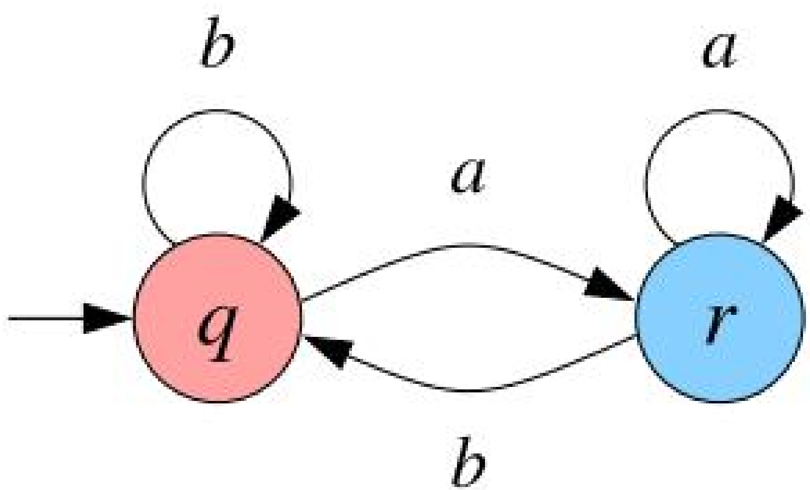
Implementing boolean operations for generalized Büchi automata

Generalized Büchi Automata

- An acceptance condition is a generalized Büchi condition if there are sets $F_1, \dots, F_k \subseteq Q$ of accepting states such that a run ρ is accepting iff it visits each of $F_1, \dots, F_k$ infinitely often.



$$F_1 = \{q\}$$
$$F_2 = \{r\}$$

From NGAs to NBAs

- Important fact:

All the sets $F_1, \ldots, F_k$ are visited infinitely often

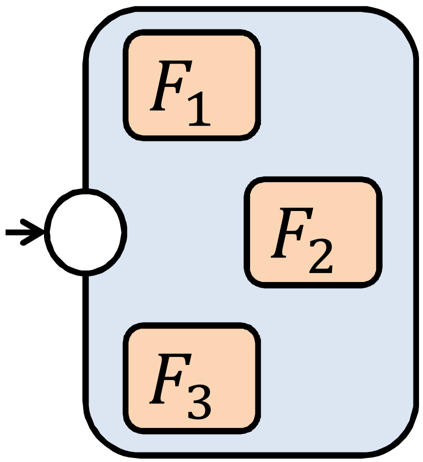
is equivalent to

F_1 is eventually visited

and for every $1 \leq i \leq k$

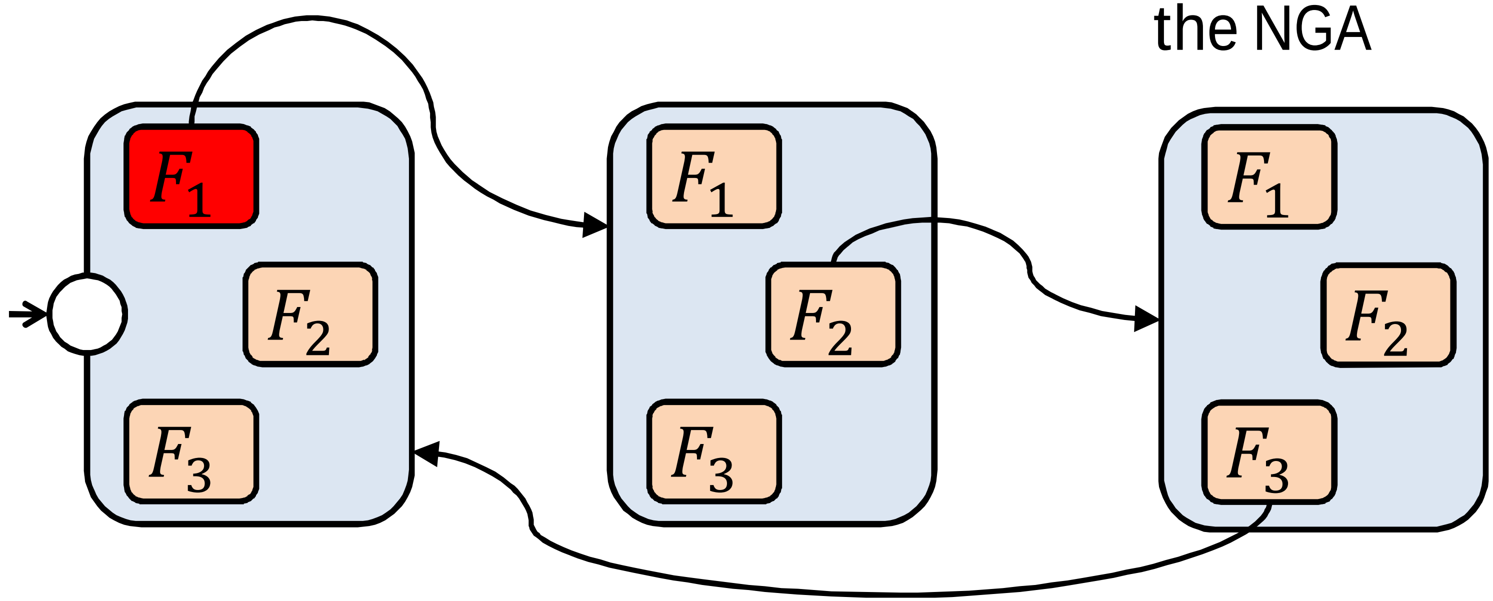
every visit to F_i is eventually followed by a visit to "$F_{i \oplus 1}$"

From NGAs to NBAs



NGA with 3 sets of accepting states

Equivalent NBA with 3 copies of the NGA



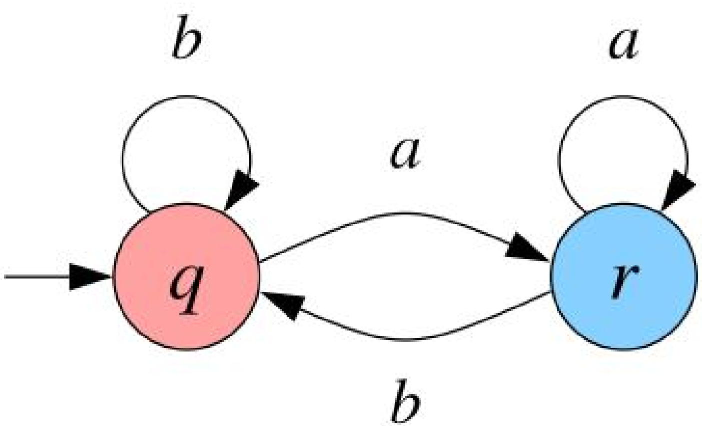
NGAtoNBA(A)

Input: NGA $A = (Q, \Sigma, Q_0, \delta, \mathcal{F})$, where $\mathcal{F} = \{F_0, \ldots, F_{m-1}\}$

Output: NBA $A' = (Q', \Sigma, \delta', Q'_0, F')$

$Q', \delta', F' \leftarrow \emptyset$; $Q'_0 \leftarrow \{[q_0, 0] \mid q_0 \in Q_0\}$
$W \leftarrow Q'_0$
while $W \neq \emptyset$ **do**
    pick $[q, i]$ **from** W
    add $[q, i]$ **to** Q'
    if $q \in F_0$ **and** $i = 0$ **then add** $[q, i]$ **to** F'
    for all $a \in \Sigma, q' \in \delta(q, a)$ **do**
        if $q \notin F_i$ **then**
            if $[q', i] \notin Q'$ **then add** $[q', i]$ **to** W
            add $([q, i], a, [q', i])$ **to** δ'
        else /* $q \in F_i$ */
            if $[q', i \oplus 1] \notin Q'$ **then add** $[q', i \oplus 1]$ **to** W
            add $([q, i], a, [q', i \oplus 1])$ **to** δ'
return $(Q', \Sigma, \delta', Q'_0, F')$

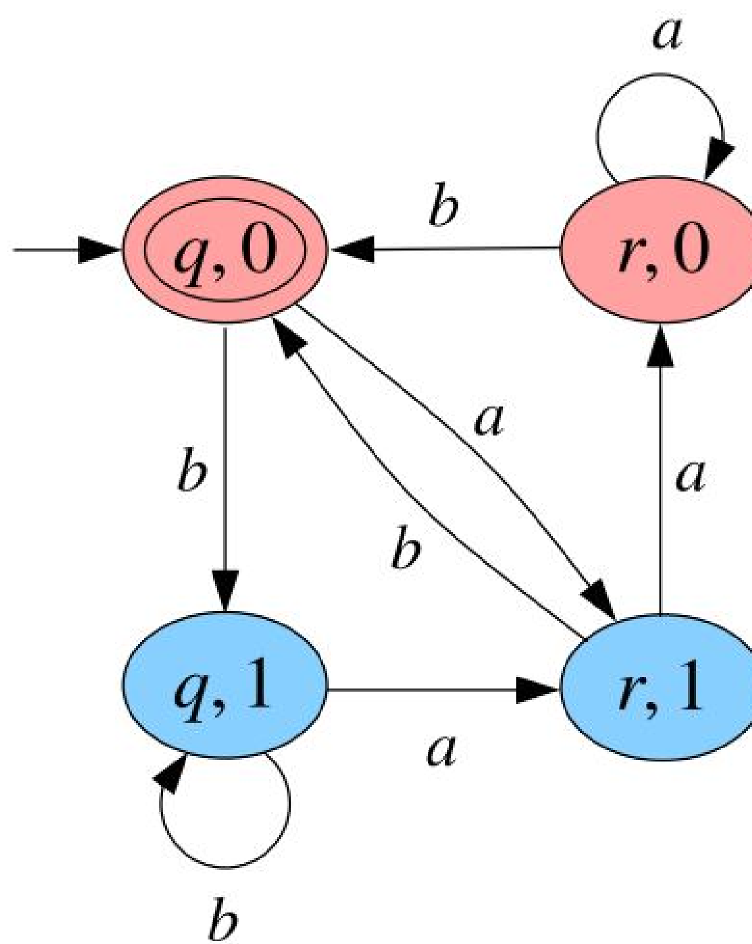
NGA



$F_1 = \{q\}$

$F_2 = \{r\}$

NBA



Union of NGA: The NBA case

- Let $A_1 = (S_1, \{F_1\})$ and $A_2 = (S_2, \{F_2\})$
- Let S be the result of putting S_1 and S_2 „side by side“

$$S \coloneqq (Q_1 \cup Q_2, \Sigma, \delta_1 \cup \delta_2, Q_{01} \cup Q_{02})$$

- Which NGA recognizes $L(A_1) \cup L(A_2)$?
  - $(S, \{F_1 \cup F_2\})$
  - $(S, \{F_1, F_2\})$

Union of NGA: Another case

- Let $A_1 = (S_1, \{ F_1^1, F_1^2\})$ and $A_2 = (S_2, \{ F_2^1, F_2^2\})$
- Let S be the result of putting S_1 and S_2 „side by side“

$$S := (Q_1 \cup Q_2, \Sigma, \delta_1 \cup \delta_2, Q_{01} \cup Q_{02})$$

- Which NGA recognizes $L(A_1) \cup L(A_2)$?
  - $(S, \{F_1^1 \cup F_2^1 \cup F_1^2 \cup F_2^2\})$
  - $(S, \{F_1^1 \cup F_2^1, F_1^2 \cup F_2^2\})$
  - $(S, \{F_1^1 \cup F_2^1, F_1^1 \cup F_2^2, F_1^2 \cup F_2^1, F_1^2 \cup F_2^2\})$

Union of NGA: The general case

- Let $A_1 = \left(S_1, \{F_1^1, \dots, F_1^k\}\right)$

  $A_2 = \left(S_2, \{F_2^1, \dots, F_2^k, F_2^{k+1}, \dots, F_2^{k+l}\}\right)$

- Let S be the result of putting S_1 and S_2 „side by side“

$$S := (Q_1 \cup Q_2, \Sigma, \delta_1 \cup \delta_2, Q_{01} \cup Q_{02})$$

- The following NGA recognizes $L(A_1) \cup L(A_2)$

$$A = \left(S, \left\{ \begin{matrix} F_1^1 \\ \cup \\ F_2^1 \end{matrix}, \dots, \begin{matrix} F_1^k \\ \cup \\ F_2^k \end{matrix}, \begin{matrix} Q_1 \\ \cup \\ F_2^{k+1} \end{matrix}, \dots, \begin{matrix} Q_1 \\ \cup \\ F_2^{k+l} \end{matrix} \right\}\right)$$

Intersection of NGA: The NBA case

- Let $A_1 = (S_1, \{F_1\})$ and $A_2 = (S_2, \{F_2\})$
- Let S be the pairing of S_1 and S_2

$$S := (Q_1 \times Q_2, \Sigma, \delta, Q_{01} \times Q_{02})$$

where $\delta([q_1, q_2], a) = \delta(q_1, a) \times \delta(q_2, a)$

- Which NGA recognizes $L(A_1) \cap L(A_2)$?
  - $(S, \{F_1 \times F_2\})$
  - $(S, \{F_1 \times Q_2, Q_1 \times F_2\})$

Intersection of NGA: The general case

- Let $A_1 = \left(S_1, \{F_1^1, \ldots, F_1^k\}\right)$, $A_2 = \left(S_2, \{F_2^1, \ldots, F_1^l\}\right)$
- Let S be the pairing of S_1 and S_2

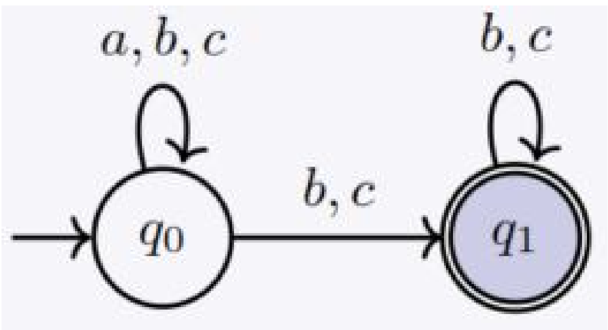
$$S \coloneqq (Q_1 \times Q_2, \Sigma, \delta, Q_{01} \times Q_{02})$$

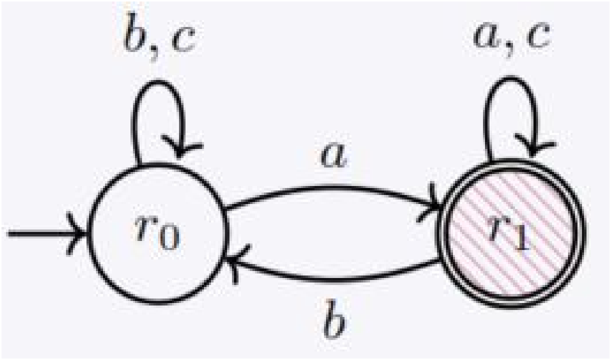
where $\delta([q_1, q_2], a) = \delta(q_1, a) \times \delta(q_2, a)$

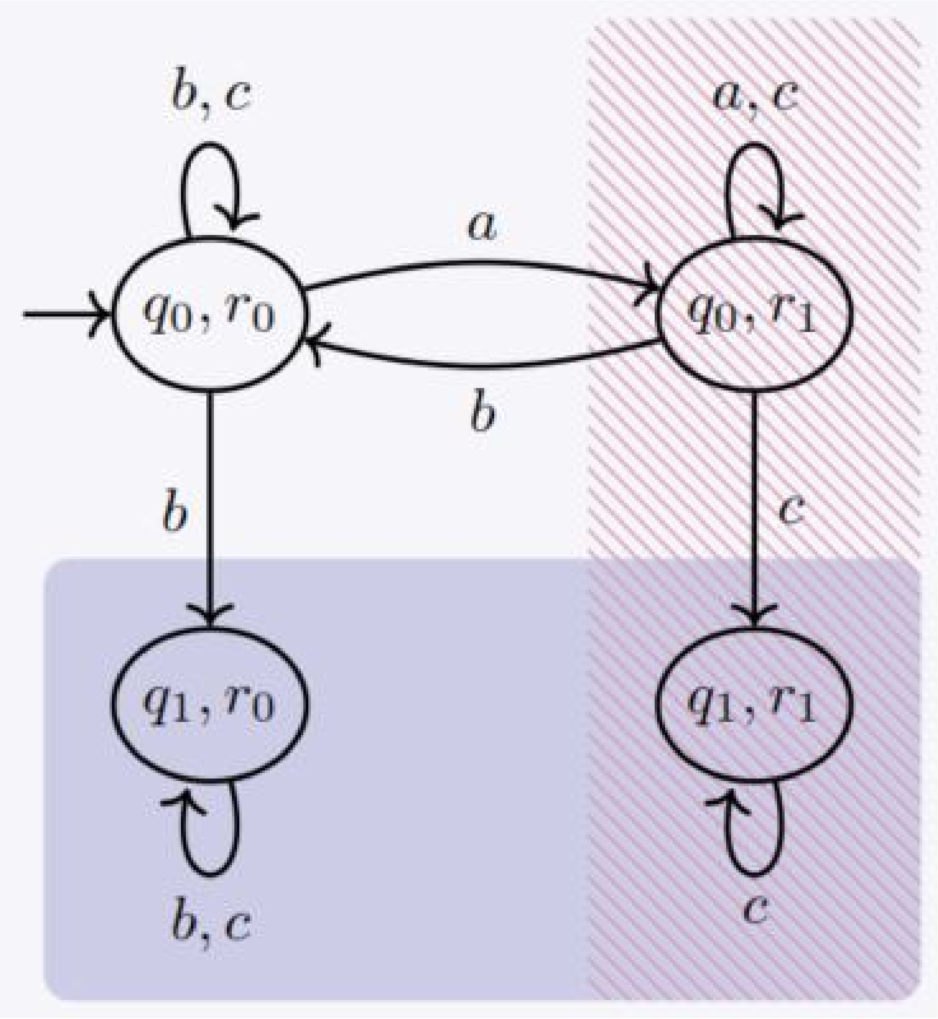
- The following NGA recognizes $L(A_1) \cap L(A_2)$:

$$\left(S, \underbrace{\{F_1^1 \times Q_2, \ldots, F_1^k \times Q_2, Q_1 \times F_2^1, \ldots, Q_1 \times F_2^l\}}_{k+l}\right)$$

Intersection of NGA: The general case







Special case

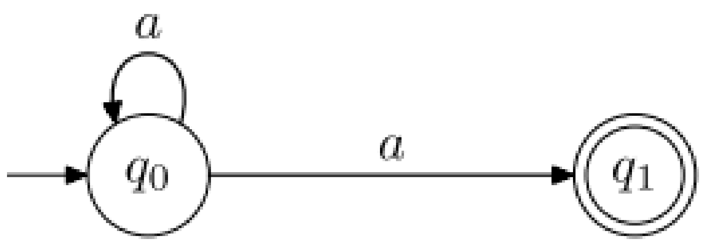
- The intersection of $(S_1, \{F_1\})$ and $(S_2, \{F_2\})$ is $([S_1, S_2], \{F_1 \times Q_2, Q_1 \times F_2\})$
- Not a NBA in general.
- However, if $F_1 = Q_1$ then $\{F_1 \times Q_2, Q_1 \times F_2\}$ can be replaced by $\{Q_1 \times F_2\}$, and so the result is again a NBA.

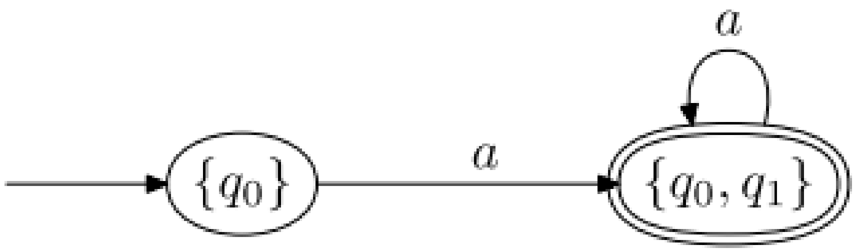
Complementation of NGA

- Given a NBA A, we construct a NBA B such that $L_\omega(B) = \overline{L_\omega(A)}$
- We can then complement a NGA by transforming it first into a NBA
- Complementation construction radically different from the one for NFAs.

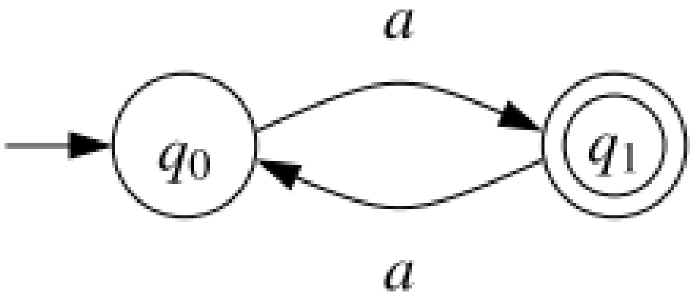
Problems

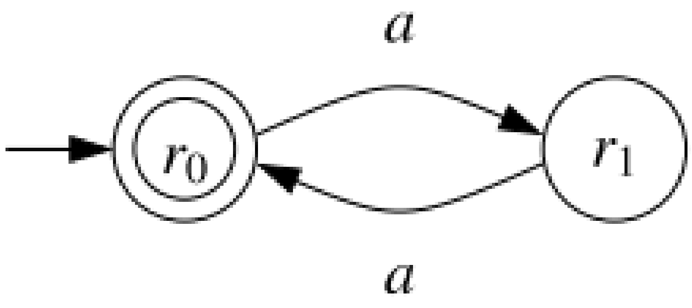
- The powerset construction does not work.





- Exchanging final and non-final states in DBAs also fails.



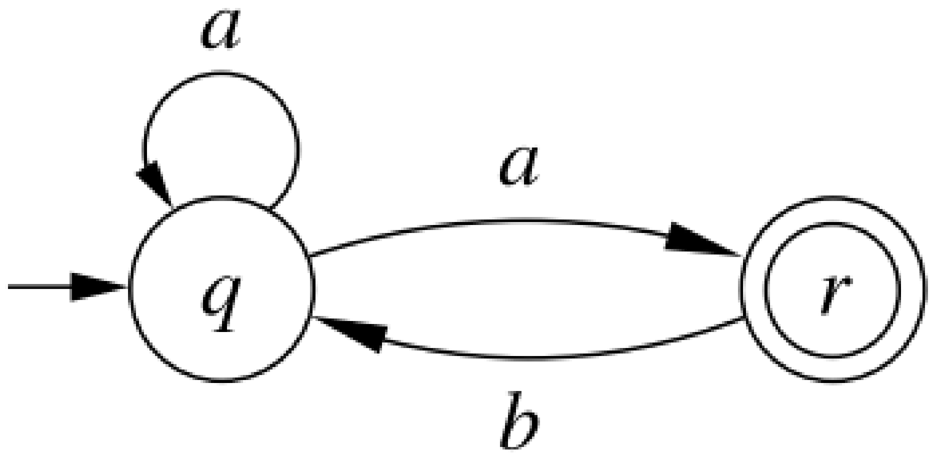


Solution

- Extend the idea used to determinize co-Büchi automata with a new component.
- Recall: a NBA accepts a word w iff some path of $dag(w)$ visits final states infinitely often.
- Goal: given NBA A, construct NBA $\bar{A}$ such that:

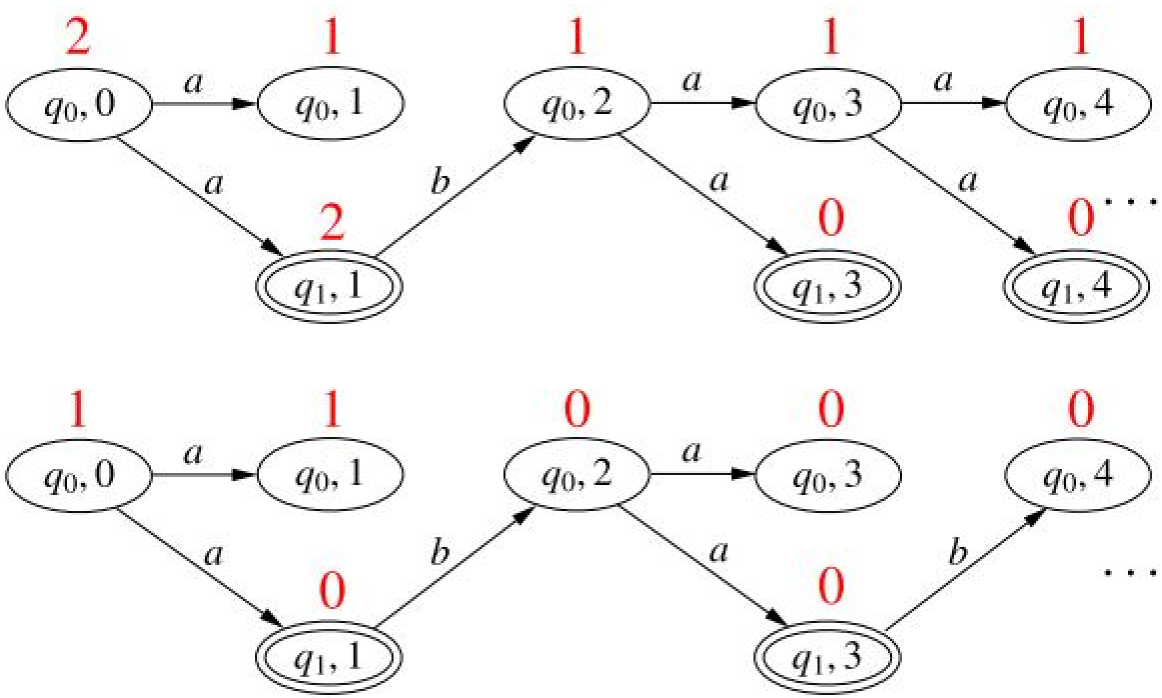
A rejects w
iff
no path of $dag(w)$ visits accepting states of A i.o.
iff
some run of $\bar{A}$ visits accepting states of $\bar{A}$ i.o.
iff
$\bar{A}$ accepts w

Running example



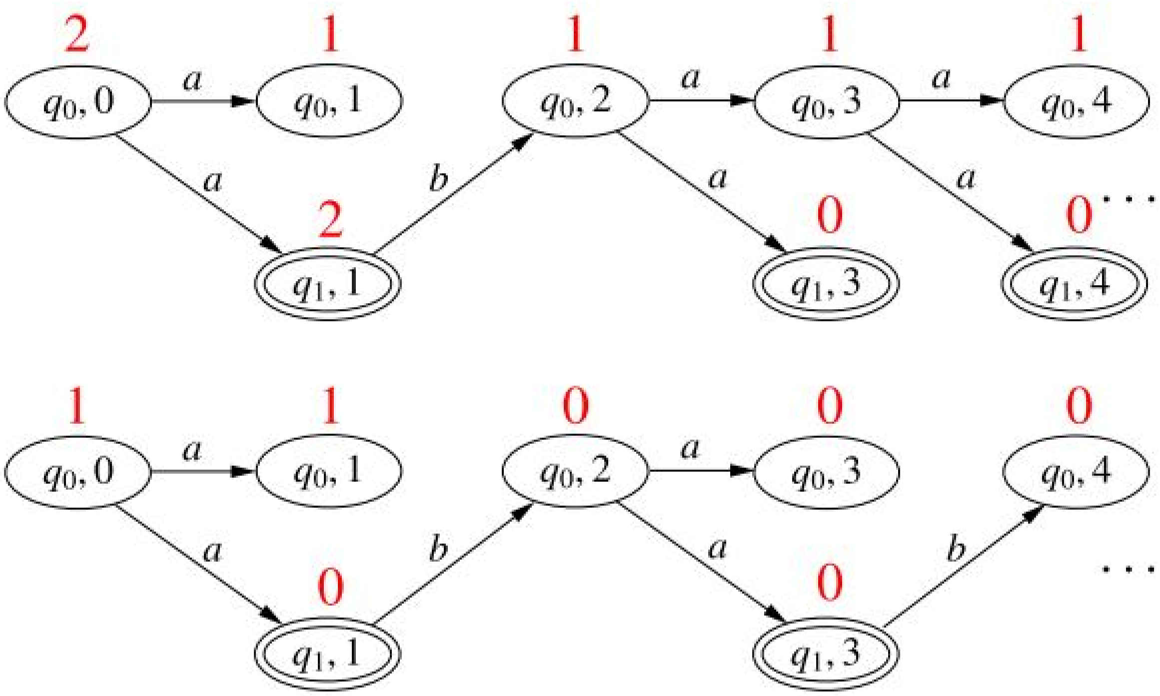
Rankings

- Mappings that associate to every node of $dag(w)$ a rank (a natural number) such that
  - ranks never increase along a path, and
  - ranks of accepting nodes are even.



Odd rankings

- A ranking is odd if every infinite path of $dag(w)$ visits nodes of odd rank i.o.



Odd rankings

Goal: given NBA A, construct NBA $\bar{A}$ such that:

A rejects w

iff

no path of $dag(w)$ visits accepting states of A i.o.

iff

$dag(w)$ has an odd ranking

iff

some run of $\bar{A}$ visits accepting states of $\bar{A}$ i.o.

iff

$\bar{A}$ accepts w

Odd rankings

Prop:

> no path of $dag(w)$ visits accepting states of A i.o.
> iff
> $dag(w)$ has an odd ranking

Further, all ranks of the odd ranking are in the range $[0,2n]$, and all states of the first level rank have rank $2n$.

Proof:

(⇐): In an odd ranking of $dag(w)$, ranks along infinite paths stabilize to odd values.

Therefore, since accepting nodes have even rank, no path of $dag(w)$ visits accepting nodes i.o.

Odd rankings

(⇒): Assume no path of $dag(w)$ visits accepting states of A i.o.

Define an odd ranking of $dag(w)$ as follows:

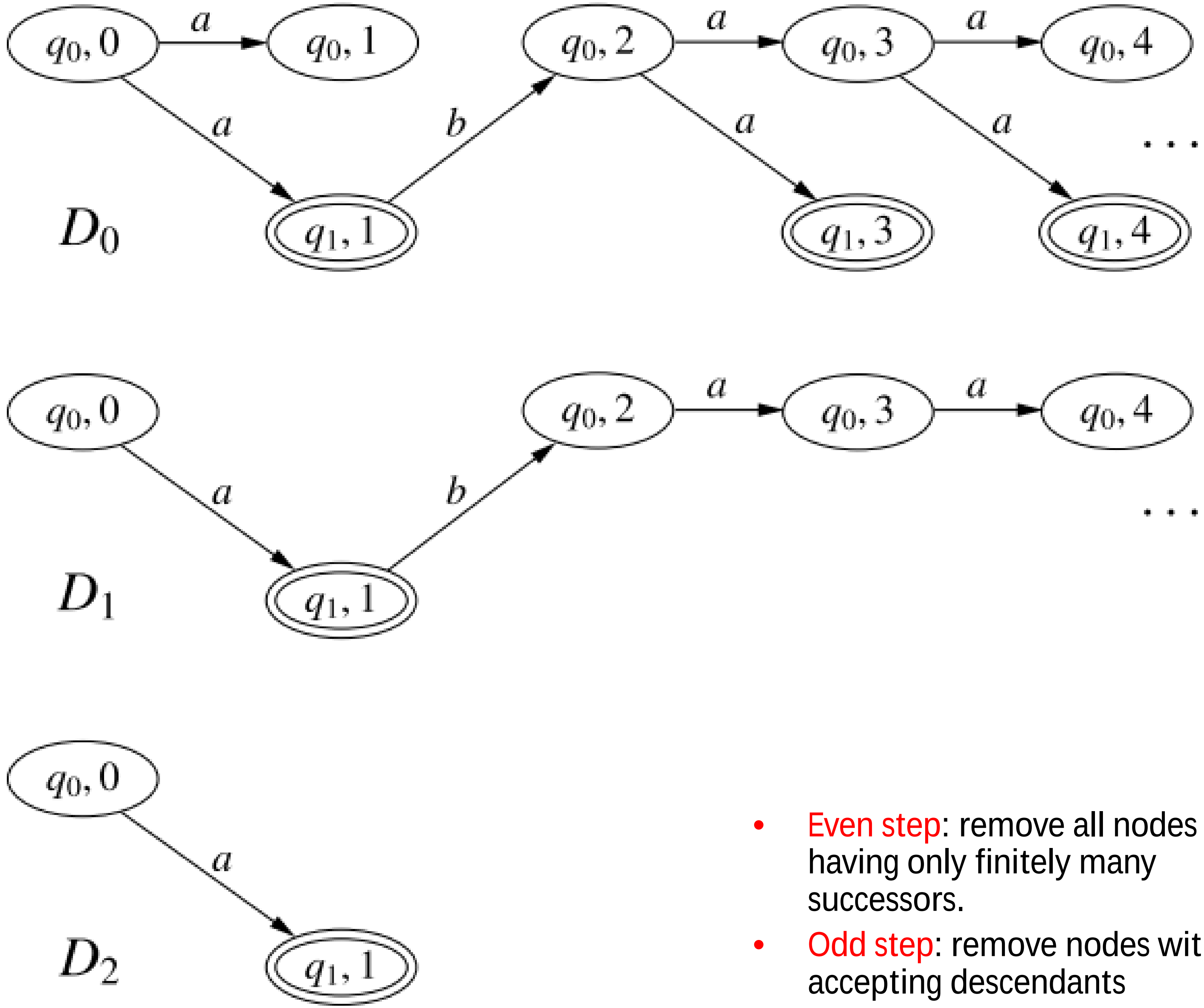
- Construct a sequence $D_0 \supseteq D_1 \supseteq D_2 \cdots \supseteq D_{2n} \supseteq D_{2n+1}$ of dags, where

a) $D_0 = dag(w)$

b) D_{2i+1} is the result of removing from D_{2i} all nodes with finitely many descendants.

c) D_{2i+2} is the result of removing all nodes of D_{2i+1} with no accepting descendants (a node is a descendant of itself).

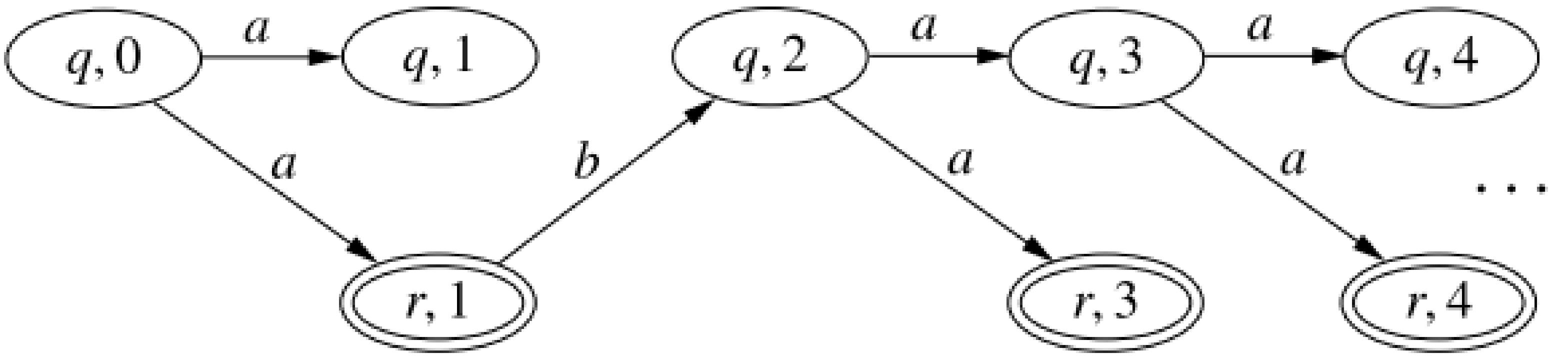
- We define the rank of a node of $dag(w)$ as the index of the unique dag D_j in the sequence such that the node belongs to D_j but not to D_{j+1}.



- Even step: remove all nodes having only finitely many successors.
- Odd step: remove nodes with no accepting descendants

- This definition of rank guarantees :
  1. Ranks along a path cannot increase.
  2. Accepting states get even ranks, because they can only be removed from dags with even index.
- It remains to prove:
  – every node gets a rank, i.e., $D_{2n+1} = \emptyset$.
- A round consists of two steps, an even step from D_{2i} to D_{2i+1}, and an odd step from D_{2i+1} to D_{2i+2}.

- Each level of a dag has a width



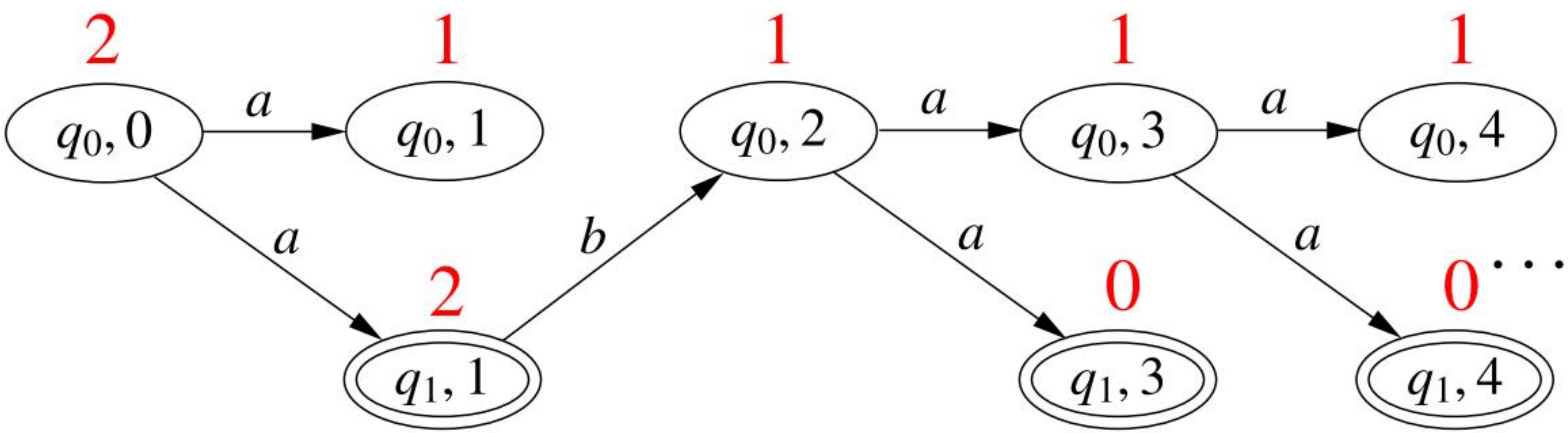
- We define the width of a dag as the largest level width that appears infinitely often.
- Each round decreases the width of the dag by at least 1.
- Since the initial width is at most n, after at most n rounds the width is 0, and then a last step removes all nodes.

- Goal:

$dag(w)$ has an odd ranking
iff
some run of $\bar{A}$ visits accepting states of $\bar{A}$ i.o.

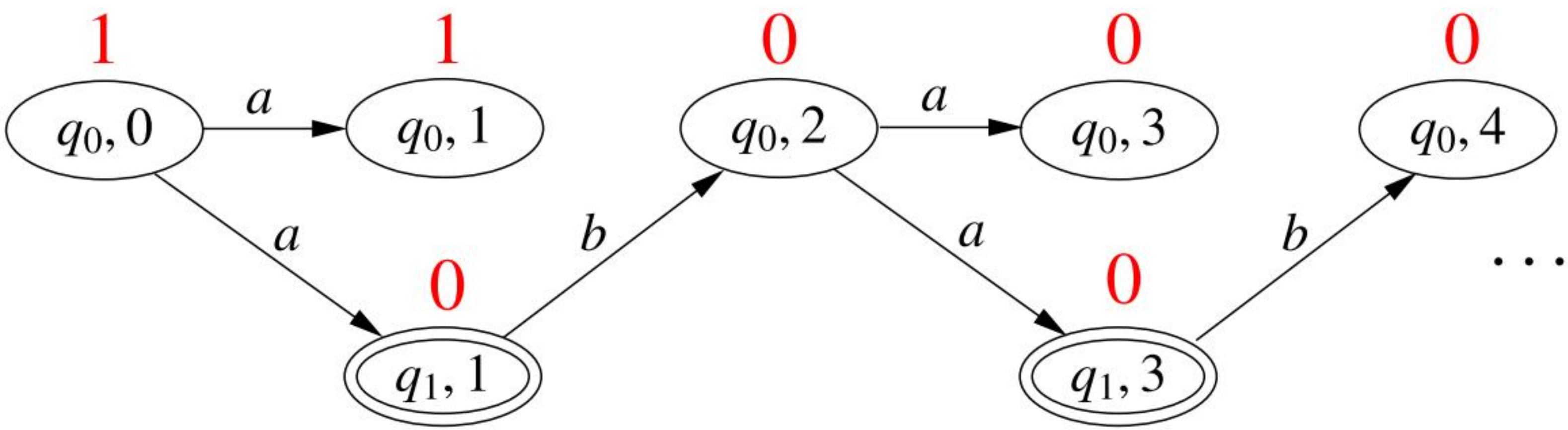
- Idea: design $\bar{A}$ so that
  - its runs on w are the rankings of $dag(w)$, and
  - its accepting runs on w are the odd rankings of $dag(w)$.

Representing rankings



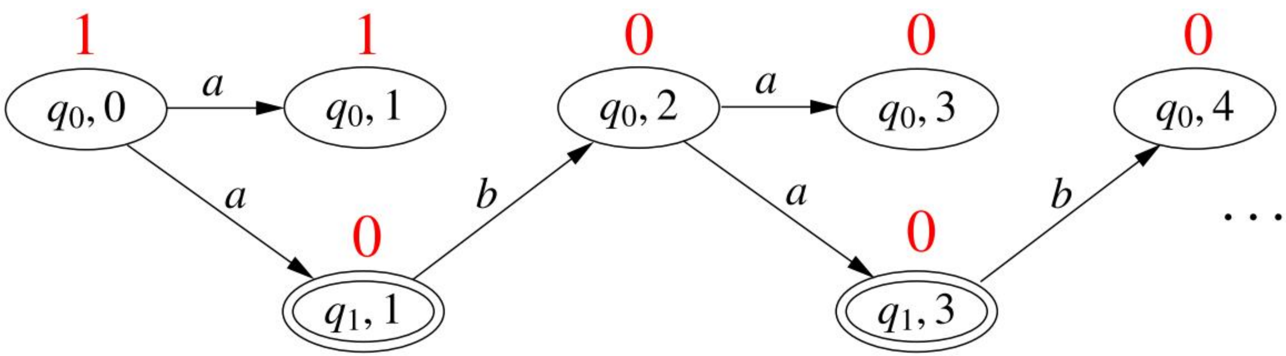
$$\begin{bmatrix}2\\ \bot\end{bmatrix} \xrightarrow{a} \begin{bmatrix}1\\ 2\end{bmatrix} \xrightarrow{b} \begin{bmatrix}1\\ \bot\end{bmatrix} \xrightarrow{a} \begin{bmatrix}1\\ 0\end{bmatrix} \xrightarrow{a} \begin{bmatrix}1\\ 0\end{bmatrix} \cdots$$

Representing rankings



$$\begin{bmatrix}1\\\bot\end{bmatrix}\xrightarrow{a}\begin{bmatrix}1\\0\end{bmatrix}\xrightarrow{b}\begin{bmatrix}0\\\bot\end{bmatrix}\xrightarrow{a}\begin{bmatrix}0\\0\end{bmatrix}\xrightarrow{b}\begin{bmatrix}0\\\bot\end{bmatrix}\cdots$$

Representing rankings



$$\begin{bmatrix}1\\ \bot\end{bmatrix} \xrightarrow{a} \begin{bmatrix}1\\ 0\end{bmatrix} \xrightarrow{b} \begin{bmatrix}0\\ \bot\end{bmatrix} \xrightarrow{a} \begin{bmatrix}0\\ 0\end{bmatrix} \xrightarrow{b} \begin{bmatrix}0\\ \bot\end{bmatrix} \cdots$$

We can determine if $\begin{bmatrix}n_1\\ n_2\end{bmatrix} \xrightarrow{l} \begin{bmatrix}n_1'\\ n_2'\end{bmatrix}$ may appear in a ranking by just looking at n_1, n_2, n_1', n_2' and l : ranks should not increase.

First draft for $\bar{A}$

- $\bar{A}$ for or a two-state A (more states analogous):
  - States: all $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ where $0 \leq x_i \leq 2n = 4$ or $x_i = \perp$ and accepting states of A get even rank or $\perp$.
  - Initial state: all states of the form $\begin{bmatrix} n_1 \\ \perp \end{bmatrix}$
  - Transitions: all $\begin{bmatrix} n_1 \\ n_2 \end{bmatrix} \xrightarrow{a} \begin{bmatrix} n_1' \\ n_2' \end{bmatrix}$ s.t . ranks do not increase
- The runs of the automaton on a word w correspond to all the rankings of $dag(w)$.
- Observe: $\bar{A}$ is a NBA even if A is a DBA, because there are many rankings for the same word.

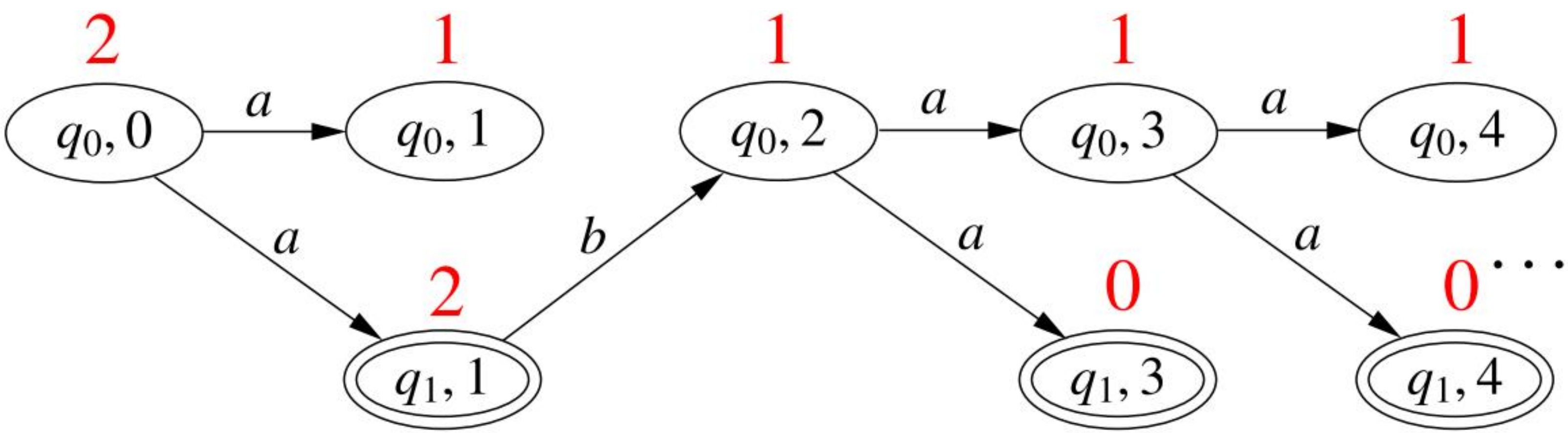
Accepting states?

- The accepting states should be chosen so that a run is accepted iff its corresponding ranking is odd.
- Problem: no way to do so when the only information of a state is the ranking.

Owing states and breakpoints

- We use owing states and breakpoints again:
  - A breakpoint of a ranking is now a level of the ranking such that no node of the level owes a visit to a node of odd rank.
  - We have again: a ranking is odd iff it has infinitely many breakpoints.
  - We enrich the states of $\bar{A}$ with a set of owing states, and choose the accepting states as those in which the set is empty.

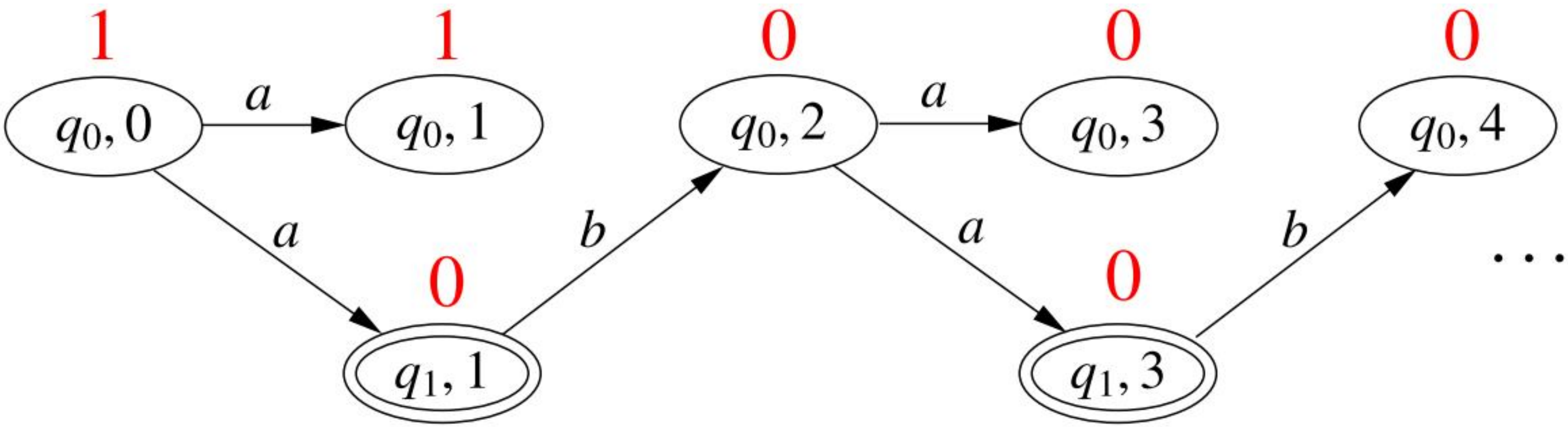
Owing states



$$\begin{bmatrix}2\\ \bot\end{bmatrix} \xrightarrow{a} \begin{bmatrix}1\\ 2\end{bmatrix} \xrightarrow{b} \begin{bmatrix}1\\ \bot\end{bmatrix} \xrightarrow{a} \begin{bmatrix}1\\ 0\end{bmatrix} \xrightarrow{a} \begin{bmatrix}1\\ 0\end{bmatrix} \cdots$$

$\emptyset$ $\{q_1\}$ $\emptyset$ $\{q_1\}$ $\emptyset$

Owing states



$$\begin{bmatrix} 1 \\ \bot \end{bmatrix} \xrightarrow{a} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \xrightarrow{b} \begin{bmatrix} 0 \\ \bot \end{bmatrix} \xrightarrow{a} \begin{bmatrix} 0 \\ 0 \end{bmatrix} \xrightarrow{b} \begin{bmatrix} 0 \\ \bot \end{bmatrix} \cdots$$

$\emptyset$ $\{q_1\}$ $\{q_0\}$ $\{q_0, q_1\}$ $\{q_0\}$

Second draft for $\bar{A}$

- For our two-state A (the case of more states is analogous):
  - States: pairs $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, O$ where $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ as in the first draft, and O is a set of owing states (of even rank)
  - Initial states: all states of the form $\begin{bmatrix} x_1 \\ \bot \end{bmatrix}, \emptyset$
  - Transitions: all $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, O \xrightarrow{a} \begin{bmatrix} x_1' \\ x_2' \end{bmatrix}, O'$ s.t. ranks don't increase and owing states are correctly updated
  - Final states: all states $\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \emptyset$

Second draft for $\bar{A}$

- The runs of $\bar{A}$ on a word w correspond to all the rankings of $dag(w)$.
- The accepting runs of $\bar{A}$ on a word w correspond to all the odd rankings of $dag(w)$.
- Therefore: $L(\bar{A}) = \overline{L(A)}$

Final $\bar{A}$ (the final touch ...)

- We can reduce the number of initial states.
- For every ranking with ranks in the range $[0,2n]$, changing the rank of all nodes of the first level to $2n$ yields again a ranking. Further, if the old ranking is odd then the new ranking is also odd.
  So we can simplify the definition of the initial states to:
  - Initial state: $\begin{bmatrix} 2n \\ \bot \end{bmatrix}, \emptyset$

An example

- We construct the complements of

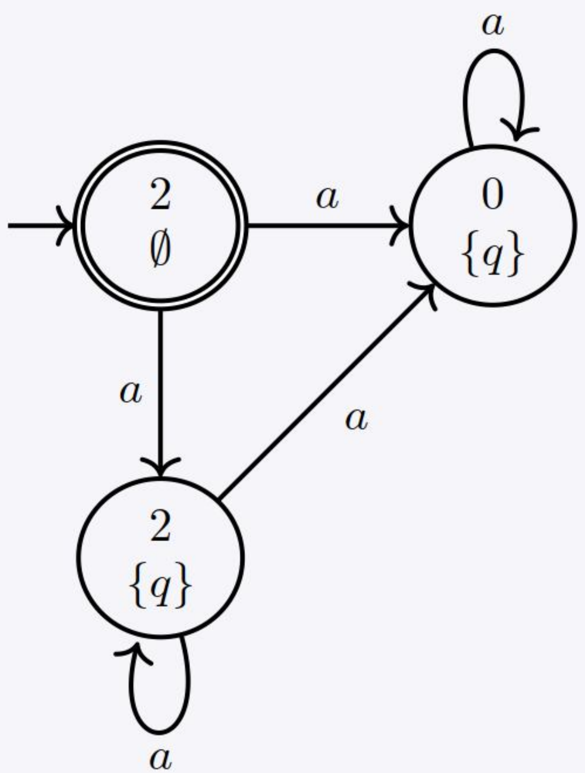
  $A_1 = (\{q\}, \{a\}, \delta, \{q\}, \{q\})$ with $\delta(q, a) = \{q\}$

  $A_2 = (\{q\}, \{a\}, \delta, \{q\}, \emptyset)$ with $\delta(q, a) = \{q\}$

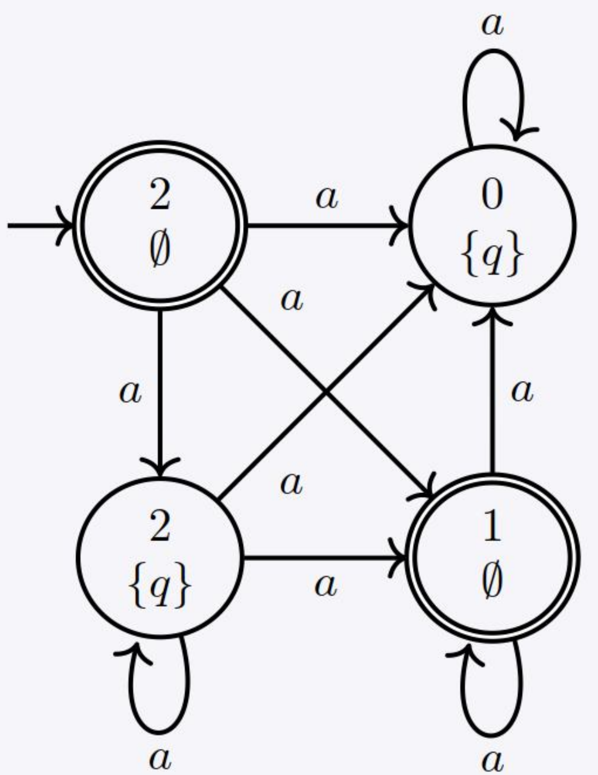
- States of $\bar{A}_1$: $\langle 0, \emptyset\rangle, \langle 2, \emptyset\rangle, \langle 0, \{q\}\rangle, \langle 2, \{q\}\rangle$
- States of $\bar{A}_2$: $\langle 0, \emptyset\rangle, \langle 1, \emptyset\rangle, \langle 2, \emptyset\rangle, \langle 0, \{q\}\rangle, \langle 2, \{q\}\rangle$
- Initial state of $\bar{A}_1$ and $\bar{A}_2$: $\langle 2, \emptyset\rangle$
- Final states of $\bar{A}_1$: $\langle 2, \emptyset\rangle$, $\langle 0, \emptyset\rangle$ (unreachable)
- Final states of $\bar{A}_2$: $\langle 2, \emptyset\rangle$, $\langle 1, \emptyset\rangle$, $\langle 0, \emptyset\rangle$ (unreachable)

An example

$\overline{A_1}$



$\overline{A_2}$



Complexity

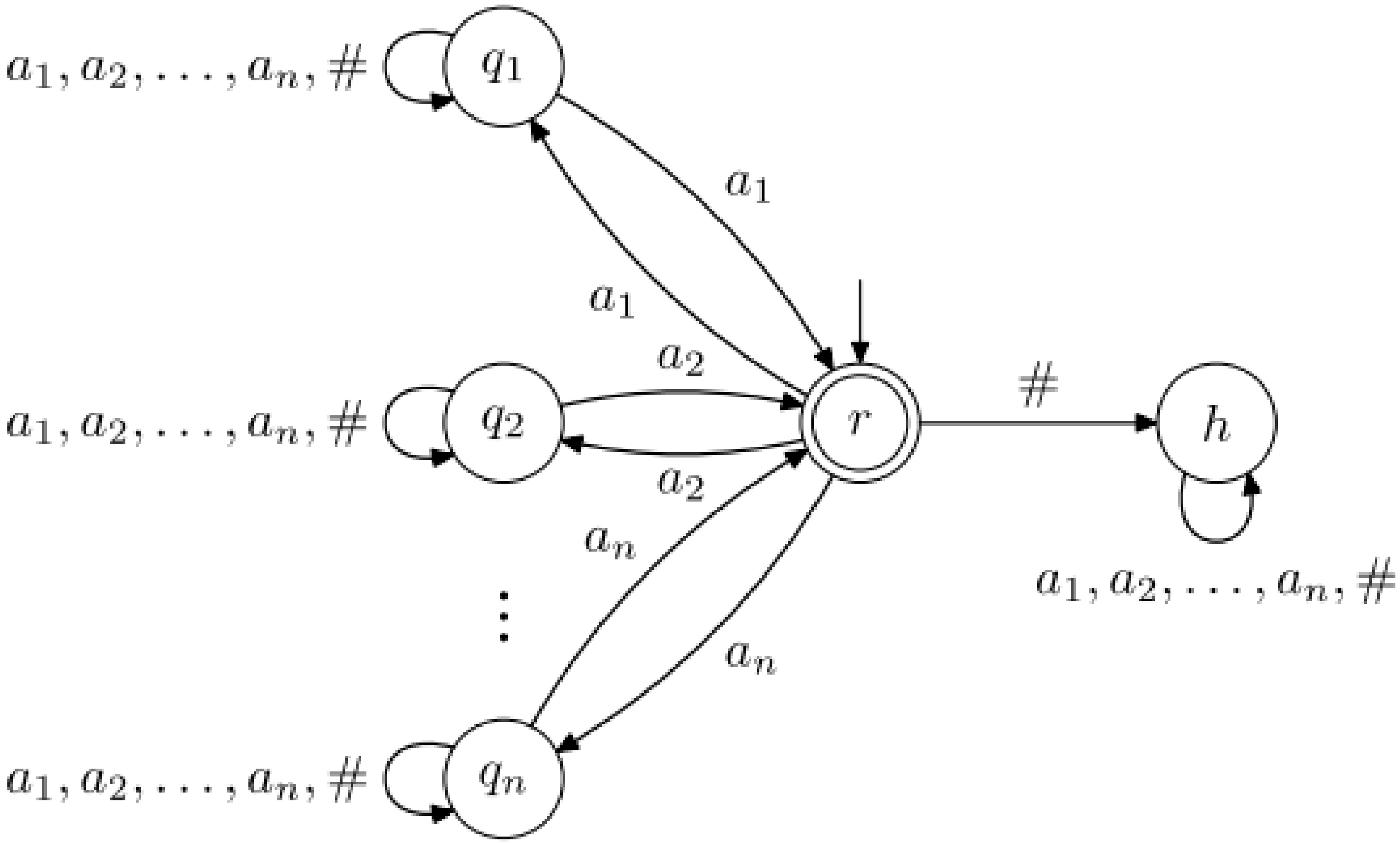
- A state consists of a level of a ranking and a set of owing states.
- A level assigns to each state a number of $[0,2n]$ or the symbol $\bot$.
- So the complement NBA has at most $(2n+2)^n \cdot 2^n \in n^{O(n)} = 2^{O(n \log n)}$ states.
- Compare with 2^n for the NFA case.
- We show that the $\log n$ factor is unavoidable.

We define a family $\{L_n\}_{n\geq 1}$ of ω-languages s.t.

- L_n is accepted by a NBA with $n+2$ states.
- Every NBA accepting $\overline{L_n}$ has at least $n! \in 2^{\Theta(n \log n)}$ states.

- The alphabet of L_n is $\Sigma_n = \{1,2,\ldots,n,\#\}$.
- Assign to a word $w \in \Sigma_n$ a graph $G(w)$ as follows:
  - Vertices: the numbers $1,2,\ldots,n$.
  - Edges: there is an edge $i \rightarrow j$ iff w contains infinitely many occurrences of ij.
- Define: $w \in L_n$ iff $G(w)$ has a cycle.

- L_n is accepted by a NBA with $n + 2$ states.



Every NBA accepting $\overline{L_n}$ has at least $n! \in 2^{\Theta(n \log n)}$ states.

- Let τ denote a permutation of $1, 2, \ldots, n$.
- We have:
  a) For every τ, the word $(\tau\,\#)^{\omega}$ belongs to $\overline{L_n}$ (i.e., its graph contains no cycle).
  b) For every two distinct τ_1, τ_2, every word containing inf. many occurrences of τ_1 and inf. many occurrences of τ_2 belongs to L_n.

Every NBA accepting $\overline{L_n}$ has at least $n! \in 2^{\Theta(n \log n)}$ states.

- Assume A recognizes $\overline{L_n}$ and let τ_1, τ_2 distinct. By (a), A has runs ρ_1, ρ_2 accepting $(\tau_1 \#)^\omega$, $(\tau_2 \#)^\omega$. The sets of accepting states visited i.o. by ρ_1, ρ_2 are disjoint.
  - Otherwise we can ``interleave‘‘ ρ_1, ρ_2 to yield an acepting run for a word with inf. many occurrences of τ_1, τ_2 , contradicting (b).
- So A has at least one accepting state for each permutation, and so at least $n!$ states.