ANNOUNCEMENT: Final will be a take-home final, given on Friday and due by end of day Monday (and great happiness ensued)

Q. Does a DFA D accept an input w?
Can you write a program to answer this question? Yes.
Can you make a Turing Machine answer this question? How?
Encode D and w as input to a TM (strings)

\[
D = (Q, \Sigma, \delta, q_0, F)
\]

\[
\begin{align*}
q_0, q_1, \ldots & \rightarrow 0,13 \\
q_0, q_0 & \rightarrow q_{17} \\
q_{14}, 1 & \rightarrow q_7
\end{align*}
\]

Input string = \( <D,w> \)

\[
D = \# q_0, q_1, \ldots \# 0,1 \# q_0, q_0, q_{17} \# q_{14}, 1, q_7 \# \ldots
\]

\[
F = w
\]

\[
\ldots \# q_0 \# q_1, q_4, q_9 \# 0110
\]

\[\text{TM 1 Language } A_{\text{DFA}} = \{ <D,w> | D \text{ is a DFA that accepts } w \}\]

is decidable

PF. Consider decide TM \( M_1 = \)

Levels of description of TMs
1. Formal: \( M = (Q, \Sigma, \Gamma, \delta, q_0, \text{acc, rej}) \)

2. Implementation level: Describe TM in English: tape, etc.

3. High level: algorithm (we can do this because the Church-Turing thesis tells us that all algorithms can be implemented as TMs)
A DFA is Turing-Decidable:

AF.

Consider decision TM $M_1 = (Q, \Sigma, \Gamma, \delta, q_0, A, \emptyset)$

ON INPUT $S$:

1. Check that $S = \langle D, w \rangle$
   (if not, reject $S$)
2. Simulate $D$ on $w$

IDEA: 3 tapes:

```
# 0 1 2 3 ...
```

```
0 1 1 0
```

Current state of DFA

```
<q_0, 9, 1>
```

Loop:

Read the state, read the input, read the transition, write the new state

At end of input:

Check current state to see if it's an accept state. Accept if yes, reject if no.

ANFA is Turing-Decidable:

$\forall NFA = \forall \langle N, w \rangle | NFA N accepts w \exists$

$M_2: \langle \text{Conver}t N \text{ to DFA } D_N \text{ and run } M_1 \text{ on } \langle D_N, w \rangle \rangle$

A REG = $\forall \langle R, w \rangle | \text{Reg exp } R \text{ generates } w \exists$

$\langle \text{Conver}t R \text{ to equivalent NFA } N_R \text{ AND run } M_2 \text{ on } \langle N_R, w \rangle \rangle$

A REG is Turing-Decidable
A_{CFG} \text{ is Turing-decidable:}

A_{CFG} = \exists <G, w> \mid \text{CFG } G \text{ generates } w

(\text{convert } G \text{ to Chomsky Normal Form:} \nw \text{ is generated in } 2|w|-1 \text{ steps})

Now bound the machine: if \nw \text{ not generated within } 2|w|-1 \text{ steps} \to \text{reject}
(\text{otherwise risk infinitely loopy TM})

A_{TM} = \exists <M, w> \mid \text{Tm } M \text{ accepts } w

Is \text{ } A_{TM} \text{ Turing-decidable?} \quad \text{NO!}

[See slides]

Reducibility and the Halting Problem

[See slides]