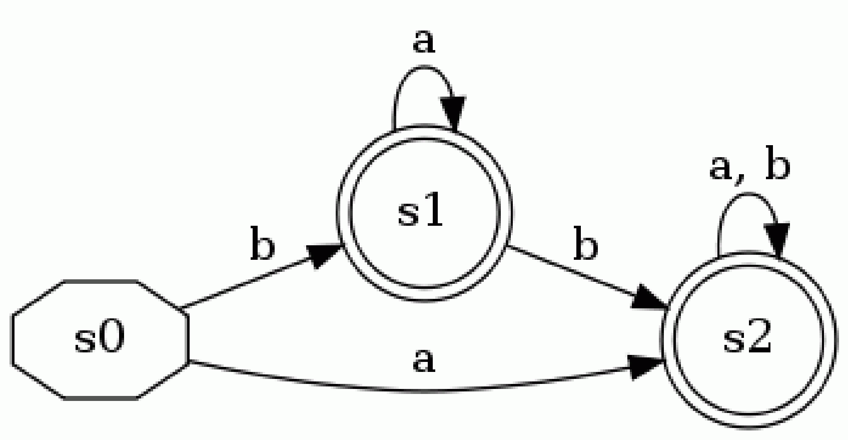
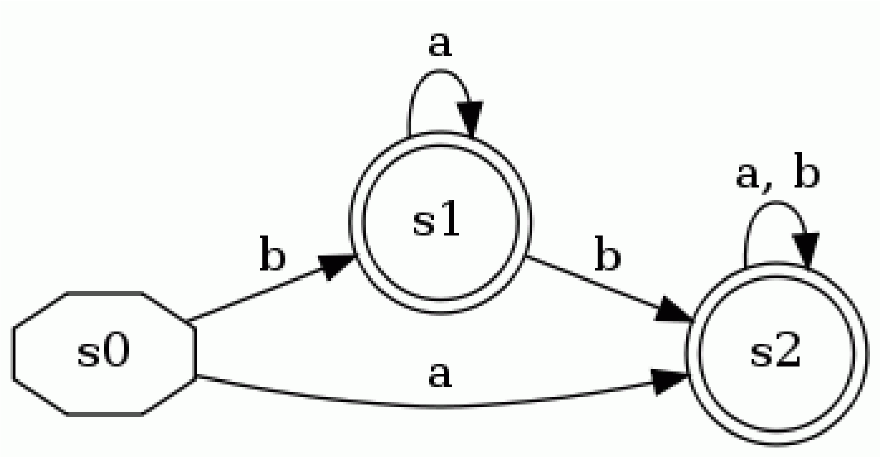
DFA Minimization

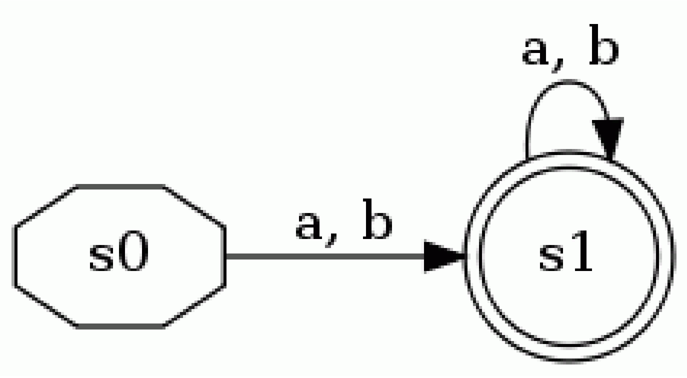
- Some states can be redundant:
  - The following DFA accepts (a|b)+
  - State s1 is not necessary



DFA Minimization

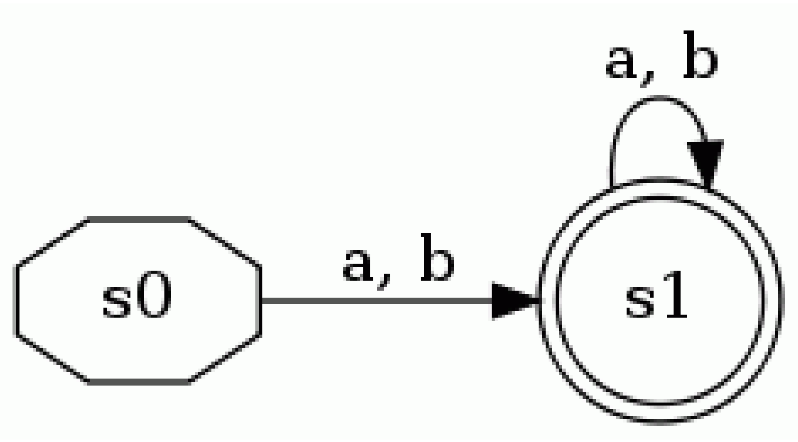
- So these two DFAs are *equivalent*:





DFA Minimization

- This is a *state-minimized* (or just *minimized*) DFA
  - Every remaining state is necessary



DFA Minimization

- The task of *DFA minimization*, then, is to automatically transform a given DFA into a state-minimized DFA
  - Several algorithms and variants are known
  - Note that this also in effect can minimize an NFA (since we know algorithm to convert NFA to DFA)

DFA Minimization Algorithm

- Recall that a DFA $M=(Q, \Sigma, \delta, q_0, F)$

- Two states *p* and *q* are distinct if
  - $p \in F$ and $q \notin F$ or vice versa, or
  - for some $\alpha \in \Sigma$, $\delta(p, \alpha)$ and $\delta(q, \alpha)$ are distinct

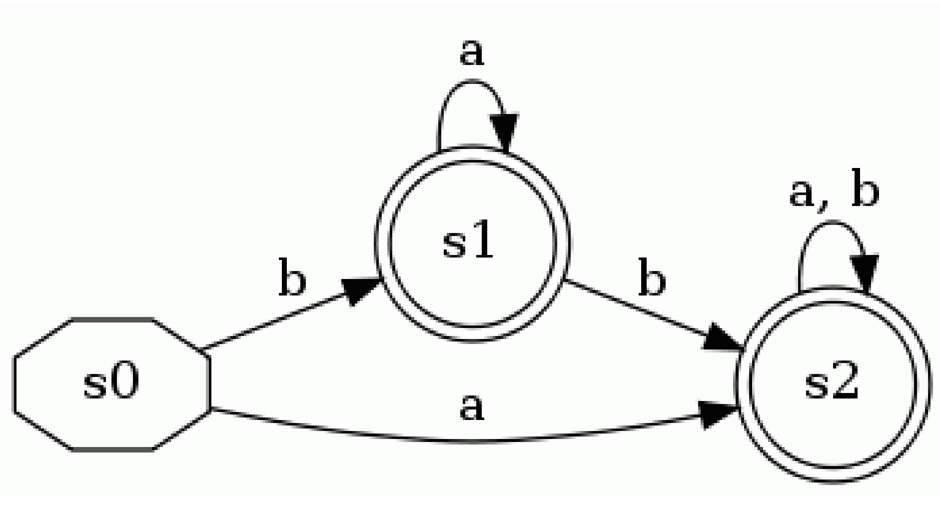
- Using this inductive definition, we can calculate which states are distinct

DFA Minimization Algorithm

- Create lower-triangular table DISTINCT, initially blank
- For every pair of states (p,q):
  - If p is final and q is not, or vice versa
    - DISTINCT(p,q) = ε
- Loop until no change for an iteration:
  - For every pair of states (p,q) and each symbol α
    - If DISTINCT(p,q) is blank and DISTINCT(δ(p,α), δ(q,α)) is not blank
      - DISTINCT(p,q) = α
- Combine all states that are not distinct

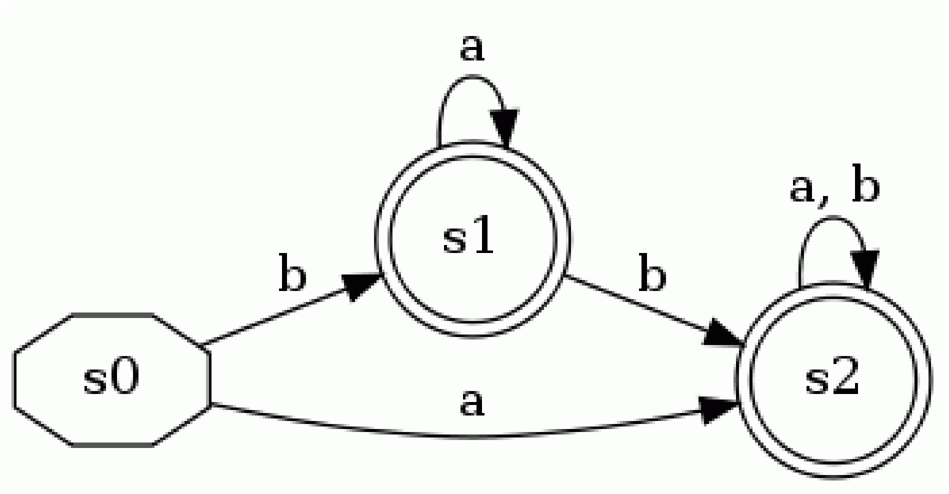
Very Simple Example

s0			
s1			
s2			
	s0	s1	s2



Very Simple Example

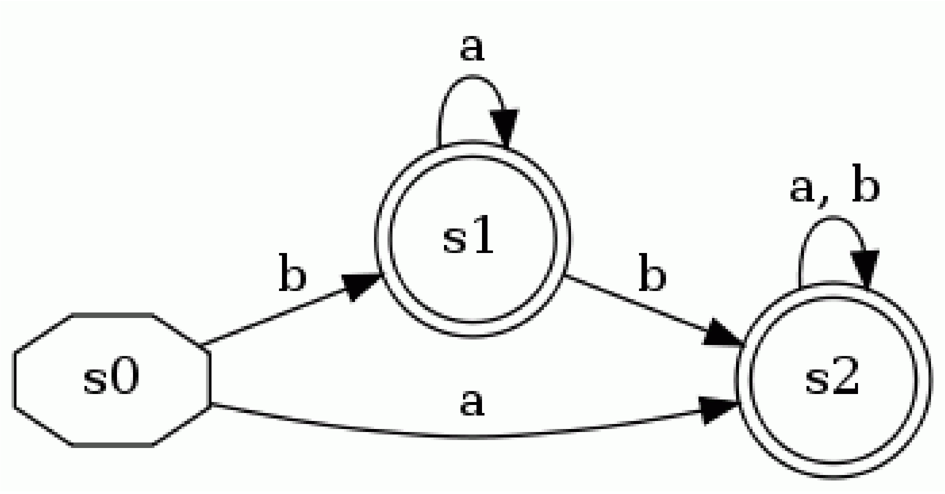
s0			
s1	ε		
s2	ε		
	s0	s1	s2



Label pairs with ε where one is a final state and the other is not

Very Simple Example

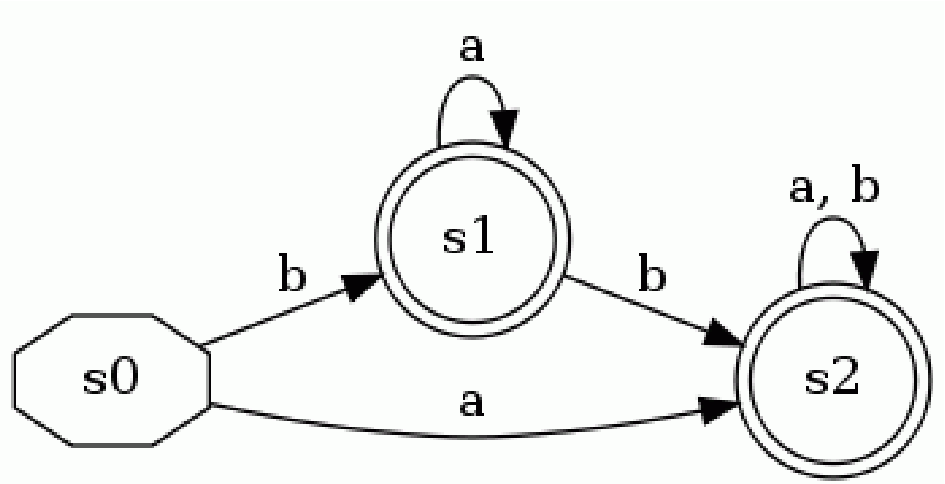
s0			
s1	ε		
s2	ε		
	s0	s1	s2



Main loop (no changes occur)

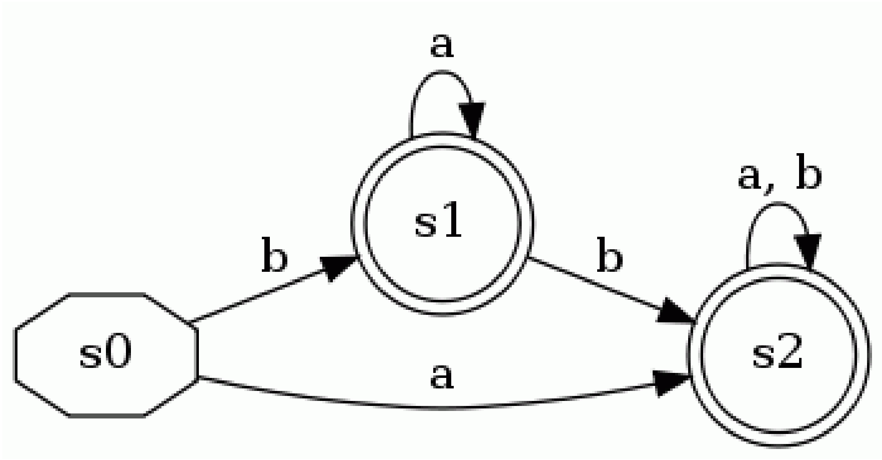
Very Simple Example

s0			
s1	ε		
s2	ε		
	s0	s1	s2

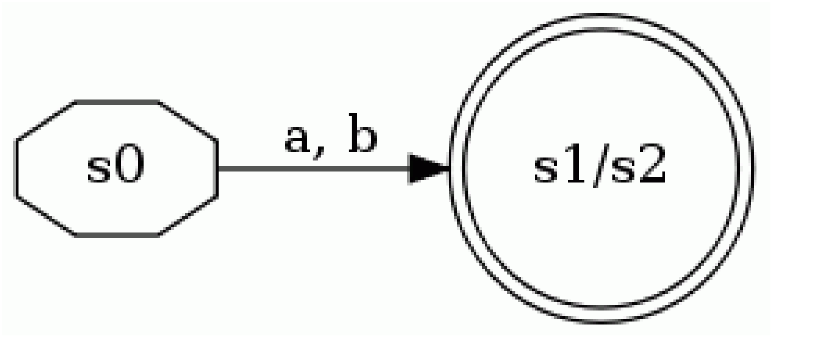


DISTINCT(s1, s2) is empty, so s1 and s2 are equivalent states

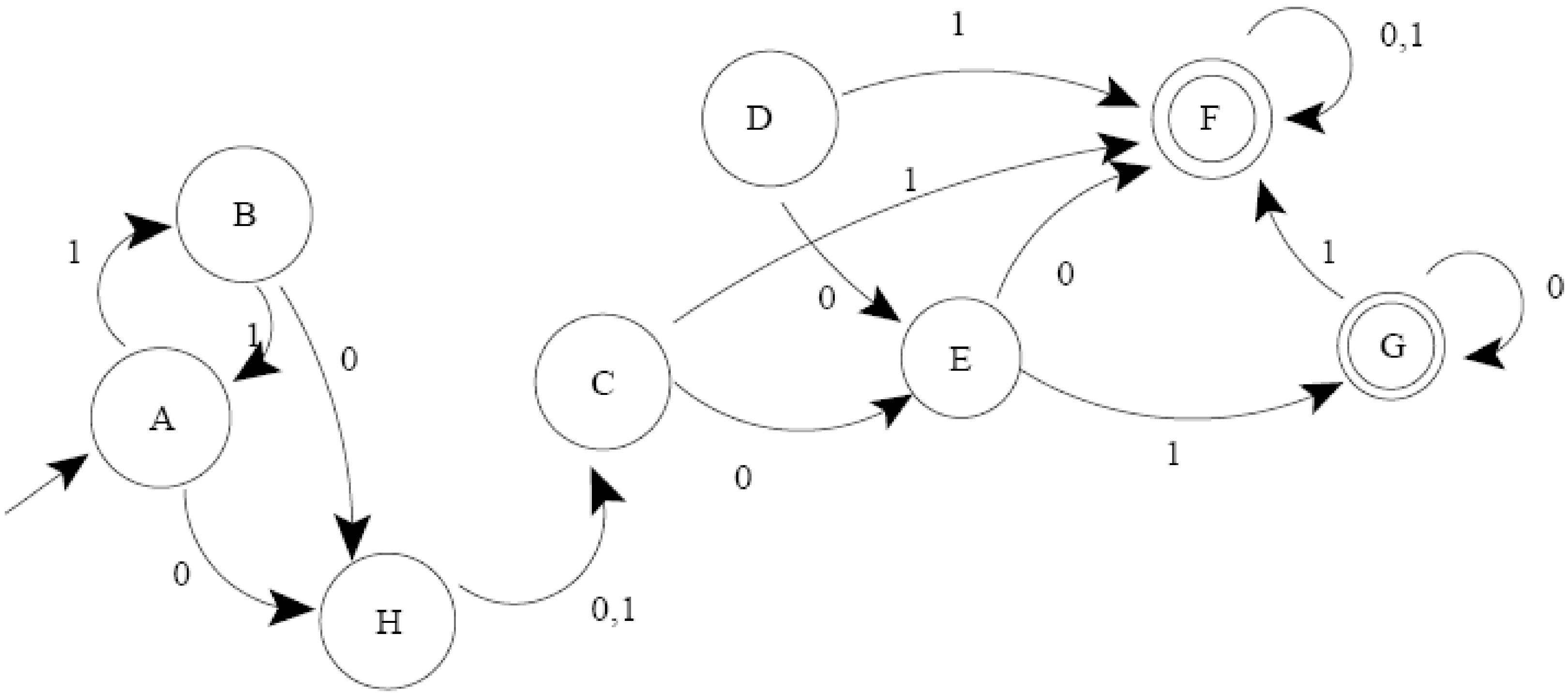
Very Simple Example



Merge s1 and s2



More Complex Example



More Complex Example

- Check for pairs with one state final and one not:

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

More Complex Example

- First iteration of main loop:

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

More Complex Example

- Second iteration of main loop:

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

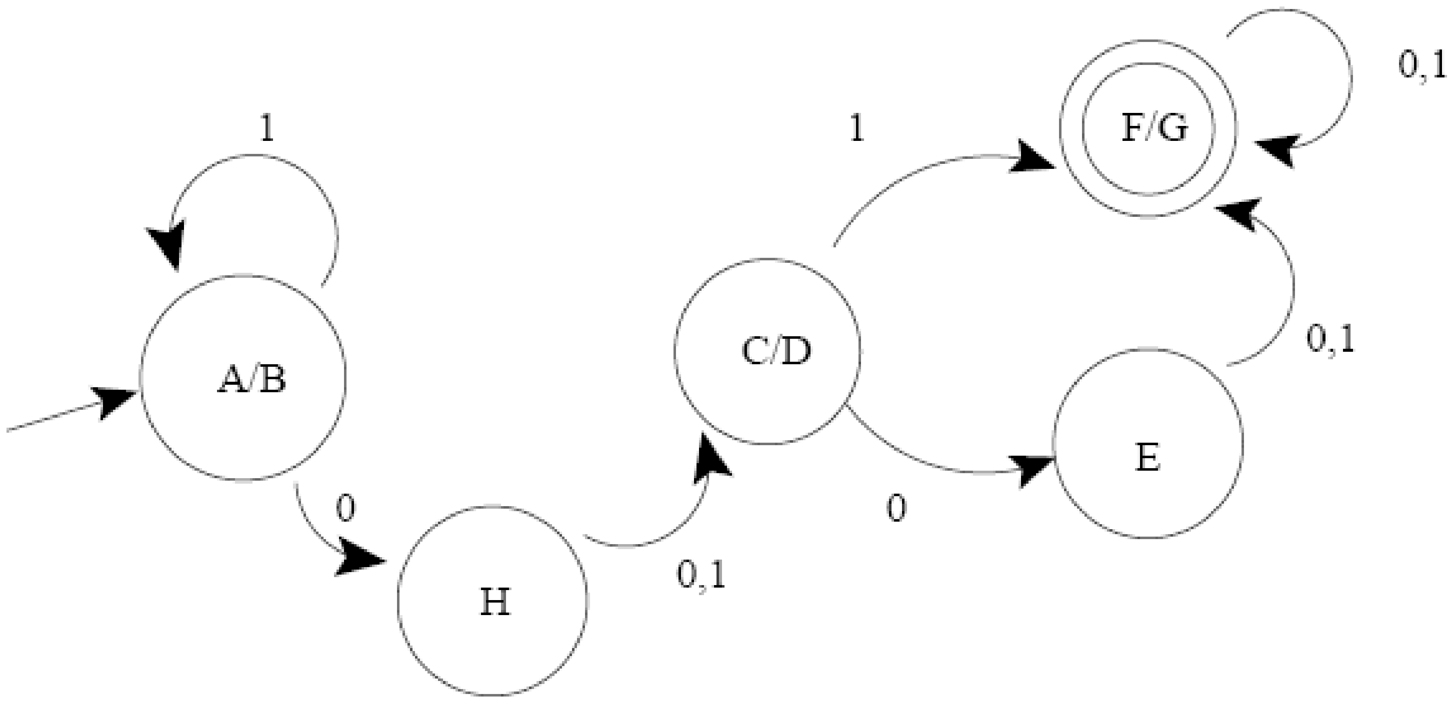
More Complex Example

- Third iteration makes no changes
  - Blank cells are equivalent pairs of states

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

More Complex Example

- Combine equivalent states for minimized DFA:



Conclusion

- DFA Minimization is a fairly understandable process, and is useful in several areas
  - Regular expression matching implementation
  - Very similar algorithm is used for compiler optimization to eliminate duplicate computations
- The algorithm described is $O(kn^2)$
  - John Hopcraft describes another more complex algorithm that is $O(k\ (n \log n)\ )$