DFA state minimization

(LECTURE 7)

Introduction

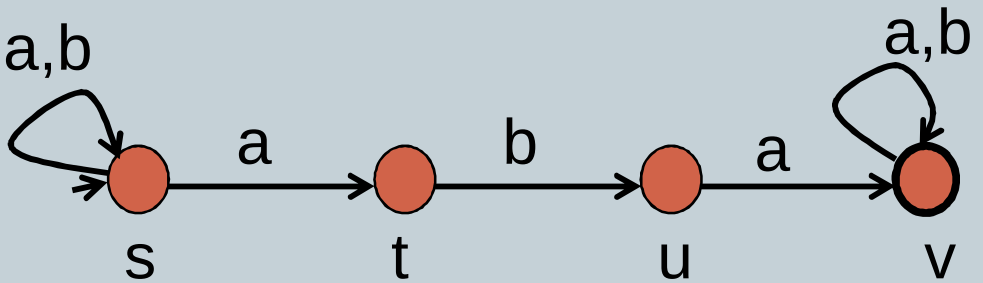
- Inaccessible states
- How to find all accessible states
- Minimization process

Motivations

Problems:

1. Given a DFA M with k states, is it possible to find an equivalent DFA M’ (I.e., L(M) = L(M’)) with state number fewer than k ?
2. Given a regular language A, how to find a machine with minimum number of states ?

Ex: A = L((a+b)*aba(a+b)*) can be accepted by the following NFA:



By applying the subset construction, we can construct a DFA M2 with 2^4=16 states, of which only 6 are accessible from the initial state {s}.

Inaccessible states

- A state p ∈ Q is said to be inaccessible (or unreachable) [from the initial state] if there exists no string x in S* s.t.
  D(s,x) = p (I.e., p ∉ {q | ∃x∈S*, D(s,x) = q }.)

Theorem: Removing inaccessible states from a machine M does not affect the language it accepts.

Pf: M = <Q,S,d, s,F> : a DFA; p : an inaccessible state

Let M’ =<Q \ {p}, S, d’, s, F\{p}> be the DFA M with p removed. Where d’:(Q\{p})xS → Q\{p} is defined by

d’(q,a) = r if d (q, a) =r and q, r∈Q \{p}.

For M and M’ it can be proved by induction on x that

for all x in S*, D (s,x) = D’ (s,x).

Hence for all x ∈ S*, x ∈ L(M) iff D(s,x) = q ∈ F

iff D’(s,x) = q ∈ F\{p} iff x ∈ L(M').

Inaccessible states are redundant

- M : any DFA with n inaccessible states $p_1, p_2, \ldots, p_n$.

Let $M_1, M_2, .., M_{n+1}$ are DFAs s.t. DFA M_{i+1} is constructed from M_i by removing p_i from M_i . I.e.,

M -rm(p_1)-> M_1 -rm(p_2)-> M_2 - … M_n -rm(p_n)-> M_n

By previous lemma: L(M) = L(M_1) = …=L(M_n) and

M_n has no inaccessible states.

- Conclusion: Removing all inaccessible sates simultaneously from a DFA will not affect the language it accepts.
- In fact the conclusion holds for all NFAs we well.

Pf: left as an exercise.

- Problem: Given a DFA (or NFA), how to find all inaccessible states ?

How to find all accessible states

- A state is said to be accessible if it is not inaccessible.

Note: the set of accessible states A(M) of a NFA M is

{q| $ x∈S*, q ∈ D(S,x) }

and hence can be defined by induction.

- Let A_k be the set of states accessible from initial states of M by at most k steps of transitions.

  I.e., A_k = {q|$ x∈S* with |x| ≤ k and q ∈ D (S,x) }
- What is the relationship b/t A(M) and A_ks ?
  - sol: A(M) = $U_{k\geq 0}$ A_k. Moreover $A_k \subseteq A_{k+1}$
- What is A_0 and the relationship b/t A_k and A_{k+1} ?

Formal definition: M=<Q,S,d, S,F> : any NFA.

- Basis: Every start state q ∈ S is accessible.($A_0 \subseteq$ A(M))
- Induction: If q is accessible and p in d (q,a) for some a ∈S, then p is accessible.

  ($A_{k+1}=A_k$ U {p | p∈ d(q,a) for some q∈A_k and a ∈ S.)

An algorithm to find all accessible states:

- REACH(M) {   // M = <Q,S,d, S ,F>

1. A = S;   // $A = A_0$
2. B = D (A) - A ;   // $B = A_1 – A_0$
3. For k = 0 to |Q| do { // $A = A_k$; $B = A_{K+1} - A_k$
4. A = A U B ;   // $A = A_{K+1}$

   B = D(B) - A;   // $B = D(B)–A=D(A_{K+1}–A_k)–A_{K+1}=A_{K+2}–A_{k+1}$;

   if B = {} then break };

5. Return(A) }

Function D(S) {   // = $U_{p\in S\ ,\ a\in S,}\ q\in d(p,a)$

1. D = {};
2. For each q in Q do

   for each a in S do

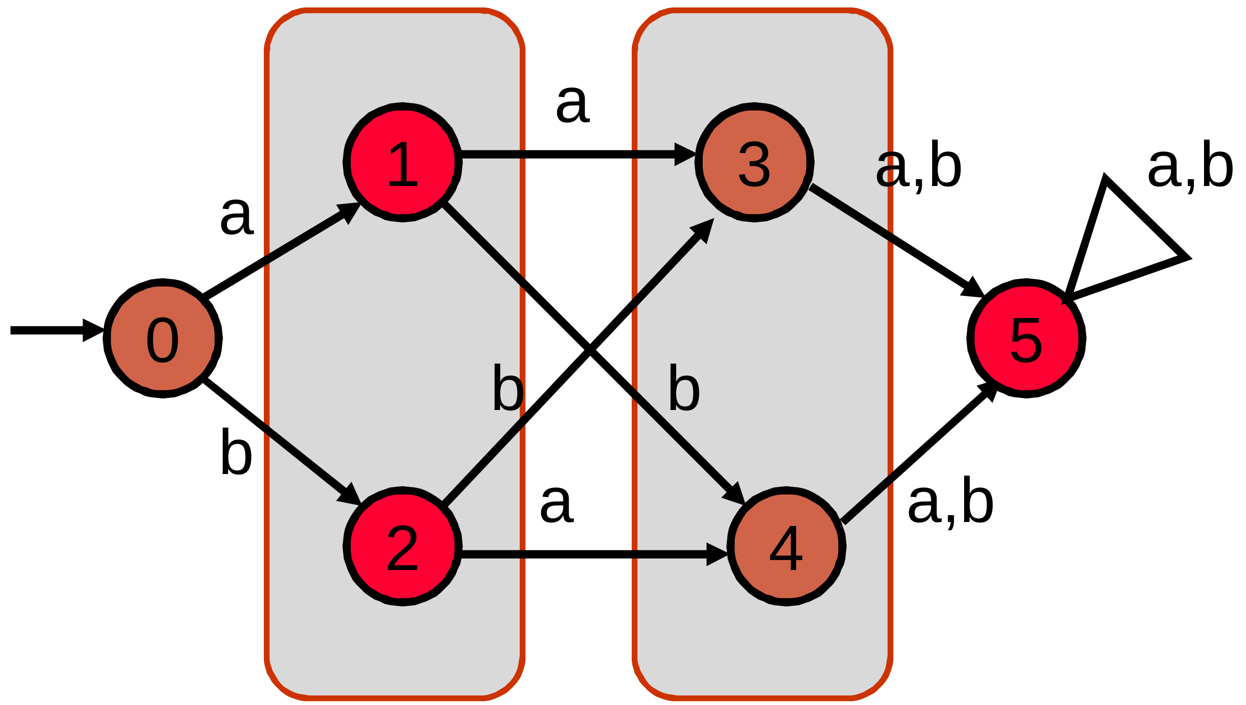
   D = D U d (q,a);

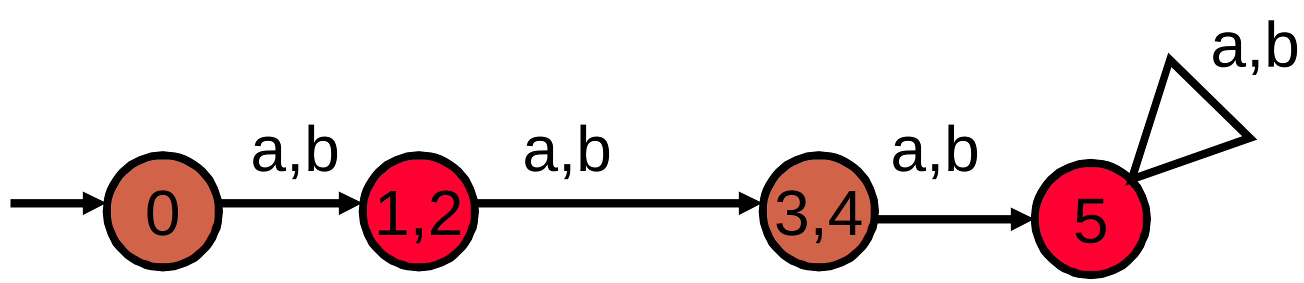
3. Return(D)}

Minimization process

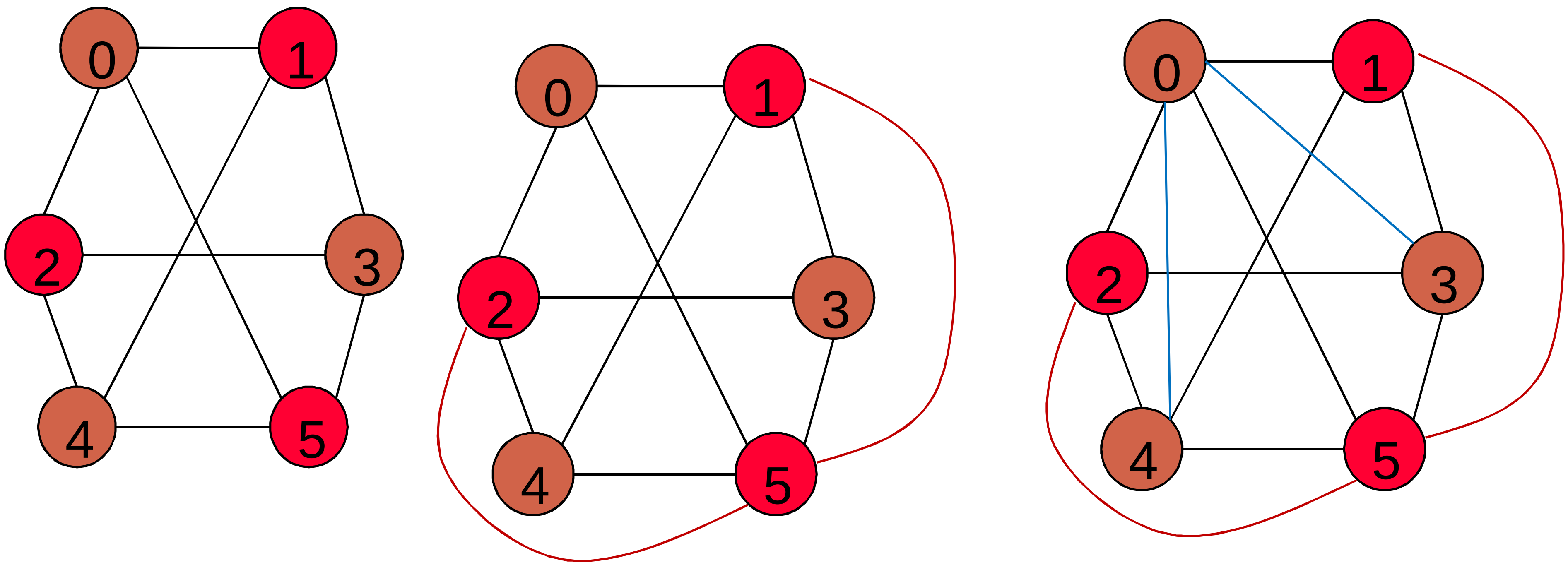
- Minimization process for a DFA:
  - 1. Remove all inaccessible states
  - 2. Merge all *equivalent* states
- What does it mean that two states are equivalent?
  - both *have the same observable behaviors* .i.e.,
  - there is no way to distinguish their difference.
- Definition: we say state p and q are *distinguishable* if there exists a string $x \in S^*$ s.t. $(D(p,x) \in F \Leftrightarrow D(q,x) \notin F)$.
  - If there is no such string, i.e. $\forall x \in S^*(D(p,x) \in F \Leftrightarrow D(q,x) \in F)$, we say p and q are equivalent (or indistinguishable).
- Example[13.2]: (next slide)
  - state 3 and 4 are equivalent.
  - States 1 and 2 are equivalent.
- Equivalents sates can be merged to form a simpler machine.

Example 13.2:





Example 13.2: Witness for states that are distinguishable



1. States b/t {0,3,4} and {1,2,5} can be distinguishsed by the empty string e.
2. States b/t {1,2} and {5} can be distinguished by a or b.
3. States b/t {0} and {3,4} can be distinguished by aa,ab, ba or bb.
4. There is no way to distinguish b/t 1 and 2, and b/t 3 and 4.

Quotient Construction

- M=(Q, S, ,ds, F): a DFA.
- ≈ : a relation on Q defined by:

  p ≈ q <=> ∀x∈S* D(p,x)∈F iff D (q,x) ∈ F
- Property: ≈ is an equivalence (i.e., reflexive, symmetric and transitive) relation.
- Hence it partitions Q into equivalence classes :
  - [p] $=_{def}$ {q ∈ Q | p ≈ q} for p ∈ Q.
  - Q/≈ $=_{def}$ {[p] | p ∈ Q} is the quotient set.
  - Every p ∈ Q belongs to exactly one class (which is [p])
  - p ≈ q iff [p]=[q] //why? since p ≈ q implies p≈r iff q≈r.
- Ex: From Ex 13.2, we have 0, 1 ≈ 2, 3 ≈ 4, 5.
  - => [0] = {0}, [1] = {1,2}, [2]={1,2}, [3]={3,4},[4]={3,4} and
  - [5] = {5}. As a result, [1] = [2] = {1,2}, [3]=[4]= {3,4} and
  - Q/≈ = { {0},{1,2},{3,4},{5}} = { [0],[1],[2],[3],[4],[5] } = {[0],[1],[3],[5] }.

the function d' is well-defined.

- Define a DFA called the quotient machine M/≈ = <Q',S, d',s',F'> where
  - Q'=Q/≈ ; s'=[s]; F'={[p] | p ∈ F}; and
  - d'([p],a)=[d (p,a)] for all p∈Q and a∈S. But well-defined?

Lem 13.5. if p ≈ q then d (p,a) ≈ d (q,a).

Hence [p]=[q] ⇒ p≈q ⇒ d(p,a) ≈ d(q,a) ⇒ [d (p,a)] = [d (q,a)]

Pf: By def. [d (p,a)] = [d(q,a)] iff d(p,a) ≈ d (q,a)

iff ∀y∈S* D(d (p,a),y) ∈ F ⇔ D(d (q,a),y) ∈ F

iff ∀ y ∈ S* D (p, ay) ∈ F ⇔ D (q,ay) ∈ F

if p ≈ q.

Lemma 13.6. p ∈ F iff [p] ∈ F'.

pf: => : trival.

<=: need to show that if q ≈ p and p ∈ F, then q ∈ F.

But this is trivial since p = D(p,e) ∈ F iff D (q, e) = q ∈ F

Properties of the quotient machine.

Lemma 13.7: ∀ x ∈ S*, D’([p],x) = [D(p,x)].

Pf: By induction on |x|.

Basis x = e: D’([p], e] = [p] = [D(p, e)].

Ind. step: Assume D’([p],x) = [D(p,x)] and let a ∈ S.

D’([p],xa) = d’(D’(p,x),a) = d’([D(p,x)],a) --- ind. hyp.

=[d(D (p,x),a)] -- def. of d’

= [D (p,xa)]. -- def. of D.

Theorem 13.8: L(M/≈) = L(M).

Pf: ∀ x ∈ S*,

x ∈ L(M/≈) iff D’(s’,x) ∈ F’

iff D’([s],x) ∈ F’ iff [D(s,x)] ∈ F’ --- lem 13.7

iff D (s,x) ∈ F --- lem 13.6

iff x ∈ L(M).

$M/\approx$ need not be merged further

- Theorem: $((M/\approx) / \approx) = M/\approx$

Pf: Denote the second $\approx$ by $\sim$. I.e.

$[p] \sim [q]$ iff $\forall x \in S^*$, $D'([p],x) \in F' \Leftrightarrow D'([q],x) \in F'$

Now

$[p] \sim [q]$

iff $\forall x \in S^*$, $D'([p],x) \in F' \Leftrightarrow D'([q],x) \in F'$ -- def.of~

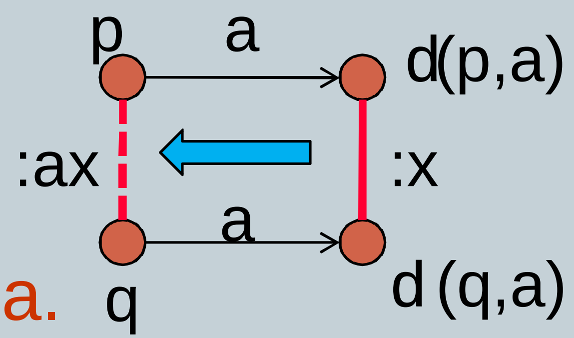
iff $\forall x \in S^*$, $[D(p,x)] \in F' \Leftrightarrow [D(q,x)] \in F'$ -- lem 13.7

iff $\forall x \in S^*$, $D(p,x) \in F \Leftrightarrow D(q,x) \in F$ -- lem 13.6

iff $p \approx q$ -- def of $\approx$

iff $[p] = [q]$ -- property of equivalence $\approx$

A minimization algorithm



1. Write down a table of all pairs {p,q}, initially unmarked.
2. mark {p,q} if p ∈ F and q ∉ F or vice versa.
3. Repeat until no additional pairs marked:
   3.1 if ∃ unmarked pair {p,q} s.t. {d(p,q), d(q,a) } is marked for some a ∈ S, then mark {p,q}.
4. When done, p ≈ q iff {p,q} is not marked.

Let M_k (k ≥ 0) be the set of pairs marked after the k-th iteration of step 3. [and M_0 is the set of pairs before step 3.]

Notes: (1) $M = U_{k \geq 0} M_k$ is the final set of pairs marked by the alg. (2) The algorithm must terminate since there are totally only C(n,2) pairs and each iteration of step 3 must mark at least one pair for it to not terminate..

An Example:

- The DFA: (Ex 13.2)

	a	b
>0	1	2
1F	3	4
2F	4	3
3	5	5
4	5	5
5F	5	5

Initial Table

1	-				
2	-	-			
3	-	-	-		
4	-	-	-	-	
5	-	-	-	-	-
	0	1	2	3	4

After step 2 (M_0)

1	M				
2	M	-			
3	-	M	M		
4	-	M	M	-	
5	M	-	-	M	M
	0	1	2	3	4

After first pass of step 3 (M_1)

1	**M**				
2	**M**	**-**			
3	**-**	**M**	**M**		
4	**-**	**M**	**M**	**-**	
5	**M**	**M**	**M**	**M**	**M**
	0	**1**	**2**	**3**	**4**

2nd pass of step 3. (M_2 & M_3)

- The result : 1 ≈ 2 and 3 ≈ 4.

1	**M**				
2	**M**	**-**			
3	**M2**	**M**	**M**		
4	**M2**	**M**	**M**	**-**	
5	**M**	**M1**	**M1**	**M**	**M**
	0	**1**	**2**	**3**	**4**

Correctness of the minimization algorithm

Let M_k ($k \geq 0$) be the set of pairs marked after the k-th itration of step 3. [and M_0 is the set of pairs befer step 3.]

Lemma: $\{p,q\} \in M_k$ iff $\exists x \in S^*$ of length $\leq k$ s.t. $D(p,x) \in F$ and $D(q,x) \notin F$ or vice versa,

Pf: By ind. on k. Basis $k = 0$. trivial.

Ind. step: $\exists\ x \in S^*$ of length $\leq k+1$ s.t. $D(p,x) \in F \Leftrightarrow D(q,x) \notin F$,

iff $\exists\ y \in S^*$ of length $\leq k$ s.t. $D(p,y) \in F \Leftrightarrow D(q,y) \notin F$, or

$\exists\ ay \in S^*$ of length $\leq k+1$ s.t. $D(d(p,a),y) \in F \Leftrightarrow D(d(q,a),y) \notin F$,

iff $\{p, q\} \in M_k$ or $\{d(p,a), d(q,a)\} \in M_k$ for some $a \in S$.

iff $\{p,q\} \in M_{k+1}$.

Theorem 14.3: The pair $\{p,q\}$ is marked by the algorithm iff not($p \approx q$) (i.e., $\exists\ x \in S^*$ s.t. $D(p,x) \in F \Leftrightarrow D(q,x) \notin F$)

Pf: not($p \approx q$) iff $\exists\ x \in S^*$ s.t. $D(p,x) \in F \Leftrightarrow D(q,x) \notin F$

iff $\{p,q\} \in M_k$ for some $k \geq 0$

iff $\{p,q\} \in M = U_{k \geq 0} M_k$.