W7 1

Is the set
$A_{DFA} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
finite or infinite?

A Finite

B Infinite

W7 2

Suppose we divide up this problem
$A_{DFA} = \{\langle B, w\rangle \mid B \text{ is a DFA that accepts input string } w\}$
into
Part A: B is a DFA, and
Part B: Given that B is a DFA, does B accept string w
Which part looks harder?

- A A
- B B

W7 3

Suppose for Part A: B is a DFA , we were to make use of the definition of a DFA.

1. Q is a finite set called the **states**,
2. Σ is a finite set called the **alphabet**,
3. $\delta :\rightarrow Q \times \Sigma \rightarrow Q$ is the **transition function**,
4. $q_0 \in Q$ is the **start state**,
5. $F \subseteq Q$ is the **set of accept states**

A If the DFA were required to be expressed in these parts, a Java program, or a Turing machine, could be written to check candidate machine descriptions, and it could give the correct answer every time.

B With the qualification that the input has to be finite, A would be right.

C A DFA could be defined that does not fit the description above.

D It would not necessarily work for NFAs.

W7 4

Suppose for Part B: Given that B is a DFA, does B accept string w, we were to make use of the definition of a computation: machine M **accepts** w if a sequence of states $r_0, r_1, \ldots, r_n$ in Q exists with three conditions:

1. $r_0 = q_0$,
2. $\delta(r_i, w_{i+1}) = r_{i+1}, \text{ for } i = 0, \ldots, n-1$, and
3. $r_n \in F$

A Given a DFA description (for q_0, δ, F, etc.), and a finite input sequence r a Java program or Turing machine could be written, to check this and answer correctly in a finite time.

B The above description of computation is not complete.

C The above description cannot work because there are multiple possibilities for exiting state q_0, one for each symbol in the machine's alphabet.

D The above description cannot work because there are multiple possibilities for exiting state q_0, at least one for each symbol in the machine's alphabet.

W7 5

Why can we pose the finite automaton empty language problem as a graph problem?

A DFAs have a finite number of states, and can represent an infinite language.

B NFAs can always be converted into DFAs.

C Breadth first search is better than depth first search if the language can be infinite.

D DFA states correspond to vertices and the transition function gives a correspondence between the language and the edges.

E NFA states correspond to vertices and the transition function maps the Cartesian product of states and alphabet symbols to a single target state of the corresponding DFA.

W7 6

If a DFA graph is connected, is its language empty?

A Yes.

B No.

C It depends.

If a PDA graph is unconnected, is its language empty?

A Yes.

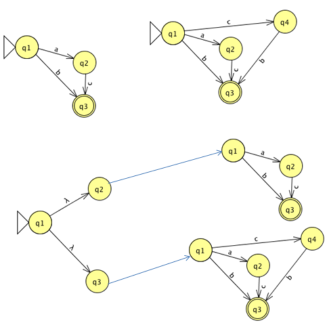
B No.

C It depends.

stop for a while

W7 8

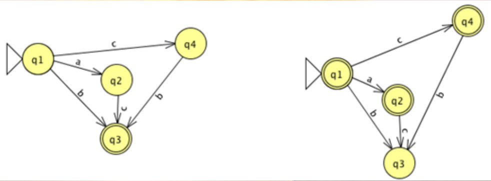
This construction proves that regular languages are closed under what operation?



- A Union
- B Intersection
- C Complementation
- D Concatenation
- E Star

W7 9

This construction proves that regular languages are closed under what operation?



A Union

B Intersection

C Complementation

D Concatenation

E Star

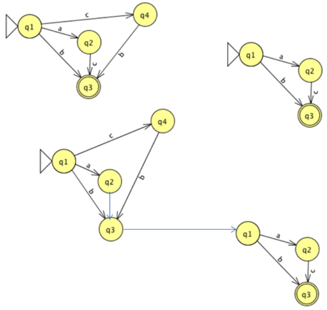
W7 10

This construction proves that regular languages are closed under what operation?
$\{A \cup B\}' = \{A' \cap B'\}$, where A and B are sets, and ' means complementation.

- A Union
- B Intersection
- C Complementation
- D Concatenation
- E Star

W7 11

This construction proves that regular languages are closed under what operation?



A Union

B Intersection

C Complementation

D Concatenation

E Star

Given the several constructions just seen, is the following a Turing machine?

$F =$ "On input $\langle A, B\rangle$, where A and B are DFAs:

1. Construct DFA $C = (L(A) \cap \overline{L(B)}) \cup (\overline{L(A)} \cap L(B))$
2. run TM T that checks whether a DFA has an empty language on $\langle C\rangle$
3. If T accepts, accept. If T rejects, reject."

A Yes

B No

stop for a while