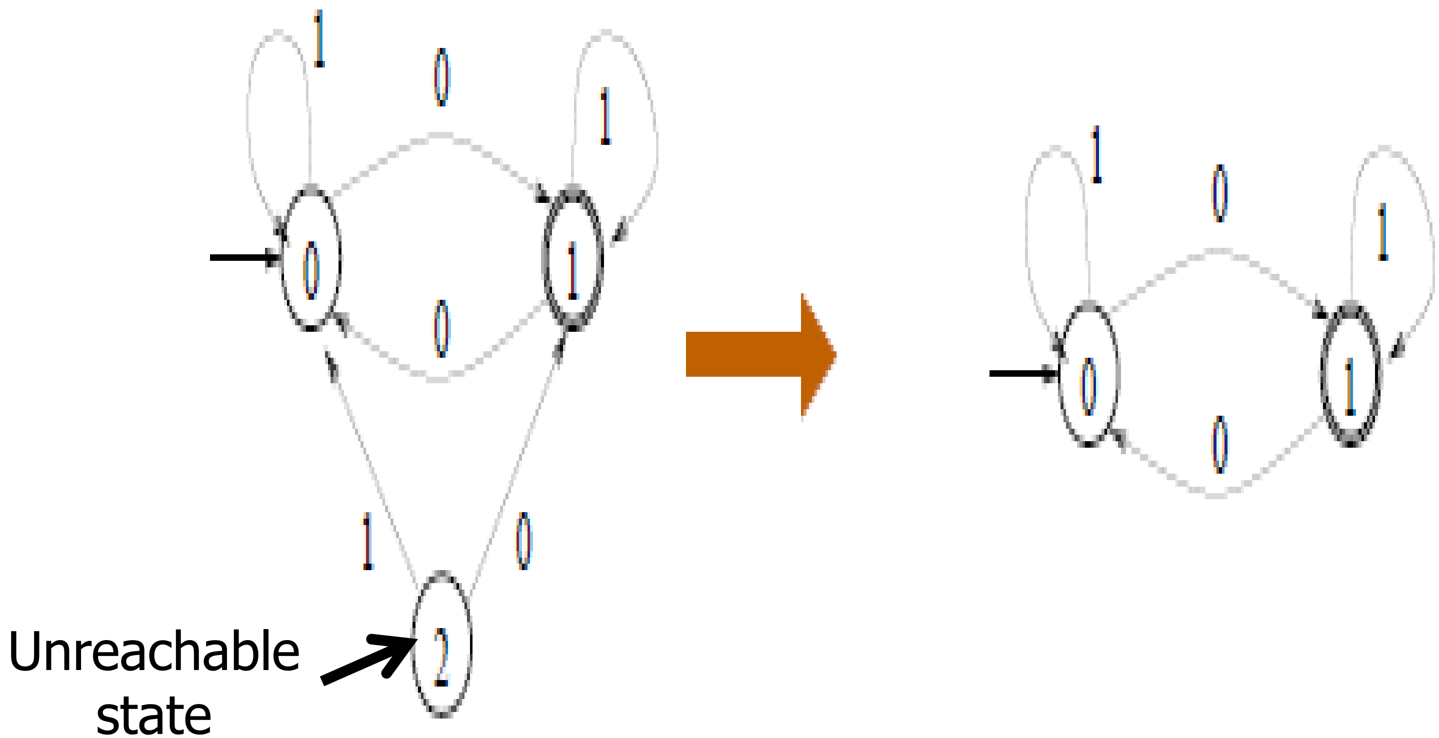
COURSE: THEORY OF AUTOMATA COMPUTATION

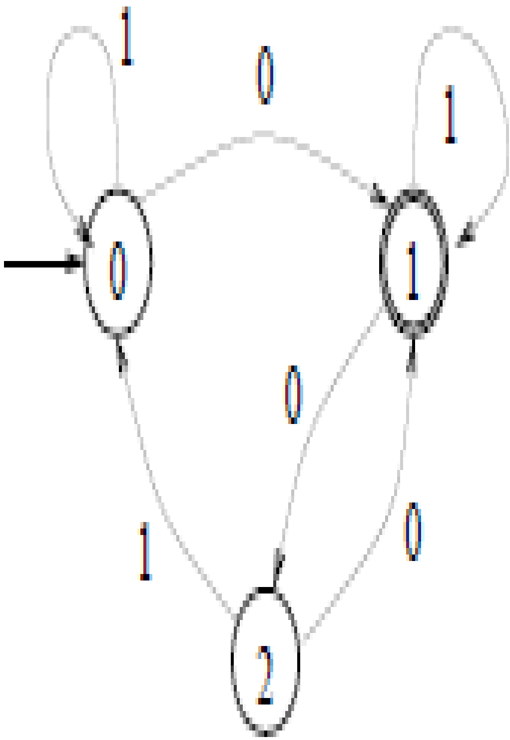
TOPICS TO BE COVERED

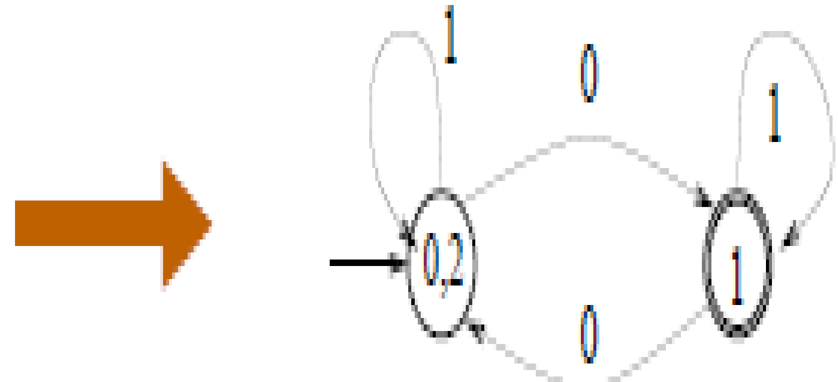
- Minimization of FA
- Minimization Algorithm
  - Guarantees smallest possible DFA for a given regular language
  - Proof of this fact.

EXAMPLE ONE



EXAMPLE TWO

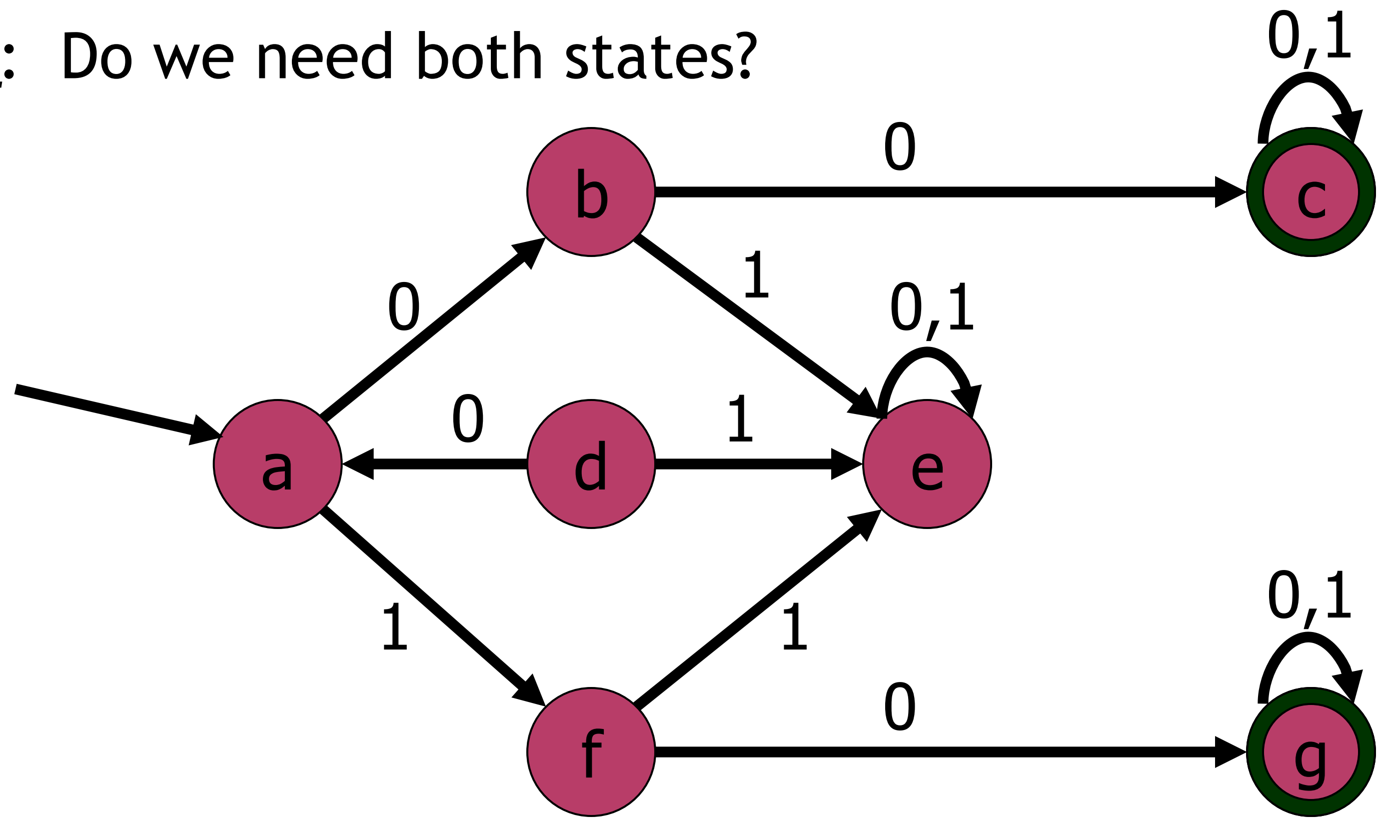




EQUIVALENT STATES. EXAMPLE

Consider the accept states c and g. They are both sinks meaning that any string which ever reaches them is guaranteed to be accepted later.

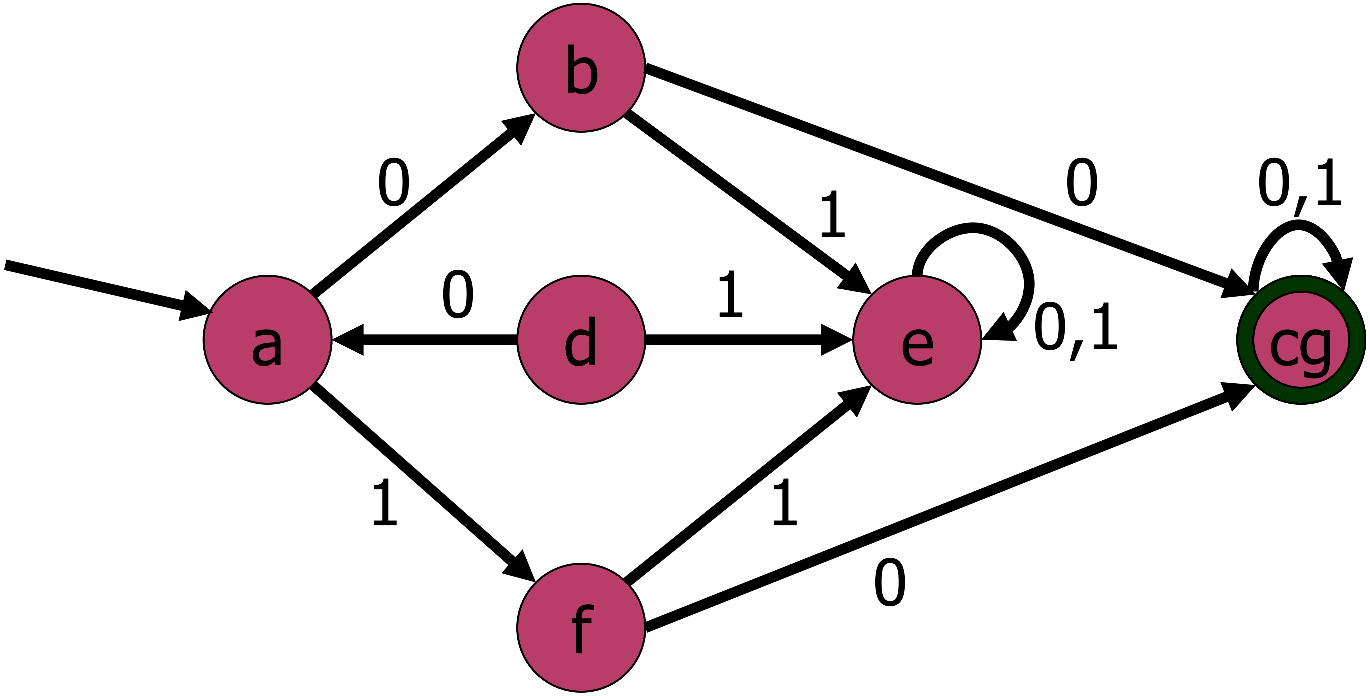
Q: Do we need both states?



EQUIVALENT STATES. EXAMPLE

A: No, they can be unified as illustrated below.

Q: Can any other states be unified because any subsequent string suffixes produce identical results?

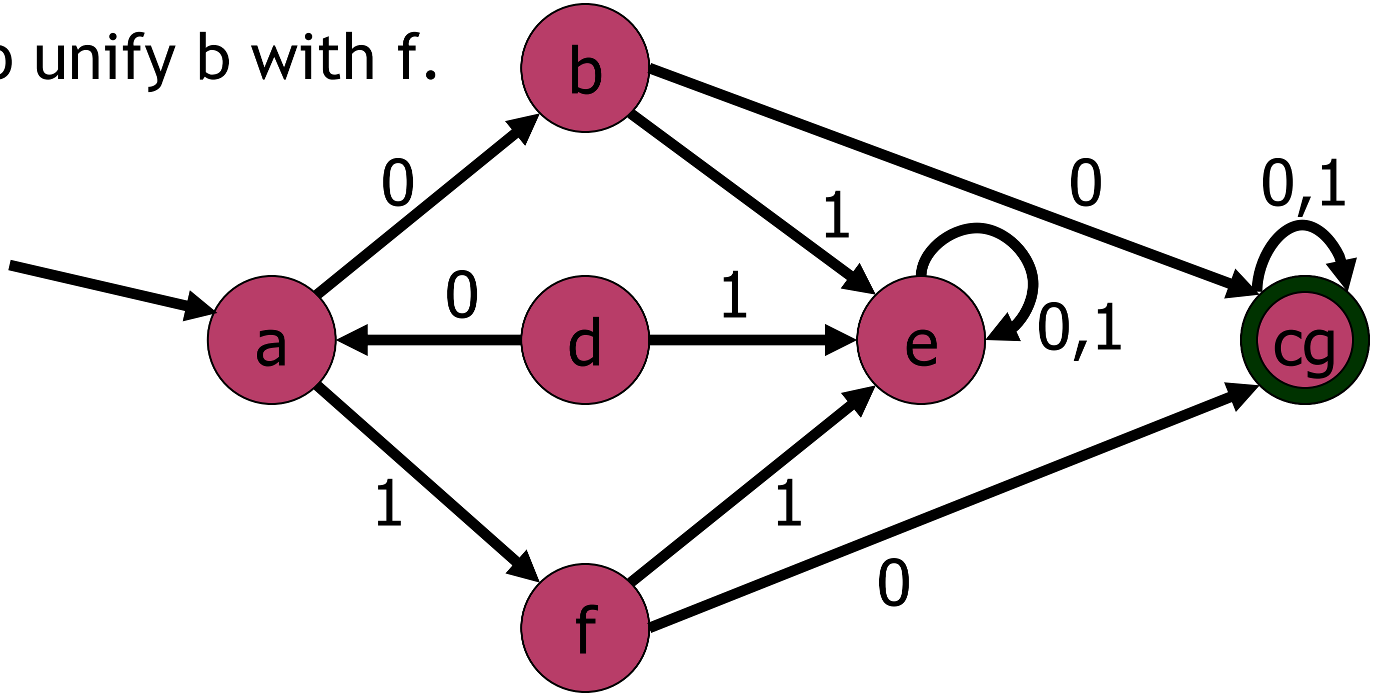


EQUIVALENT STATES. EXAMPLE

A: Yes, b and f. Notice that if you're in b or f then:

1. if string ends, reject in both cases
2. if next character is 0, forever accept in both cases
3. if next character is 1, forever reject in both cases

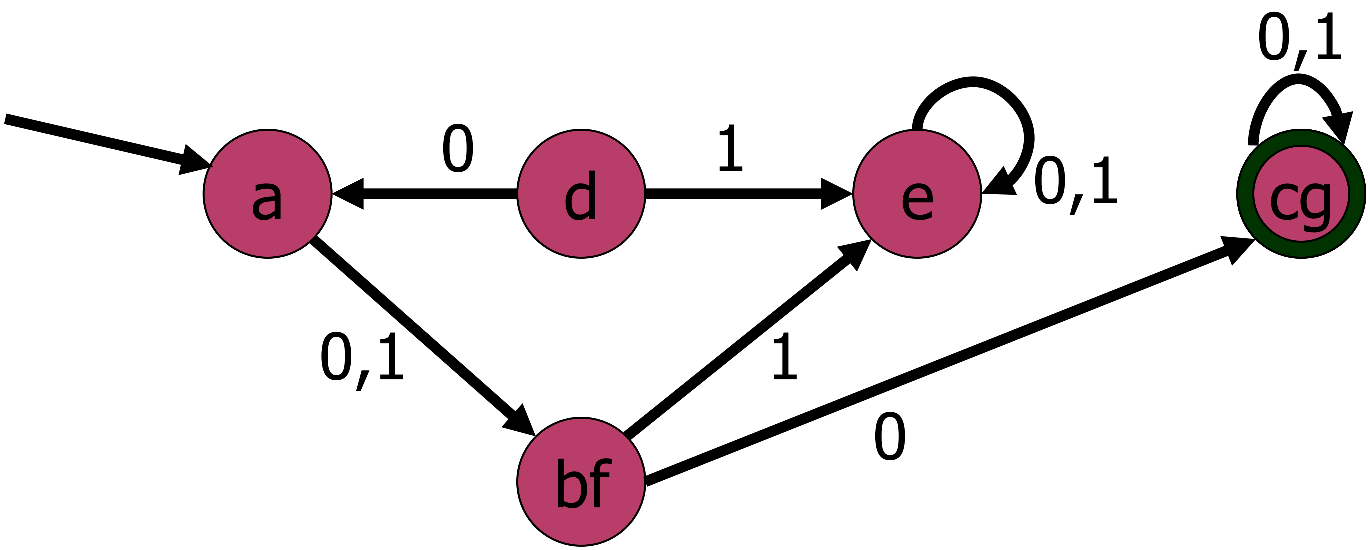
So unify b with f.



EQUIVALENT STATES. EXAMPLE

Intuitively two states are equivalent if all subsequent behavior from those states is the same.

Q: Come up with a formal characterization of state equivalence.



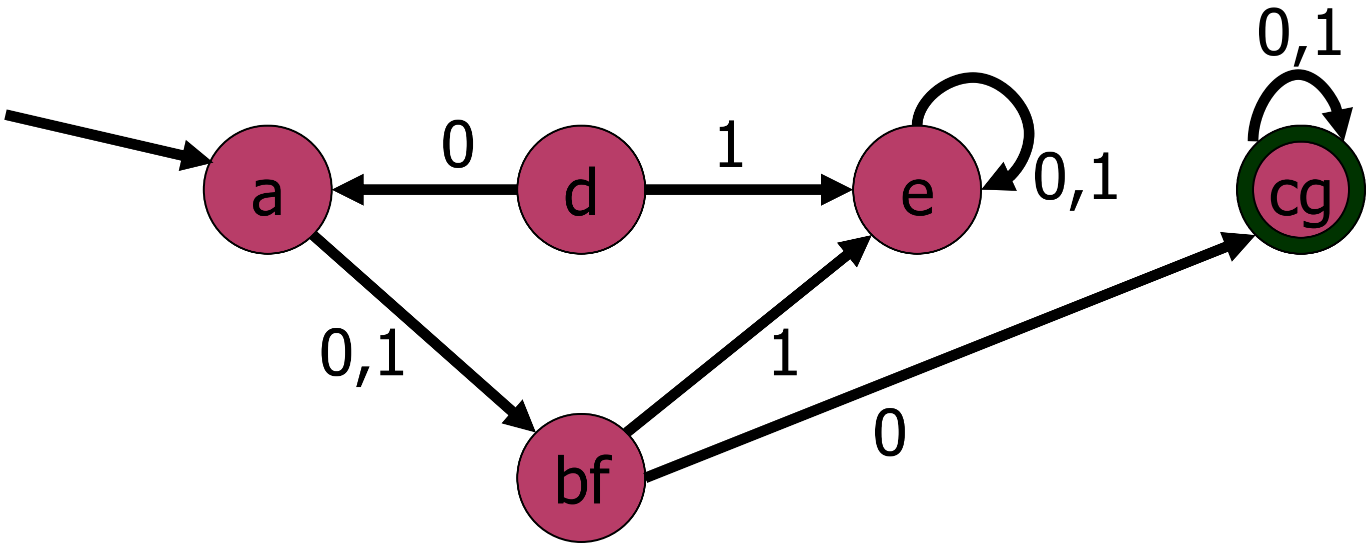
EQUIVALENT STATES. DEFINITION

DEF: Two states q and q' in a DFA $M = (Q, \Sigma, \delta, q_0, F)$ are said to be ***equivalent*** (or ***indistinguishable***) if for all strings $u \in \Sigma^*$, the states on which u ends on when read from q and q' are both accept, or both non-accept.

Equivalent states may be glued together without affecting M's behavior.

FINISHING THE EXAMPLE

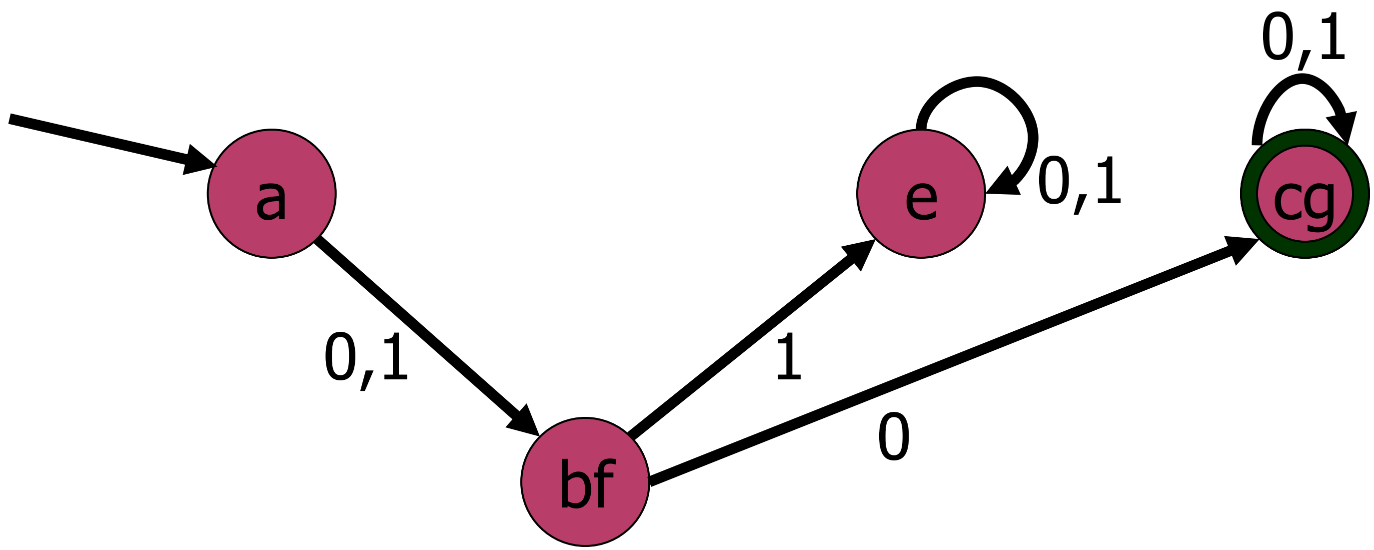
Q: Any other ways to simplify the automaton?



USELESS STATES

A: Get rid of d.

Getting rid of unreachable ***useless states*** doesn't affect the accepted language.



MINIMIZATION ALGORITHM. GOALS

DEF: An automaton is ***irreducible*** if

- it contains no useless states, and
- no two distinct states are equivalent.

The goal of minimization algorithm is to create irreducible automata from arbitrary ones. Later: remarkably, the algorithm actually produces smallest possible DFA for the given language, hence the name "minimization".

The minimization algorithm *reverses* previous example. Start with least possible number of states, and create new states when forced to.

Explain with a game:

Minimization Algorithm. (Partition Refinement) Code

DFA minimize(DFA (Q, Σ, δ, q_0, F))

remove any state q unreachable from q_0

Partition P = {F, Q - F }

boolean Consistent = false

while(Consistent == false)

Consistent = true

for(every Set $S \in P$, char $a \in \Sigma$, Set $T \in P$)

Set temp = $\{q \in T \mid \delta(q,a) \in S\}$

if (temp != Ø && temp != T)

Consistent = false

$P = (P - T) \cup \{\text{temp}, T - \text{temp}\}$

return defineMinimizor((Q, Σ, δ, q_0, F), P)

Minimization Algorithm. (Partition Refinement) Code

DFA defineMinimizor

$(\text{DFA } (Q, \Sigma, \delta, q_0, F), \text{ Partition } P)$

Set $Q' = P$

State q'_0 = the set in P which contains q_0

$F' = \{ S \in P \mid S \subseteq F \}$

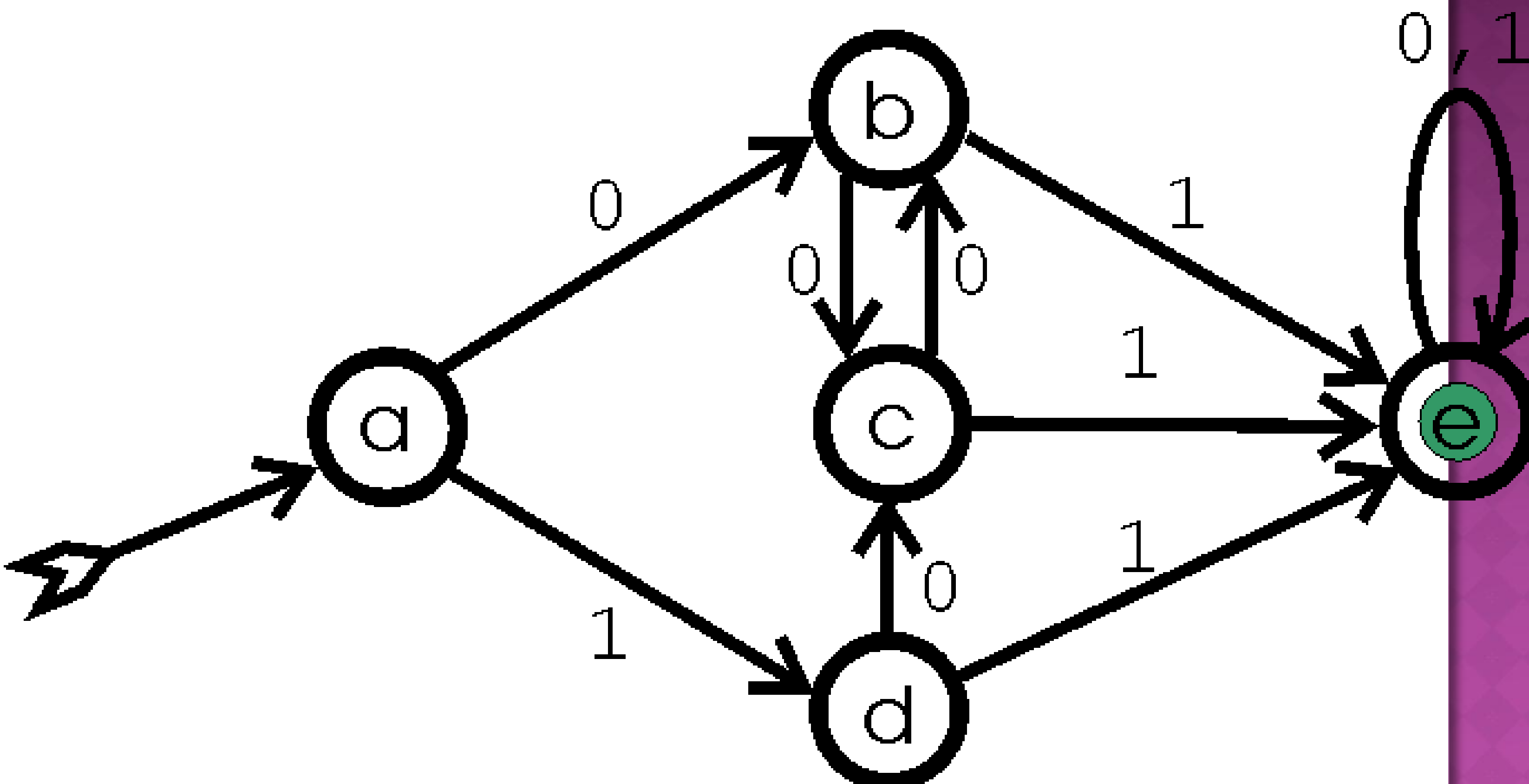
for (each $S \in P$, $a \in \Sigma$)

define $\delta'(S,a)$ = the set $T \in P$ which contains the states $\delta'(S,a)$

return $(Q', \Sigma, \delta', q'_0, F')$

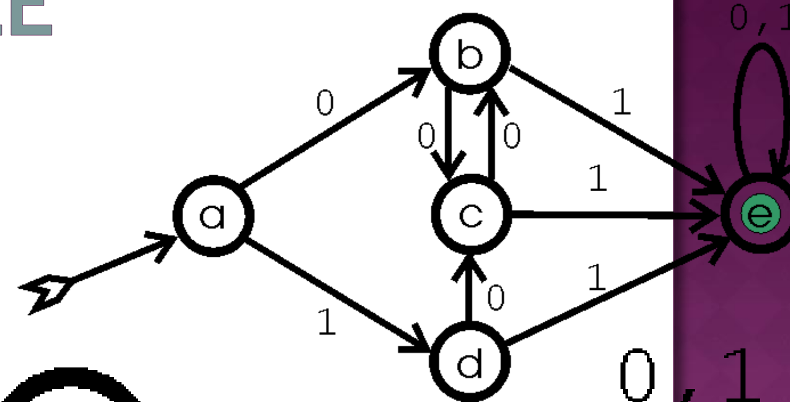
MINIMIZATION EXAMPLE

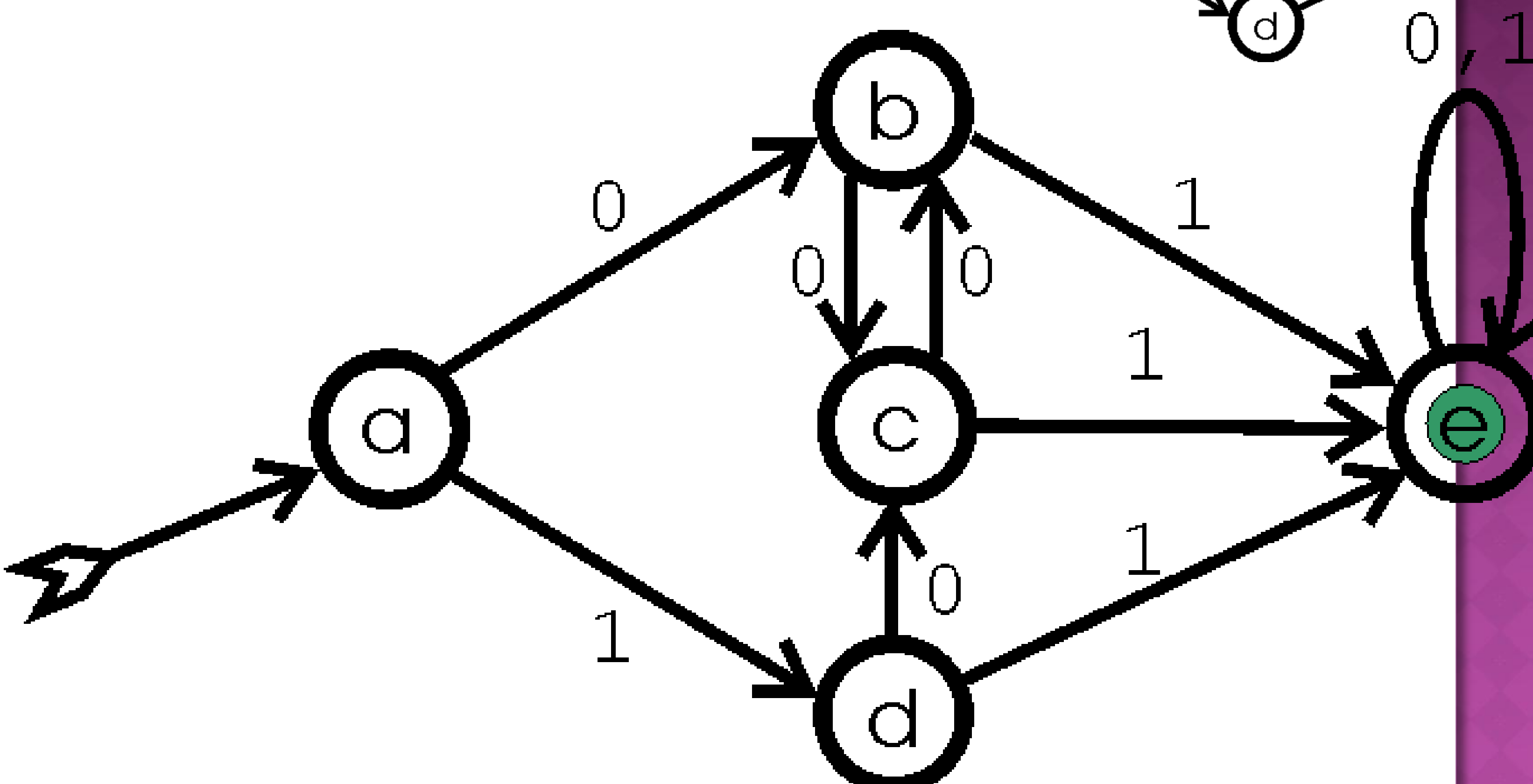
Start with a DFA



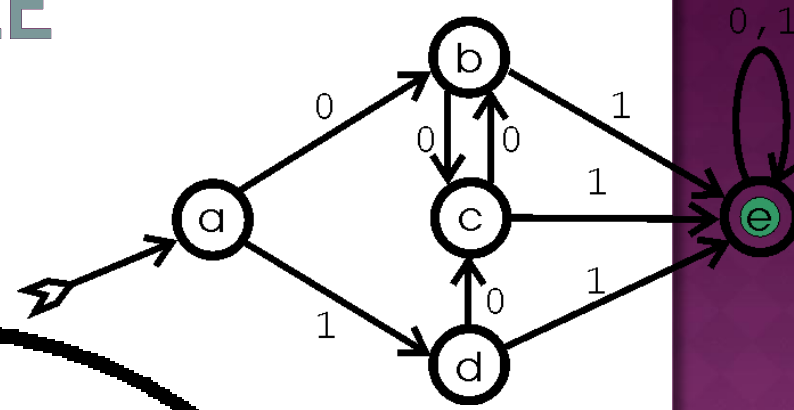
MINIMIZATION EXAMPLE

Miniature version →





MINIMIZATION EXAMPLE

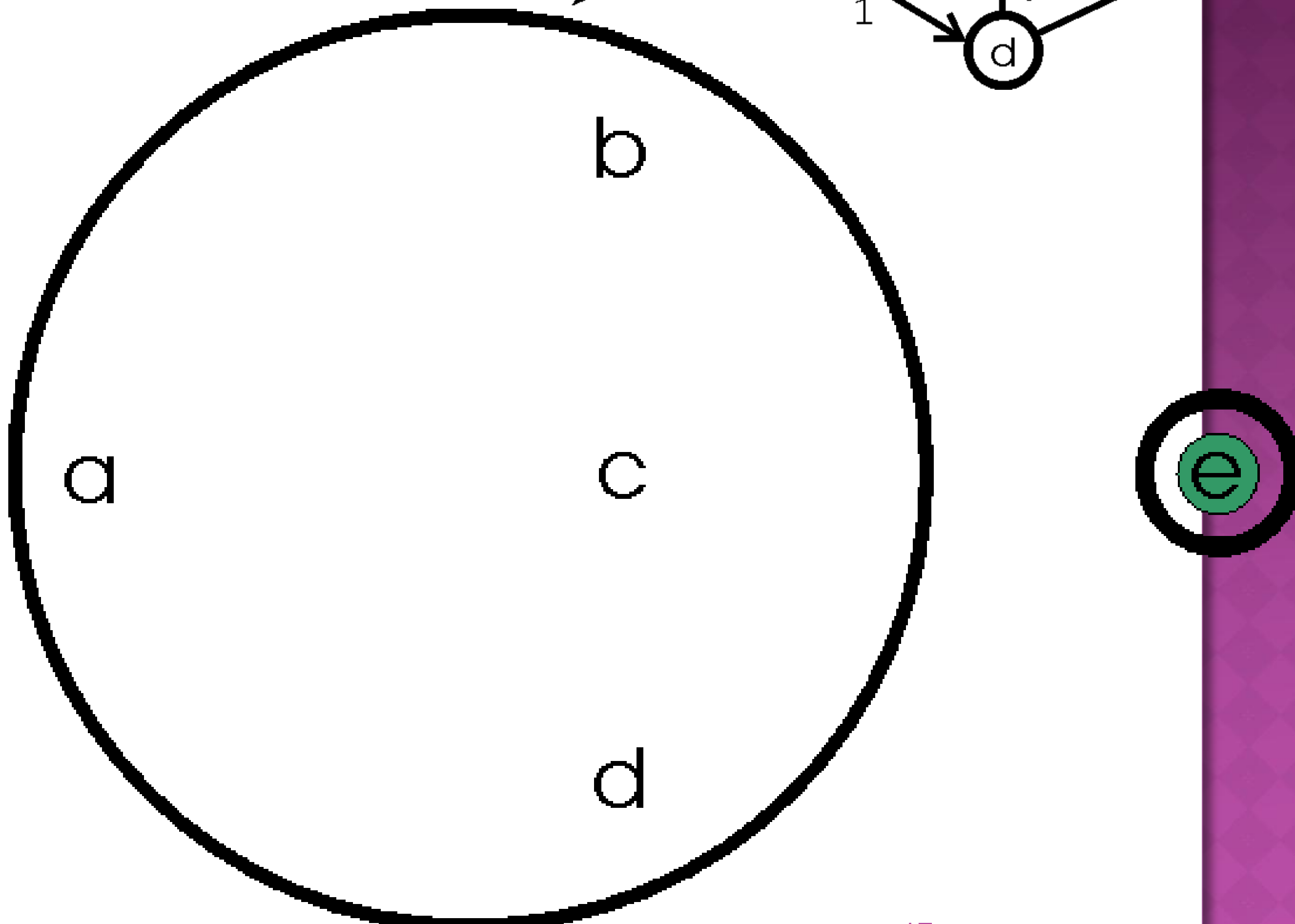


Split into two teams.

ACCEPT

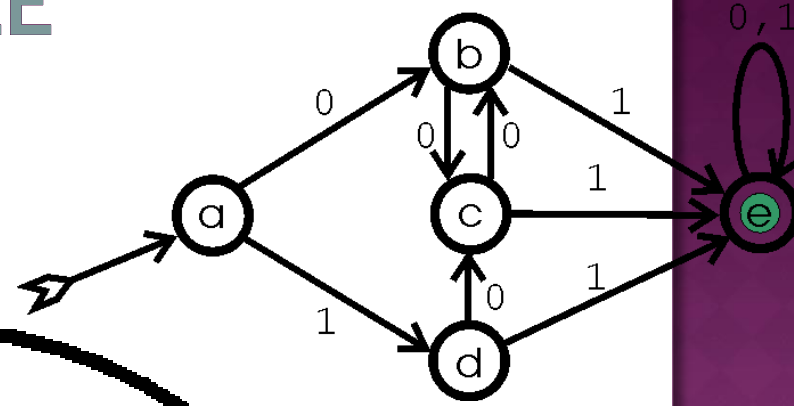
vs.

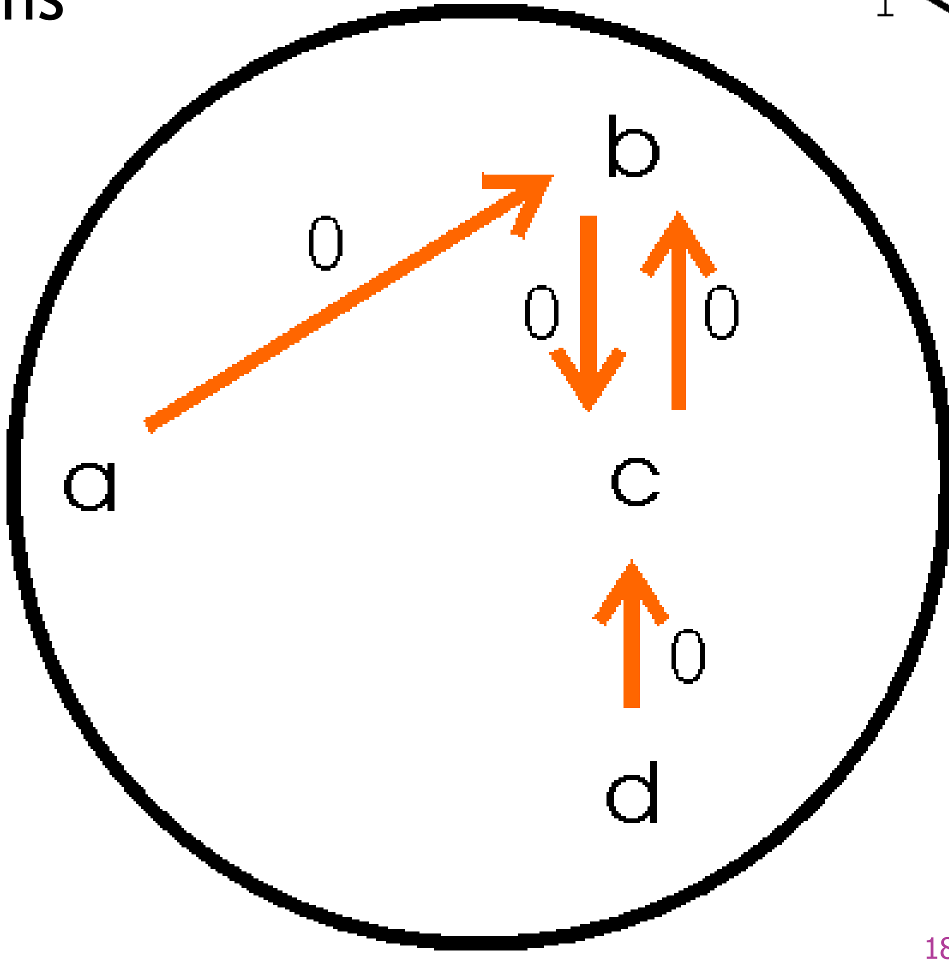
REJECT



MINIMIZATION EXAMPLE

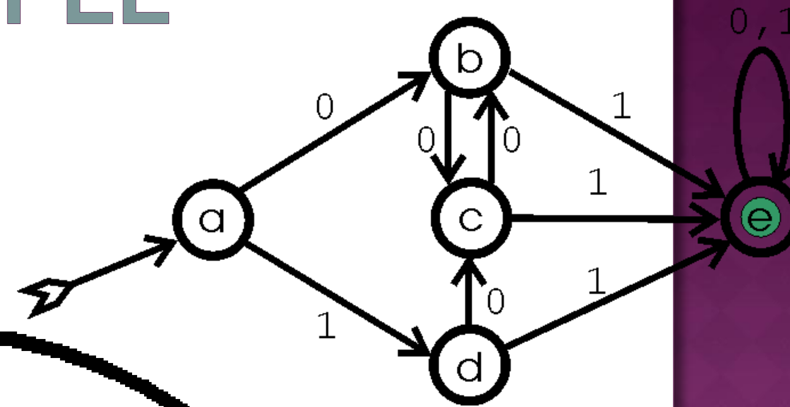
0-label doesn't split up any teams

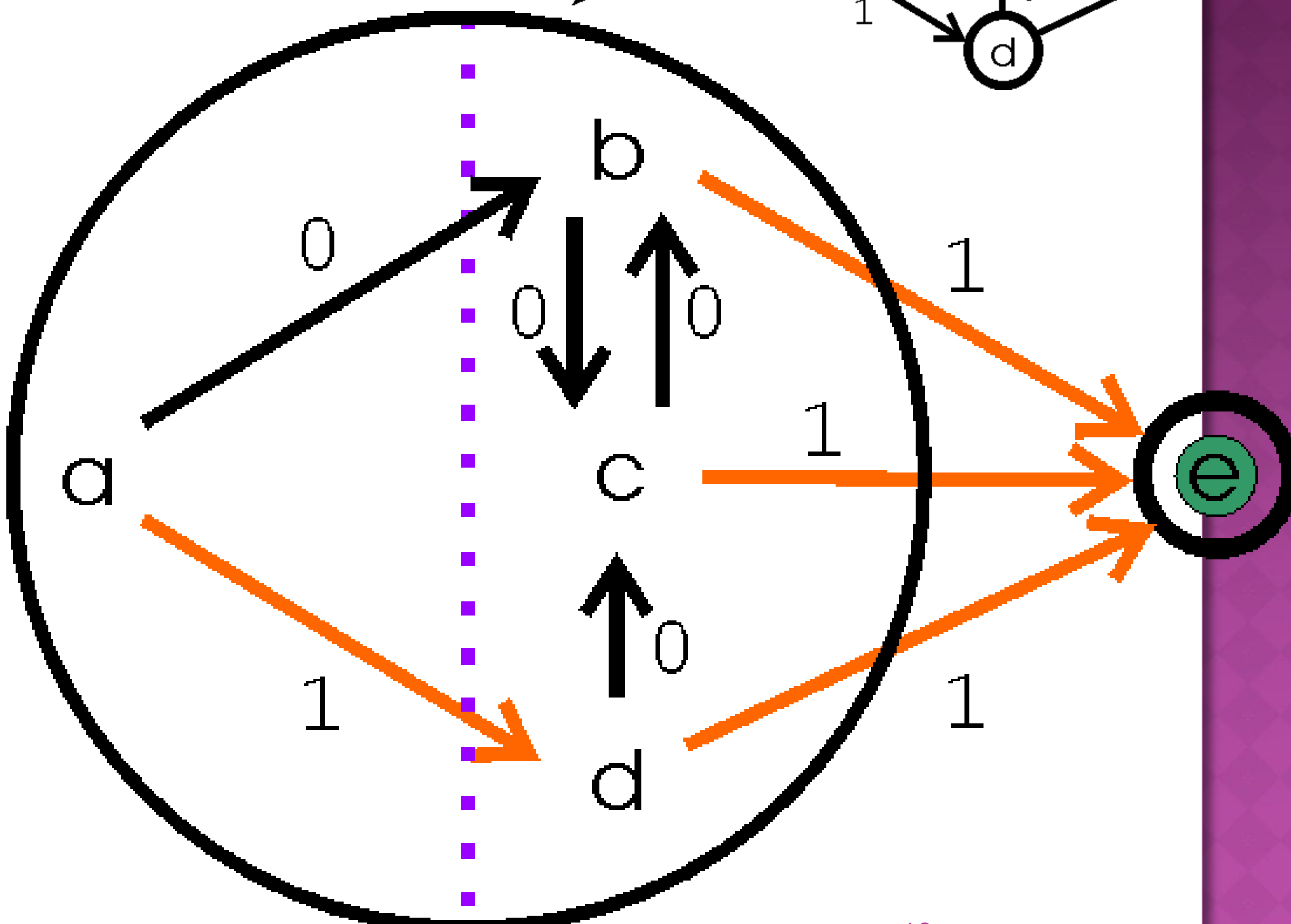




MINIMIZATION EXAMPLE

1-label splits up REJECT's

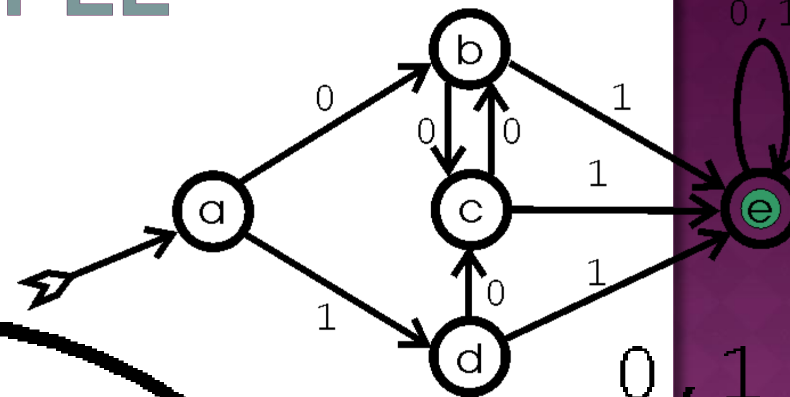


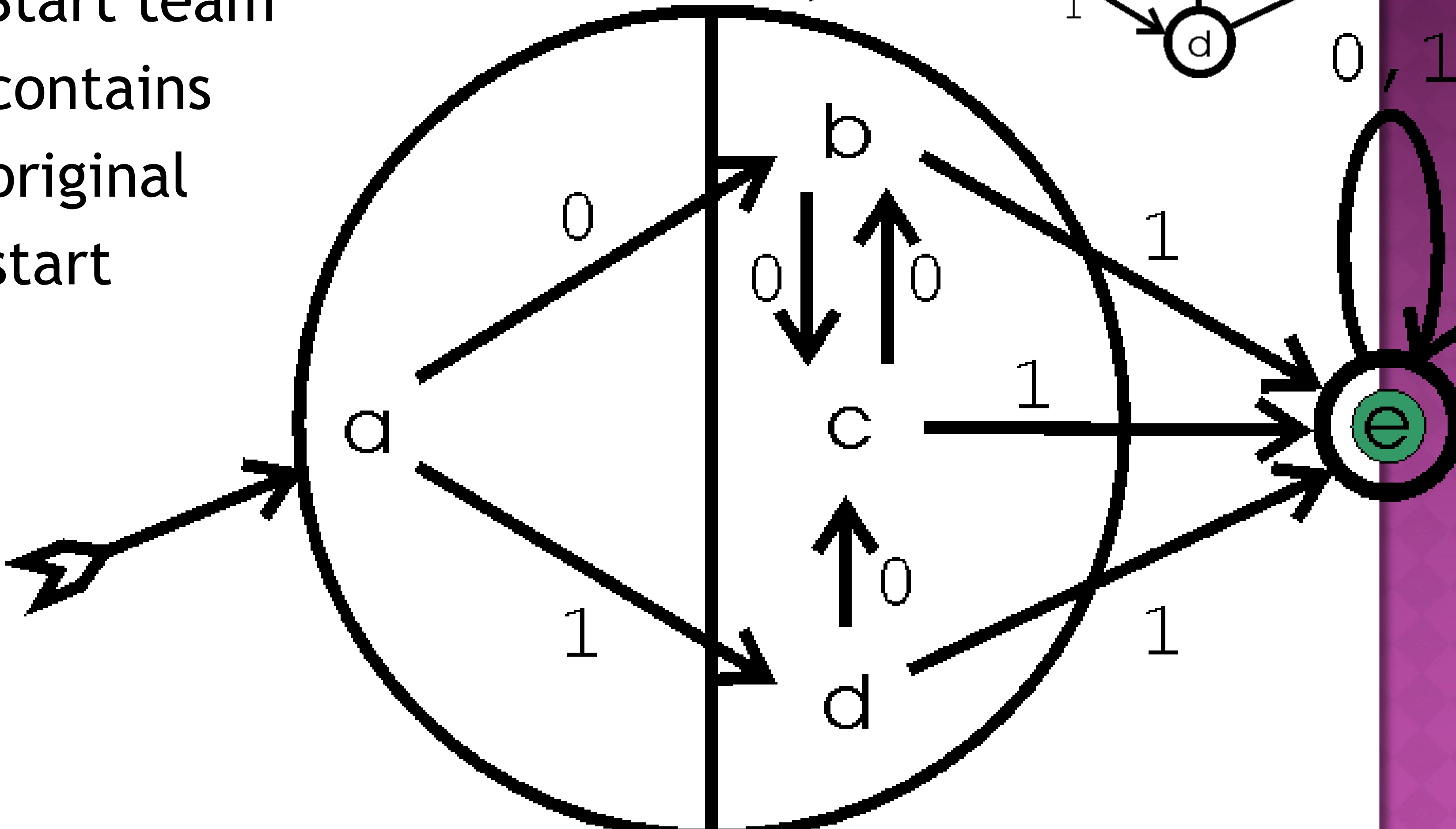


MINIMIZATION EXAMPLE

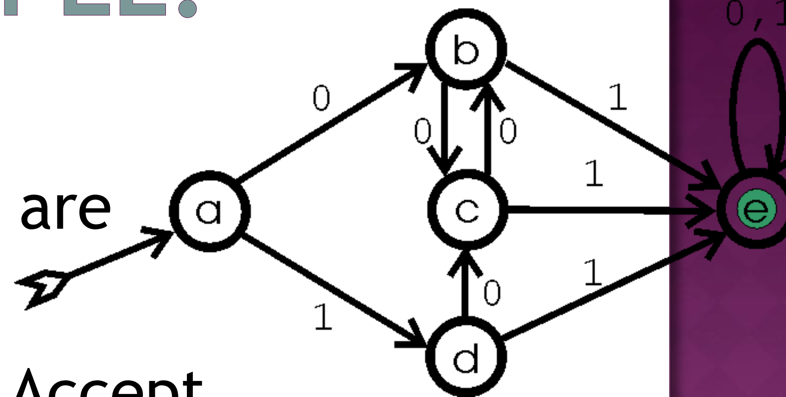
No further splits. HALT!

Start team contains original start

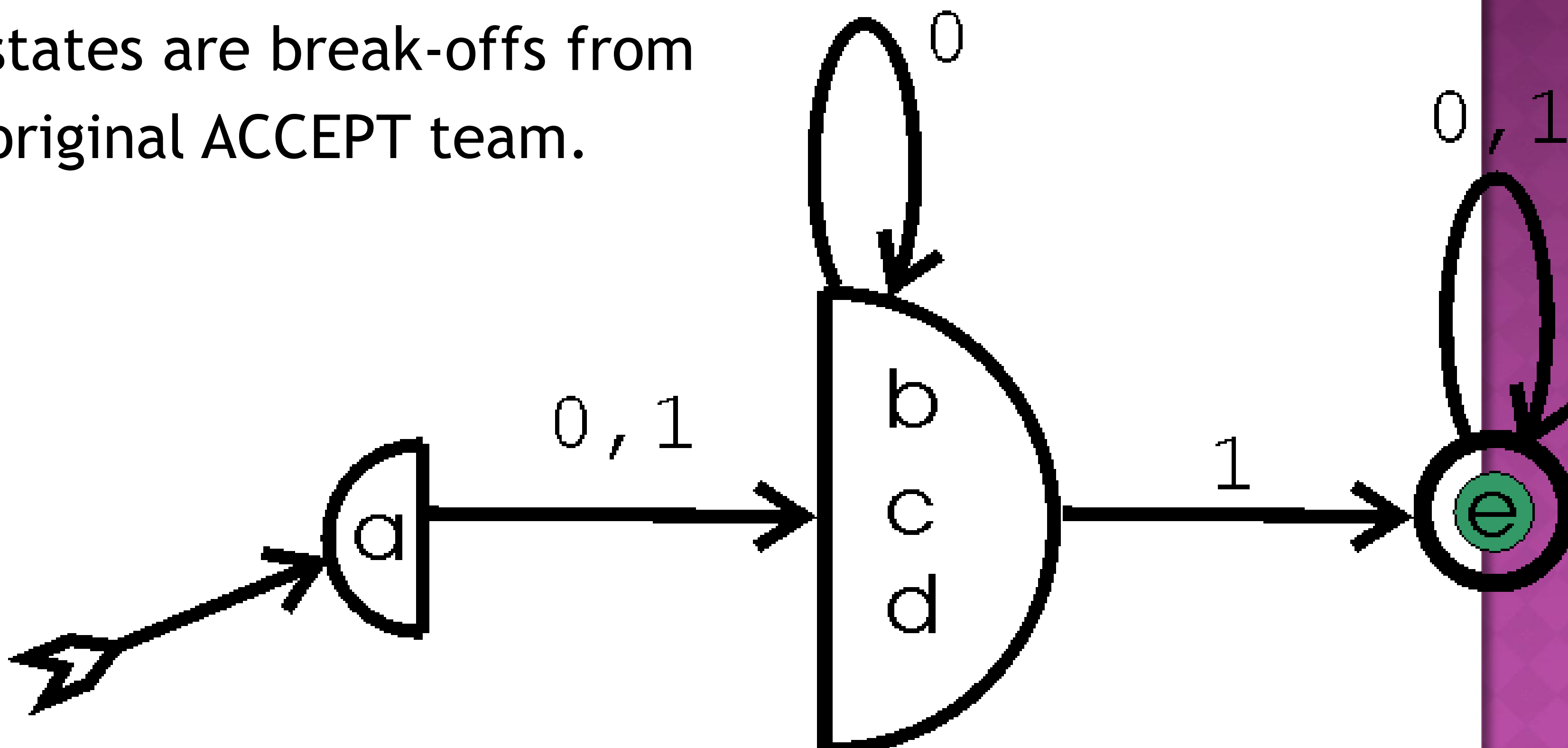




MINIMIZATION EXAMPLE. END RESULT

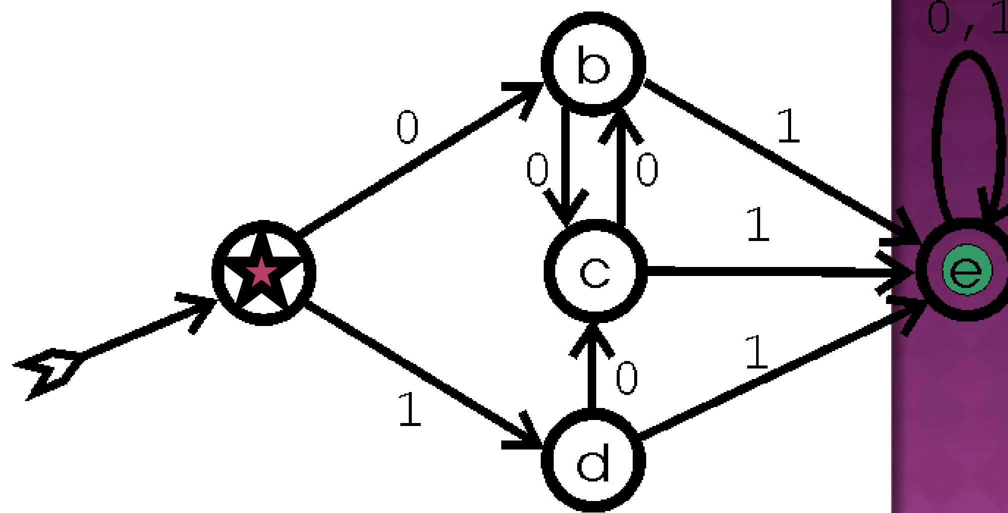


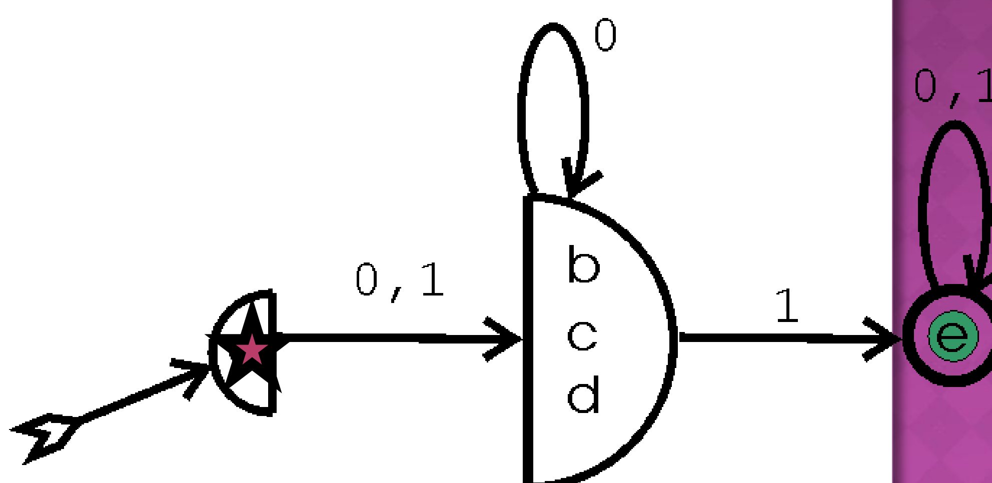
States of the minimal automata are remaining teams. Edges are consolidated across each team. Accept states are break-offs from original ACCEPT team.



MINIMIZATION EXAMPLE. COMPARE

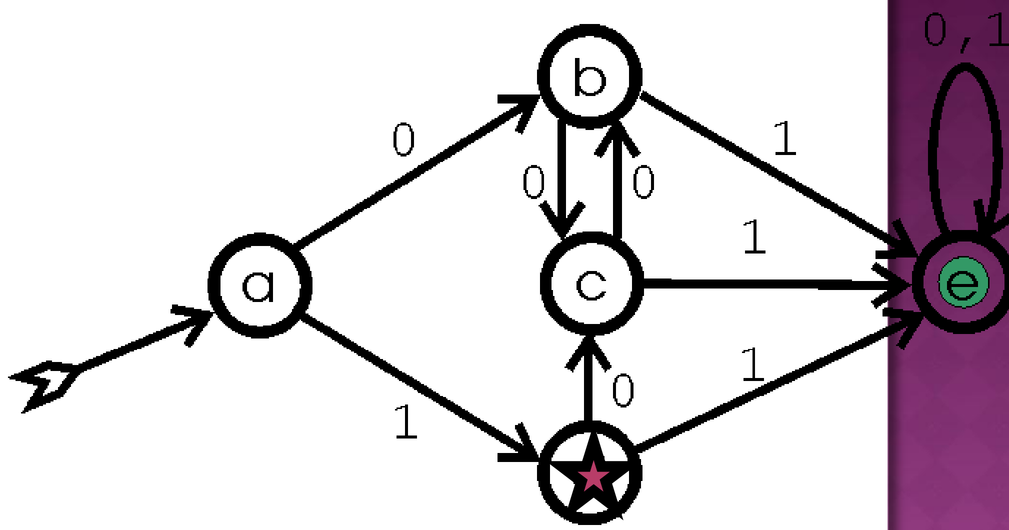
100100101

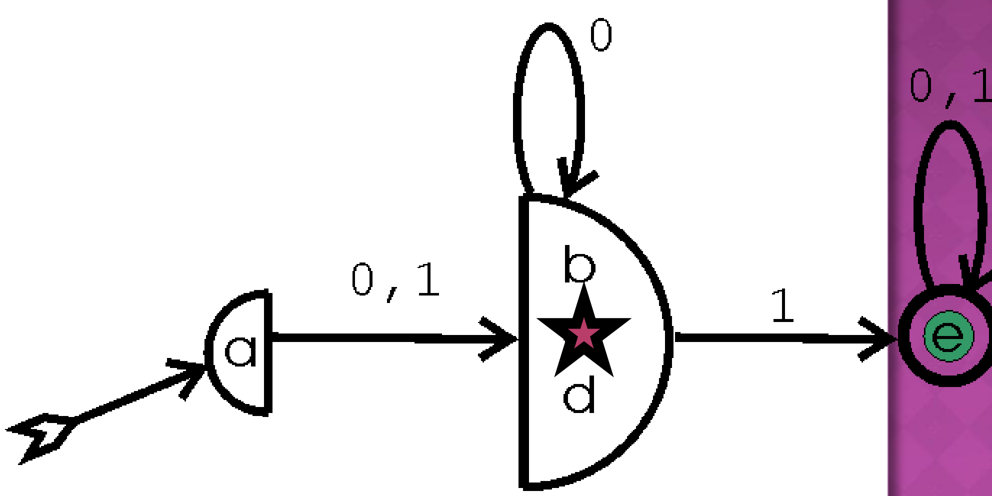




MINIMIZATION EXAMPLE. COMPARE

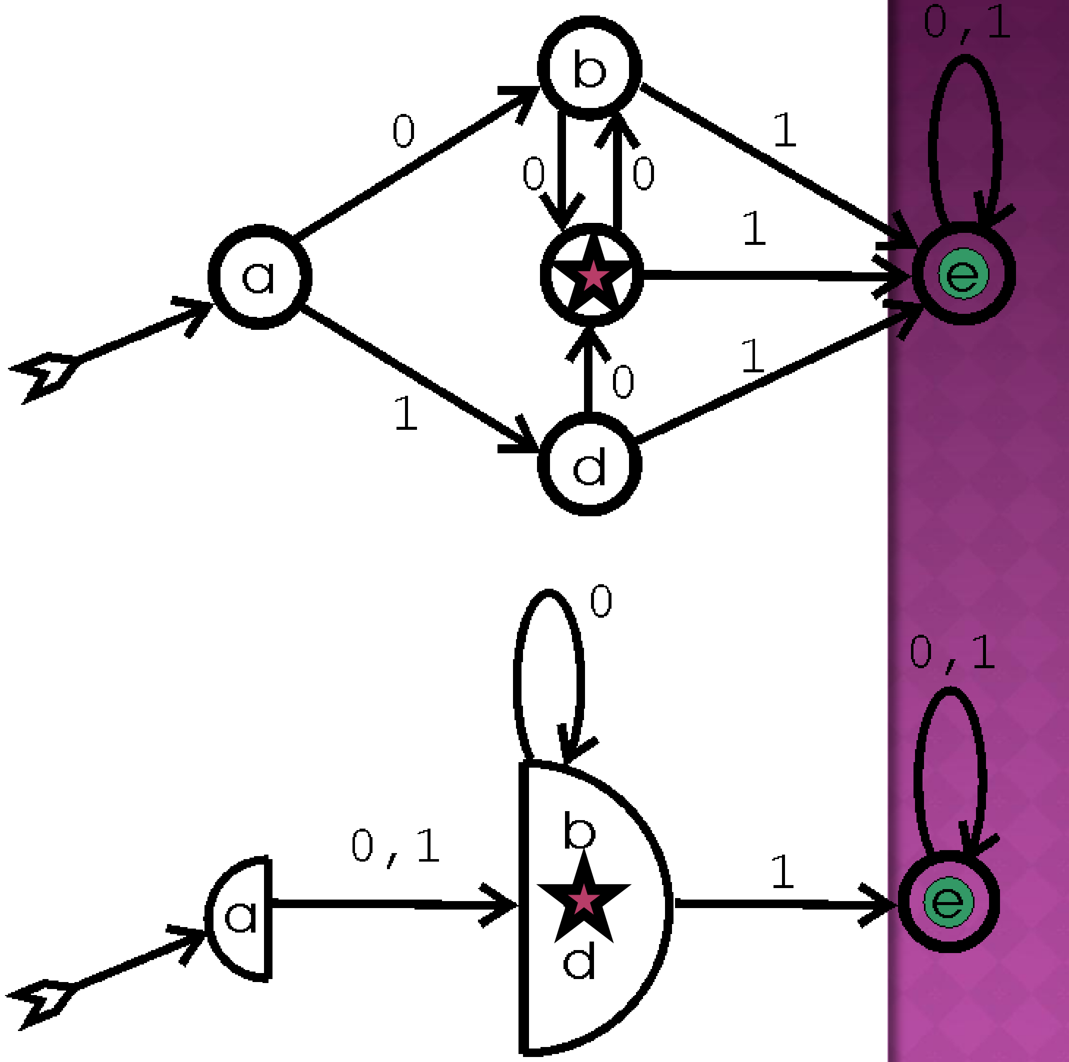
100100101





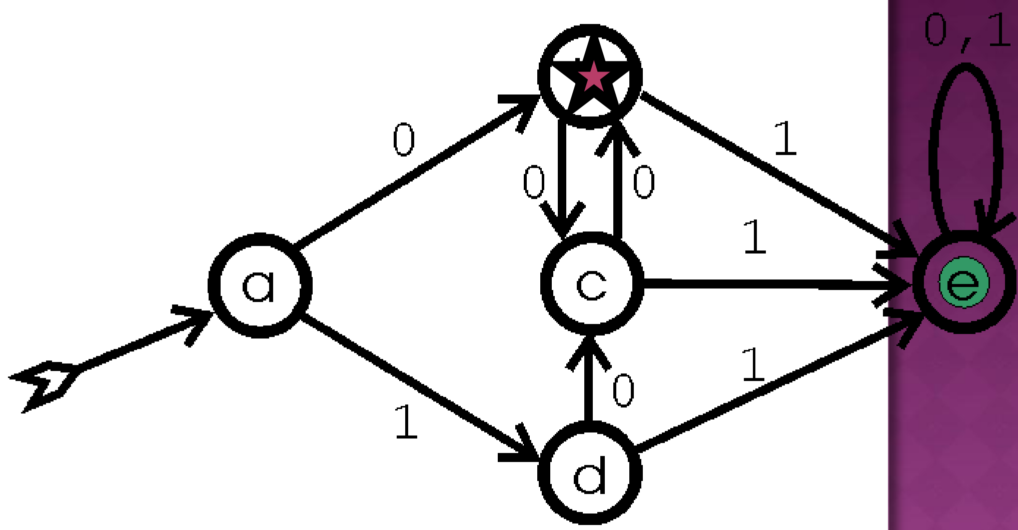
MINIMIZATION EXAMPLE. COMPARE

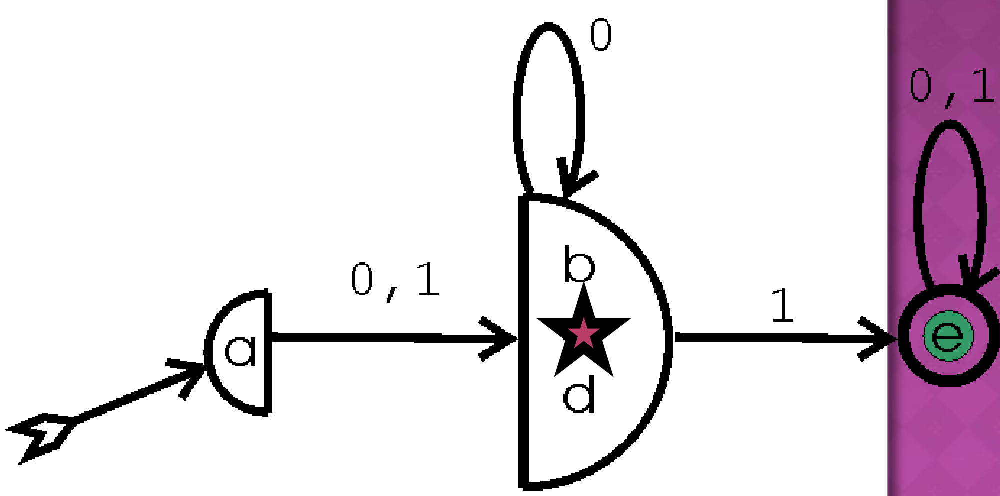
100100101



MINIMIZATION EXAMPLE. COMPARE

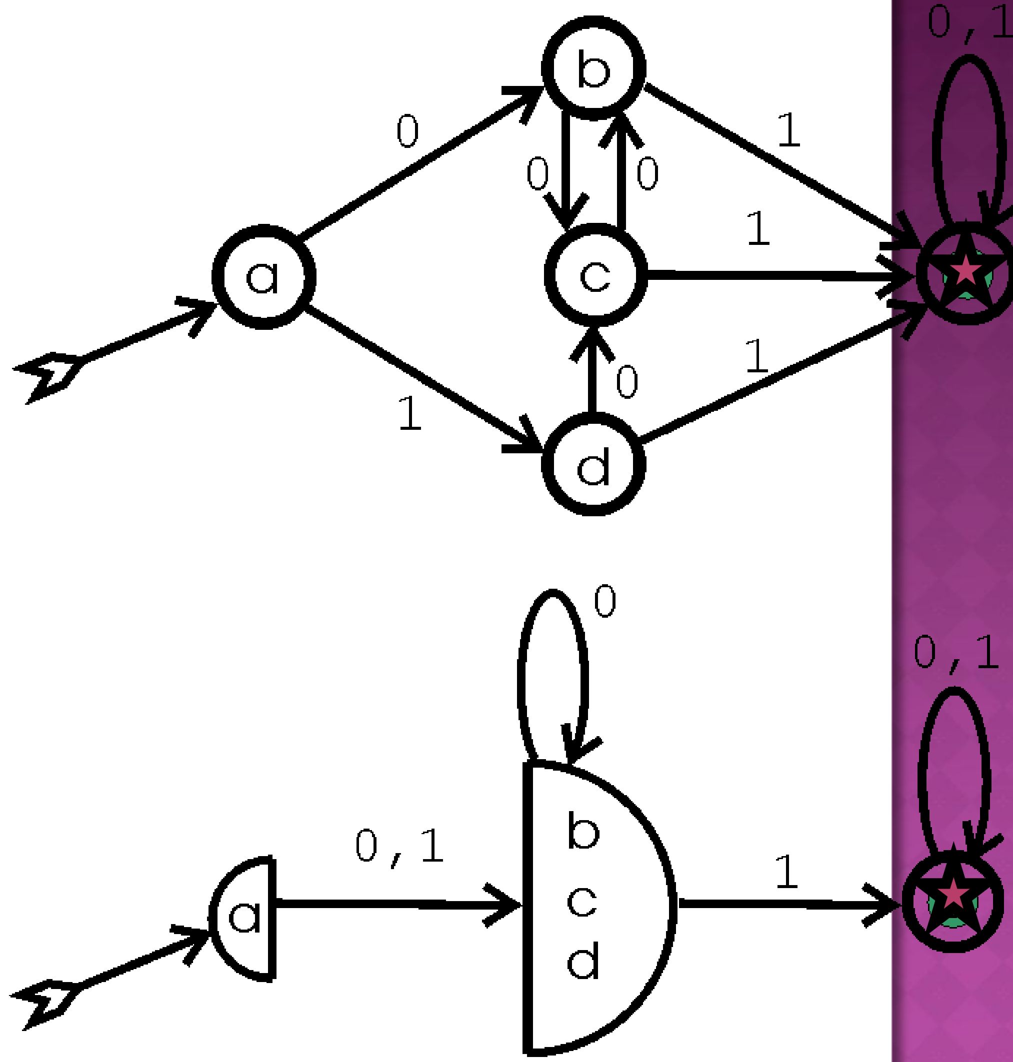
100100101





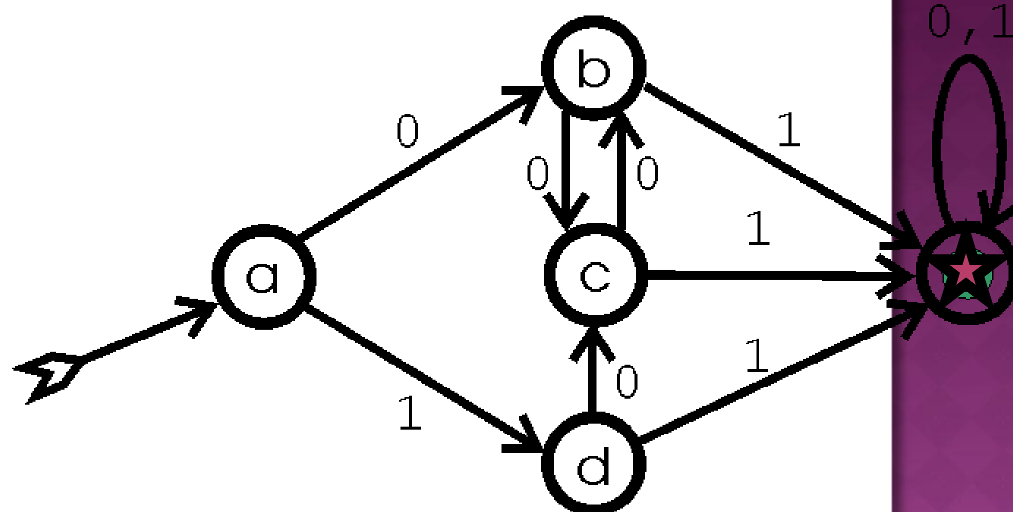
MINIMIZATION EXAMPLE. COMPARE

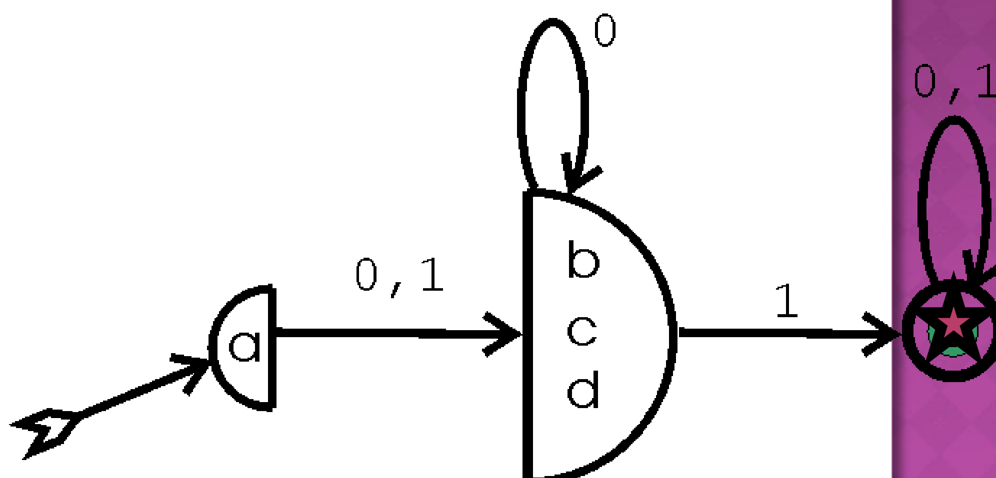
100100101



MINIMIZATION EXAMPLE. COMPARE

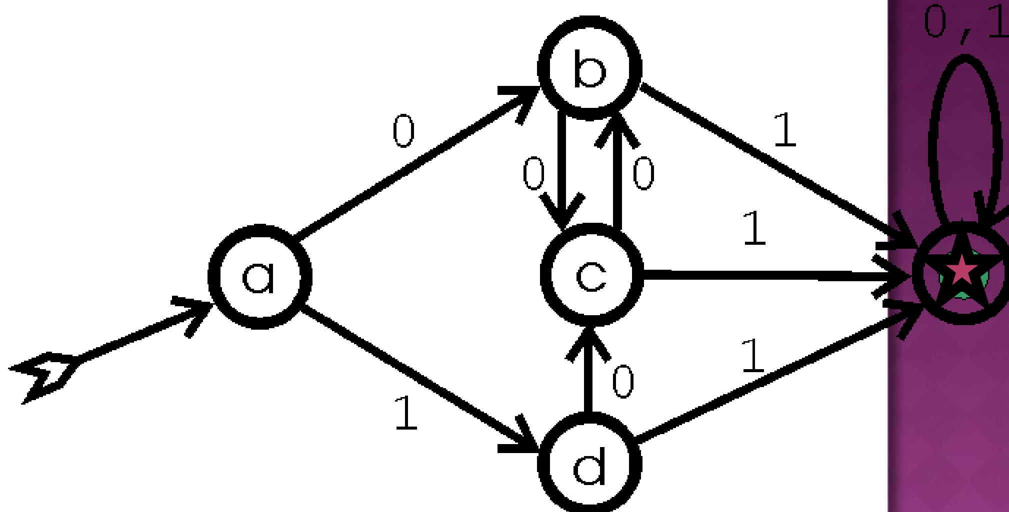
100100101

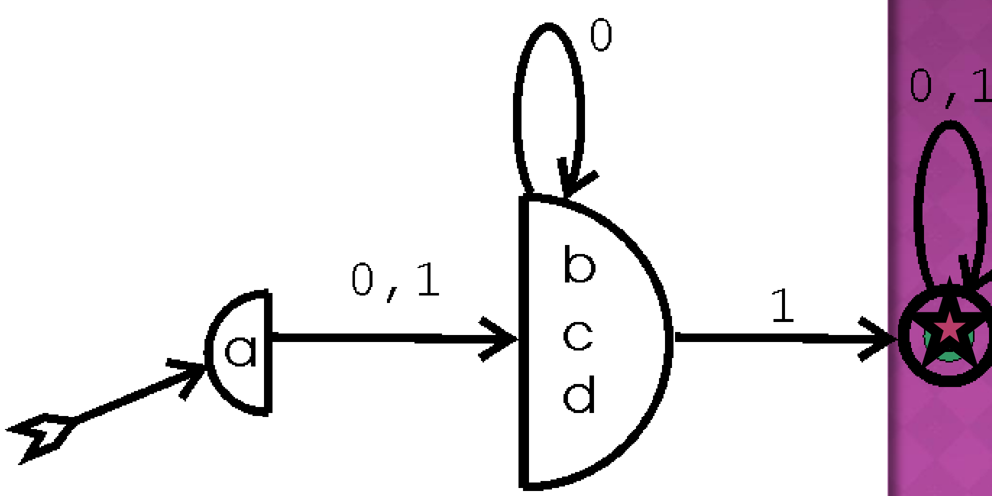




MINIMIZATION EXAMPLE. COMPARE

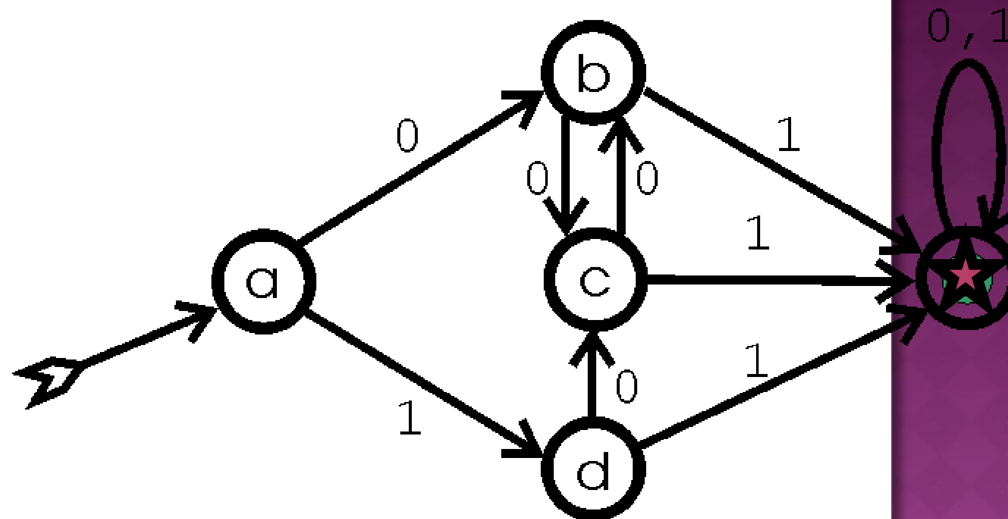
100100101

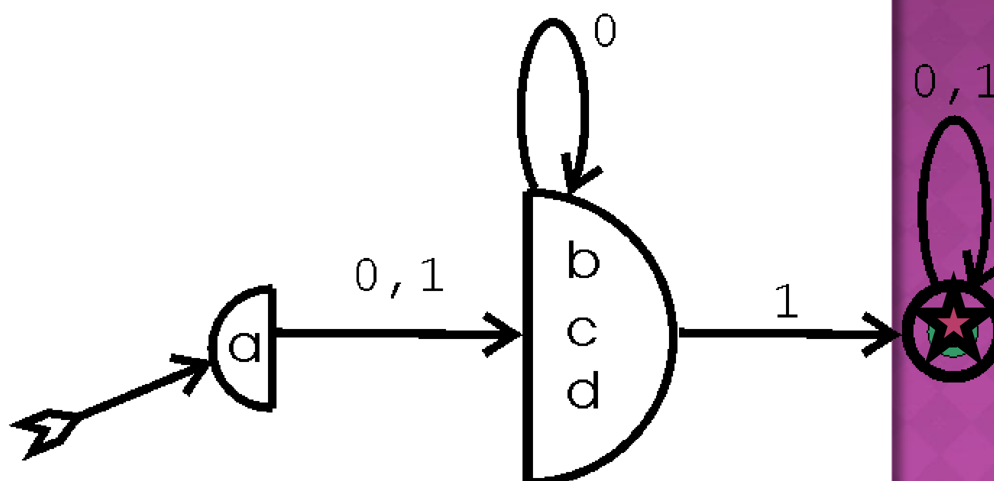




MINIMIZATION EXAMPLE. COMPARE

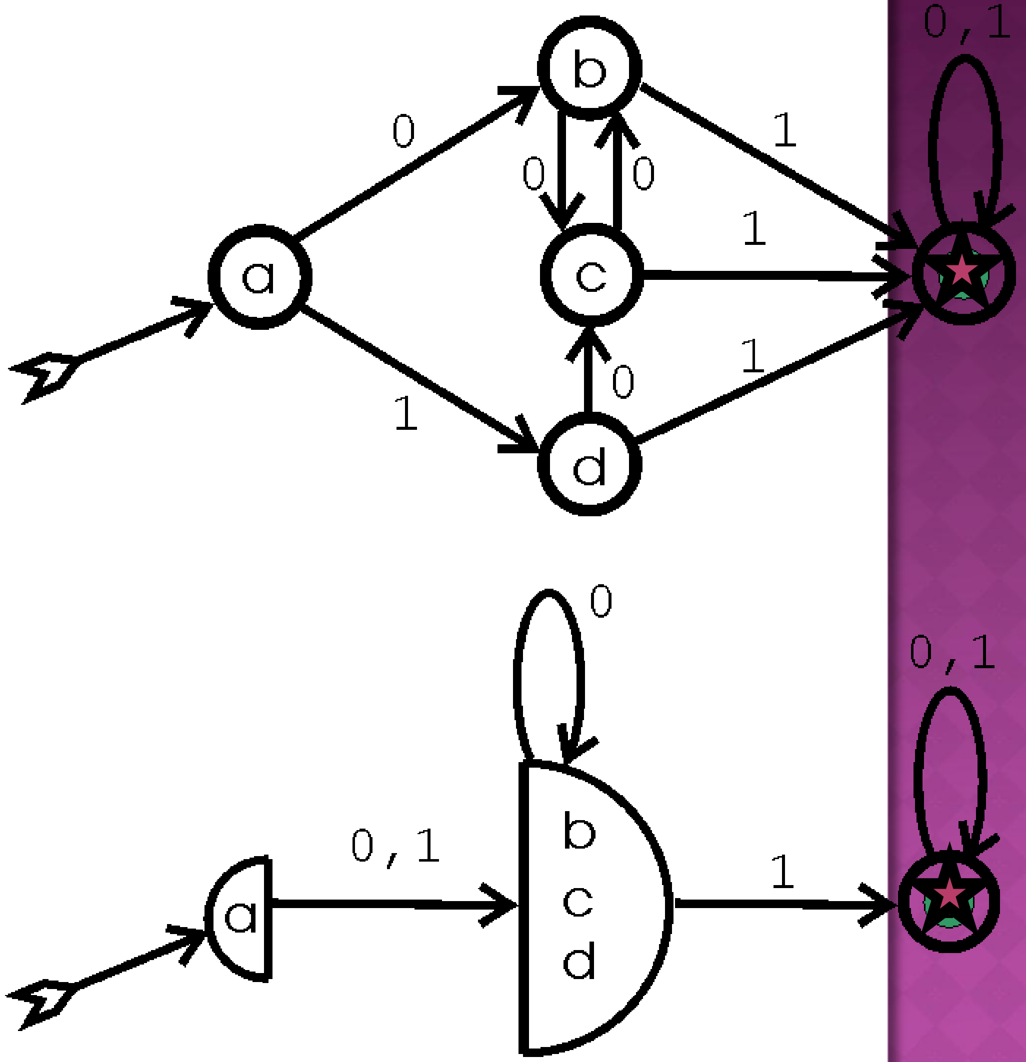
100100101





MINIMIZATION EXAMPLE. COMPARE

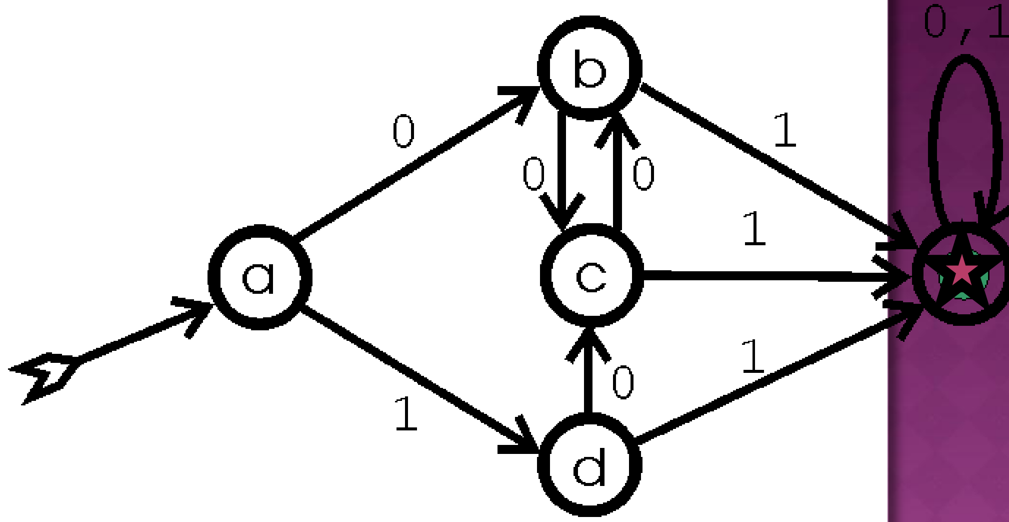
100100101

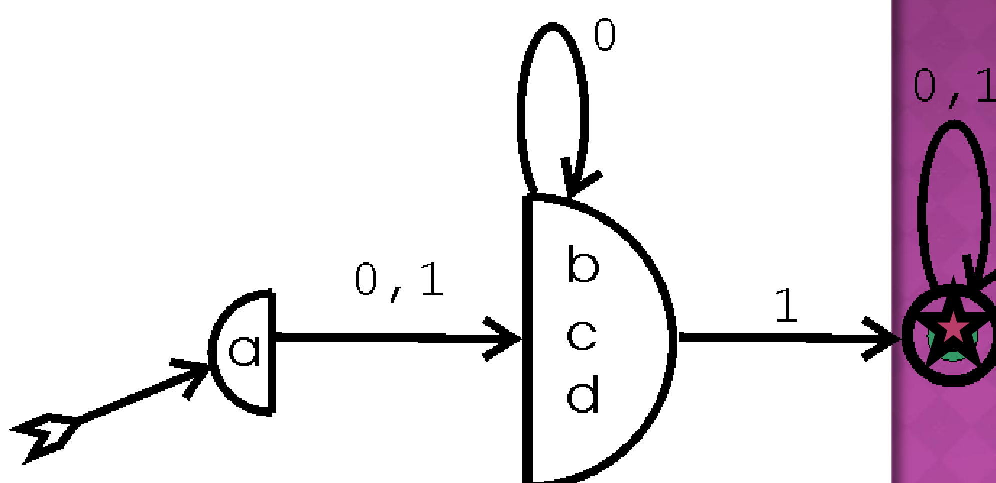


MINIMIZATION EXAMPLE. COMPARE

100100101

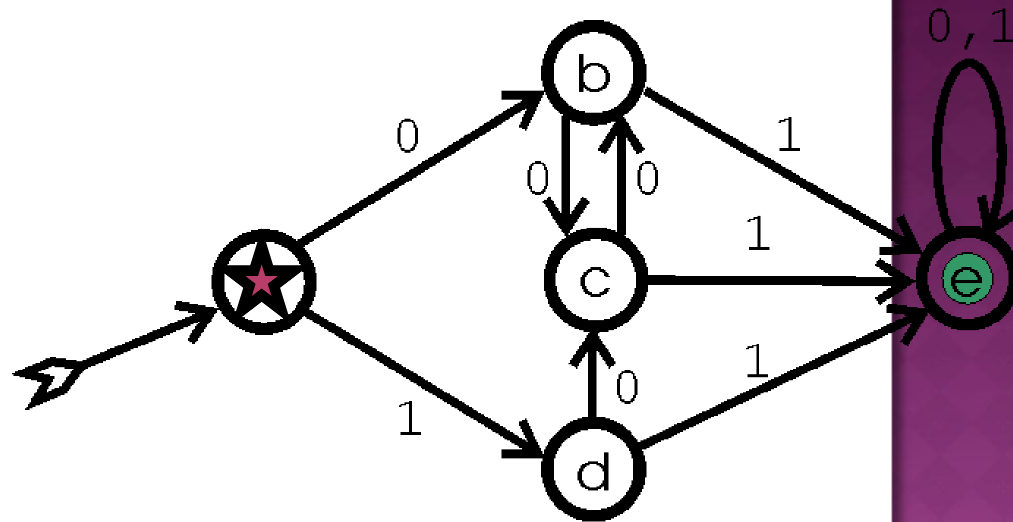
ACCEPTED.

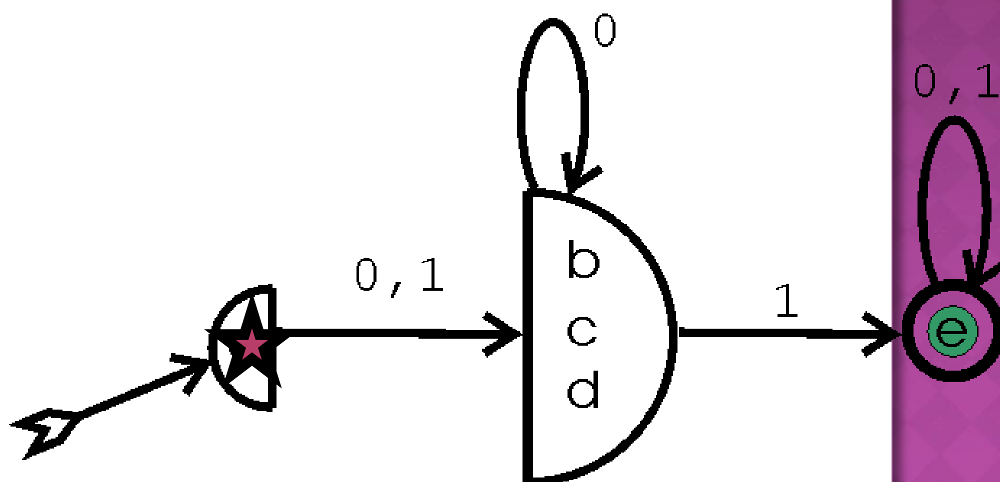




MINIMIZATION EXAMPLE. COMPARE

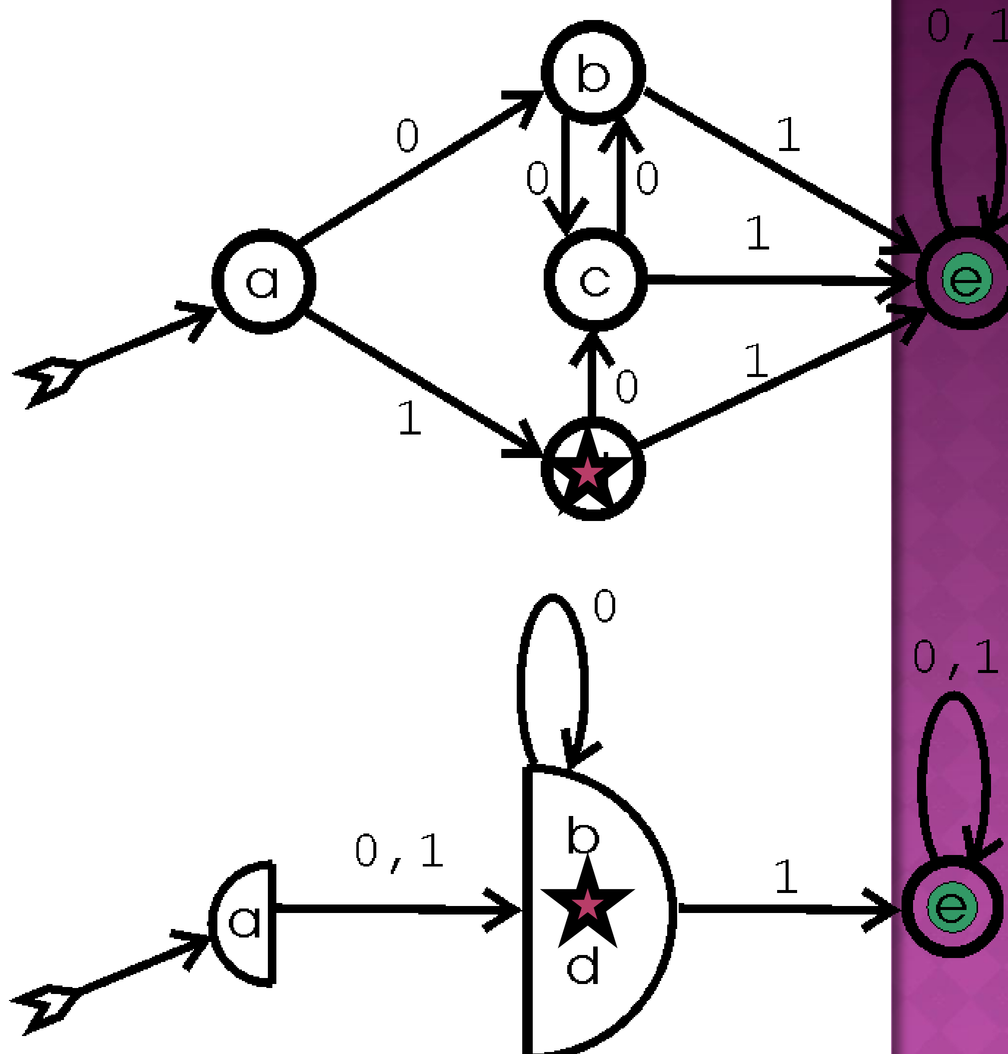
10000





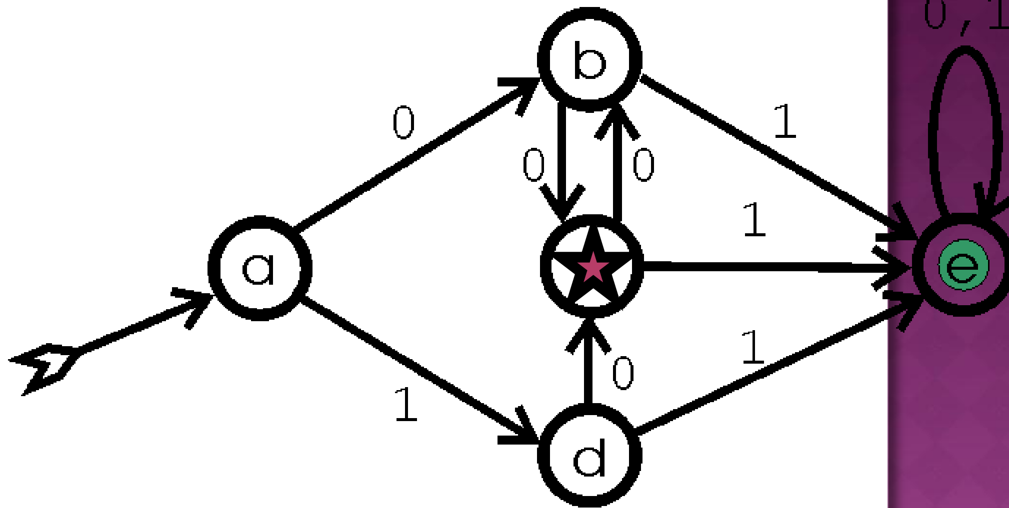
MINIMIZATION EXAMPLE. COMPARE

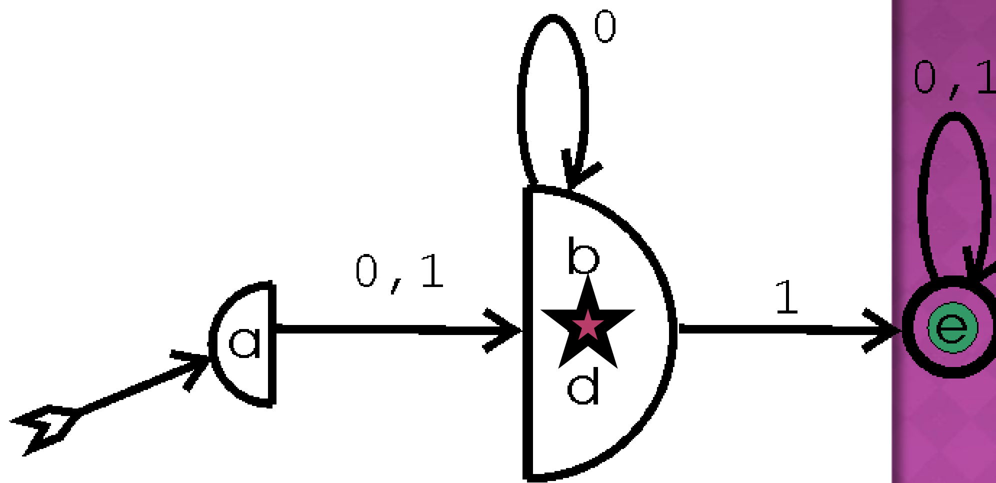
10000



MINIMIZATION EXAMPLE. COMPARE

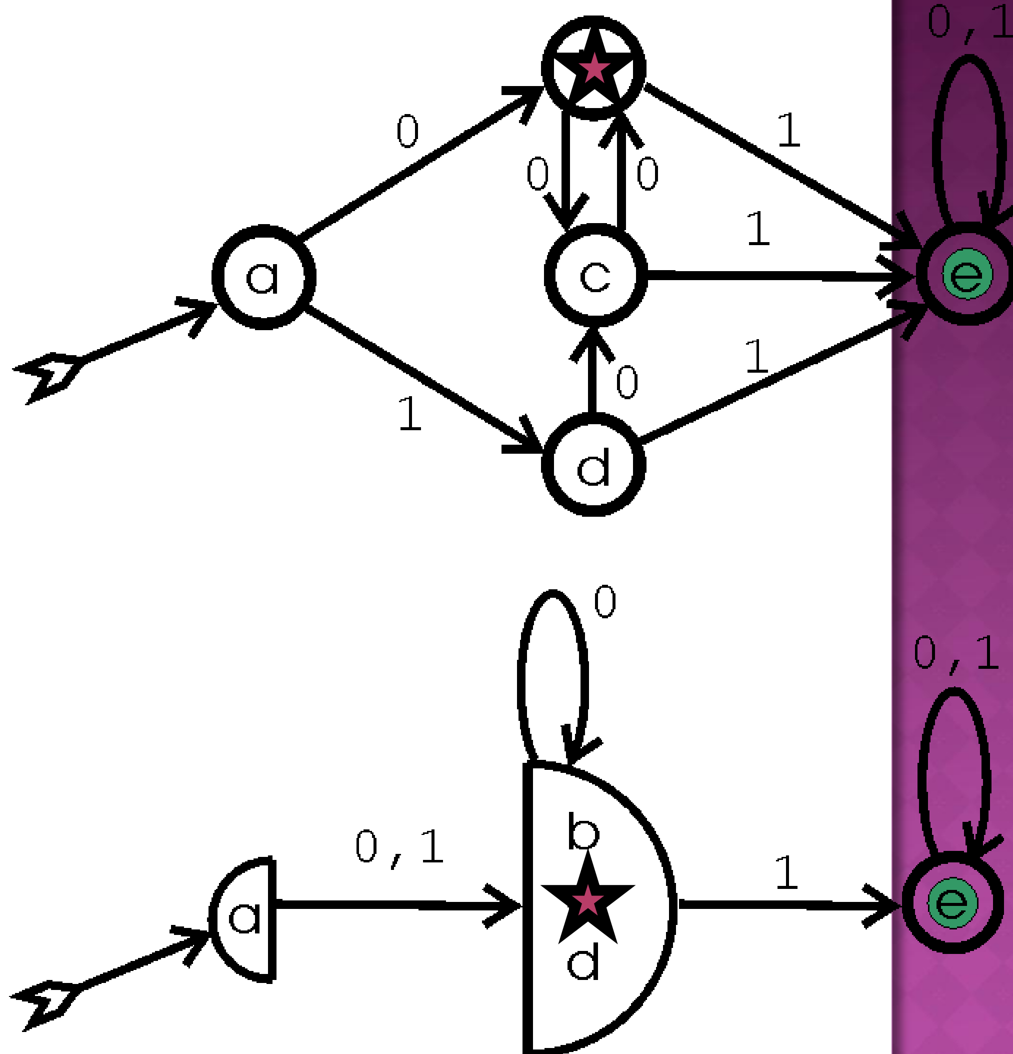
10000





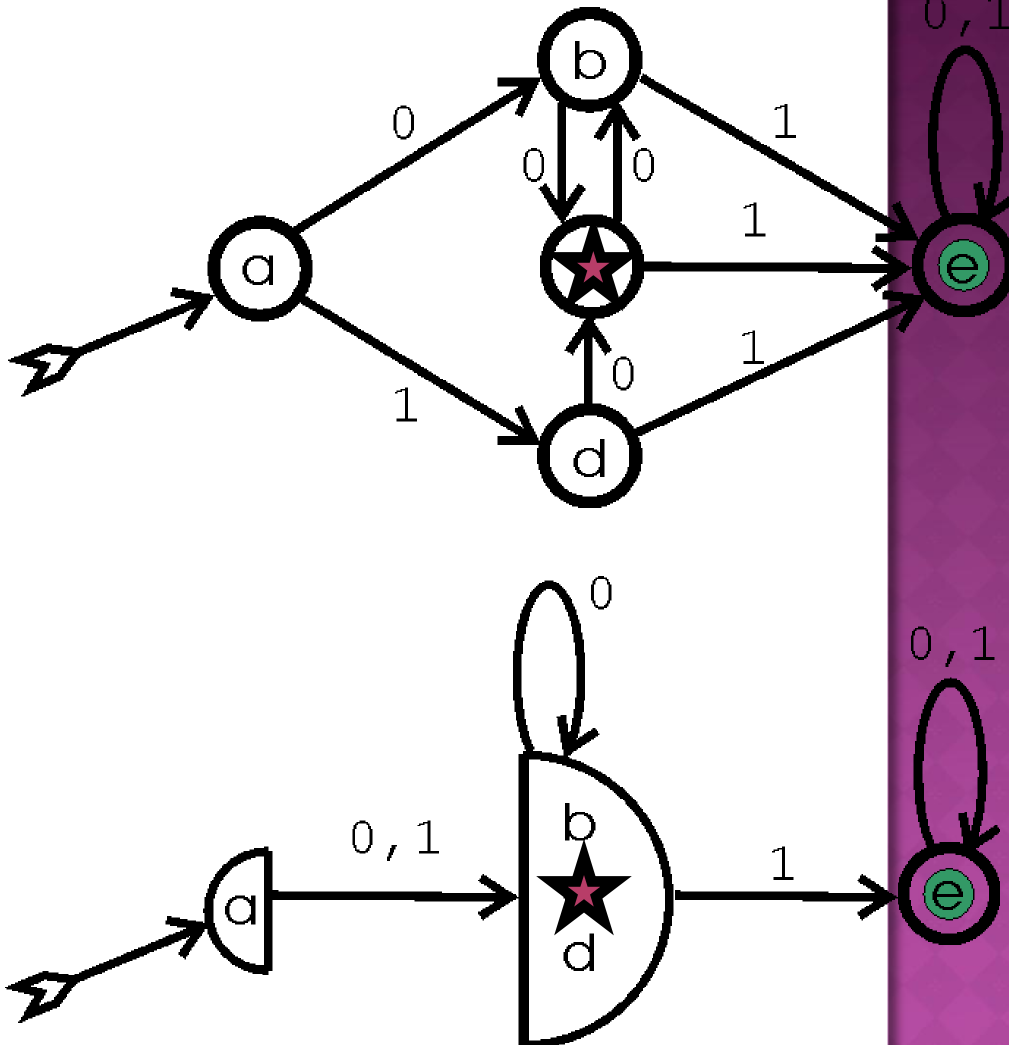
MINIMIZATION EXAMPLE. COMPARE

10000



MINIMIZATION EXAMPLE. COMPARE

10000



MINIMIZATION EXAMPLE. COMPARE

10000

REJECT.