DFA State minimization

- How to construct a DFA for a given regular language with as few states as possible?
- Given two minimal DFAs for the same language, are they same?

L_k = the set of all binary strings with the kth last symbol 1.

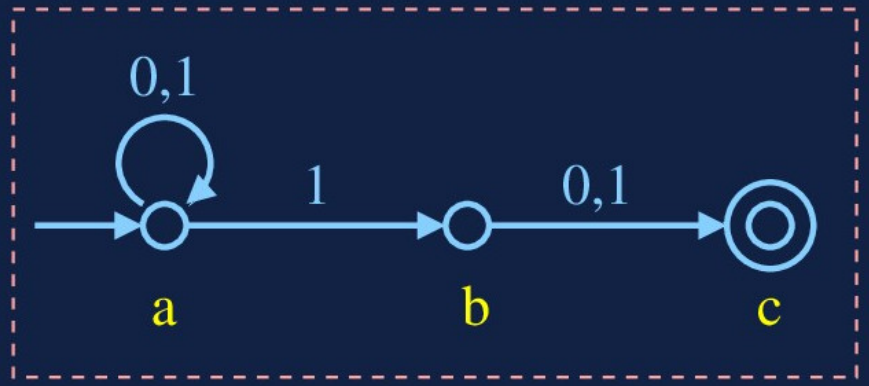
No DFA with less than 2^k states can accept L_k.

We know a design of a DFA for L_k with exactly 2^k states.

How to minimize the state count of a DFA?

- Throw away all the unreachable states
- Collapse equivalent states
  - What is equivalence
  - How to collapse
    - Is the procedure safe?
    - The language must remain the same.
    - How long shall we keep on collapsing?

Subset construction ⟶ DFA



NFA

$L_2 = \{ w \in \{0,1\}^* \mid$ the second last symbol is $1 \}$

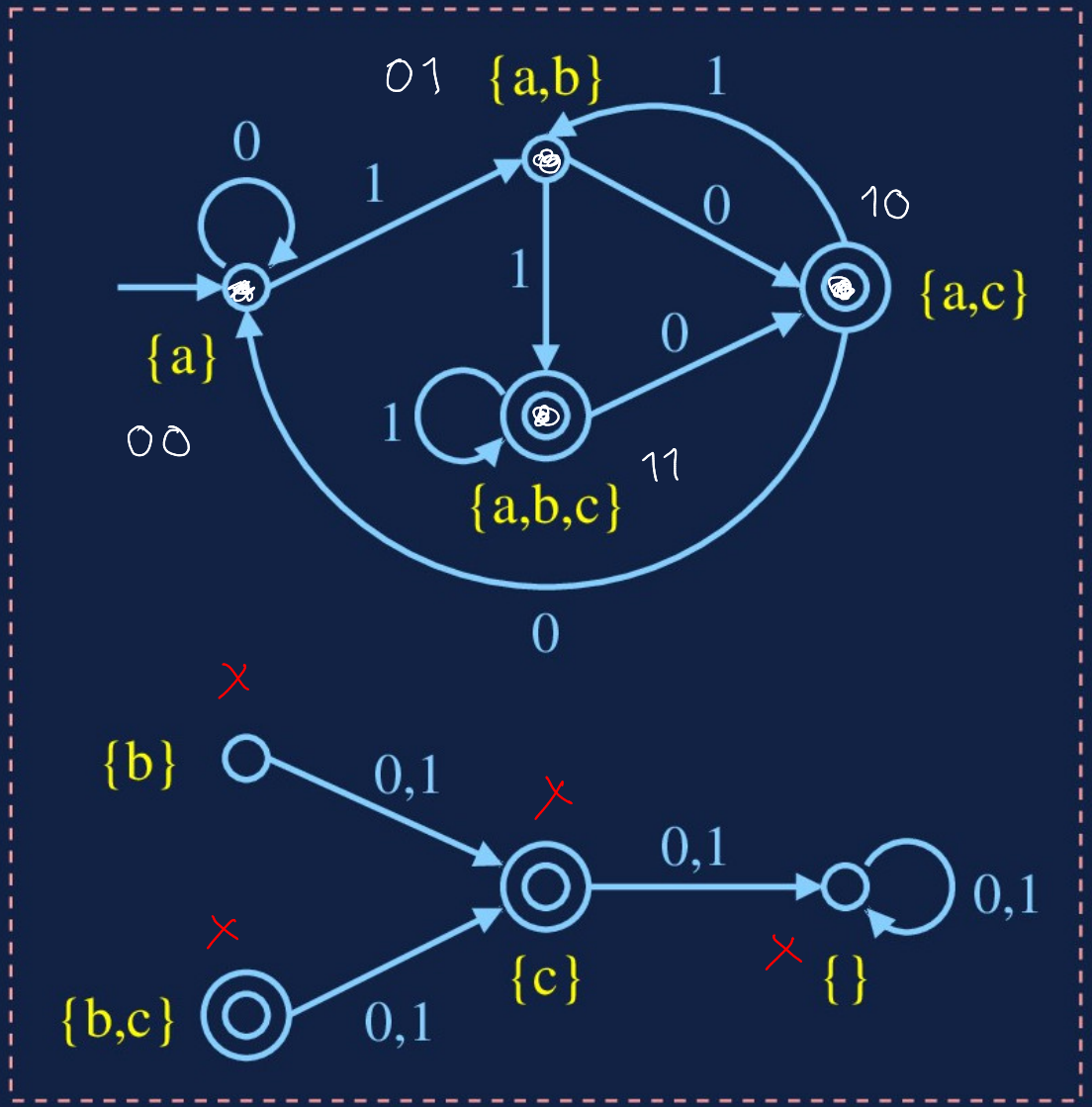
Subset construction ⟶ $2^3 = 8$ states

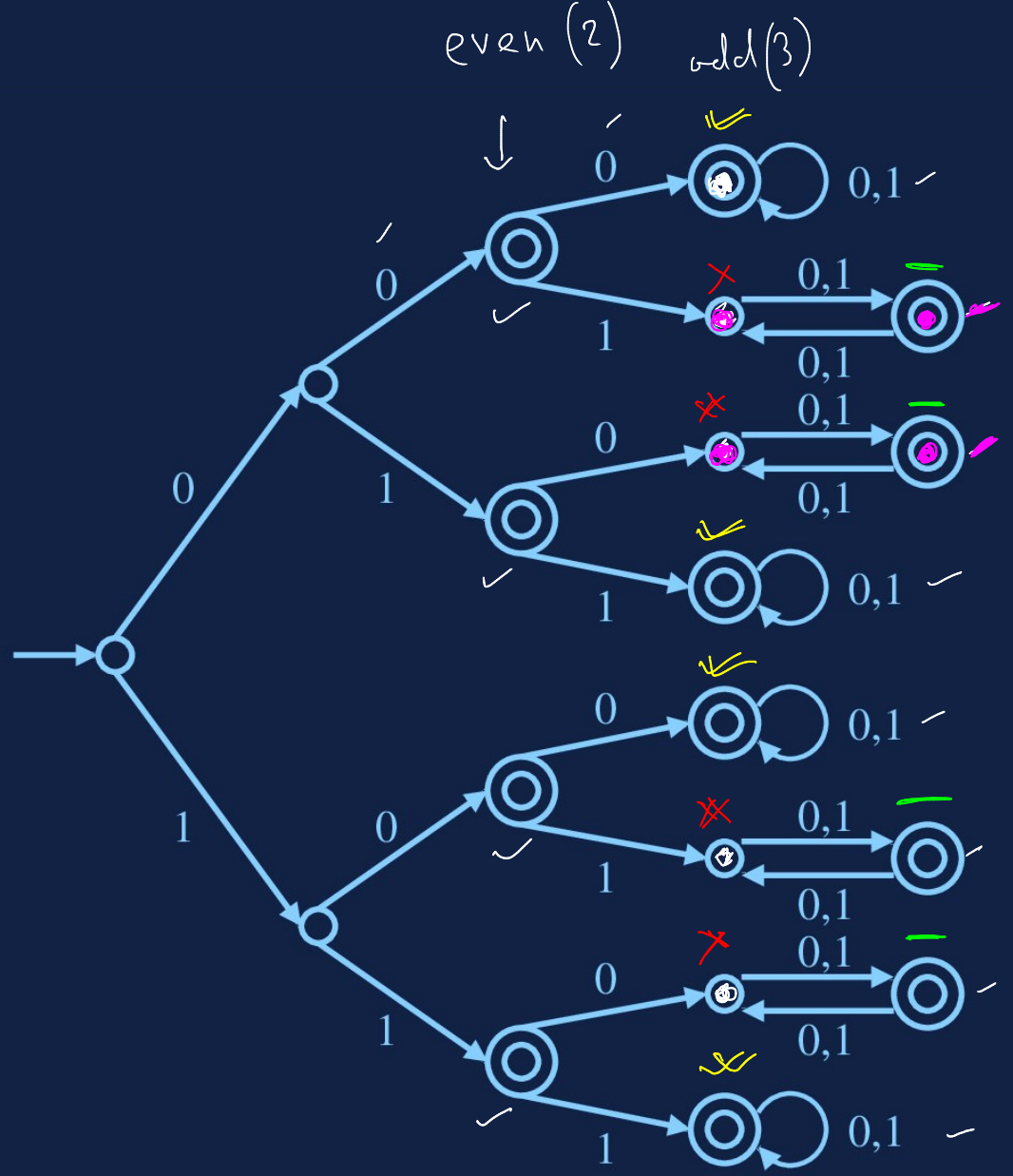
unreachable states

Graph traversal DFS / BFS starting from the start state

Throw away all unvisited states

— Solved





accepts all non-empty binary strings x such that

either $|x|$ is even

or the second and the third symbols of x are the same.

Final → whatever the remaining input is, the m/c stays in that state.

↳ can be collapsed

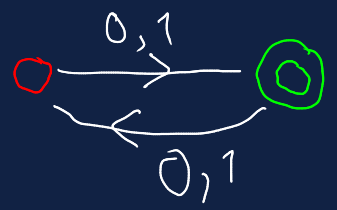
Not final — even — reject
odd — accept

↳ can also be collapsed

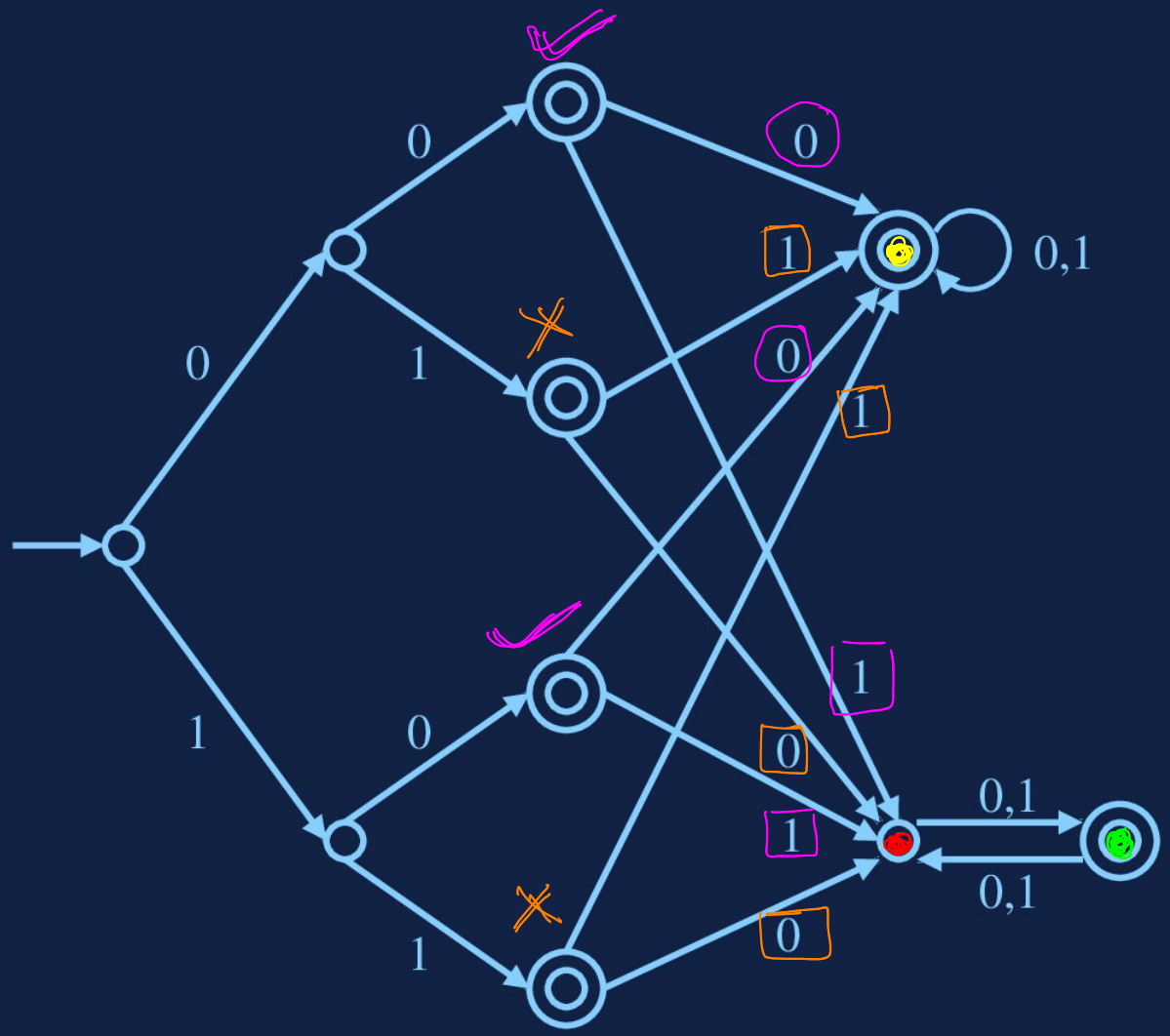
the reject or accept states are not important

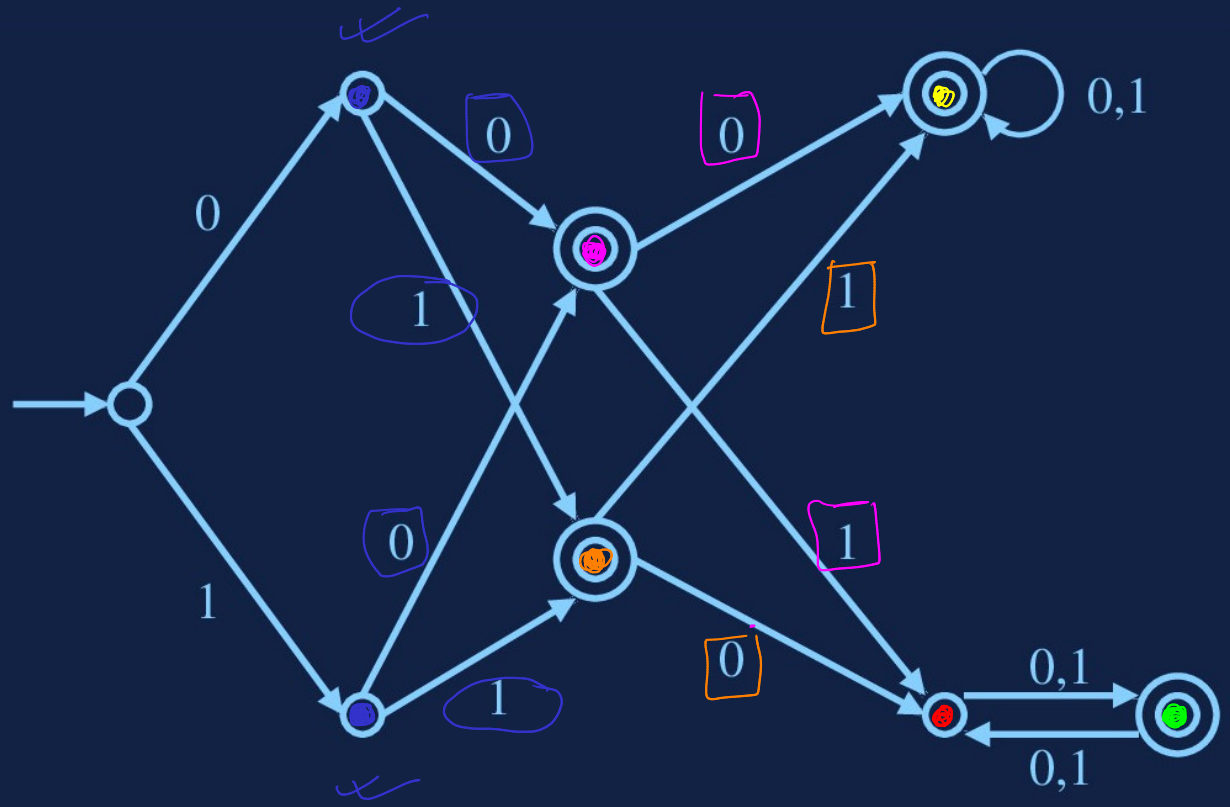
Final — even - accept, odd - reject

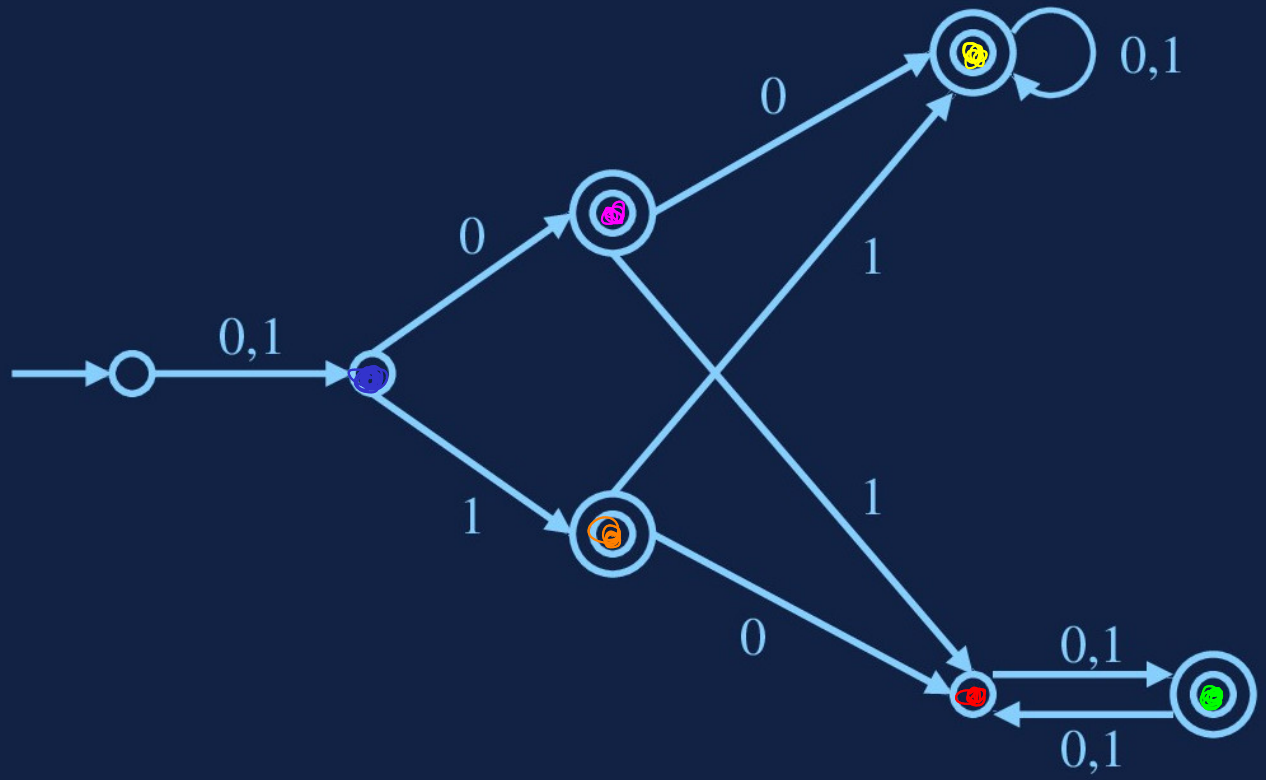
↳ can also be collapsed —



— No luck
— Rule


0
1
0
1
0
1
0
1
0
1
0
1
0
1
0,1
0,1
0,1


0
1
0
1
0
1
0
1
0
1
0,1
0,1
0,1



Exercise:

Are there any more collapsing possibilities?

Defn: $p, q \in Q$ are called **equivalent**, denoted $p \approx q$, if

$$\forall x \in \Sigma^* \left[\hat{\delta}(p, x) \in F \iff \hat{\delta}(q, x) \in F \right]$$

$\approx$ is an equivalence relation on Q.

$$p \approx p, \quad p \approx q \Rightarrow q \approx p, \quad p \approx q, q \approx r \Rightarrow p \approx r$$

Equivalence classes $[p]$ for $p \in Q$.

$$Q' = Q/\approx \; = \{ [p] \mid p \in Q \}$$

$$s' = [s], \quad F' = \{ [p] \mid p \in F \}$$

$$\delta'([p], a) = [\delta(p, a)]$$

$\rightarrow$ safe

$D = (Q, \Sigma, \delta, s, F)$

$D' = (Q', \Sigma, \delta', s', F')$

$D' = D/\approx$

Shrinking each equiv class to a single state.

Quotient construction

- δ' is well-defined

$$p \approx q \quad ([p] = [q]), \quad a \in \Sigma$$

$$\Rightarrow \delta(p,a) \approx \delta(q,a) \quad ([\delta(p,a)] = [\delta(q,a)])$$

$$\delta'([p],a) = \delta'([q],a)$$

Proof: $p \approx q$

$$\forall x \quad \hat{\delta}(p,x) \in F \Leftrightarrow \hat{\delta}(q,x) \in F$$

$x = ay$

$$\hat{\delta}(p,x) = \hat{\delta}(p,ay) = \hat{\delta}(\delta(p,a),y)$$

$$\hat{\delta}(q,x) = \hat{\delta}(\delta(q,a),y).$$

$$\Rightarrow \delta(p,a) \approx \delta(q,a)$$

Generalize

For any string x and for $p \approx q$,

$$\hat{\delta}(p, x) \approx \hat{\delta}(q, x)$$

Proof: Induction on $|x|$ [Exercise]

- $\delta'([p], x) = [\delta(p, x)]$ [Corollary]
- $p \in F \Leftrightarrow [p] \in F'$

Proof: "$\Rightarrow$" Obvious

"$\Leftarrow$" $p \notin F$, but $[p] \in F' \Rightarrow [p] = [q] \in F'$ for some $q \in F$.

$p \approx q$ $\Downarrow$ $(p \notin F$ and $q \in F)$

Theorem: $\mathcal{L}(D) = \mathcal{L}(D')$.

Proof:

$x \in \mathcal{L}(D)$

$\Leftrightarrow \hat{\delta}(s, x) \in F$

$\Leftrightarrow [\hat{\delta}(s, x)] \in F'$

$\Leftrightarrow \hat{\delta}'([s], x) \in F'$

$\Leftrightarrow x \in \mathcal{L}(D')$.

$D' \rightarrow$ collapsed automata

Exercise: Apply the collapsing procedure on D'

$[p] \approx [q]$

$\Rightarrow \forall x \left[\hat{\delta}'([p], x) \in F' \Leftrightarrow \hat{\delta}'([q], x) \in F' \right]$

Show that: $[p] \sim [q] \Leftrightarrow [p] = [q]$.