Now we have algorithms
We want to check problems solvable or not by computers
Need a TM to decide it
i.e., accept/reject in a finite number of steps
We will show some examples
Acceptance Problems for DFA I

\[ A_{DFA} = \{ \langle B, w \rangle \mid B \text{ is a DFA that accepts } w \} \]

- \( \langle B, w \rangle \) is the input
  
  Note that a DFA can be represented as a string \((Q, \Sigma, \ldots)\)

- Is \( A_{DFA} \) decidable?

- Idea: input \( \langle B, w \rangle \)
  
  1. simulate \( B \) on \( w \)
  2. ends in an accept state \( \Rightarrow \) accept
     
     otherwise \( \Rightarrow \) reject
Proof of $A_{DFA}$ I

- Put
  
  $$B = \langle Q, \Sigma, \delta, q_0, F \rangle$$

  into a tape

- Check if $w \in \Sigma^*$ and $B$ a valid DFA

- Simulate $w$ according to $\delta$

- After processing the last element of $w$, check if in a final state
\[ A_{NFA} = \{ \langle B, w \rangle \mid B \text{ is an NFA that accepts } w \} \]

- We can convert \( B \) to a DFA and use the procedure for \( A_{DFA} \)
- It’s like to use the procedure for \( A_{DFA} \) as a subroutine
$A_{REX} = \{ \langle R, w \rangle \mid R : \text{regular expression generates } w \}$

- It’s similar
- We convert $R$ to a DFA first
- Recall that we had a procedure to convert $R$ to an NFA. Then we can convert the NFA to a DFA
- The key is that the conversion is a finite procedure
$E_{DFA} = \{ \langle A \rangle \mid A : DFA, L(A) = \emptyset \}$

- i.e. $A$ accepts nothing
- Idea:
  DFA accepts something
  $\iff$ reaching a final state from $q_0$ after several links
- procedure
  1. mark $q_0$
repeat until no new state marked
mark all

\[ a \rightarrow b, \]

where \( a \) has been marked

if no \( q \in F \) marked, accept. otherwise, reject

Example: a state diagram with 3 nodes and the following connections

\[ 1 \rightarrow 2, 3 \]
Marked states in running the procedure

1
12
12

- Each iteration: at least one new state marked
- At most $n$ iterations: $n \neq \# \text{ states}$
\( EQ_{DFA} \)

\[ EQ_{DFA} = \{ \langle A, B \rangle \mid A, B : DFAs, L(A) = L(B) \} \]

- \( EQ_{DFA} \) is decidable
- Idea for the proof:
  - Let a DFA \( C \) be the exclusive or of \( A \) and \( B \)
  - If \( L(A) = L(B) \)
  - then \( L(C) = \emptyset \)
Exclusive or of $A$ and $B$

(latex source from https://texample.net/tikz/examples/set-operations-illustrated-with-venn-diagrams/)
Formally

\[ L(C) = (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B)) \]

- \(B\) DFA \(\Rightarrow\) so is \(\overline{B}\)
- \(A, B\) DFA \(\Rightarrow\) so is \(A \cup B, A \cap B\)
- We then use \(E_{DFA}\) to check if \(L(C) = \emptyset\) or not