## CSE596 Lecture Wed. Oct. 9

I forgot to snapshot the statement of Rice's Theorem and its proof before erasing the left board, but here is the same proof from last year's notes. The statement is that the only decidable index sets are those of the empty class (which gives the empty language) and of RE (which is Sigma^\*).

Proof: Given Ie when C+O. and C+RE, so there is a G.e. language A such that ither: @ OEC. A & C (\*) Take on MA st. LMA) = A. or DA EC, Ø & C. Build LM, W) C+, M'= (Sim M(W)) iF on! Whenit Whenit a(UTIB Run MA(X)) O iff MA does.  $L_{M,W} \in A_{FM} \Rightarrow L(M') = A \Rightarrow M' \notin I_{e} : A_{FM} \neq M'$   $L_{M,W} \notin A_{FM} \Rightarrow L(M') = 0 \Rightarrow M' \in I_{e} : A_{FM} \neq M'$ Fn (ax (a)  $\begin{array}{c} fn (ave (b) \\ (M,w) \in A_{fm} \Rightarrow L(m') = A \Rightarrow ((m')) \in \mathcal{C} \Rightarrow M' \in Ie^{-A_{fm} \leq m_{e}^{2}} \\ (M,w) \in A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin A_{fm} \Rightarrow L(m') = Q \Rightarrow M' \notin Ie since Q \notin \mathcal{C} in This \\ (ave ) \notin \mathcal{C} in This \\ (ave ) \notin \mathcal{C} in This \\ (ave ) \# (ave ) \oplus \mathcal{C} in This \\ (ave ) \# (ave ) \oplus \mathcal{C} in This \\ (ave ) \# (ave ) \# (ave ) \oplus \mathcal{C} in This \\ (ave ) \# (ave ) \# (ave ) \# (ave ) \oplus \mathcal{C} in This \\ (ave ) \# (ave ) \# (ave ) \oplus \mathcal{C} in This \\ (ave ) \# (ave ) \#$ En race (b) Either way, Ze if undecidable. 18 Example: C = REG case (b) applies with A = { palindrames }, so IREG is underidable.

You can see the conclusions about the reductions in the right-hand board snap as the lecture continued:

There is a similar treasen For sets of programs P of the form computations J= S(P): some property T that depends only on the or proofs!)1 Partial function TP = Z partial Z computed by P. Note $<math display="block">L(M) is the same as the function L(K) = \begin{cases} 1 & if M accepts x \\ (Partial) \end{cases}$ RE Either way, Ic or JIT expresses an extensional property of programs. Historial Import " Every nontrivial extensional property of programs is underilable FmI, What about Intensional properties, meaning how code is written? Note: IREN Consider programs PI I input X Impati Every proget Ic < M, W) CF P= (Similat M(W)) When are doing behavior B if only behavior B if and when it Allep's (Similat M(W)) When are doing behavior B if and when it Allep's JUMP BACK lie. élective program behavior INST: An assembly program F and an input x to P. (P,x) QUES: Does P(x) ever jump C LM, W)& AM =) Vx, p'(x) numy Do behavior B Either way, I.e. D does B. sing TMs are simulate is undecidable. D without doing B B back to an earlier-numbered line & code? Decidable? nessential for compatition. Escontial?

Ignore that part in red pen in the last snap:

3: For every formal entity E (machine or grammer a ote) that is capable of checking computations the (non-) emptiness problem for E and much else is under lable There is a similar theorem (or proofs!) 1 J= 3(P): som an orm Proof by Map M C Emi partial funchin TP: 2 5 X1 X2 --- X5 Example -[(M) is the same as the E= a Two-Tupe DFA Either way, Ip or JT has its 2 which receives its input Historial Import "Every non erral to the IO alphabet Em is able to check the local conditions for Io to I, Z on both tapes. What about Intensional then I, Im Iz, and so on. QVF of M, Consider programs P Either tape head can more R or stay (S) It accepts if and when it finds an acc JD together with {#? J with and of M as it states i Jobs 1= J so XellM). Thus LIEm) = Ø < M, W) CIP P= but cannot change a char. (m, W) (AT => Vx P(x) des B An En I C (MN)& Am = Vx, P'(2) north Eliter way I C dos B, sing The are simulat is underidable. D without dring B ⇒ LIM) ≠ Ø so Ein ≤ empthed to En