What is an Optimal DFA?

If DFA D has n states and accepts the set of strings M from the alphabet Σ, **and** if there does not exist an equivilant DFA D' with $n' < n$ states, then D is optimal and unique.

EQUIVILANT DFA: D' accepts **only** M.

Why does “optimal” matter?

We are less concerned about number of **transitions** in a DFA than the the number of **states**. Why?

Why does “optimal” matter?

We are less concerned about number of **transitions** in a DFA than the the number of **states**. Why?

The canonical representation of a DFA in computer memory is the **transition table** $T[state][char]$ (you have been reading along, yes?).

You can’t make a state row smaller, it must be able to represent a transition for every character in Σ — A state with 1 transition consumes as much memory as a state with $|\Sigma|$ transitions.

The only way to decrease the memory footprint of $T[\cdot][\cdot]$ is to **decrease the number of states**. If the alphabet is large ($|\Sigma|$ large), this can be a substantial memory savings.

Merging DFA States — initialize

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Set $M = \emptyset$

Stack $L = < (\{0,1,\ldots,17\},\Sigma), (\{18,19,\ldots,23\},\Sigma) >$

```
procedure MergeStates( accepts DFA D defined by
    transition table T[·][·] )
returns a potentially new T[·][·]

T[row][·] uniquely identifies one state of D, and
each T[r][c] identifies the unique transition from
state r to state T[r][c] on input character c∈Σ.

let M be an empty set
let L be an empty stack
push ({accepting states of D},Σ) onto L
push ({non-accepting states of D},Σ) onto L
```

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — $S,C \leftarrow$ pop L

Set $M = \emptyset$

Stack $L = <(\{18,19,\ldots,23\},\Sigma)>$

$S = \{0,1,\ldots,17\} \quad C = \Sigma$

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

Merging DFA States — partition by a

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Set $M = \emptyset$

Stack $L = < (\{18, 19, \ldots, 23\}, \Sigma) >$

$S = \{0, 1, \ldots, 17\}$ $C = \{$b,c,q,r,s,t,u,v$\}$ $c =$ a

$X_1 = \{0\}$

$X_2 = \{1, \ldots, 17\}$

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — push $\{X_i, C\}$ back to L

Set $M = \emptyset$

Stack $L = < (\{1,\ldots,17\}, \{\text{b,c,q,r,s,t,u,v}\}), (\{18,19,\ldots,23\}, \Sigma) >$

Where did the $X_1 = \{0\}$ go?

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

Merging DFA States — partition by b

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Set $M = \emptyset$

Stack $L = < (\{18, 19, \ldots, 23\}, \Sigma) >$

$S = \{1, 2, \ldots, 17\}$ $C = \{$c,q,r,s,t,u,v$\}$ $c =$ b

$X_1 = \{1, \ldots, 16\}$

$X_2 = \{17\}$

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — push $\{X_i, C\}$ back to L

Set $M = \emptyset$

Stack $L = < (\{1,\ldots,16\}, \{\text{c,q,r,s,t,u,v}\}), (\{18,19,\ldots,23\}, \Sigma) >$

$X_2 = \{17\}$ discarded by $|X_i| > 1$ conditional.

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

Merging DFA States — after c, q partitioning

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Set $M = \emptyset$

Stack $L = < (\{1,2,\ldots,14\}, \{\text{r,s,t,u,v}\}), (\{18,19,\ldots,23\}, \Sigma) >$

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

Merging DFA States — partition by `r`

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Set $M = \emptyset$

Stack $L = < (\{18, 19, \ldots, 23\}, \Sigma) >$

$S = \{1, 2, \ldots, 14\}$ $C = \{\text{s,t,u,v}\}$ $c = \text{r}$

$X_1 = \{1, \ldots, 7, 9, \ldots, 12\}$

$X_2 = \{8, 14\}$ $X_3 = \{13\}$

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — push $\{X_i, C\}$ back to L

Set $M = \emptyset$

Stack $L = <(\{8,14\},\{\text{s,t,u,v}\}),(\{1,\ldots,7,9,\ldots,12\},\{\text{s,t,u,v}\}),$
$(\{18,19,\ldots,23\},\Sigma)>$

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — M grows!

Set $M = \{\{8,14\}\}$

Stack $L = < (\{1,\ldots,7,9,\ldots,12\},\{\text{s,t,u,v}\}),$
$(\{18,19,\ldots,23\},\Sigma) >$

State set $\{8,14\}$ was the top of the stack, with no other character transition differences. It will be added to M after character v partitioning ($C = \emptyset$)

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — partition by s

Set $M = \{\{8,14\}\}$

Stack $L = < (\{18,19,\ldots,23\},\Sigma) >$

$S = \{1,\ldots,7,9,\ldots,12\}$ $C = \{$t,u,v$\}$ $c =$ s

$X_1 = \{1,2,4,5,6,9,10\}$

$X_2 = \{3,7\}$ $X_3 = \{11,12\}$

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — push $\{X_i, C\}$ back to L

Set $M = \{\{8, 14\}\}$

Stack $L = <(\{11,12\}, \{t,u,v\}), (\{3,7\}, \{t,u,v\}),$
$(\{1,2,4,5,6,9,10\}, \{t,u,v\}), (\{18,19,\ldots,23\}, \Sigma)>$

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — M grows!

Set $M = \{\{8,14\},\{11,12\},\{3,7\}$

Stack $L = <(\{1,2,4,5,6,9,10\},\{\text{t,u,v}\}),(\{18,19,\ldots,23\},\Sigma)>$

State sets $\{11,12\}$ and $\{3,7\}$ were on the top of the stack and neither had any other character transition differences. They will be added to M after character v partitioning ($C = \emptyset$)

The state set $\{1,2,4,5,6,9,10\}$ has no other similarities over the characters $\{\text{t,u,v}\}$.
$T[6][\text{t}] = 5$ will discard state 6, $T[2][\text{u}] \neq T[5][\text{u}] \neq T[10][\text{u}]$ will discard states 2, 5 and 10, and
$T[1][\text{v}] \neq T[4][\text{v}] \neq T[9][\text{v}]$ will discard the remaining states.

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — M grows!

Set $M = \{\{8,14\},\{11,12\},\{3,7\}$

Stack $L = <(\{1,2,4,5,6,9,10\},\{\text{t,u,v}\}),(\{18,19,\ldots,23\},\Sigma)>$

State sets $\{11,12\}$ and $\{3,7\}$ were on the top of the stack and neither had any other character transition differences. They will be added to M after character v partitioning ($C = \emptyset$)

The state set $\{1,2,4,5,6,9,10\}$ has no other similarities over the characters $\{\text{t,u,v}\}$.
$T[6][\text{t}] = 5$ will discard state 6, $T[2][\text{u}] \neq T[5][\text{u}] \neq T[10][\text{u}]$ will discard states 2, 5 and 10, and
$T[1][\text{v}] \neq T[4][\text{v}] \neq T[9][\text{v}]$ will discard the remaining states.

Can we predict how the accepting state set in L will alter M?

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	14	•	•	•	•	8	7	•	6	•
	15	•	•	•	14	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Merging DFA States — $|L| = 0$

Set $M = \{\{8,14\},\{11,12\},\{3,7\},\{18,19\},\{20,23\},\{21,22\}\}$

Stack $L = <$ empty $>$

```
repeat (
  S,C ← pop L
  remove an element c from C
  Partition states s in S by T[s][c] into sets
    X1,X2,X3,...,Xk
  foreach ( Xi of X1,X2,X3,...,Xk with |Xi| > 1 ) do (
    if ( C = ∅ ) then (
      add Xi to M
    ) else (
      push (Xi,C) onto L
    )
  )
) while ( |L| > 0 )
```

Merging DFA States — Merge Rows 8 and 14

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	12	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	12	•	•	•
	15	•	•	•	8	13	12	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Set $M = \{\{8,14\},\{11,12\},\{3,7\},\{18,19\},\{20,23\},\{21,22\}\}$

Stack $L =<$ empty $>$

$S = \{8,14\}$

Remove row 14, change 14s to 8s in cells.

```
foreach ( S∈M ) do (
  merge rows of T[·][·] identified by states in S,
    fixing up transitions to these states as well!
  if ( starting state of D∈S ) then (
    mark the newly merged row as the
    starting state of D
  )
)
```

Merging DFA States — Merge Rows 11 and 12

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	7	•	•	•	•	•	3	2	•	•
	8	•	•	•	•	8	7	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	11	•	•	•
	15	•	•	•	8	13	11	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Set $M = \{\{8,14\},\{11,12\},\{3,7\},\{18,19\},\{20,23\},\{21,22\}\}$

Stack $L = <$ empty $>$

$S = \{11,12\}$

Remove row 12, change 12s to 11s in cells.

```
foreach ( S∈M ) do (
  merge rows of T[·][·] identified by states in S,
    fixing up transitions to these states as well!
  if ( starting state of D∈S ) then (
    mark the newly merged row as the
    starting state of D
  )
)
```

Merging DFA States — Merge Rows 3 and 7

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	19
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	8	•	•	•	•	8	3	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	11	•	•	•
	15	•	•	•	8	13	11	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	19	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Set $M = \{\{8,14\},\{11,12\},\{3,7\},\{18,19\},\{20,23\},\{21,22\}\}$

Stack $L = <$ empty $>$

$S = \{3,7\}$

Remove row 7, change 7s to 3s in cells.

```
foreach ( S∈M ) do (
  merge rows of T[·][·] identified by states in S,
    fixing up transitions to these states as well!
  if ( starting state of D∈S ) then (
    mark the newly merged row as the
    starting state of D
  )
)
```

Merging DFA States — Merge Rows 18 and 19

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	18
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	22
	8	•	•	•	•	8	3	•	6	•
	9	•	•	•	•	•	•	•	•	23
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	11	•	•	•
	15	•	•	•	8	13	11	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21
◎	22	•	•	•	•	•	•	•	•	21
◎	23	•	•	•	•	•	•	•	•	•

Set $M = \{\{8,14\},\{11,12\},\{3,7\},\{18,19\},\{20,23\},\{21,22\}\}$

Stack $L = <$ empty $>$

$S = \{18,19\}$

Remove row 19, change 19s to 18s in cells.

```
foreach ( S∈M ) do (
  merge rows of T[·][·] identified by states in S,
    fixing up transitions to these states as well!
  if ( starting state of D∈S ) then (
    mark the newly merged row as the
    starting state of D
  )
)
```

Merging DFA States — . . . so on and so forth.

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	18
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	21
	8	•	•	•	•	8	3	•	6	•
	9	•	•	•	•	•	•	•	•	20
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	11	•	•	•
	15	•	•	•	8	13	11	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21

Set $M = \{\{8,14\},\{11,12\},\{3,7\},\{18,19\},\{20,23\},\{21,22\}\}$

Stack $L = <$ empty $>$

```
foreach ( S∈M ) do (
  merge rows of T[·][·] identified by states in S,
    fixing up transitions to these states as well!
  if ( starting state of D∈S ) then (
    mark the newly merged row as the
    starting state of D
  )
)
return T[·][·]
```

	State	a	b	c	q	r	s	t	u	v
→	0	17	•	•	•	•	•	•	•	•
	1	•	•	•	•	•	•	•	•	18
	2	•	•	•	•	•	•	•	1	•
	3	•	•	•	•	•	3	2	•	•
	4	•	•	•	•	•	•	•	•	20
	5	•	•	•	•	•	•	•	4	•
	6	•	•	•	•	•	•	5	•	21
	8	•	•	•	•	8	3	•	6	•
	9	•	•	•	•	•	•	•	•	20
	10	•	•	•	•	•	•	•	9	•
	11	•	•	•	•	•	11	10	•	•
	13	•	•	•	•	13	11	•	•	•
	15	•	•	•	8	13	11	•	•	•
	16	•	•	15	•	•	•	•	•	•
	17	•	16	•	•	•	•	•	•	•
◎	18	•	•	•	•	•	•	•	•	18
◎	20	•	•	•	•	•	•	•	•	•
◎	21	•	•	•	•	•	•	•	•	21

Merging DFA States — . . . not done yet

Set $M = \{\{8,14\},\{11,12\},\{3,7\},\{18,19\},\{20,23\},\{21,22\}\}$
Stack $L =<$ empty $>$

Consider states 4 and 9, these can now be merged!

Also notice that states 18 and 21 could actually be considered identical, because they both transition back onto themselves.

We correct the first issue by running the $MergeStates$ algorithm on a DFA until the number of states does not change.

The second issue could be addressed by more complicated algorithms, but we won't pursue these in this course.

MergeStates until $|D|$ does not change

Let DFA D be defined by transition table $T[\cdot][\cdot]$.
$T[row][\cdot]$ uniquely identifies one state of D, and
each $T[r][c]$ identifies the unique transition from
state r to state $T[r][c]$ on input character $c \in \Sigma$.

```
repeat (
  T' ← MergeStates( T )
  if ( |T| = |T'| ) then (
    break loop
  ) else (
    T ← T'
  )
)
```

$T'[\cdot][\cdot]$ is now a well (near?) optimized *DFA* equivalent
to D with a reasonable number of effective states.

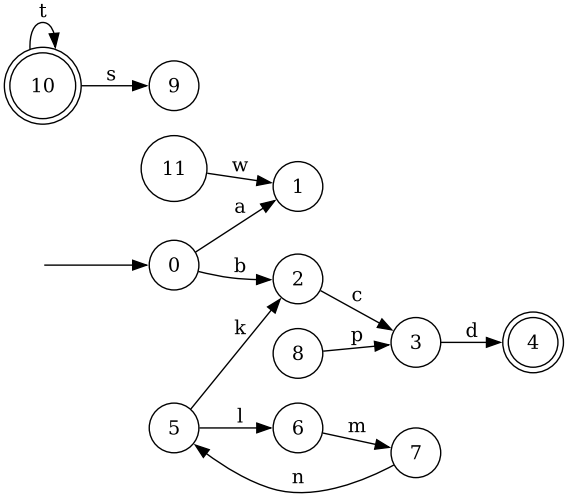
(Dead or unreachable states may still exist.)

Dead and Unreachable Stages

Unreachable States do not have a transition path **from the starting state**

Dead States do not have a transition path **to an accepting state**.

Which states in this DFA are of what type?



Dead and Unreachable Stages

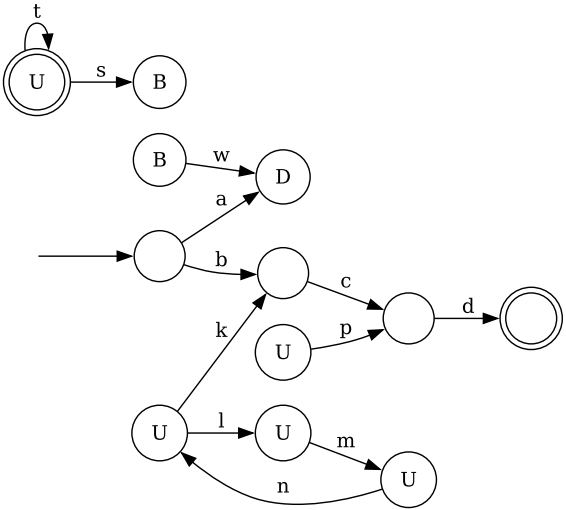
Unreachable States do not have a transition path **from the starting state**

Dead States do not have a transition path **to an accepting state**.

Which states in this DFA are of what type?

B means both dead and unreachable!

Algorithms for pruning out dead and unreachable states are left as an exercise (NFAMATCH).



What are equivilant DFA states?

Define in your own (group) terms. . .

What are equivilant DFA states?

Define in your own (group) terms. . .

Two identical rows of the DFA $T[state][char]$ table with the same **accepting attribute value**.