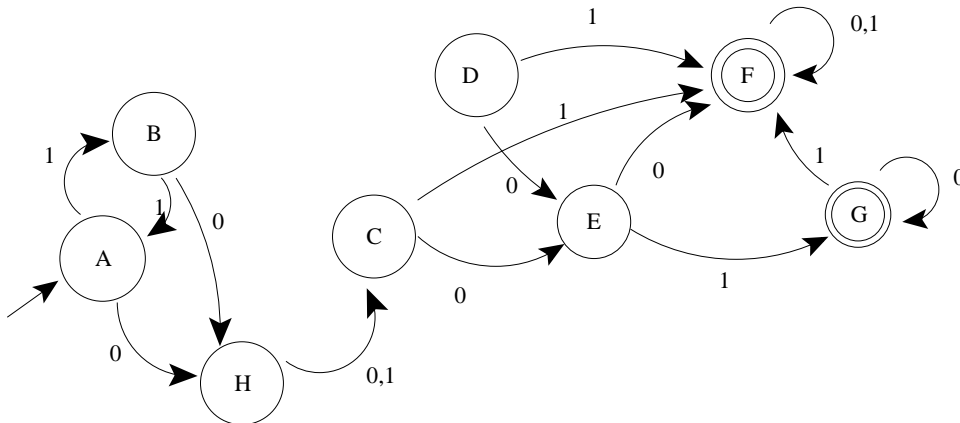


CS 273: Intro to Theory of Computation, Spring 2007

DFA minimization

This DFA is more complex than it needs to be:



The DFA minimization algorithm marks which states are distinct. States not marked as distinct can then be merged, to create a simpler DFA.

Suppose $M = (Q, \Sigma, \delta, q_0, F)$ is a DFA. We define distinctness inductively. Specifically, two states p and q in M are distinct if

- a) $p \in F$ and $q \notin F$, or vice versa, or
- b) for some $a \in \Sigma$, $\delta(p, a)$ and $\delta(q, a)$ are distinct

The algorithm for marking distinct states follows this inductive definition. Create a table DISTINCT with an entry for each pair of states. Table cells are initially blank.

(1) For every pair of states (p, q)

If p is final and q is not, or vice versa,

Set DISTINCT(p, q) to be ϵ

(2) Loop until there is no change in the table contents

For each pair of states (p, q) and each character a in the alphabet

if DISTINCT(p, q) is empty and DISTINCT($\delta(p, a), \delta(q, a)$) is not empty

Set DISTINCT(p, q) to be a .

(3) Two states p and q are distinct iff DISTINCT(p, q) is not empty.

After step (1):

b							
c							
d							
e							
f	ϵ	ϵ	ϵ	ϵ	ϵ		
g	ϵ	ϵ	ϵ	ϵ	ϵ		
h						ϵ	ϵ
	a	b	c	d	e	f	g

After one iteration of step (2):

b							
c	1	1					
d	1	1					
e	0	0	0	0			
f	ϵ	ϵ	ϵ	ϵ	ϵ		
g	ϵ	ϵ	ϵ	ϵ	ϵ		
h			1	1	0	ϵ	ϵ
	a	b	c	d	e	f	g

After the second iteration of step (2):

b							
c	1	1					
d	1	1					
e	0	0	0	0			
f	ϵ	ϵ	ϵ	ϵ	ϵ		
g	ϵ	ϵ	ϵ	ϵ	ϵ		
h	1	1	1	1	0	ϵ	ϵ
	a	b	c	d	e	f	g

Third iteration of step (2) makes no changes to the table, so we halt. The cells (a, b) , (c, d) and (f, g) are still empty, so these pairs of states are not distinct. Merging them produces the following simpler DFA recognizing the same language.

