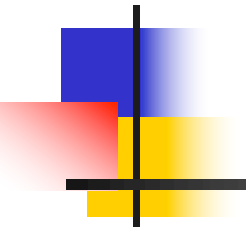


Course 9
Regular Languages Properties (cont'd)

The structure and the content of the lecture is based on http://www.eecs.wsu.edu/~ananth/CptS317/Lectures/index.htm

Excursion: Previous lecture

Minimization of a DFA by Table Filling Algorithm

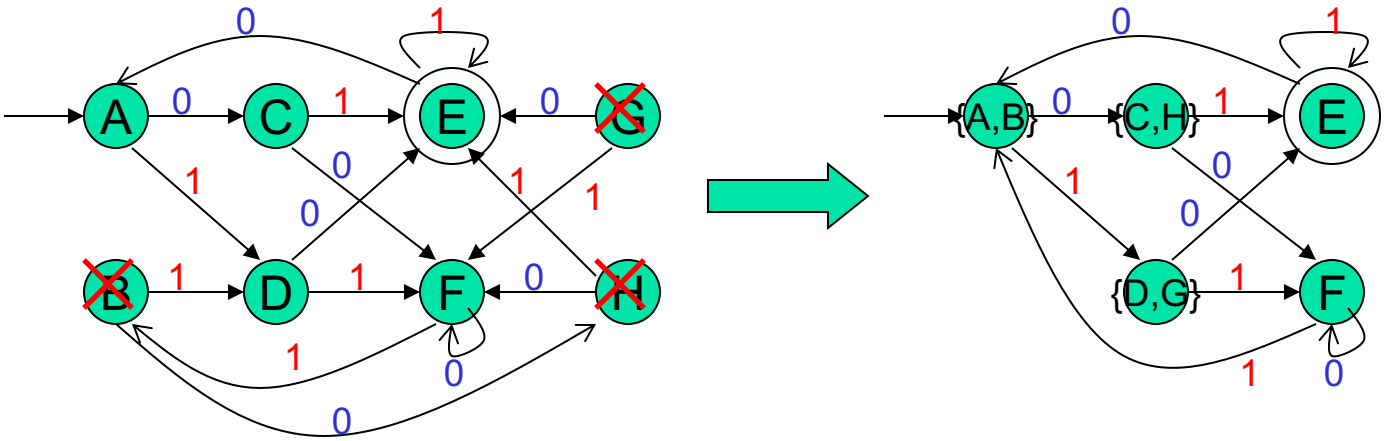
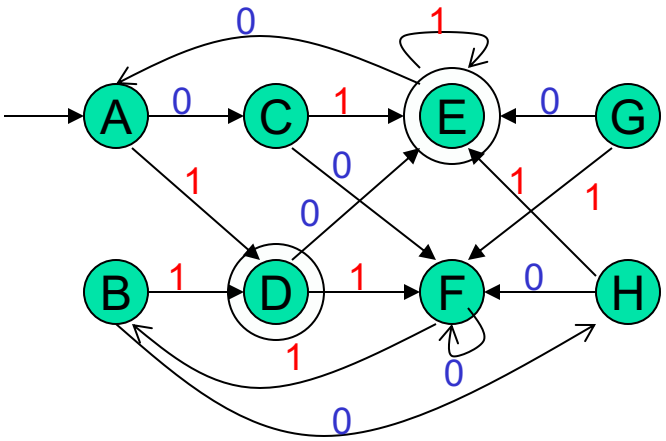
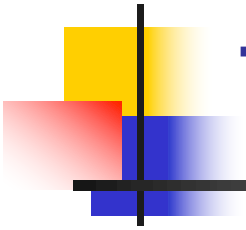


Table Filling Algorithm – special case



	A	B	C	D	E	F	G	H
A	=							
B		=						
C			=					
D				=				
E				?	=			
F						=		
G							=	
H								=

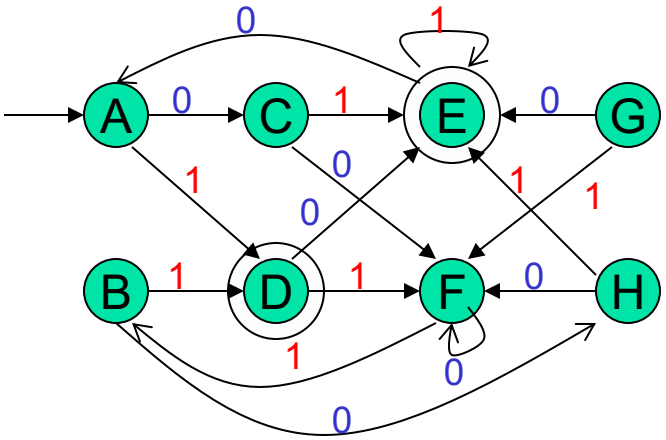
Q) What happens if the input DFA has more than one final state?
Can all final states initially be treated as equivalent to one another?



Topics

1) Minimization of DFAs by state equivalence method
2) Equivalence of DFAs
3) Decision properties of regular languages

DFA Minimization by state equivalence method



0-Equivalence: {A,B,C,F,G,H} {D,E}

1-Equivalence {A,B,C,H} {F} {G} {D} {E}

δ(A,0) = C
δ(B,0) = H
δ(A,1) = D
δ(B,1) = D
δ(C,0) = F
δ(C,1) = E
δ(G,0) = E

δ(F,0) = F
δ(F,1) = B
δ(G,1) = F

δ(H,0) = F = δ(C,0)
δ(H,1) = E = δ(C,1)
δ(D,0) = E
δ(E,0) = A

2-Equivalence

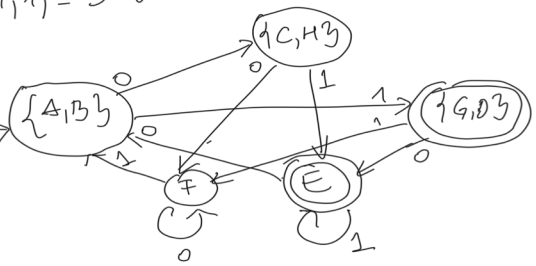
{A,B} {C,H} {F} {G,D} {E}

δ(A,0) = C
δ(H,0) = F
δ(C,0) = F
δ(F,0) = F
δ(G,0) = E

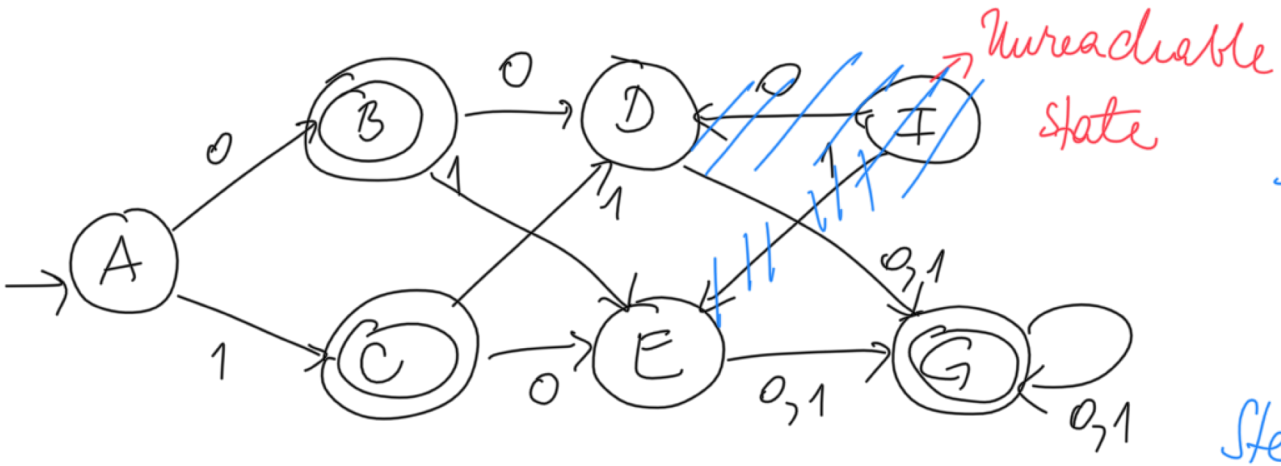
δ(H,1) = E
δ(C,1) = E
δ(F,1) = B

δ(D,1) = F
δ(G,1) = F

3-Equivalence
sets don't change!



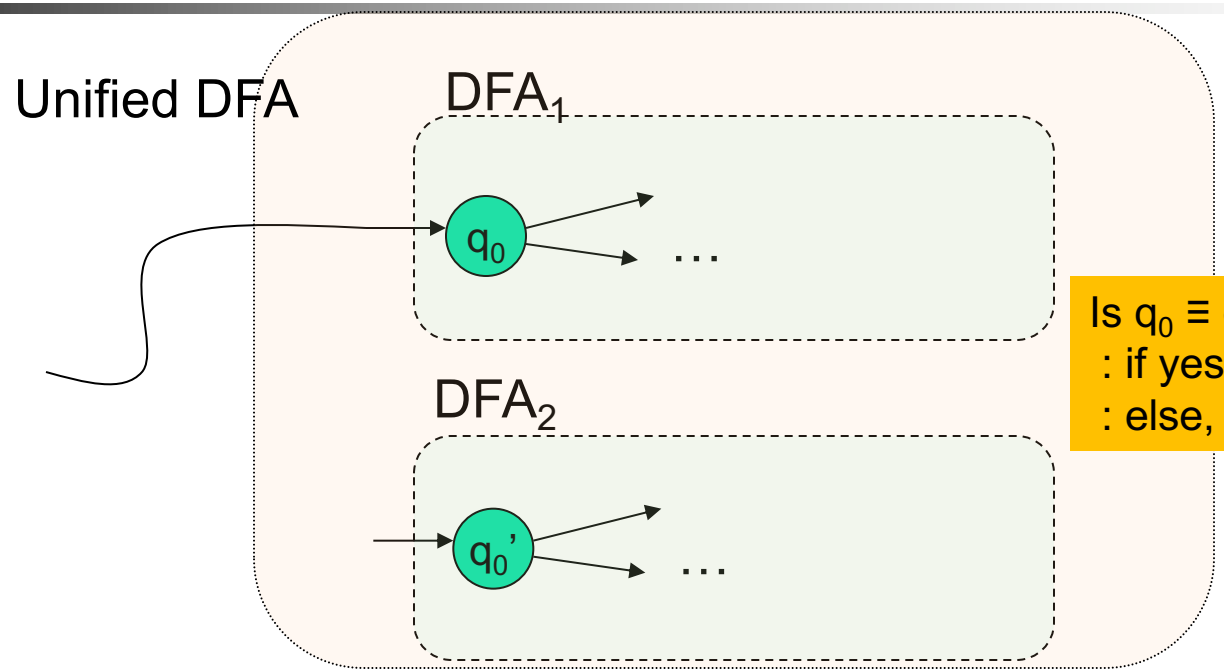
DFA Minimization with unreacheable states



Step 1. Eliminate the unreach. State

Step 2. Proceed with State equivalence or table filling methods.

Are Two DFAs Equivalent?



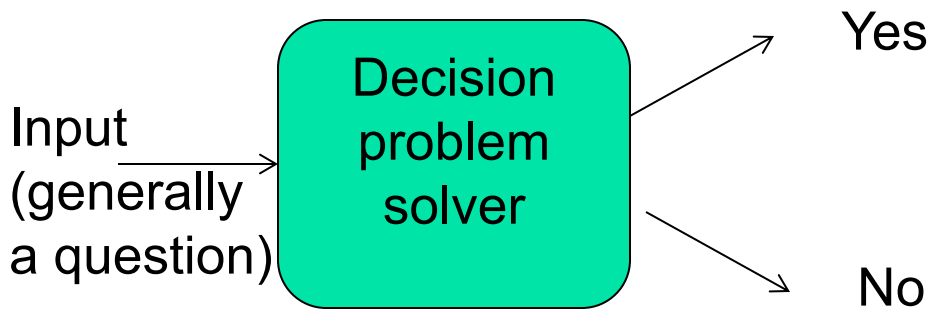
1. Make a new dummy DFA by just putting together both DFAs
2. Run table-filling/equivalence state algorithm on the unified DFA
3. *IF* the start states of both DFAs are found to be equivalent,

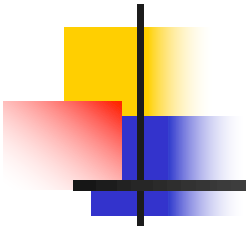
   THEN: $DFA_1 \equiv DFA_2$

   ELSE: different

Decision properties of regular languages

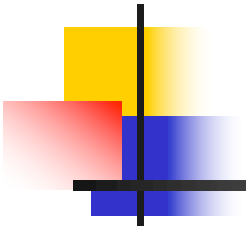
Any "decision problem" looks like this:





Membership question

- Decision Problem: Given L, is w in L?
- Possible answers: Yes or No
- Approach:
  1. Build a DFA for L
  2. Input w to the DFA
  3. If the DFA ends in an accepting state, then yes; otherwise no.



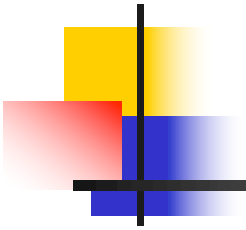
Emptiness test

- Decision Problem: Is L=Ø ?
- Approach:

  On a DFA for L:

  1. From the start state, run a *reachability* test, which returns:
     1. success: if there is at least one final state that is reachable from the start state
     2. failure: otherwise
  2. L=Ø if and only if the reachability test fails

How to implement the reachability test?



Finiteness

- Decision Problem: Is L finite or infinite?
- Approach:

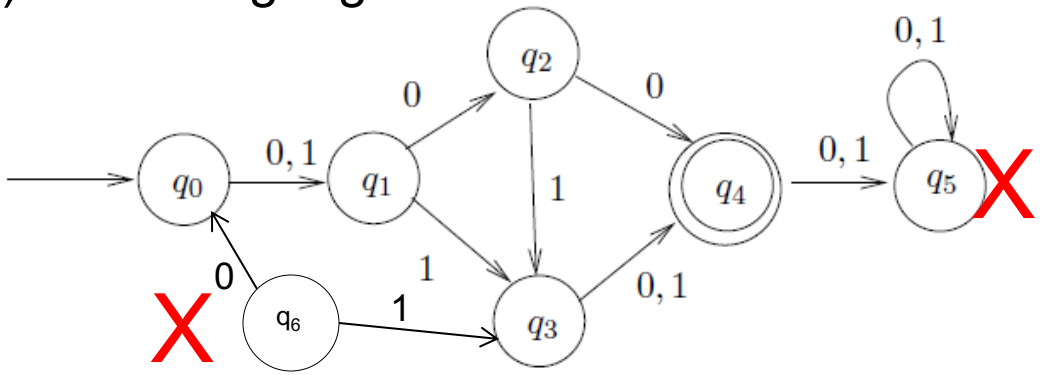
  On a DFA for L:

  1. Remove all states unreachable from the start state
  2. Remove all states that cannot lead to any accepting state.
  3. After removal, check for cycles in the resulting FA
  4. L is finite if there are no cycles; otherwise it is infinite
- Another approach
  - Build a regular expression and look for Kleene closure

How to implement steps 2 and 3?

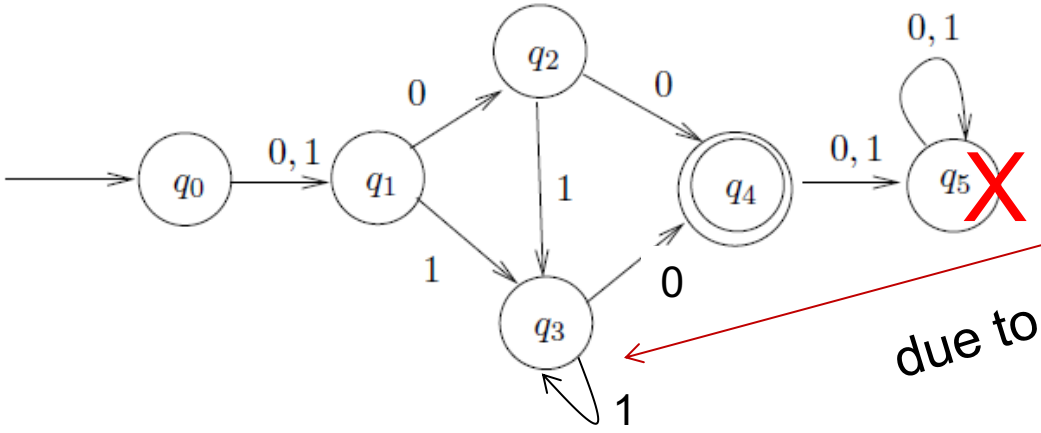
Finiteness test - examples

Ex 1) Is the language of this DFA finite or infinite?



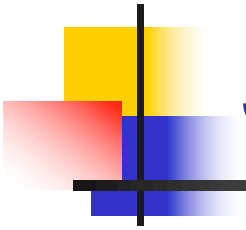
FINITE

Ex 2) Is the language of this DFA finite or infinite?



INFINITE

Equivalence & Minimization of DFAs



Summary

- Simplification of DFAs
  - How to remove unreachable states?
  - How to identify and collapse equivalent states?
  - How to minimize a DFA?
  - How to tell whether two DFAs are equivalent?