Church-Turing Thesis

"Any computation or algorithm can be expressed as a TM or λ-calculus term."

$\forall c \in$ "computations" . $(\exists t \in TM.\ M(c)=M(t)) \wedge (\exists m \in \lambda.\ M(m)=M(c))$

"computation" = TM = λ-calculus

axiom of faith = provable = =

MTM $\rightarrow$ ✓ λ + number

= $\leftarrow$? should be possible

NTM = λ + class

"problems" = ALL | "algorithms/ideas" = Σ_1 | = | = "programming languages or software"

ITM = C $\leftrightarrow$ = Ocaml

= "hardware" = Racket

compcert

= x86 = Rust

= ~~RAM~~ = JS

= DFC

useful

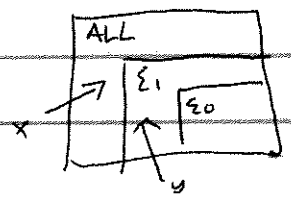
ALL $\subseteq \Sigma_0 \subseteq \Sigma_1$ = possible

ALL = Σ_0 $\rightarrow$ All questions have efficient algorithms to answer

ALL = $\Sigma_1 \wedge \Sigma_0 \neq \Sigma_1$ $\rightarrow$ All questions are possible to answer, but not efficiently

ALL $\neq \Sigma_1 \wedge \Sigma_0 \neq \Sigma_1$ $\rightarrow$ Some questions have no answers and some alg. are inherently not useful

reality $\neq \Sigma_0$



21-2/ Polynomial Root Problem

Polynomial over n variable $(x_1 \dots x_n)$

$$= \sum_{i=0}^{n!} \alpha_i x_1^{c_{1,i}} \cdot \dots \cdot x_n^{c_{n,i}}$$

$$3xy + 4x^2y^3 + 2x^{99}y^{1000}$$

Find a value for all variables s.t. the poly = 0

$ax^2 + bx + c$ $\quad x = \frac{b \pm \sqrt{b^2 - 4ac}}{2a}$ (quadratic eqn, Jay forgets)

Matijasevič's Theorem

If the poly has 1 var, then the root $\in \left[-k \frac{c_{max}}{c_1}, +k \frac{c_{max}}{c_1} \right]$

k = the # of terms

c_{max} = largest coefficient (abs val)

c_1 = coefficient of highest degree

Σ_0 (polynomial) = root

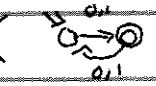
Try all numbers in range

21-3

A_x = acceptance problems

$A_{DFA} = \{ w \in \{0,1\}^* \mid w$ encodes the pair (B, x) where B is a DFA and x is $\in B.\Sigma^*$ and B accepts $x \}$

$\{ \langle (B,x) \rangle \mid B \in DFA$ and $x \in \Sigma^*$ and B accepts $x \}$

$\langle$  , 010 $\rangle \in A_{DFA}$

$\langle$ '' , 01 $\rangle \notin A_{DFA}$

$\langle$ 0^*1^*00 , 00011100 $\rangle \in A_{DFA}$

$\langle$ '' , ε $\rangle \notin A_{DFA}$

	new (interp)	old (compiler)
$A_{DFA}(\langle B, w \rangle) :=$	$A_{DFA} \in \Sigma_0$	$REG \subsetneq \Sigma_0$

copy w to tape 1

copy q_0 to tape 2

→ look at first char on tape 1 → if no char, look at F and see if tape 2 $\in$

consult δ of that and tape 2

update tape 2

go back

yes → accept; no → reject

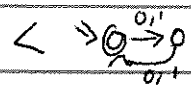
$A_{NFA} \in \Sigma_0$ $\quad A_{NFA} = \{ \langle B, w \rangle \mid B \in NFA, w \in \Sigma^*, w \in \mathcal{L}(B) \}$

$:=$ compile B into $B' \in DFA$, use A_{DFA}

$A_{REX} \in \Sigma_0$ = compile to an NFA and use A_{NFA}

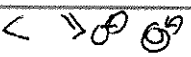
21-4/ Emptiness problems = E_x

$E_{DFA} = \{ \langle B \rangle \mid B \in DFA \text{ and } L(B) = \emptyset \}$



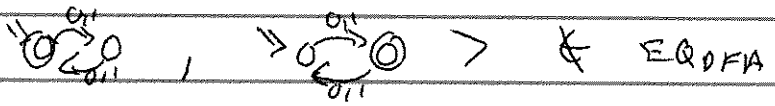
$\in E_{DFA}$ do a graph search (linear)

if F reachable, then no

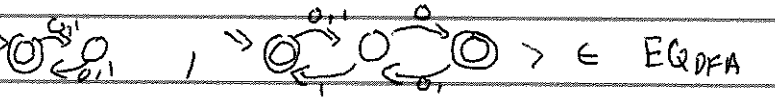

$\in E_{DFA}$ o.w. yes

Equality problems = EQ_x

$EQ_{DFA} = \{ \langle A, B \rangle \mid A \in DFA, B \in DFA, L(A) = L(B) \}$



$\notin EQ_{DFA}$



$\in EQ_{DFA}$