A - Acceptance Problem for DFAs.

INSTANCE: A DFA \( M = (Q, \Sigma, \delta, s, F) \) and an argument \( x \in \Sigma^* \)

QUESTION: Does \( M \) accept \( x \)?

The language of the problem is the set of instances for which the answer is "yes". Often the same name of the problem is used for the language.

A DFA = \{ \langle M, x \rangle \}: \ M \text{ accepts } x, \ ie. \ x \in L(M) \ ? \ . \ Could \ write \ L(\text{A DFA} \text{ but that's quirky.} \\

Fact: The A DFA (language is decidable. To prove we can sketch a decider in pseudocode.

Decider: On input \( w = \langle M, x \rangle \),

1. Run the Turing Kit with \( M \) loaded and \( x \) on the tape.

2. It will halt since \( M \) is a DFA.

3. Hence the simulation will either accept \( x \), in which case your routine accepts \( \langle M, x \rangle \); else it will reject, so your routine holds and rejects.

EDFA: INST: Just a DFA \( M \) \ (no "x") 

Ques: Is \( L(M) = \emptyset \)?

Language = \{ \langle M \rangle : L(M) = \emptyset \}

NE \( \text{DFA} \) = \{ \langle M \rangle \mid \exists x \colon \text{DFA but not technically the complement}\}

As a language, this is \{ \langle M \rangle : L(M) \neq \emptyset \} \sim \text{not technically the complement, the strings that are not valid } \langle M \rangle \text{ belong to neither
Because the complement of a decidable language is decidable, and because in every language except C++ we can decide whether a program code will compile, a full template meta-programming tool decide E DFA it suffices to give a decider for N E DFA.

**FACT**: \( L(M) \neq \emptyset \iff \) there is a path in the graph of \( M \) from start to some accepting state.

**NB**: If \( F = \{ s \} \) and no other strings besides \( s \) get accepted, we still have \( L(M) = \{ s \} \neq \emptyset \), so \( L(M) \) is in the N E DFA language. The path in this case has 0 steps. The computation on \( s \) is just (S).

We can enumerate all states reachable from \( s \) by paths using BFS, which was already exemplified for the NFA to DFA construction. In this case, however, the \( n \) possible states of \( M \) that might be seen is known in advance. So BFS runs in \( O(n^2) \) time, no "explosion". Thus BFS gives an (efficient) decider for N E DFA, hence also E DFA.

How about \( \forall N \in \text{NFA} : L(N) \neq \emptyset \)?

The same FACT holds for NFA's \( N \), so we can use the same BFS proc. Thus ENFA is (efficiently) decidable too.

How about \( \forall M \in \text{DFA} : L(M) = \Sigma^* \)? We can reduce to the already solved E DFA problem.

1. Convert \( M \) to \( M' \) (efficient: \( F + F' = Q \setminus F \)) s.t. \( L(M') = \sim L(M) \).
2. Run our E DFA decider on \( (M') \), and accept iff it all works.

How about \( \forall N \in \text{NFA} : L(N) = \Sigma^* \)? Decidable but likely not efficiently. The problem is NP-hard (Ch.7)
\[ EQ_{DFA} = \{ \langle M_1, M_2 \rangle : M_1 \text{ and } M_2 \text{ are DFAs and } L(M_1) = L(M_2) \}. \]

Key Idea: \( L(M_1) = L(M_2) \iff L(M_1) \bigtriangleup L(M_2) = \emptyset. \)

**Decision:** 1. Build \( M_3 \) as at right.
2. Run the \( EQ_{DFA} \) decision on \( \langle M_3 \rangle \).
3. Accept \( \langle M_1, M_2 \rangle \) if \( step \ 2 \) accepts.

\( NEQ_{DFA} \) is (essentially) the complement of \( EQ_{DFA} \), hence decidable.

But \( EQ_{NFA} \) is reducible from \( ALL_{NFA} \), \( EQ_{NFA} = \{ \langle N_1, N_2 \rangle : N_1 \text{ and } N_2 \text{ are NFAs by building } N_1 = N, N_2 = \begin{array}{c}
\circ \\
\circ \\
\circ \\
\circ \\
\circ
\end{array} \text{ which is a NFA.} \text{ and } L(N_1) = L(N_2) \}, \)

Nevertheless, is decidable by converting \( N_1 \) and \( N_2 \) into equivalent DFAs \( M_1 \) and \( M_2 \), then using the \( EQ_{DFA} \) decision. Just not efficient in general.

**\( E_{CFG} \)**

**INST:** A context-free grammar \( G = (V, \Sigma, R, \text{S}) \)

**QUEST:** Is \( L(G) = \emptyset \)?

**Essential Complement** \( NE_{CFG} = \{ \langle G \rangle : G \text{ is a CFG and } L(G) \neq \emptyset \} \).

Is \( NE_{CFG} \) decidable? Is it efficiently decidable?

By a loop "like" \( \Theta \)?

**Similar Problem**

\( \varepsilon_{CFG} \)

**INST:** \( G \)

**QUEST:** Is \( \varepsilon \in L(G) \), i.e., does \( S \Rightarrow^* \varepsilon ? \)