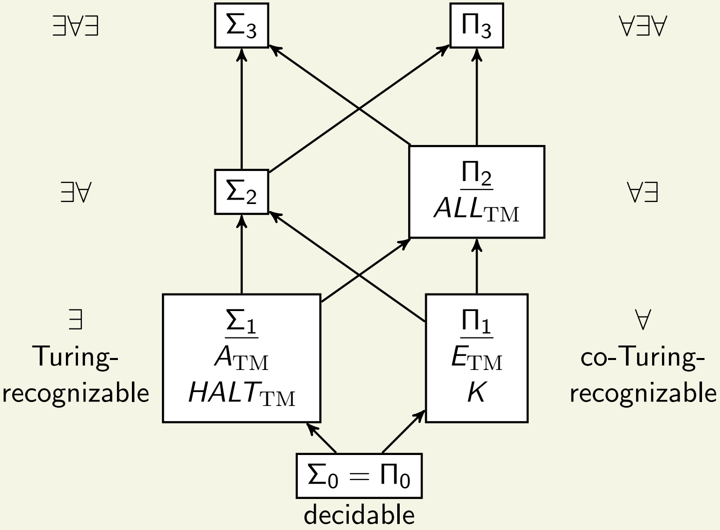
We have

$$\begin{aligned} \langle M \rangle \in ALL_{\mathrm{TM}} \quad &\Longleftrightarrow \quad L(M) = \Sigma^* \\ &\Longleftrightarrow \quad (\forall x)\ M \text{ accepts } x \\ &\Longleftrightarrow \quad (\forall x)(\exists t)\ M \text{ accepts } x \text{ in } t \text{ steps} \\ &\Longleftrightarrow \quad (\forall x)(\exists t)\ \langle M, x, t \rangle \in D \end{aligned}$$

where

$$D = \{\langle M, x, t \rangle \text{ accepts } x \text{ in } t \text{ steps}\}$$

is a decidable predicate. We used $\forall\exists$ with D, so $ALL_{\mathrm{TM}} \in \Pi_2$.



The Arithmetical Hierarchy

Infinite Problem for TMs

The infinite problem for TMs is

$$INF_{\mathrm{TM}} = \{\langle M \rangle \mid L(M) \text{ is infinite}\}.$$

Given $\langle M \rangle$, the problem is to determine whether M accepts infinitely many strings.

INF_{TM} is neither Turing-recognizable nor co-Turing-recognizale.

Classifying $\mathit{INF}_{\mathrm{TM}}$

We have

$$\langle M \rangle \in \mathit{INF}_{\mathrm{TM}} \iff L(M) \text{ is infinite}$$

Classifying $\mathit{INF}_{\mathrm{TM}}$

We have

$$\langle M \rangle \in \mathit{INF}_{\mathrm{TM}} \iff L(M) \text{ is infinite}$$

$$\iff (\forall n)(\exists x)\ |x| \geq n \text{ and } M \text{ accepts } x$$

Classifying INF_{TM}

We have

$$\begin{aligned}
\langle M\rangle \in INF_{\mathrm{TM}} \quad &\iff \quad L(M) \text{ is infinite} \\
&\iff \quad (\forall n)(\exists x)\ |x| \geq n \text{ and } M \text{ accepts } x \\
&\iff \quad (\forall n)(\exists x)(\exists t) \quad \begin{array}{l} |x| \geq n \text{ and} \\ M \text{ accepts } x \text{ in } t \text{ steps} \end{array}
\end{aligned}$$

Classifying INF_{TM}

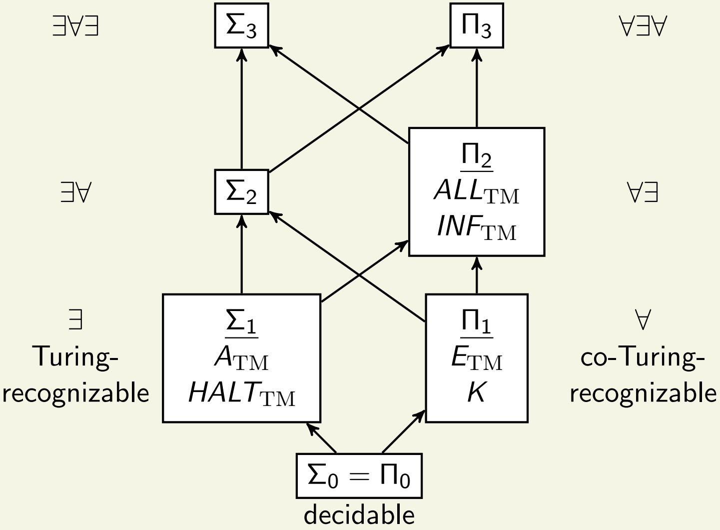
We have

$$
\begin{aligned}
\langle M\rangle \in INF_{\mathrm{TM}} \quad &\iff \quad L(M) \text{ is infinite} \\
&\iff \quad (\forall n)(\exists x)\ |x| \geq n \text{ and } M \text{ accepts } x \\
&\iff \quad (\forall n)(\exists x)(\exists t) \quad \begin{array}{l} |x| \geq n \text{ and} \\ M \text{ accepts } x \text{ in } t \text{ steps} \end{array} \\
&\iff \quad (\forall n)(\exists \langle x, t\rangle)\ \langle M, n, \langle x, t\rangle\rangle \in D,
\end{aligned}
$$

where

$$D = \{\langle M, n, \langle x, t\rangle\rangle \ |x| \geq n \text{ and } M \text{ accepts } x \text{ in } t \text{ steps}\}$$

is a decidable predicate. We used $\forall\exists$ with D, so $INF_{\mathrm{TM}} \in \Pi_2$.



The Arithmetical Hierarchy

Finite Problem for TMs

The finite problem for TMs is

$$\mathit{FIN}_{\mathrm{TM}} = \{\langle M \rangle \mid L(M) \text{ is finite}\}.$$

Given $\langle M \rangle$, the problem is to determine whether M accepts only finitely many strings.

$\mathit{FIN}_{\mathrm{TM}}$ is neither Turing-recognizable nor co-Turing-recognizale.

Classifying $\mathit{FIN}_{\mathrm{TM}}$

We have

$$\langle M\rangle \in \mathit{FIN}_{\mathrm{TM}} \quad \Longleftrightarrow \quad L(M) \text{ is finite}$$

Classifying FIN_{TM}

We have

$$\langle M \rangle \in FIN_{\mathrm{TM}} \iff L(M) \text{ is finite}$$

$$\iff (\exists n) M \text{ accepts no string } x \text{ with } |x| \geq n$$

Classifying FIN_{TM}

We have

$$\begin{aligned}
\langle M \rangle \in FIN_{\mathrm{TM}} \quad &\iff \quad L(M) \text{ is finite} \\
&\iff \quad (\exists n) M \text{ accepts no string } x \text{ with } |x| \geq n \\
&\iff \quad (\exists n)(\forall x)(\forall t) \quad \begin{array}{l} |x| \geq n \text{ or } M \text{ does not} \\ \text{accept } x \text{ in } t \text{ steps} \end{array}
\end{aligned}$$

Classifying FIN_{TM}

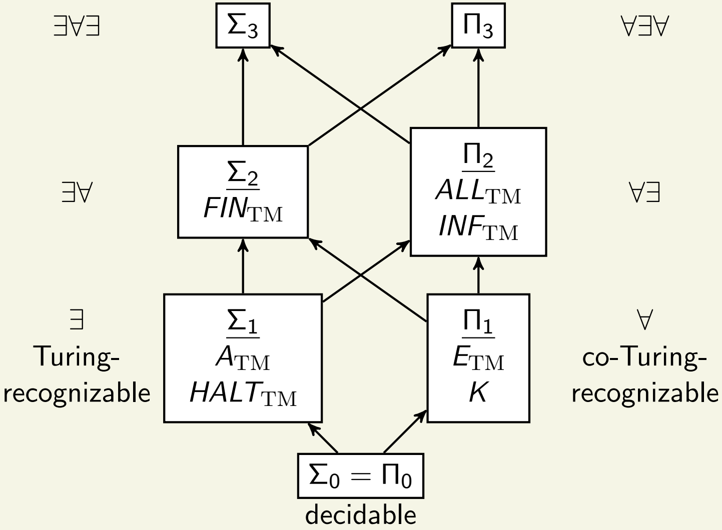
We have

$$\begin{aligned}
\langle M\rangle \in FIN_{\mathrm{TM}} \quad &\iff \quad L(M) \text{ is finite} \\
&\iff \quad (\exists n) M \text{ accepts no string } x \text{ with } |x| \geq n \\
&\iff \quad (\exists n)(\forall x)(\forall t) \quad \begin{array}{l} |x| \geq n \text{ or } M \text{ does not} \\ \text{accept } x \text{ in } t \text{ steps} \end{array} \\
&\iff \quad (\exists n)(\forall \langle x, t\rangle)\ \langle M, n, \langle x, t\rangle\rangle \in D,
\end{aligned}$$

where

$$D = \{\langle M, n, \langle x, t\rangle\rangle \ |x| \geq n \text{ or } M \text{ does not accept } x \text{ in } t \text{ steps}\}$$

is a decidable predicate. We used $\exists\forall$ with D, so $FIN_{\mathrm{TM}} \in \Sigma_2$.



The Arithmetical Hierarchy

Equivalence Problem

Recall the equivalence problem for TMs:

$$EQ_{\mathrm{TM}} = \{\langle M, N\rangle \mid M \text{ and } N \text{ are TMs and } L(M) = L(N)\ \}$$

Given $\langle M, N\rangle$, the problem is to determine whether M and N accept exactly the same strings.

We showed EQ_{TM} is neither Turing-recognizable nor co-Turing-recognizale.

Classifying EQ_{TM}

We have

$$\langle M, N \rangle \in EQ_{\mathrm{TM}} \iff L(M) = L(N)$$

Classifying EQ_{TM}

We have

$$
\begin{aligned}
\langle M, N\rangle \in EQ_{\mathrm{TM}} \quad &\iff \quad L(M) = L(N) \\
&\iff \quad (\forall x) \quad \begin{array}{l} M(x) = N(x) \text{ or} \\ M(x) \text{ and } N(x) \text{ both do not halt} \end{array}
\end{aligned}
$$

Classifying EQ_{TM}

We have

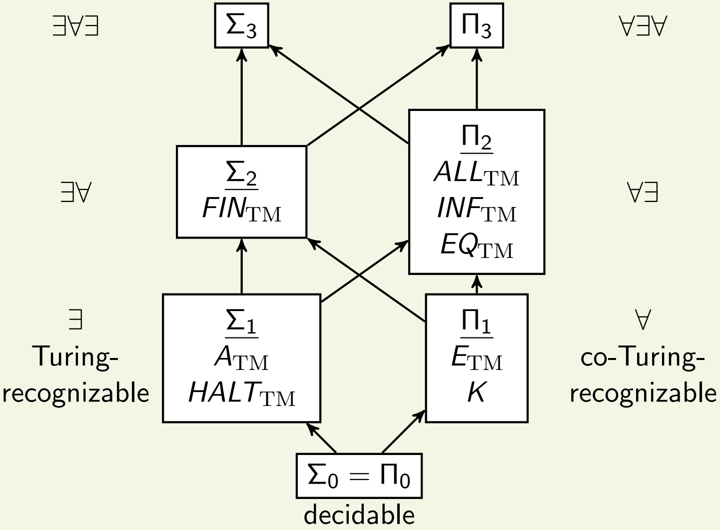
$$\begin{aligned}
\langle M, N\rangle \in EQ_{\mathrm{TM}} \quad &\iff \quad L(M) = L(N) \\
&\iff \quad (\forall x) \quad \begin{array}{l} M(x) = N(x) \text{ or} \\ M(x) \text{ and } N(x) \text{ both do not halt} \end{array} \\
&\iff \quad (\forall x)(\exists t) \quad \begin{array}{l} M(x) \text{ and } N(x) \text{ halt within } t \text{ steps} \\ \quad \text{and give the same answer,} \\ \text{or } M(x) \text{ and } N(x) \text{ both do not halt} \end{array}
\end{aligned}$$

Classifying EQ_{TM}

We have

$$
\begin{aligned}
\langle M, N\rangle \in EQ_{\mathrm{TM}} \quad &\Longleftrightarrow \quad L(M) = L(N) \\
&\Longleftrightarrow \quad (\forall x) \quad \begin{array}{l} M(x) = N(x) \text{ or} \\ M(x) \text{ and } N(x) \text{ both do not halt} \end{array} \\
&\Longleftrightarrow \quad (\forall x)(\exists t) \quad \begin{array}{c} M(x) \text{ and } N(x) \text{ halt within } t \text{ steps} \\ \text{and give the same answer,} \\ \text{or } M(x) \text{ and } N(x) \text{ both do not halt} \end{array} \\
&\Longleftrightarrow \quad (\forall x, n)(\exists t) \quad \begin{array}{c} M(x) \text{ and } N(x) \text{ halt within } t \text{ steps} \\ \text{and give the same answer,} \\ \text{or } M(x) \text{ and } N(x) \text{ both do not} \\ \text{halt within } n \text{ steps} \end{array}
\end{aligned}
$$

This is a decidable predicate with $\forall\exists$, so $EQ_{\mathrm{TM}} \in \Pi_2$.



The Arithmetical Hierarchy

Regularity Problem for TMs

Recall

$REGULAR_{\mathrm{TM}} = \{\langle M \rangle \mid M \text{ is a TM and } L(M) \text{ is a regular language}\}.$

Given $\langle M \rangle$, the problem is determine whether M may be replaced by a DFA that accepts the same language.

$REGULAR_{\mathrm{TM}}$ is neither Turing-recognizable nor co-Turing-recognizale.

We have

$$\begin{aligned}&\langle M\rangle \in \mathit{REGULAR}_{\mathrm{TM}}\\ &\qquad\iff \quad (\exists D)\ D \text{ is a DFA and } L(M) = L(D)\end{aligned}$$

We have

$$
\begin{aligned}
\langle M \rangle \in \mathit{REGULAR}_{\mathrm{TM}} & \\
\iff \quad & (\exists D)\ D \text{ is a DFA and } L(M) = L(D) \\
\iff \quad & (\exists D)(\forall x) \\
& D(x) \text{ accepts } \leftrightarrow M(x) \text{ accepts}
\end{aligned}
$$

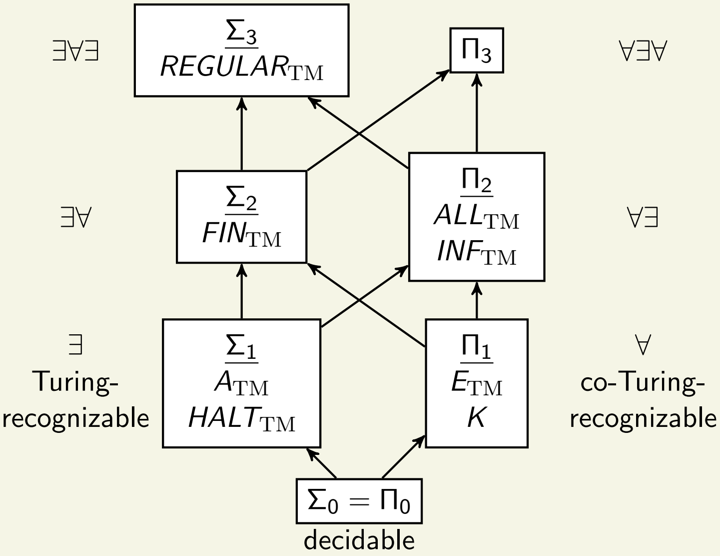
We have

$$
\begin{aligned}
\langle M \rangle \in \mathit{REGULAR}_{\mathrm{TM}} & \\
\iff & \quad (\exists D)\ D \text{ is a DFA and } L(M) = L(D) \\
\iff & \quad (\exists D)(\forall x) \\
& \quad D(x) \text{ accepts } \leftrightarrow M(x) \text{ accepts} \\
\iff & \quad (\exists D)(\forall x) \\
& \quad [D(x) \text{ accepts } \wedge M(x) \text{ accepts}] \vee \\
& \quad [D(x) \text{ rejects } \wedge M(x) \text{ does not accept}]
\end{aligned}
$$

We have

$$
\begin{aligned}
\langle M\rangle \in \mathit{REGULAR}_{\mathrm{TM}} & \\
\iff & \quad (\exists D)\ D \text{ is a DFA and } L(M) = L(D) \\
\iff & \quad (\exists D)(\forall x) \\
& \quad D(x) \text{ accepts } \leftrightarrow M(x) \text{ accepts} \\
\iff & \quad (\exists D)(\forall x) \\
& \quad [D(x) \text{ accepts } \wedge M(x) \text{ accepts}] \vee \\
& \quad [D(x) \text{ rejects } \wedge M(x) \text{ does not accept}] \\
\iff & \quad (\exists D)(\forall x, u)(\exists t) \\
& \quad [D(x) \text{ accepts } \wedge M(x) \text{ accepts in } t \text{ steps}] \vee \\
& \quad [D(x) \text{ rejects } \wedge M(x) \text{ does not accept} \\
& \qquad\qquad \text{in } u \text{ steps}]
\end{aligned}
$$

This is a decidable predicate with $\exists\forall\exists$, so $\mathit{REGULAR}_{\mathrm{TM}} \in \Sigma_3$.



The Arithmetical Hierarchy

Infinite Complement Problem for TMs

The infinite complement problem for TMs is

$$INFCOMP_{\mathrm{TM}} = \{\langle M \rangle \mid L(M)^c \text{ is infinite}\}.$$

Given $\langle M \rangle$, the problem is to determine whether there are infinitely many strings that M does not accept.

$INFCOMP_{\mathrm{TM}}$ is neither Turing-recognizable nor co-Turing-recognizale.

Classifying $INFCOMP_{\mathrm{TM}}$

We have

$$\langle M \rangle \in INFCOMP_{\mathrm{TM}} \iff L(M)^c \text{ is infinite}$$

Classifying $\mathit{INFCOMP}_{\mathrm{TM}}$

We have

$$
\begin{aligned}
\langle M \rangle \in \mathit{INFCOMP}_{\mathrm{TM}} \quad &\iff \quad L(M)^c \text{ is infinite} \\
&\iff \quad (\forall n)(\exists x) \quad \begin{array}{l} |x| \geq n \text{ and} \\ M \text{ does not accept } x \end{array}
\end{aligned}
$$

Classifying $\mathit{INFCOMP}_{\mathrm{TM}}$

We have

$$\begin{aligned}
\langle M\rangle \in \mathit{INFCOMP}_{\mathrm{TM}} &\iff L(M)^c \text{ is infinite} \\
&\iff (\forall n)(\exists x) \quad |x| \geq n \text{ and } M \text{ does not accept } x \\
&\iff (\forall n)(\exists x)(\forall t) \quad |x| \geq n \text{ and } M \text{ does not accept } x \text{ in } t \text{ steps}
\end{aligned}$$

Classifying $\mathit{INFCOMP}_{\mathrm{TM}}$

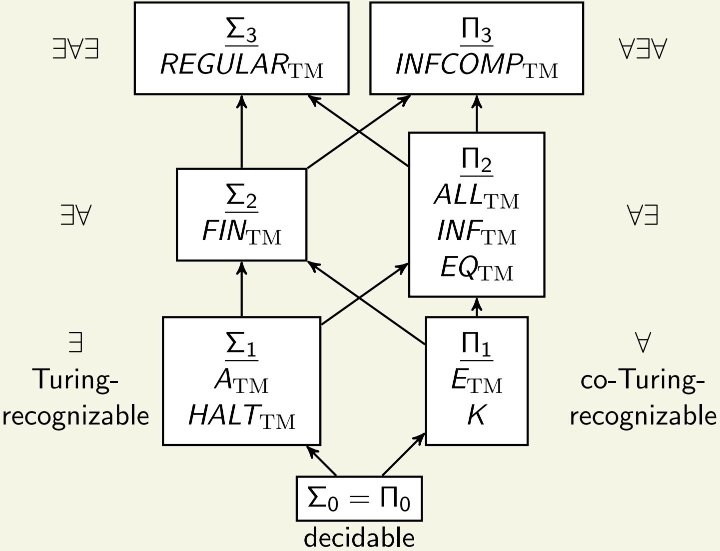
We have

$$\langle M\rangle \in \mathit{INFCOMP}_{\mathrm{TM}} \iff L(M)^c \text{ is infinite}$$

$$\iff (\forall n)(\exists x) \quad \begin{array}{l} |x| \geq n \text{ and} \\ M \text{ does not accept } x \end{array}$$

$$\iff (\forall n)(\exists x)(\forall t) \quad \begin{array}{l} |x| \geq n \text{ and} \\ M \text{ does not accept } x \\ \text{in } t \text{ steps} \end{array}$$

This a decidable predicate with $\forall\exists\forall$, so $\mathit{INFCOMP}_{\mathrm{TM}} \in \Pi_3$.



The Arithmetical Hierarchy