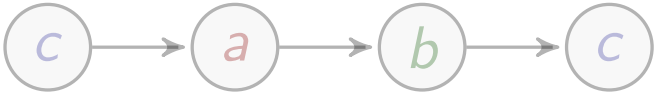
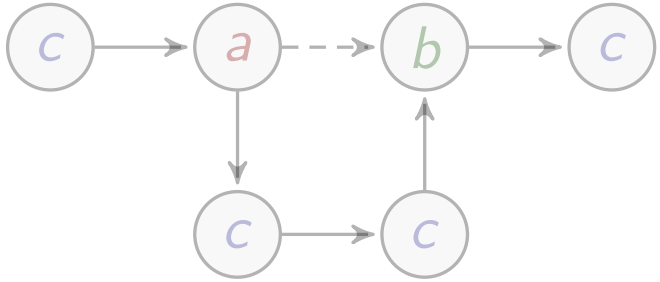


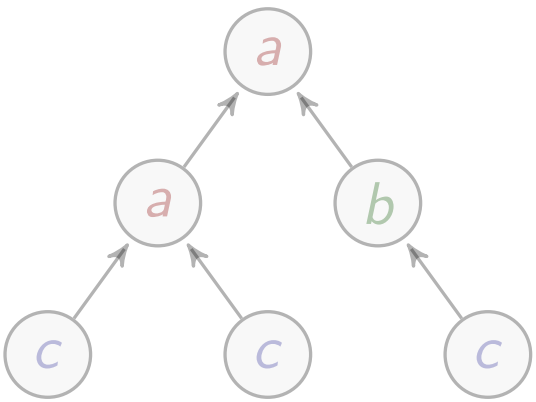


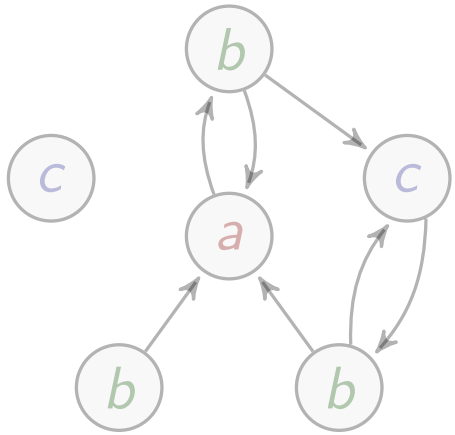
Distributed Graph Automata

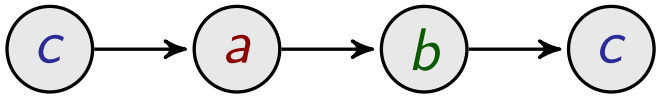
Fabian Reiter

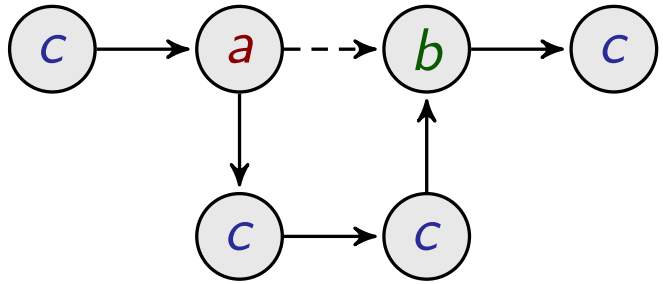
LIAFA, Université Paris Diderot

July 6, LICS 2015







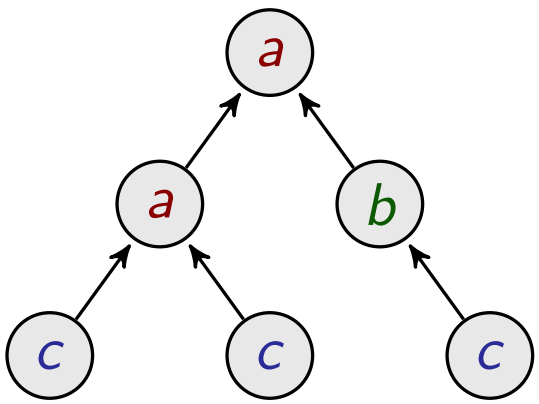


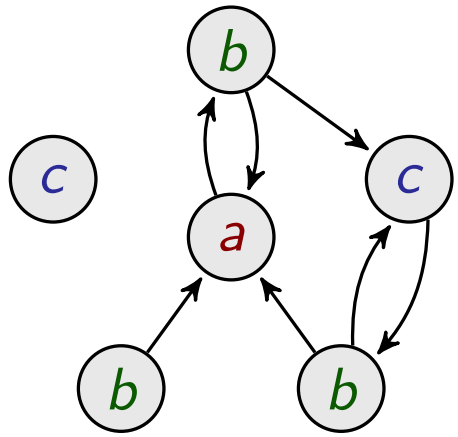
Distributed Graph Automata

Fabian Reiter

LIAFA, Université Paris Diderot

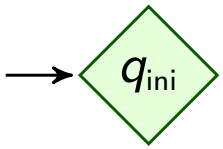
July 6, LICS 2015



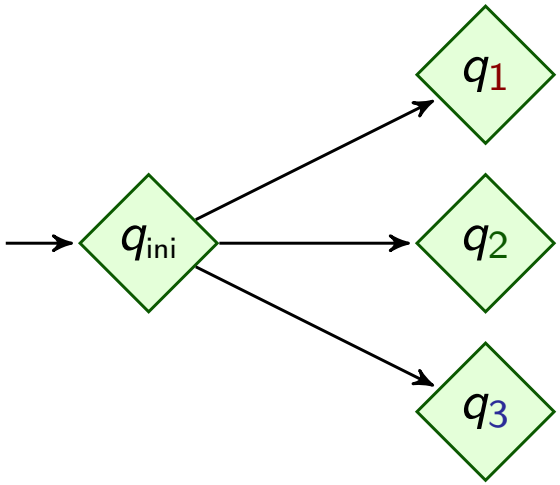


Automaton for 3-Colorability

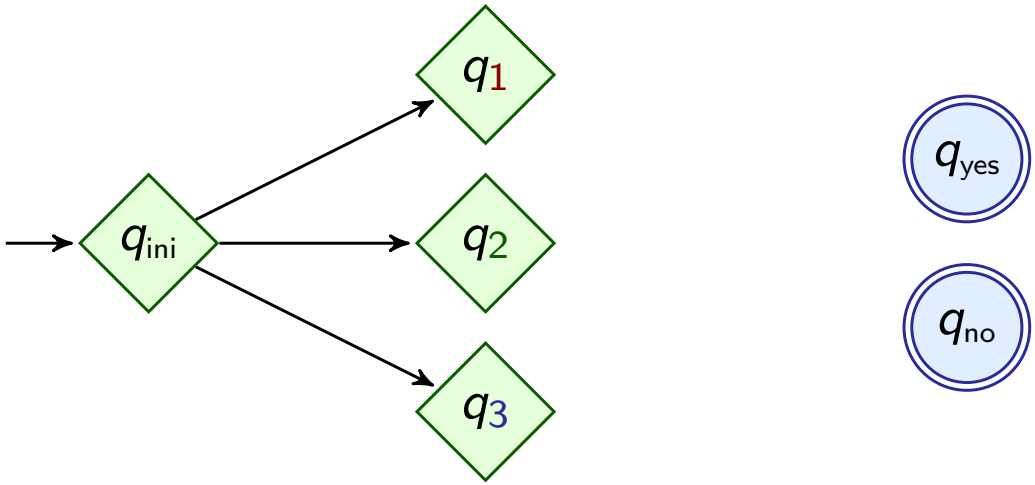
Automaton for 3-Colorability



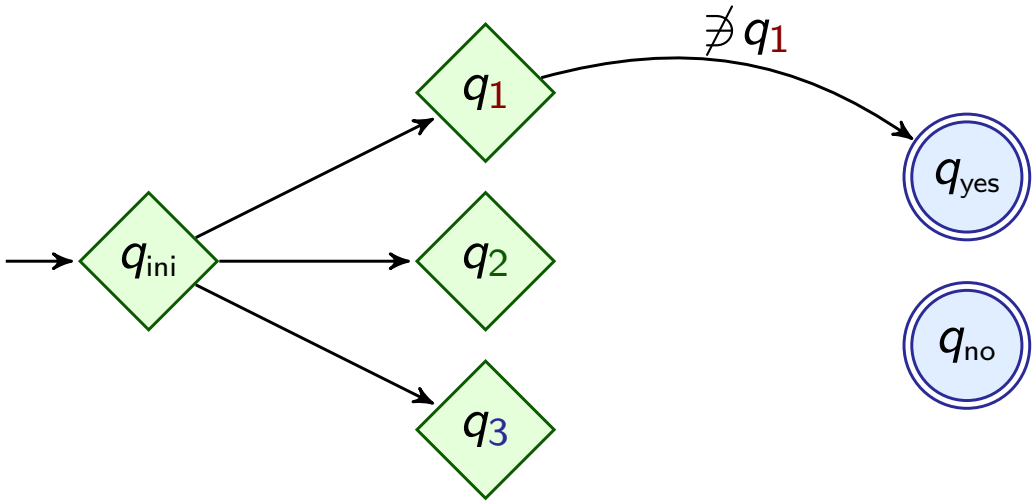
Automaton for 3-Colorability



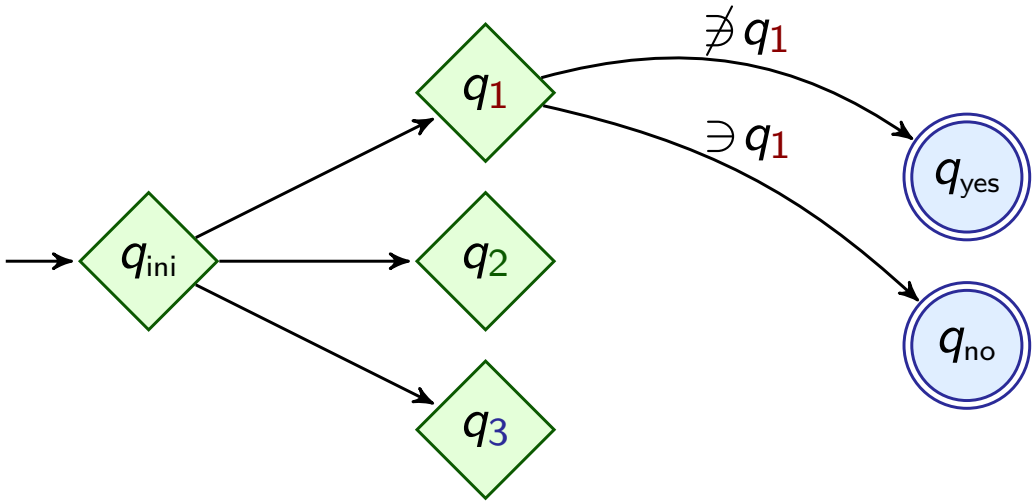
Automaton for 3-Colorability



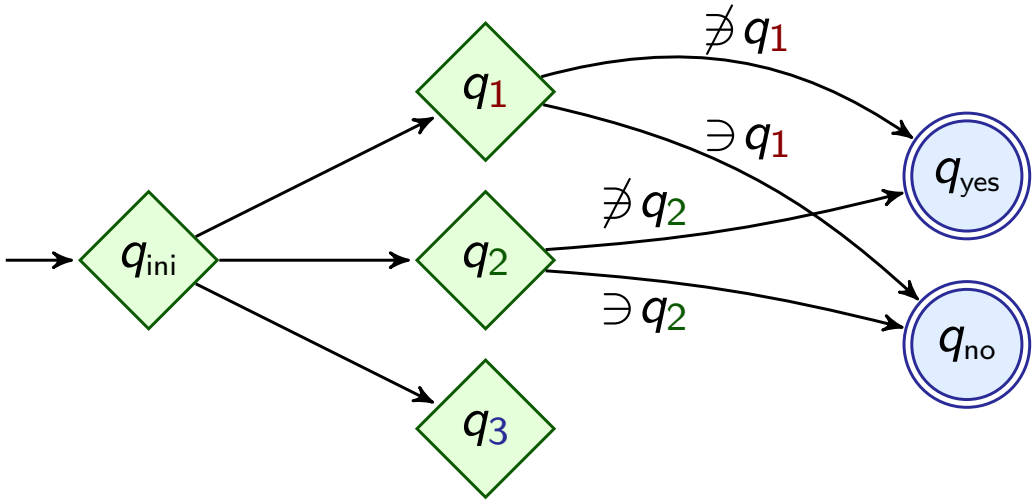
Automaton for 3-Colorability



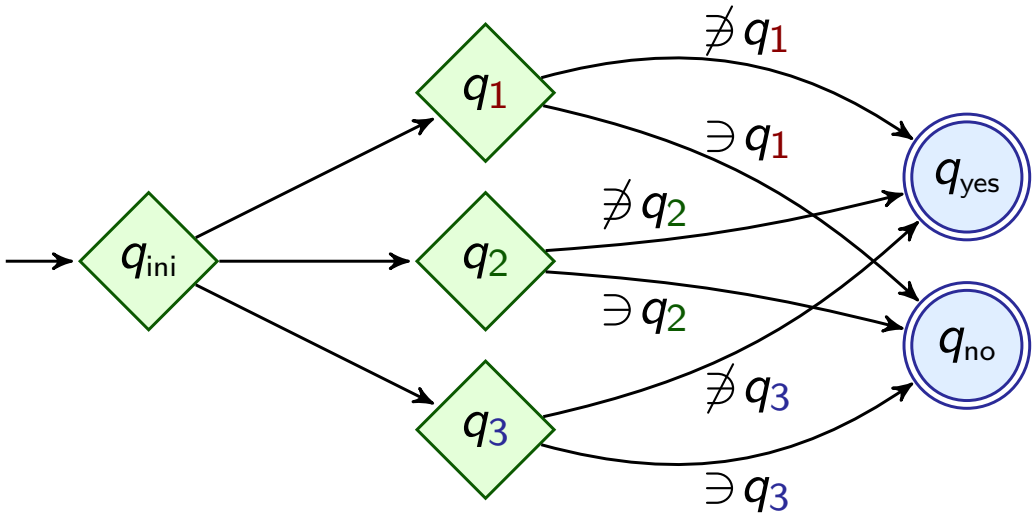
Automaton for 3-Colorability



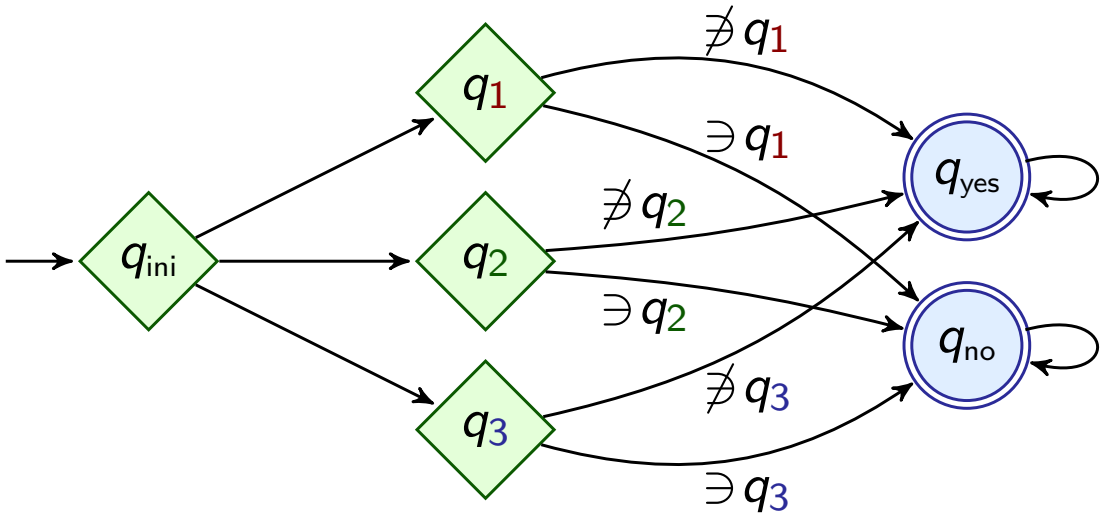
Automaton for 3-Colorability



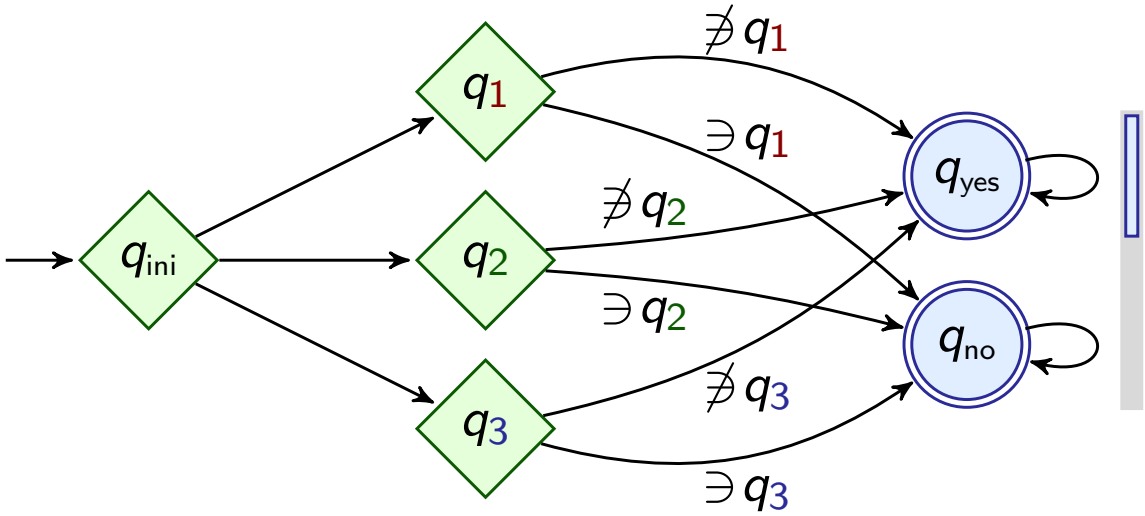
Automaton for 3-Colorability



Automaton for 3-Colorability



Automaton for 3-Colorability



ADGA – Ingredients

ADGA – Ingredients

- Synchronous distributed algorithm

ADGA – Ingredients

- Synchronous distributed algorithm
- Finite-state machines on the nodes

ADGA – Ingredients

- Synchronous distributed algorithm
- Finite-state machines on the nodes
- Constant running time

ADGA – Ingredients

- Synchronous distributed algorithm
- Finite-state machines on the nodes
- Constant running time
- Aggregation of states into sets

ADGA – Ingredients

- Synchronous distributed algorithm
- Finite-state machines on the nodes
- Constant running time
- Aggregation of states into sets
- Alternating automaton

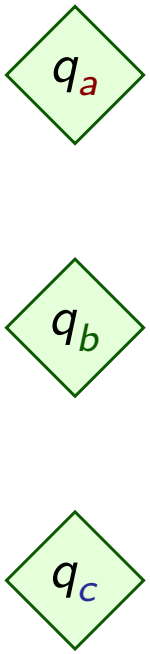
ADGA – Ingredients

- Synchronous distributed algorithm
- Finite-state machines on the nodes
- Constant running time
- Aggregation of states into sets
- Alternating automaton

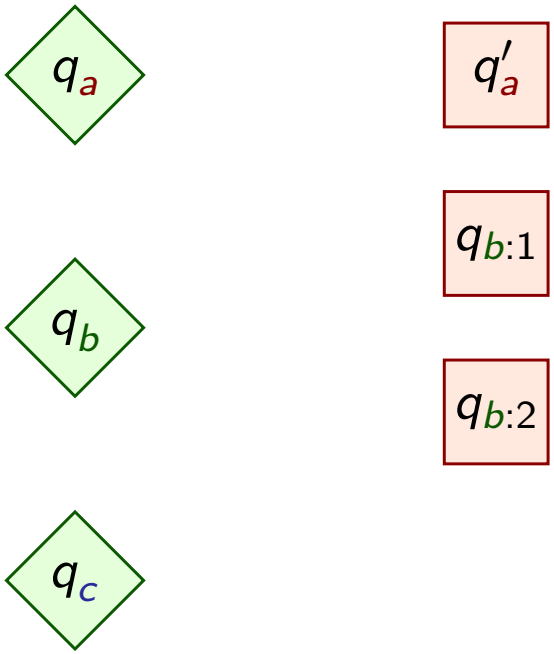
ADGA: Alternating Distributed Graph Automaton

ADGA – Definition by Example

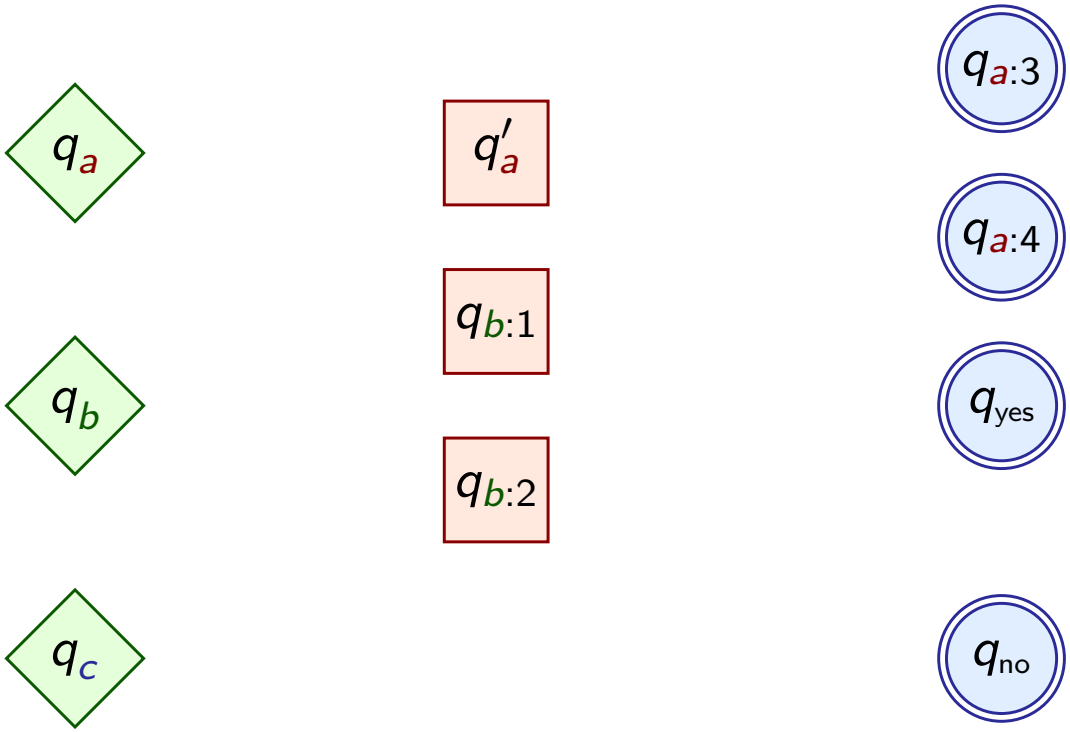
ADGA – Definition by Example



ADGA – Definition by Example

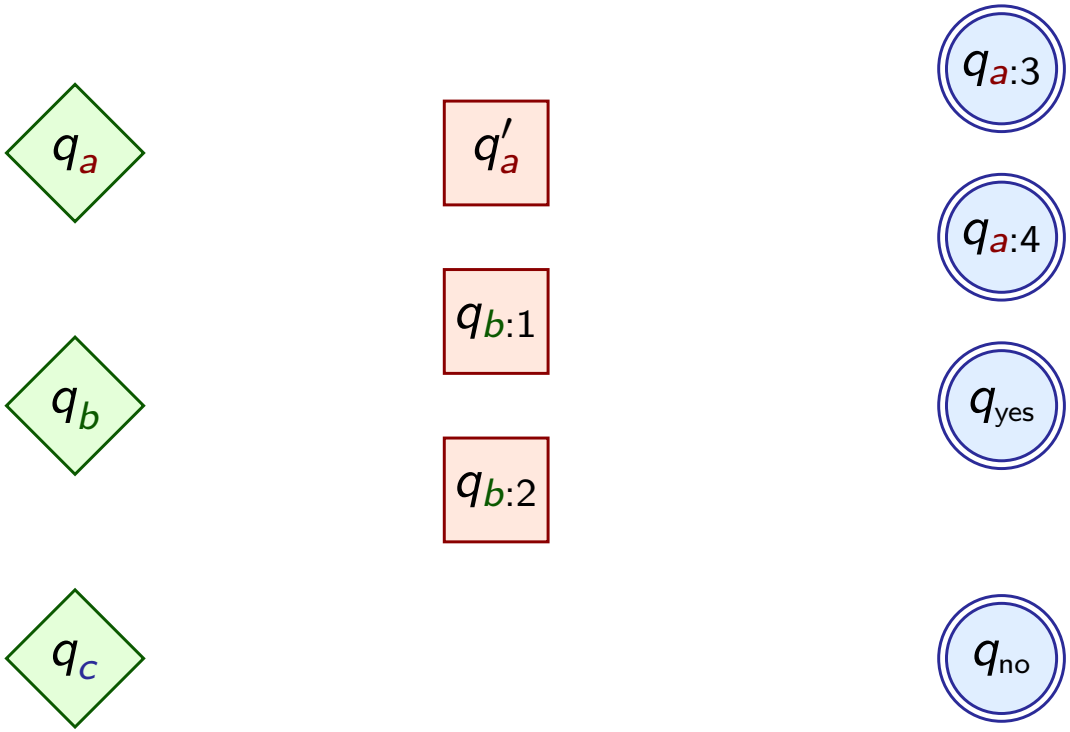


ADGA – Definition by Example

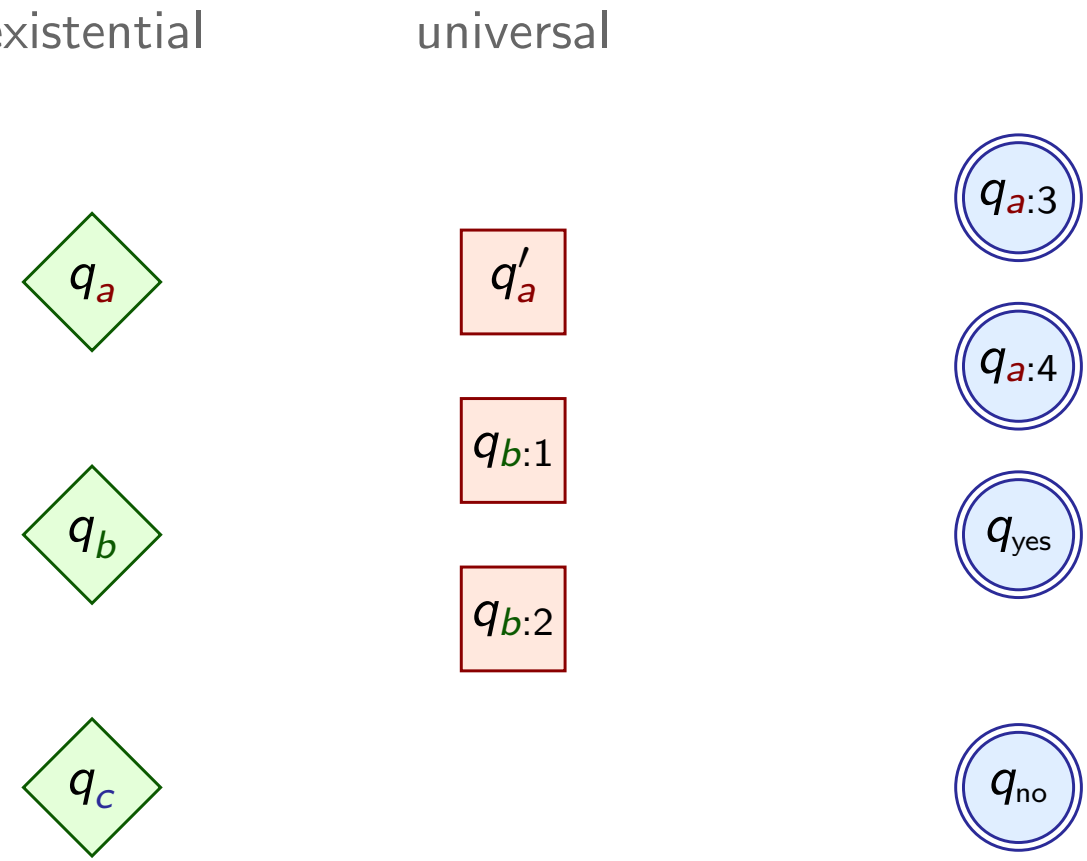


ADGA – Definition by Example

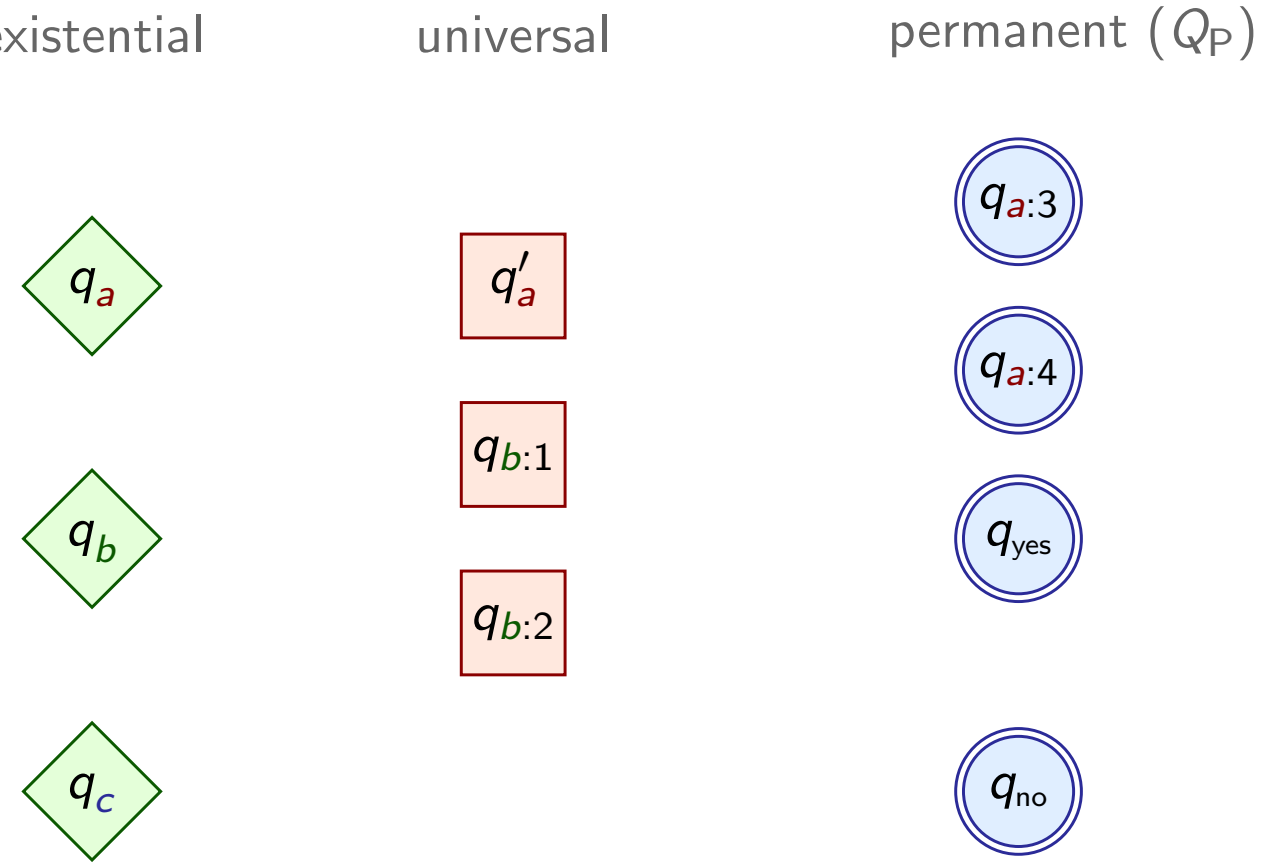
existential



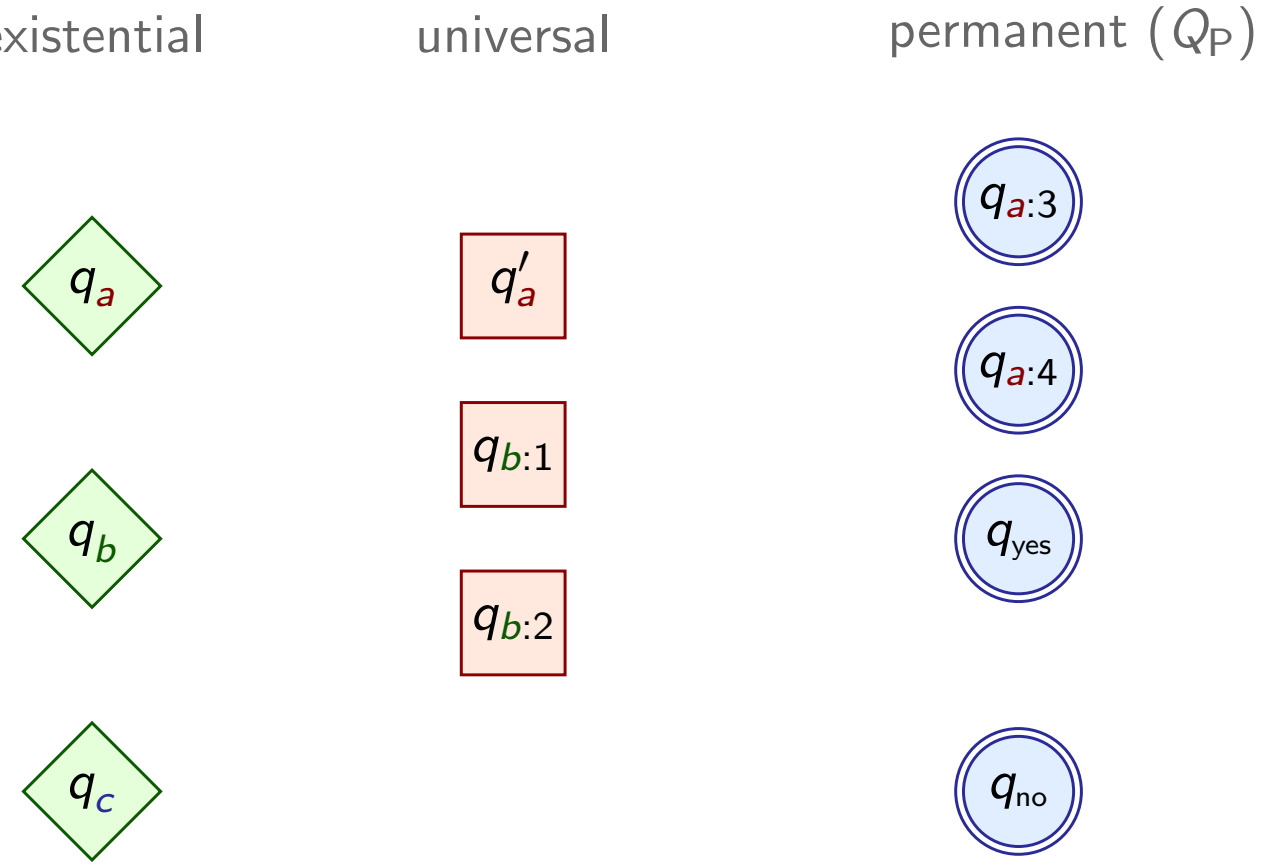
ADGA – Definition by Example



ADGA – Definition by Example

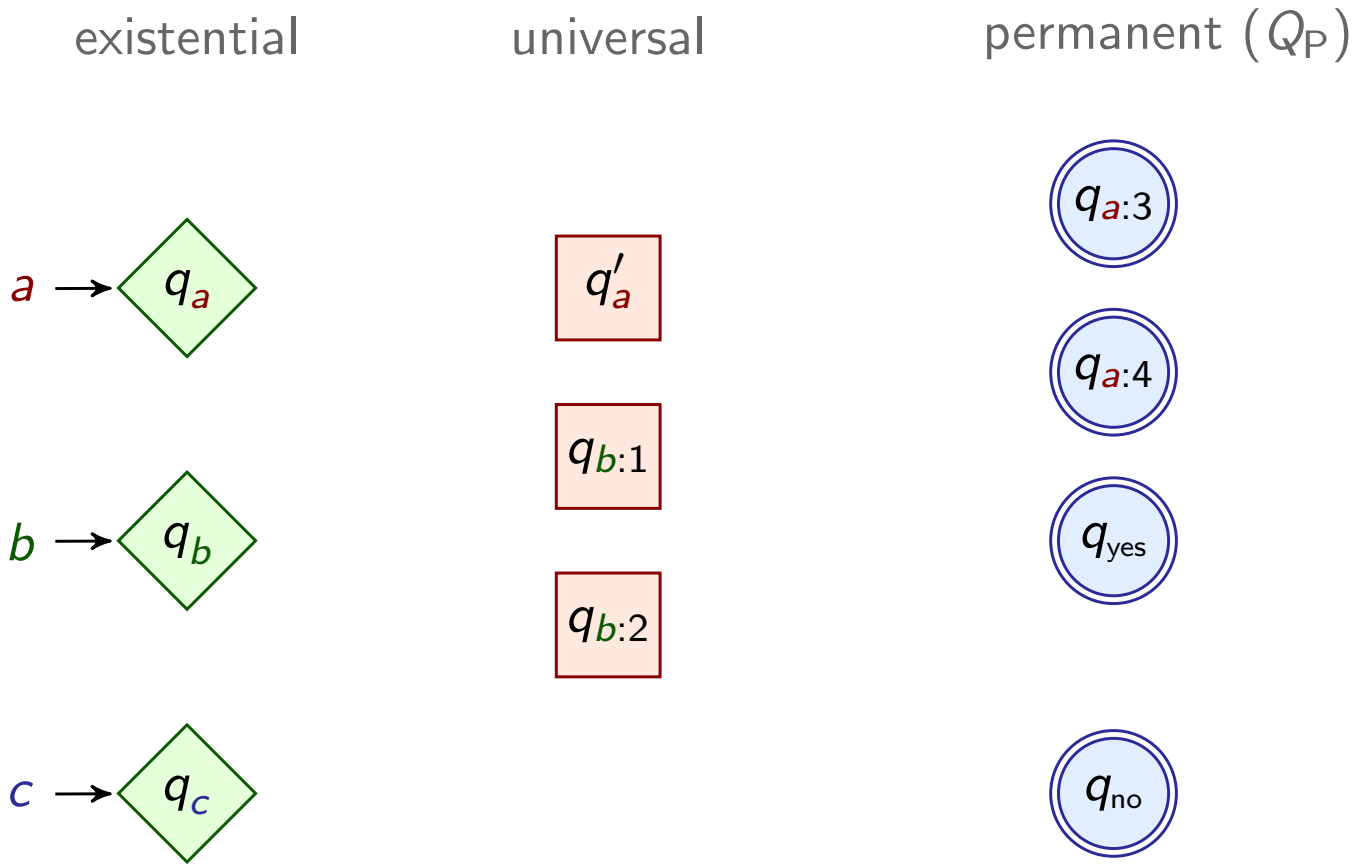


ADGA – Definition by Example



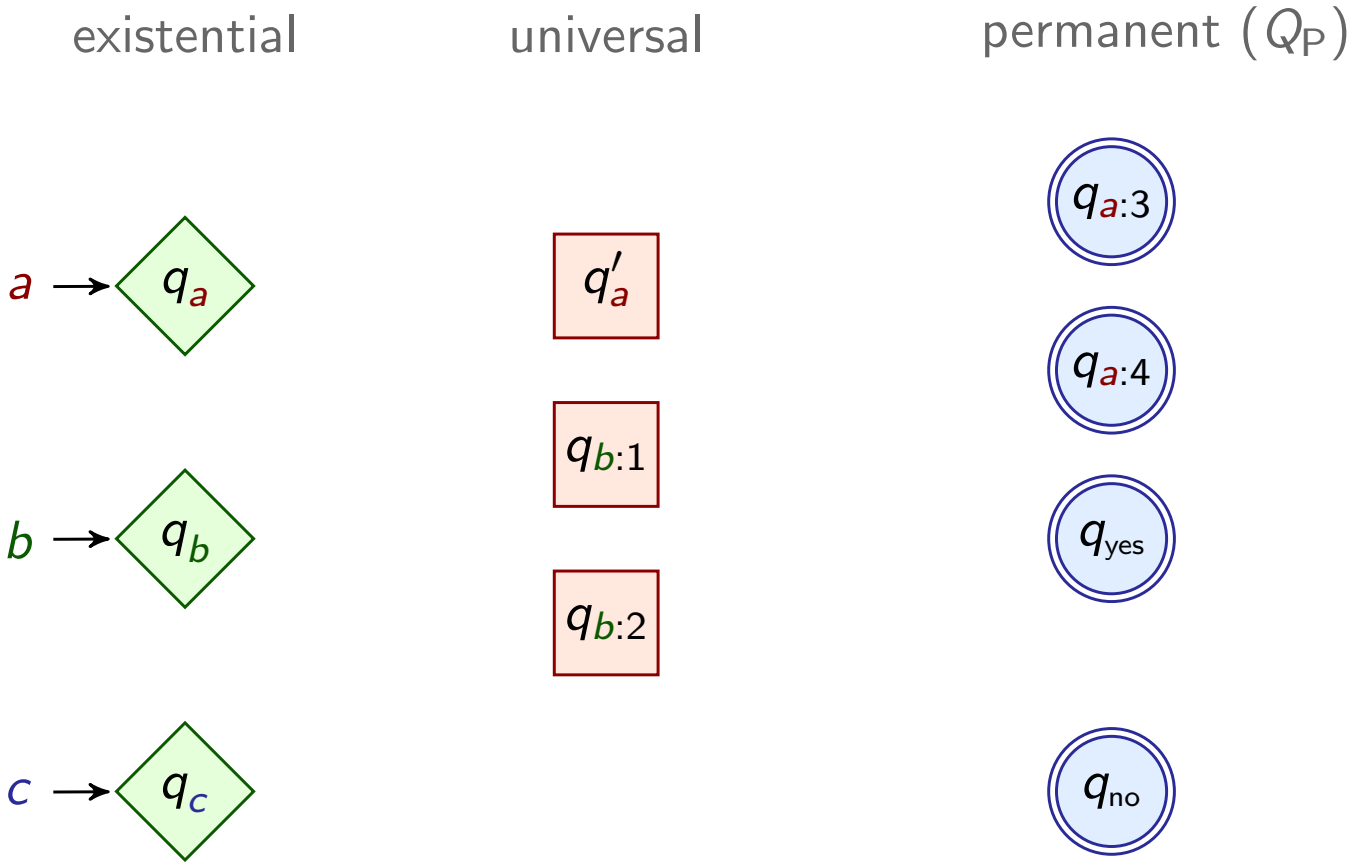
$\sigma : \Sigma \to Q$
(initialization)

ADGA – Definition by Example



$\sigma : \Sigma \to Q$
(initialization)

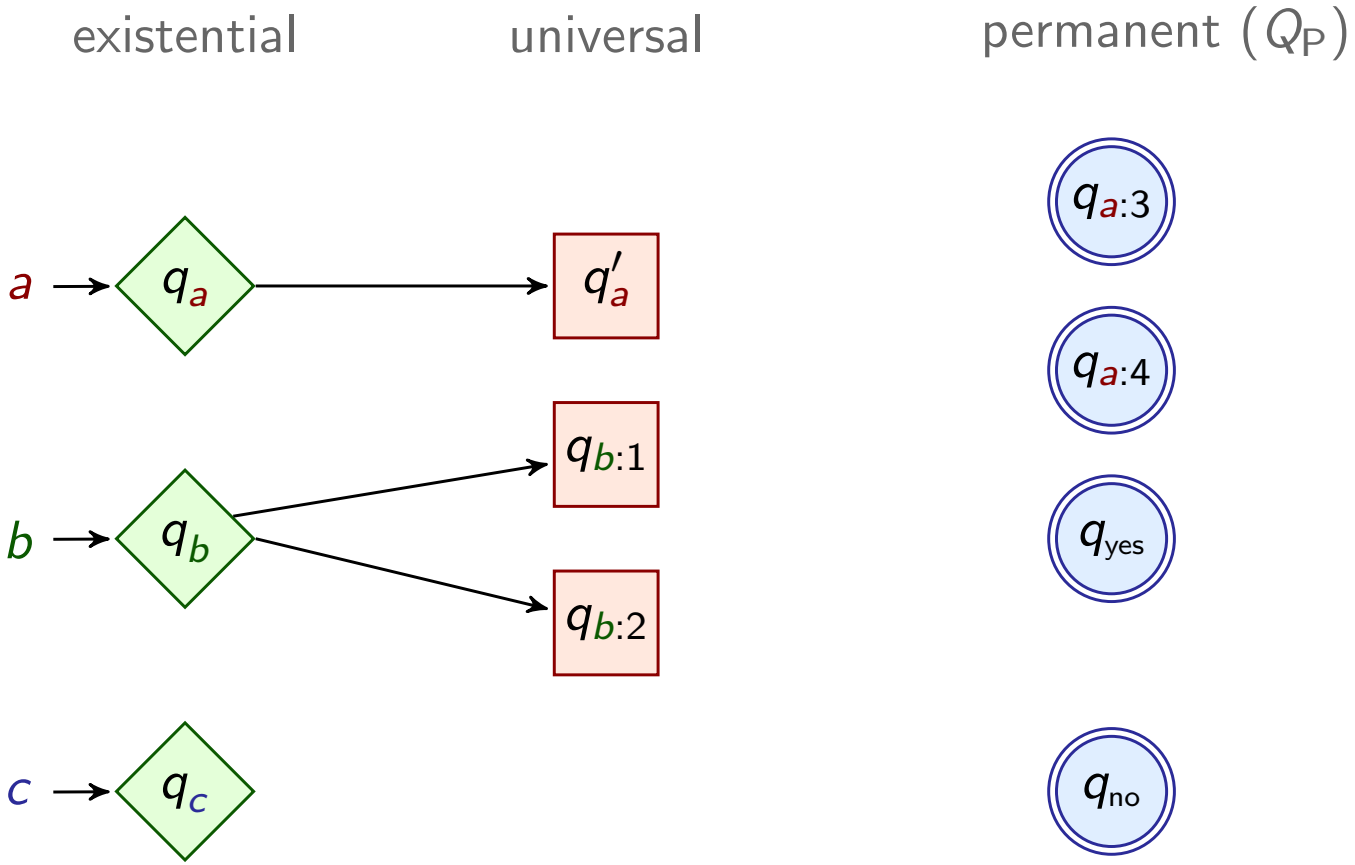
ADGA – Definition by Example



$\sigma : \Sigma \to Q$
(initialization)

$\delta : Q \times 2^Q \to 2^Q$
(transition)

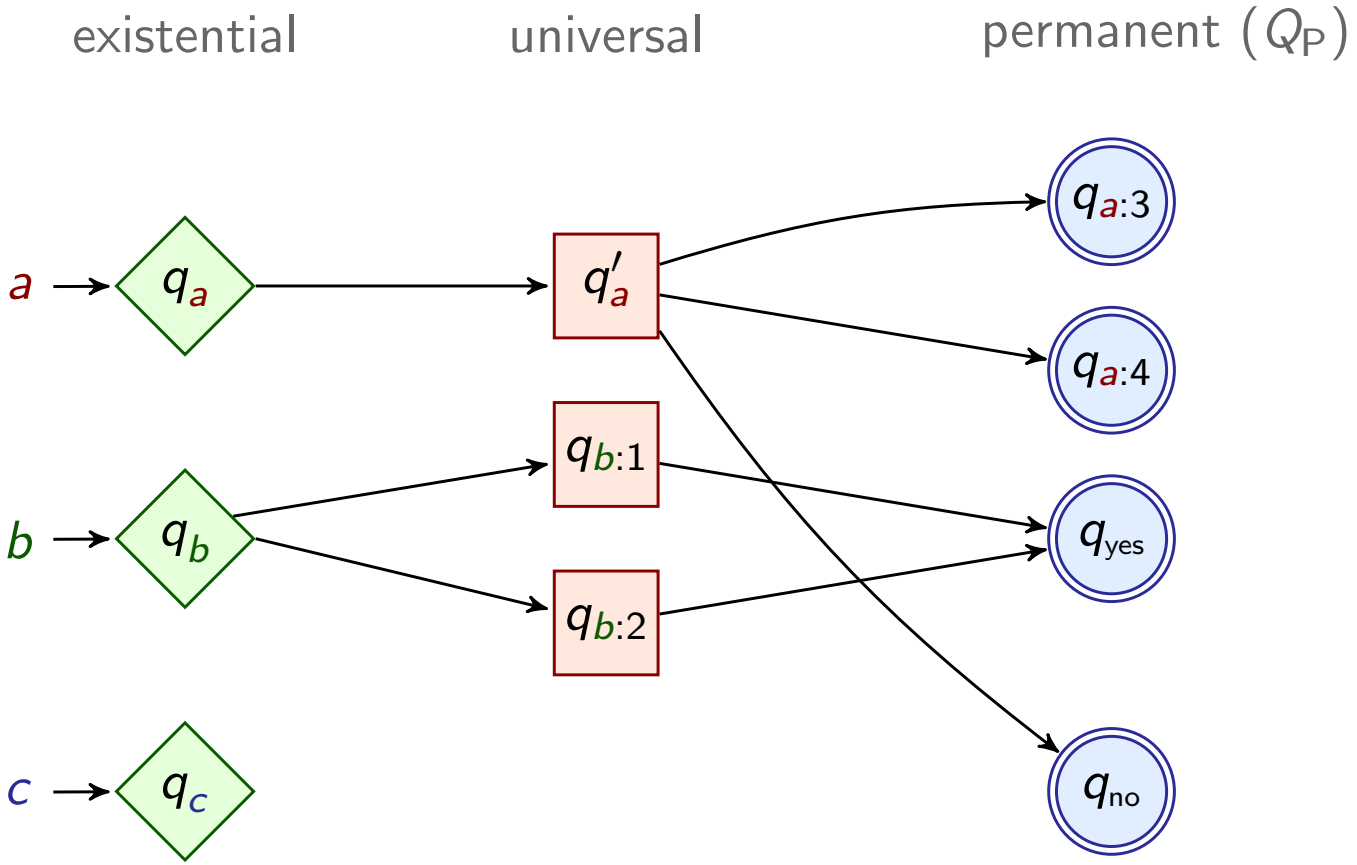
ADGA – Definition by Example



$\sigma : \Sigma \to Q$
(initialization)

$\delta : Q \times 2^Q \to 2^Q$
(transition)

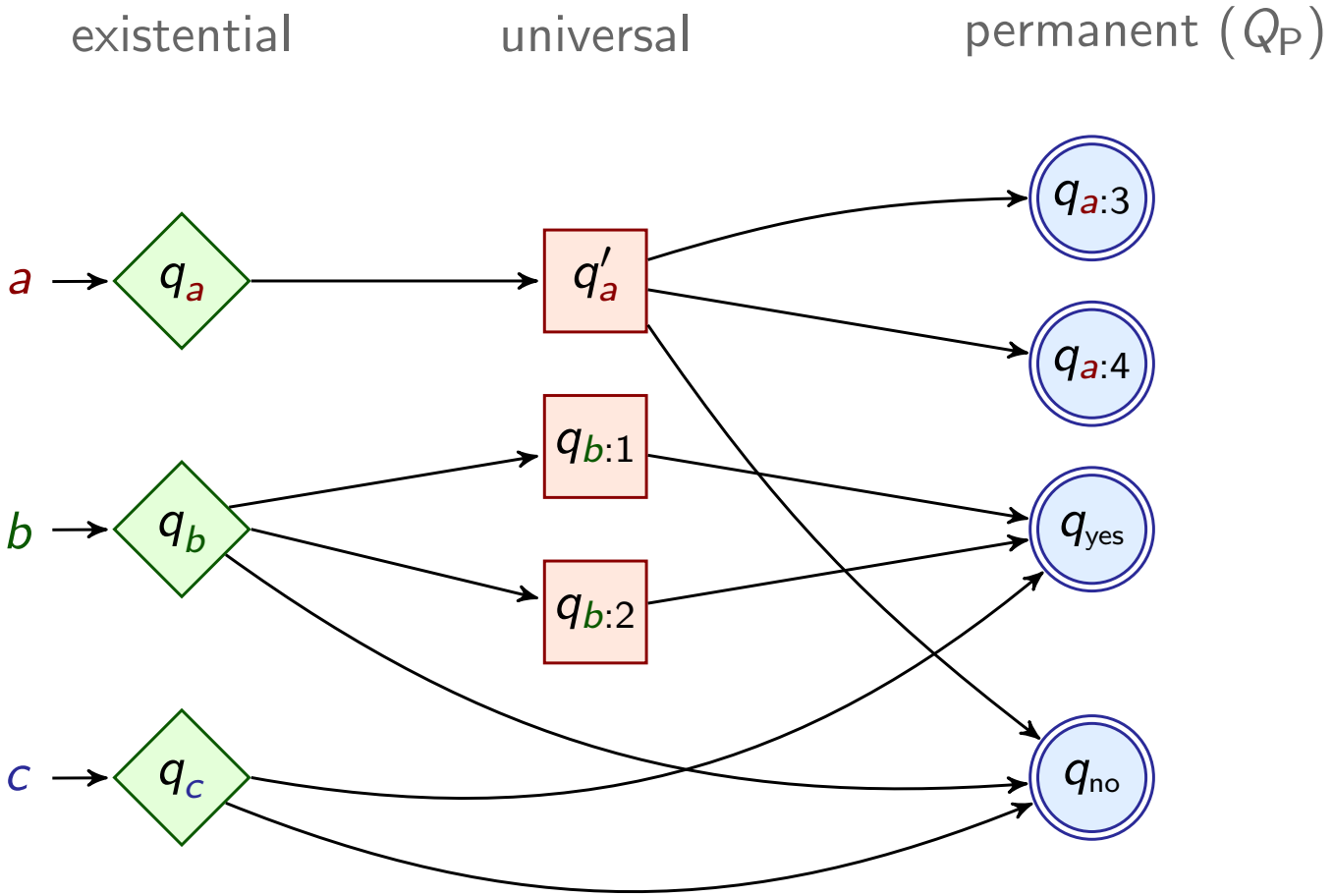
ADGA – Definition by Example



$\sigma : \Sigma \to Q$
(initialization)

$\delta : Q \times 2^Q \to 2^Q$
(transition)

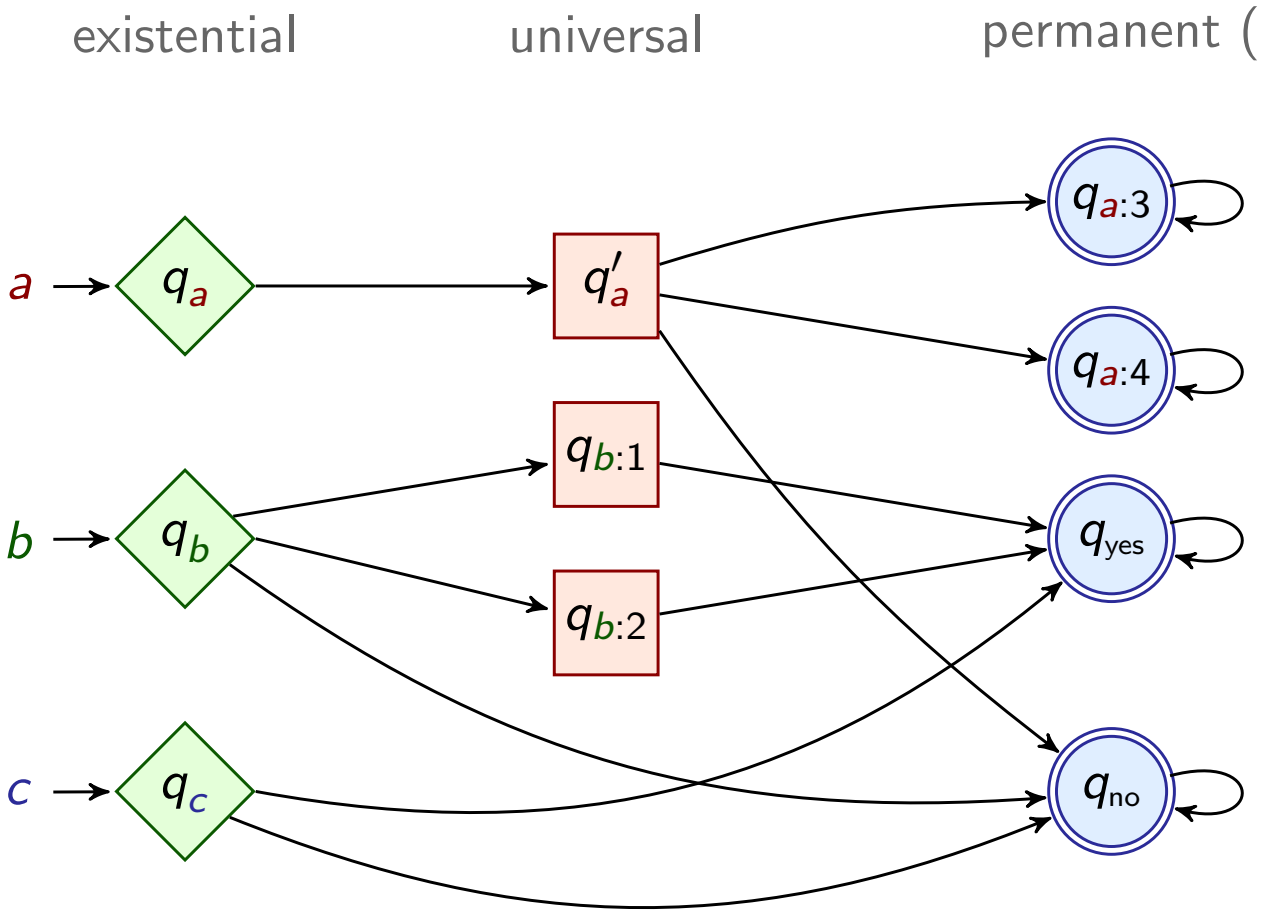
ADGA – Definition by Example



$\sigma : \Sigma \to Q$
(initialization)

$\delta : Q \times 2^Q \to 2^Q$
(transition)

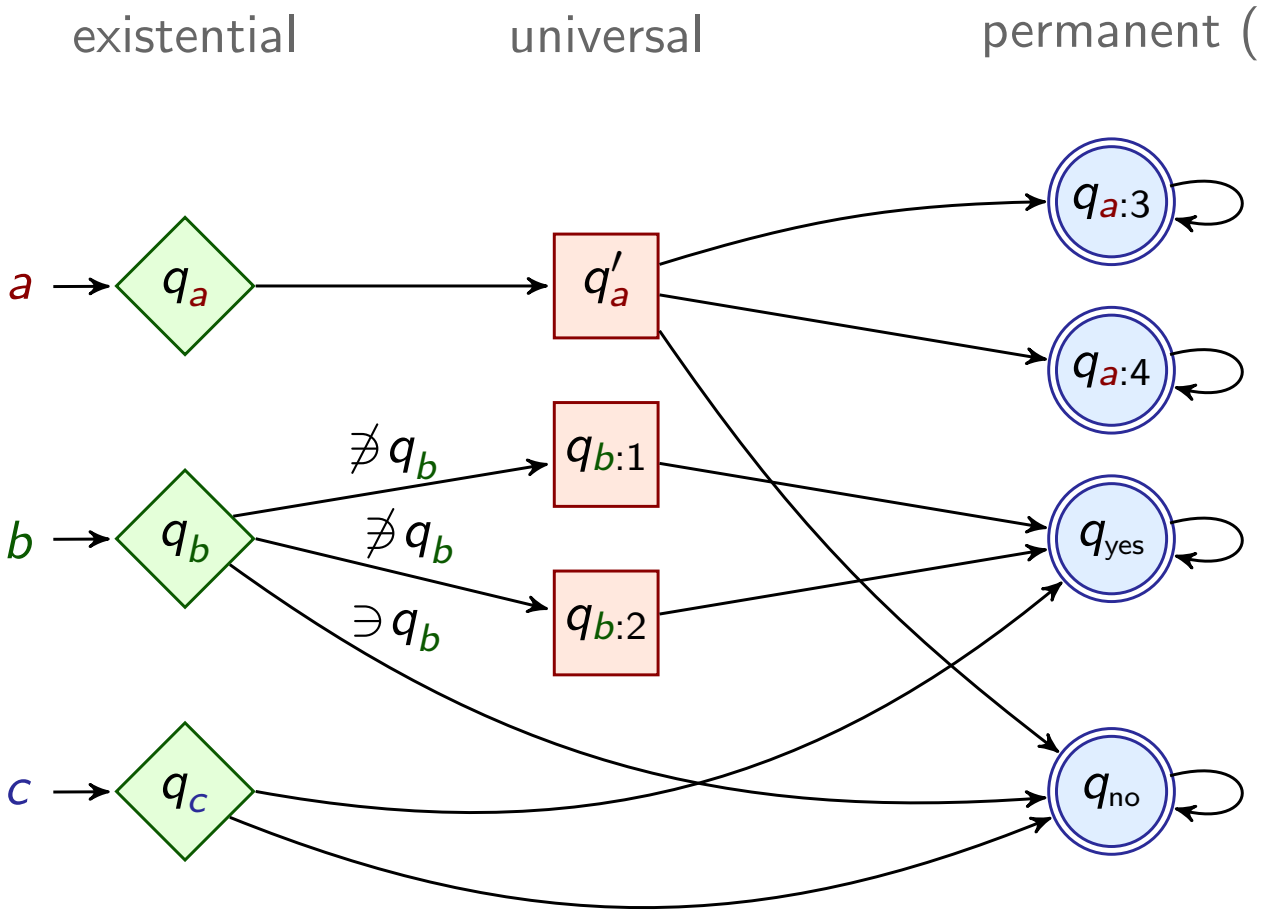
ADGA – Definition by Example



$\sigma : \Sigma \to Q$
(initialization)

$\delta : Q \times 2^Q \to 2^Q$
(transition)

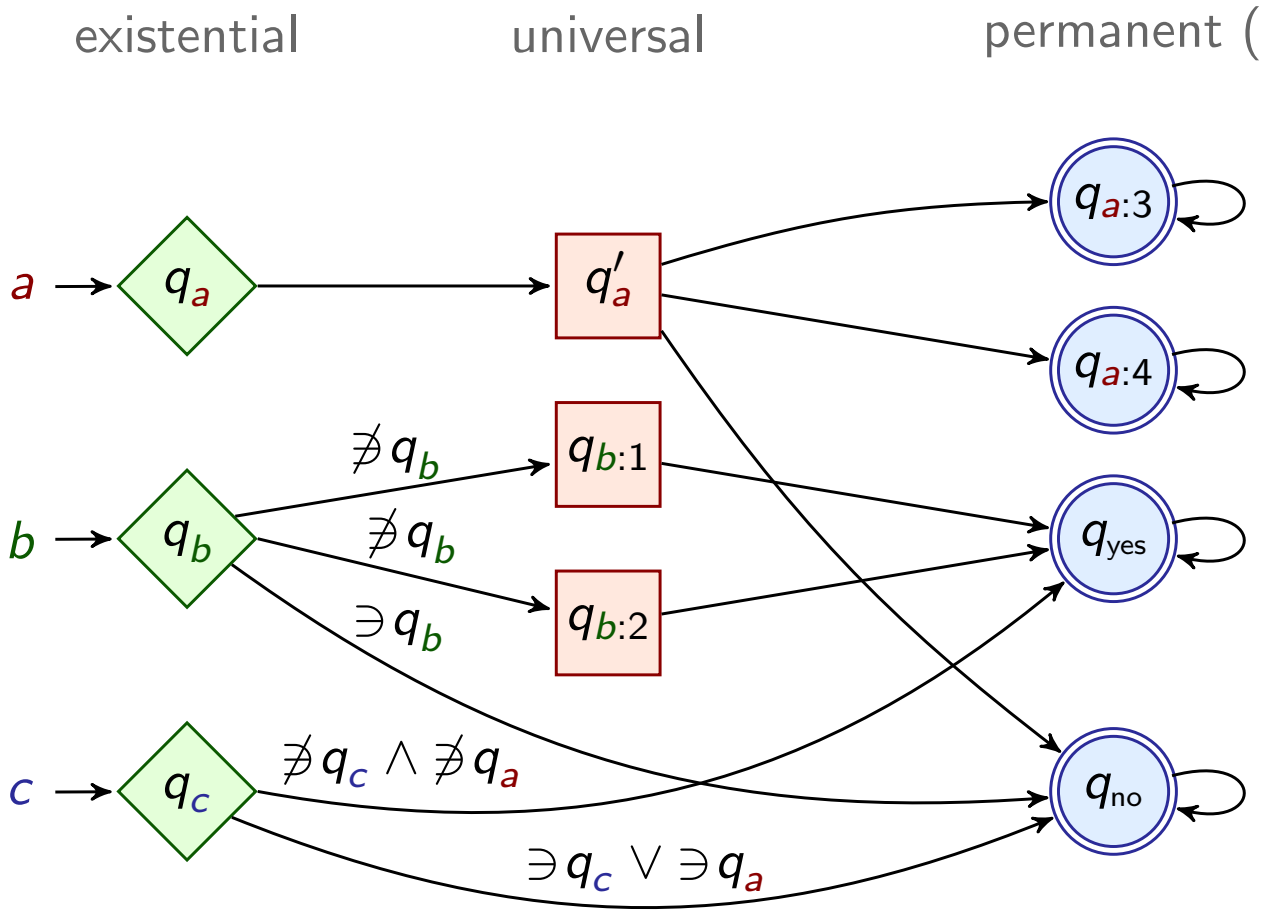
ADGA – Definition by Example



$\sigma : \Sigma \to Q$
(initialization)

$\delta : Q \times 2^Q \to 2^Q$
(transition)

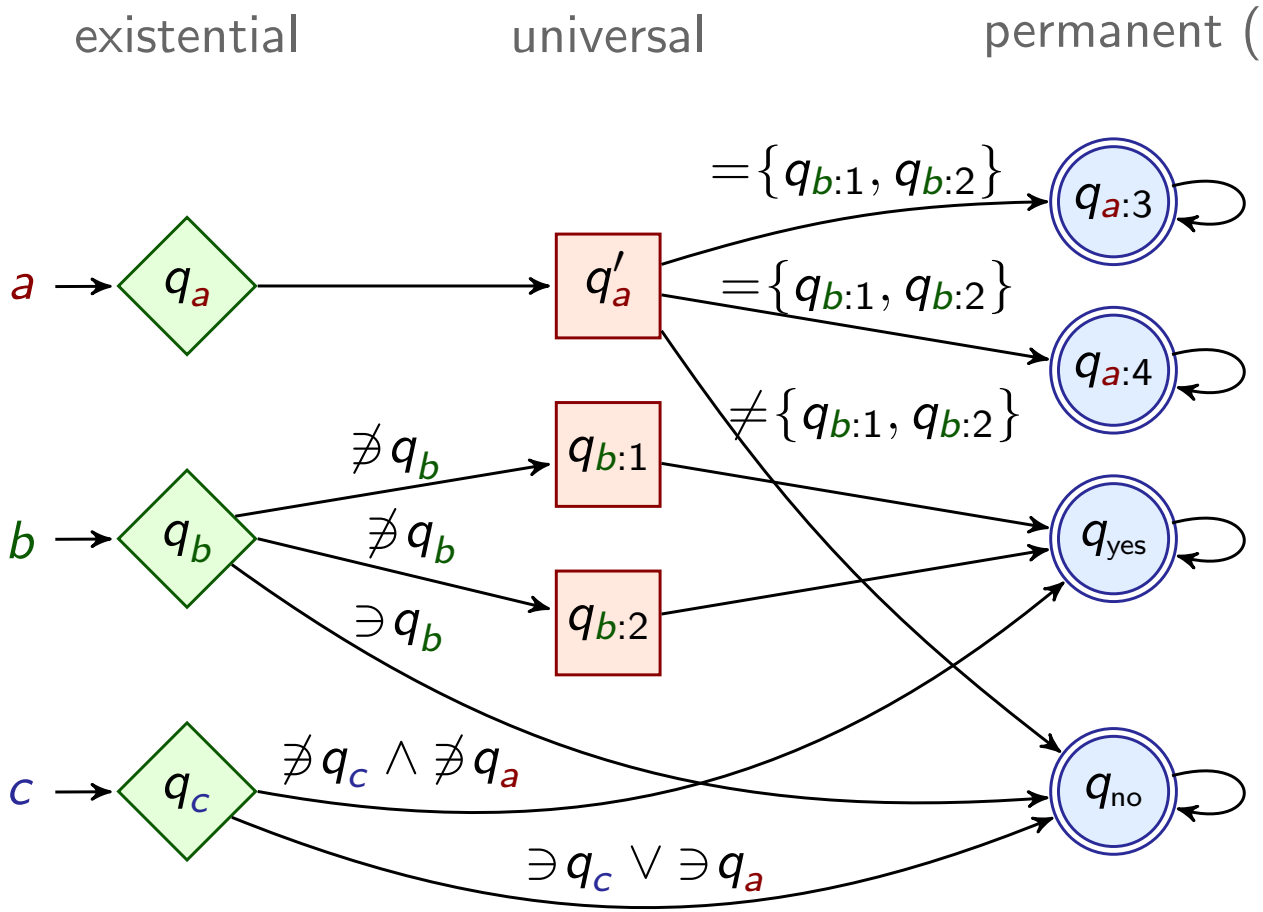
ADGA – Definition by Example



$\sigma : \Sigma \to Q$
(initialization)

$\delta : Q \times 2^Q \to 2^Q$
(transition)

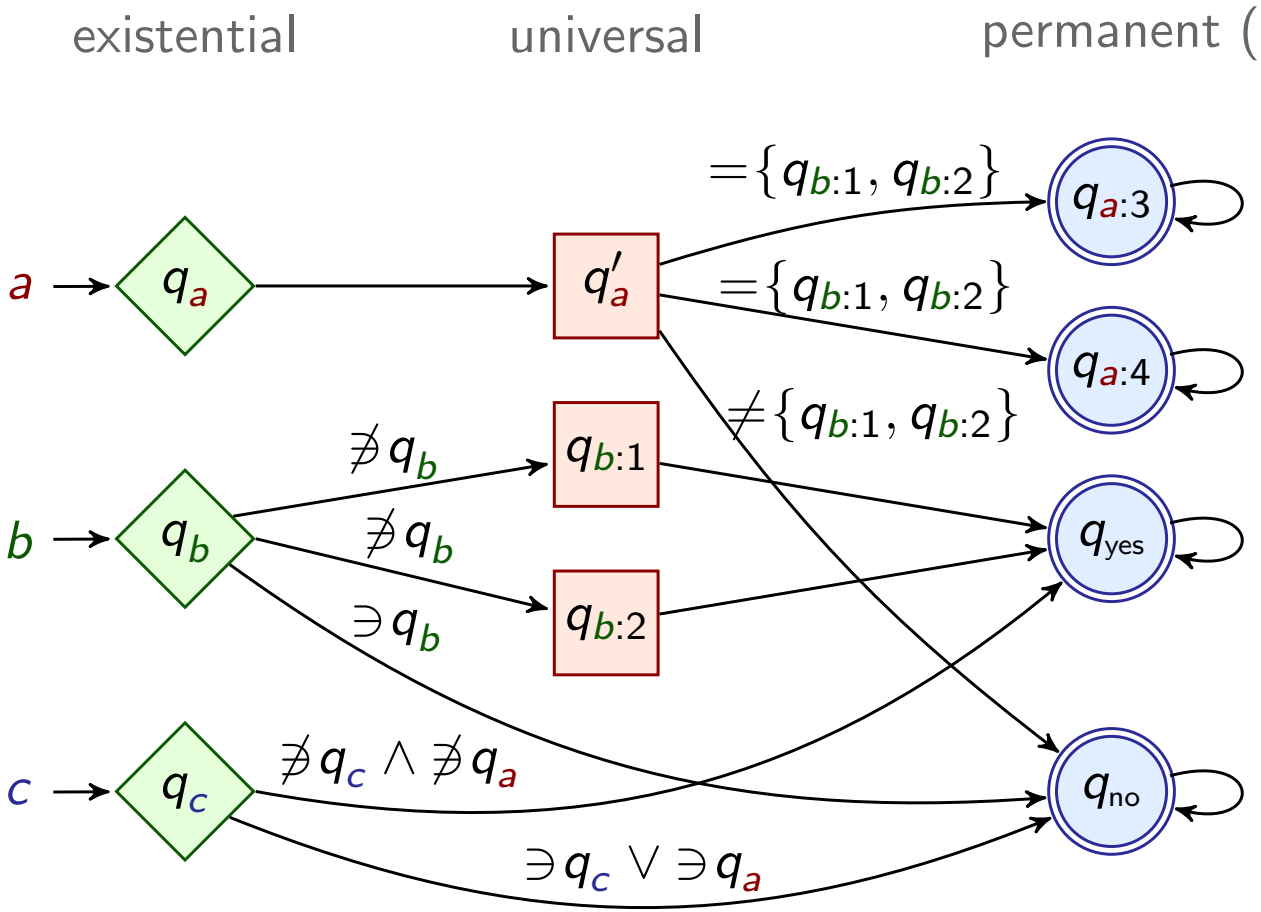
ADGA – Definition by Example



$\sigma : \Sigma \to Q$
(initialization)

$\delta : Q \times 2^Q \to 2^Q$
(transition)

ADGA – Definition by Example

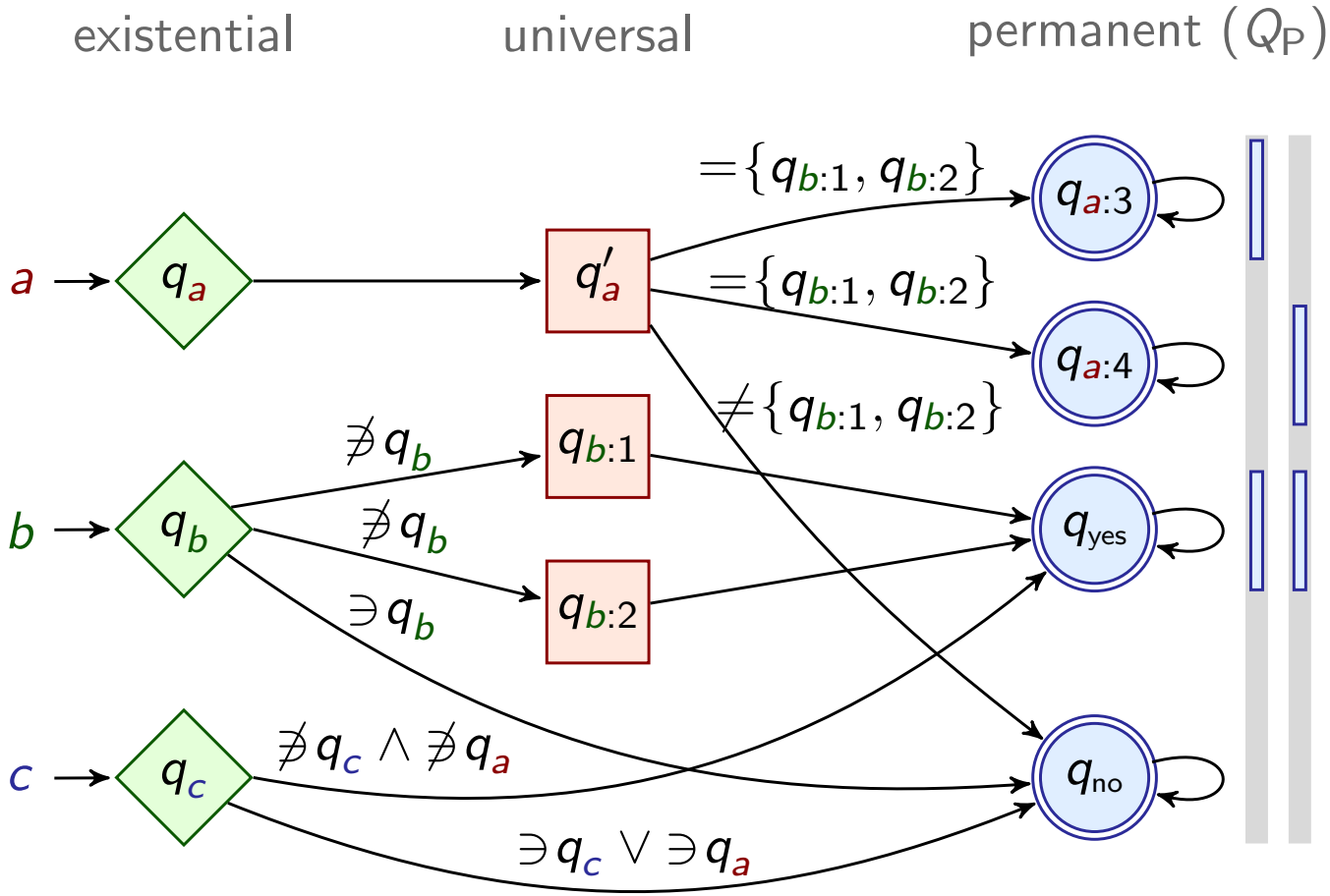


$\sigma : \Sigma \to Q$
(initialization)

$\delta : Q \times 2^Q \to 2^Q$
(transition)

$\mathcal{F} \subseteq 2^{Q_P}$
(acceptance)

ADGA – Definition by Example

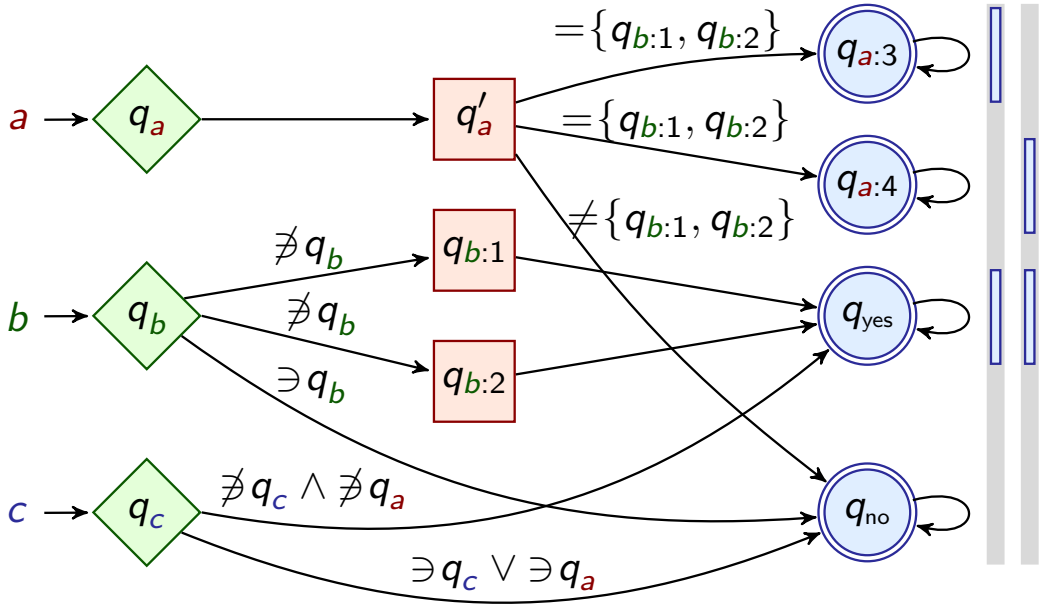


$\sigma : \Sigma \to Q$
(initialization)

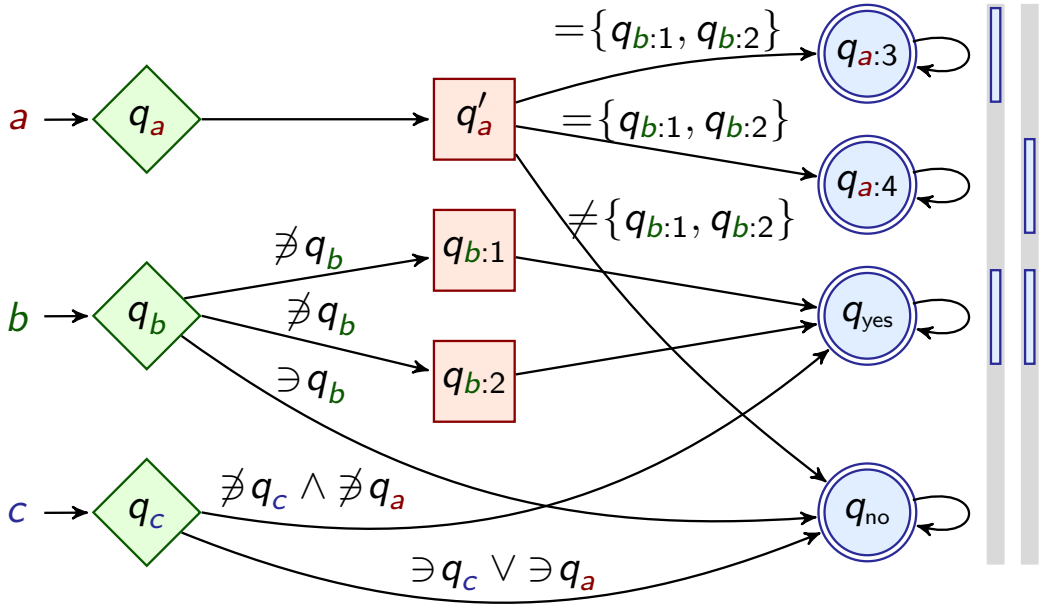
$\delta : Q \times 2^Q \to 2^Q$
(transition)

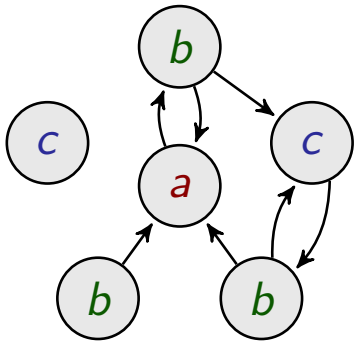
$\mathcal{F} \subseteq 2^{Q_P}$
(acceptance)

ADGA – Run

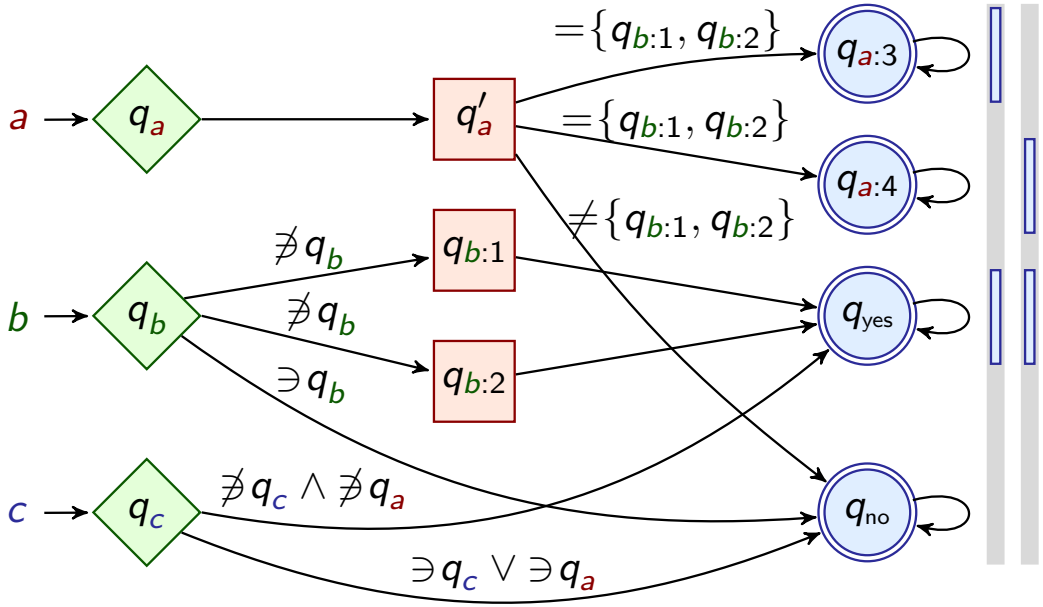

a
b
c
q_a
q_b
q_c
q'_a
$q_{b:1}$
$q_{b:2}$
$q_{a:3}$
$q_{a:4}$
q_{yes}
q_{no}
$= \{q_{b:1}, q_{b:2}\}$
$= \{q_{b:1}, q_{b:2}\}$
$\neq \{q_{b:1}, q_{b:2}\}$
$\not\ni q_b$
$\not\ni q_b$
$\ni q_b$
$\not\ni q_c \wedge \not\ni q_a$
$\ni q_c \vee \ni q_a$

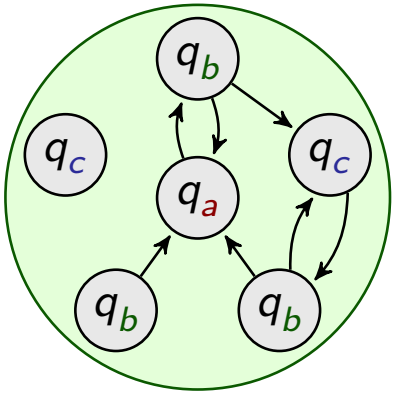
ADGA – Run


a
q_a
q'_a
$=\{q_{b:1}, q_{b:2}\}$
$q_{a:3}$
$=\{q_{b:1}, q_{b:2}\}$
$q_{a:4}$
$\neq\{q_{b:1}, q_{b:2}\}$
b
q_b
$\not\ni q_b$
$q_{b:1}$
$\not\ni q_b$
$q_{b:2}$
$\ni q_b$
q_{yes}
c
q_c
$\not\ni q_c \wedge \not\ni q_a$
q_{no}
$\ni q_c \vee \ni q_a$

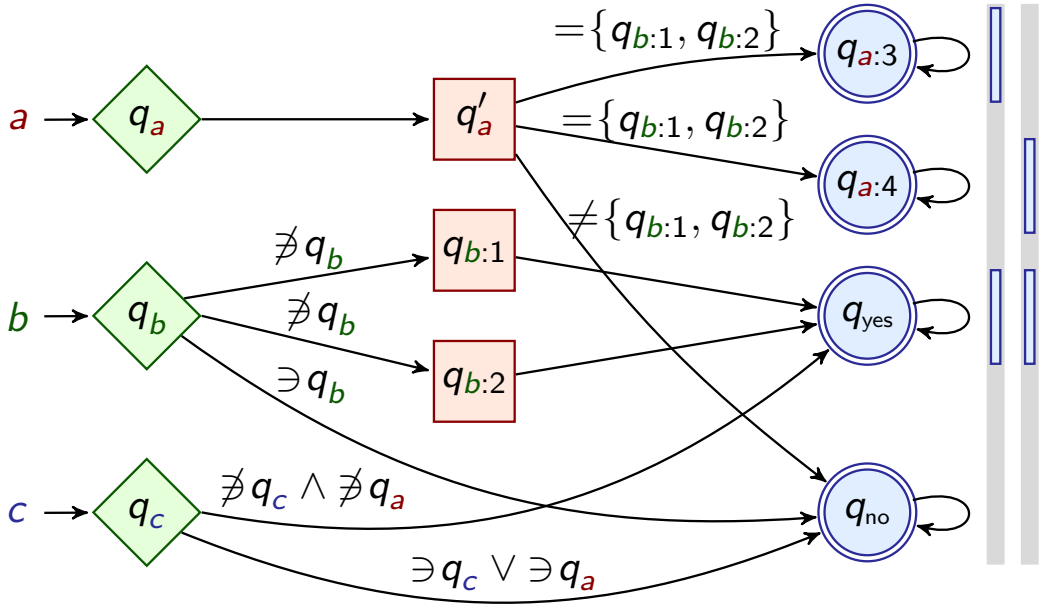

b
c
c
a
b
b

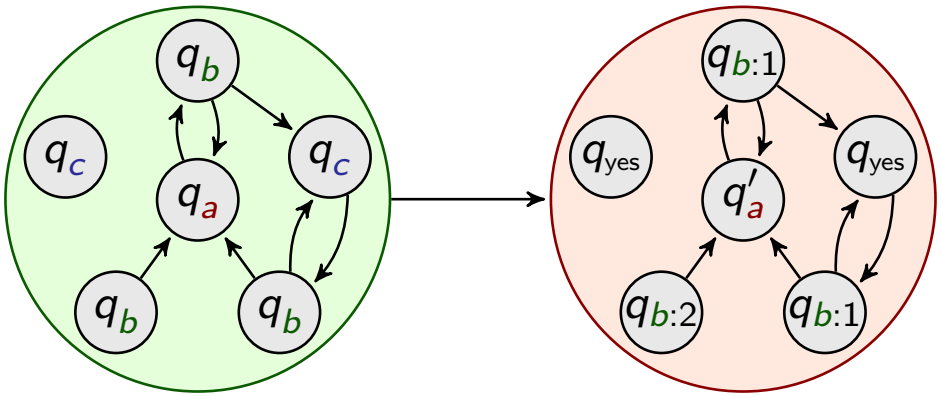
ADGA – Run


a
q_a
q'_a
$=\{q_{b:1}, q_{b:2}\}$
$q_{a:3}$
$=\{q_{b:1}, q_{b:2}\}$
$q_{a:4}$
$\neq\{q_{b:1}, q_{b:2}\}$
b
q_b
$\not\ni q_b$
$q_{b:1}$
$\not\ni q_b$
$q_{b:2}$
$\ni q_b$
q_{yes}
c
q_c
$\not\ni q_c \wedge \not\ni q_a$
q_{no}
$\ni q_c \vee \ni q_a$

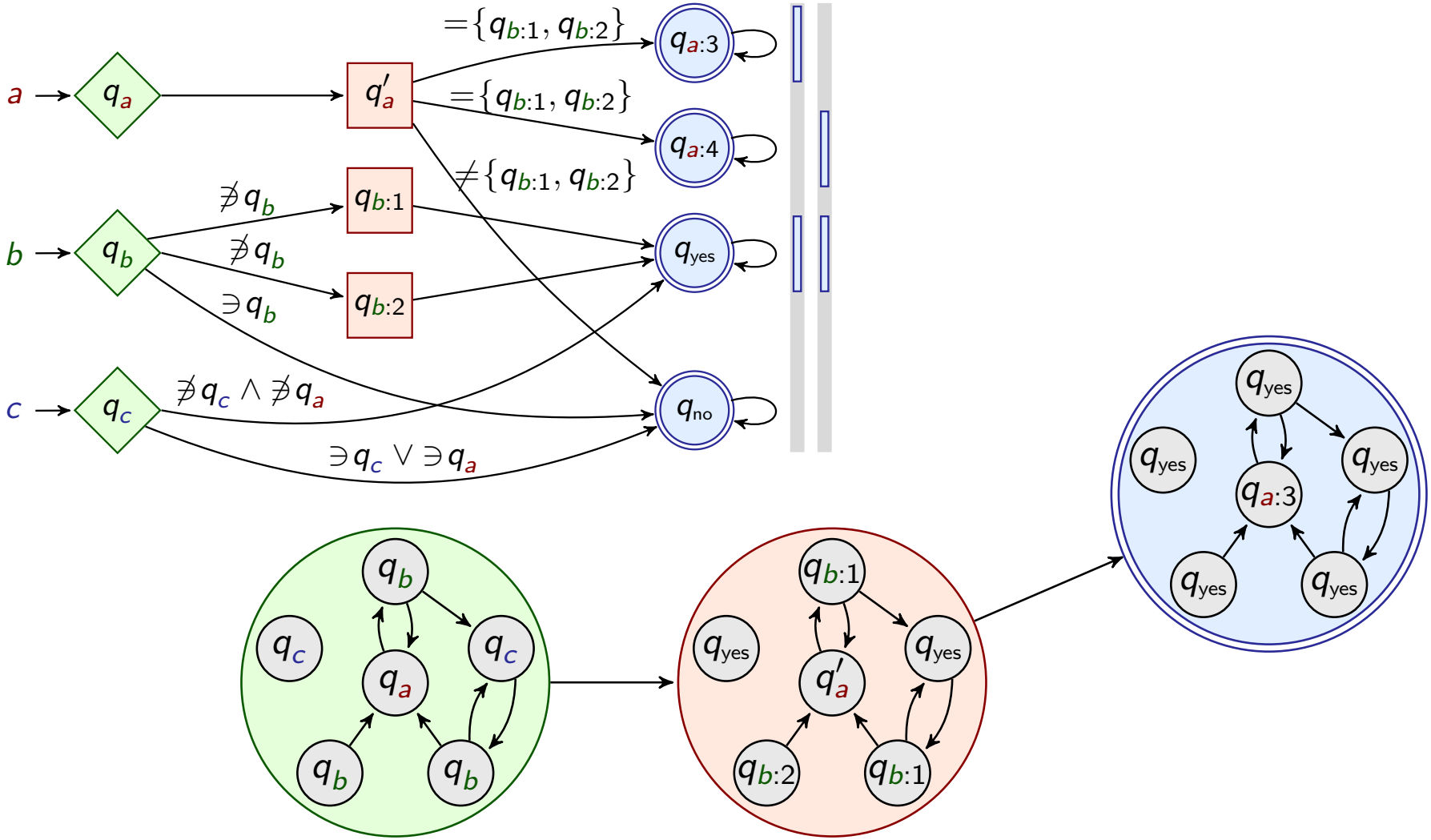

q_b
q_c
q_c
q_a
q_b
q_b

ADGA – Run

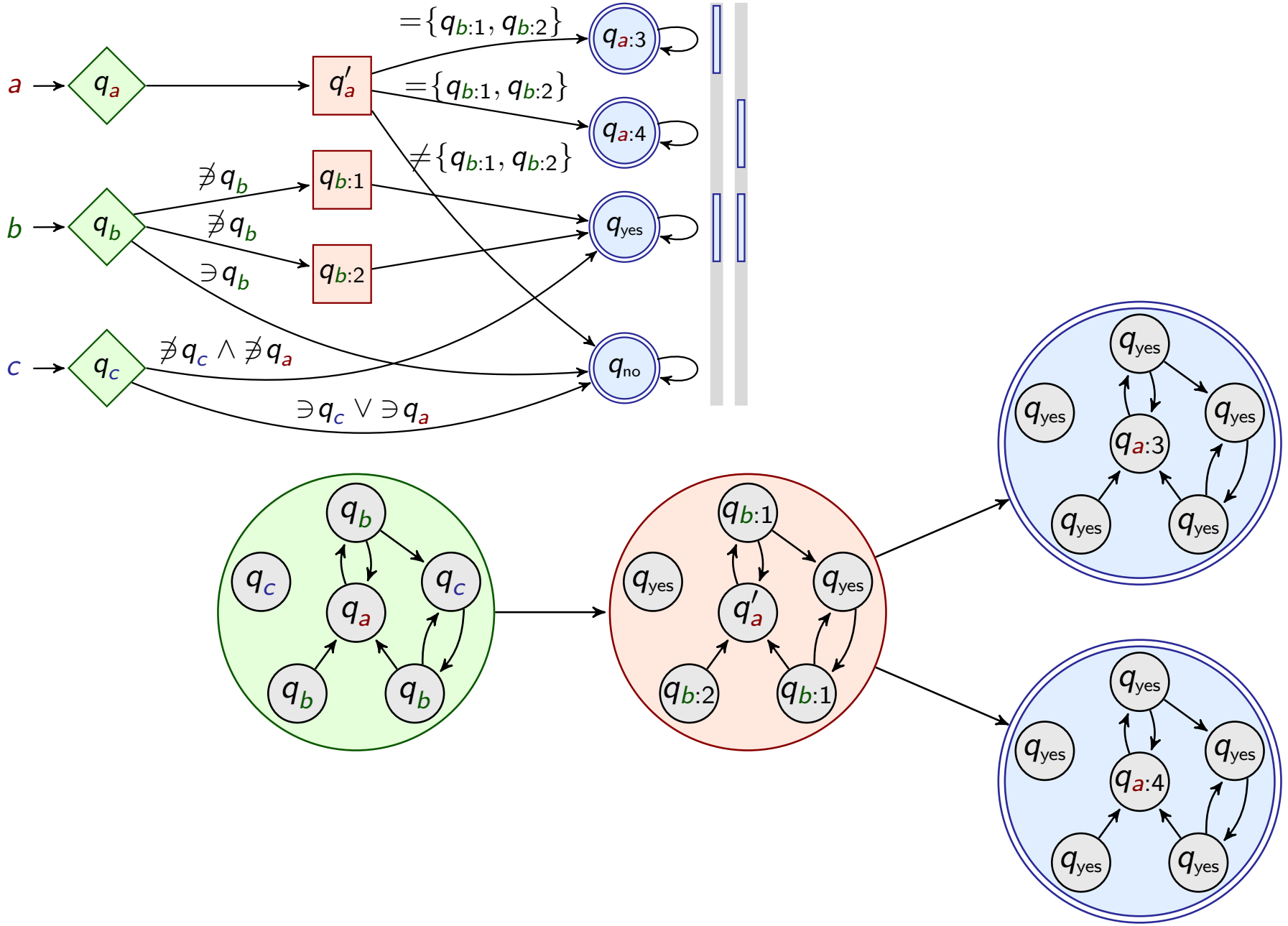

a
b
c
q_a
q_b
q_c
q'_a
$q_{b:1}$
$q_{b:2}$
$q_{a:3}$
$q_{a:4}$
q_{yes}
q_{no}
$= \{q_{b:1}, q_{b:2}\}$
$= \{q_{b:1}, q_{b:2}\}$
$\neq \{q_{b:1}, q_{b:2}\}$
$\not\ni q_b$
$\not\ni q_b$
$\ni q_b$
$\not\ni q_c \wedge \not\ni q_a$
$\ni q_c \vee \ni q_a$


q_b
q_c
q_c
q_a
q_b
q_b
$q_{b:1}$
q_{yes}
q_{yes}
q'_a
$q_{b:2}$
$q_{b:1}$

ADGA – Run


a
b
c
q_a
q_b
q_c
q'_a
$q_{b:1}$
$q_{b:2}$
$q_{a:3}$
$q_{a:4}$
q_{yes}
q_{no}
$= \{q_{b:1}, q_{b:2}\}$
$= \{q_{b:1}, q_{b:2}\}$
$\neq \{q_{b:1}, q_{b:2}\}$
$\not\ni q_b$
$\not\ni q_b$
$\ni q_b$
$\not\ni q_c \wedge \not\ni q_a$
$\ni q_c \vee \ni q_a$

ADGA – Run


a
b
c
q_a
q_b
q_c
q'_a
$q_{b:1}$
$q_{b:2}$
$q_{a:3}$
$q_{a:4}$
q_{yes}
q_{no}
$= \{q_{b:1}, q_{b:2}\}$
$= \{q_{b:1}, q_{b:2}\}$
$\neq \{q_{b:1}, q_{b:2}\}$
$\not\ni q_b$
$\not\ni q_b$
$\ni q_b$
$\not\ni q_c \wedge \not\ni q_a$
$\ni q_c \vee \ni q_a$

Main Result

Theorem ($\mathcal{L}_{\mathsf{ADGA}} = \mathcal{L}_{\mathsf{MSO}}$)

A graph language is ADGA-recognizable iff it is MSO-definable. There are effective translations in both directions.

DGA Variants

ADGA: Alternating

NDGA: Nondeterministic

DDGA: Deterministic

DGA Variants

$$\mathcal{L}_{\text{DDGA}} \subset \mathcal{L}_{\text{NDGA}} \subset \mathcal{L}_{\text{ADGA}}$$

ADGA: Alternating

NDGA: Nondeterministic

DDGA: Deterministic

DGA Variants

$$\mathcal{L}_{\text{DDGA}} \subset \mathcal{L}_{\text{NDGA}} \subset \mathcal{L}_{\text{ADGA}}$$

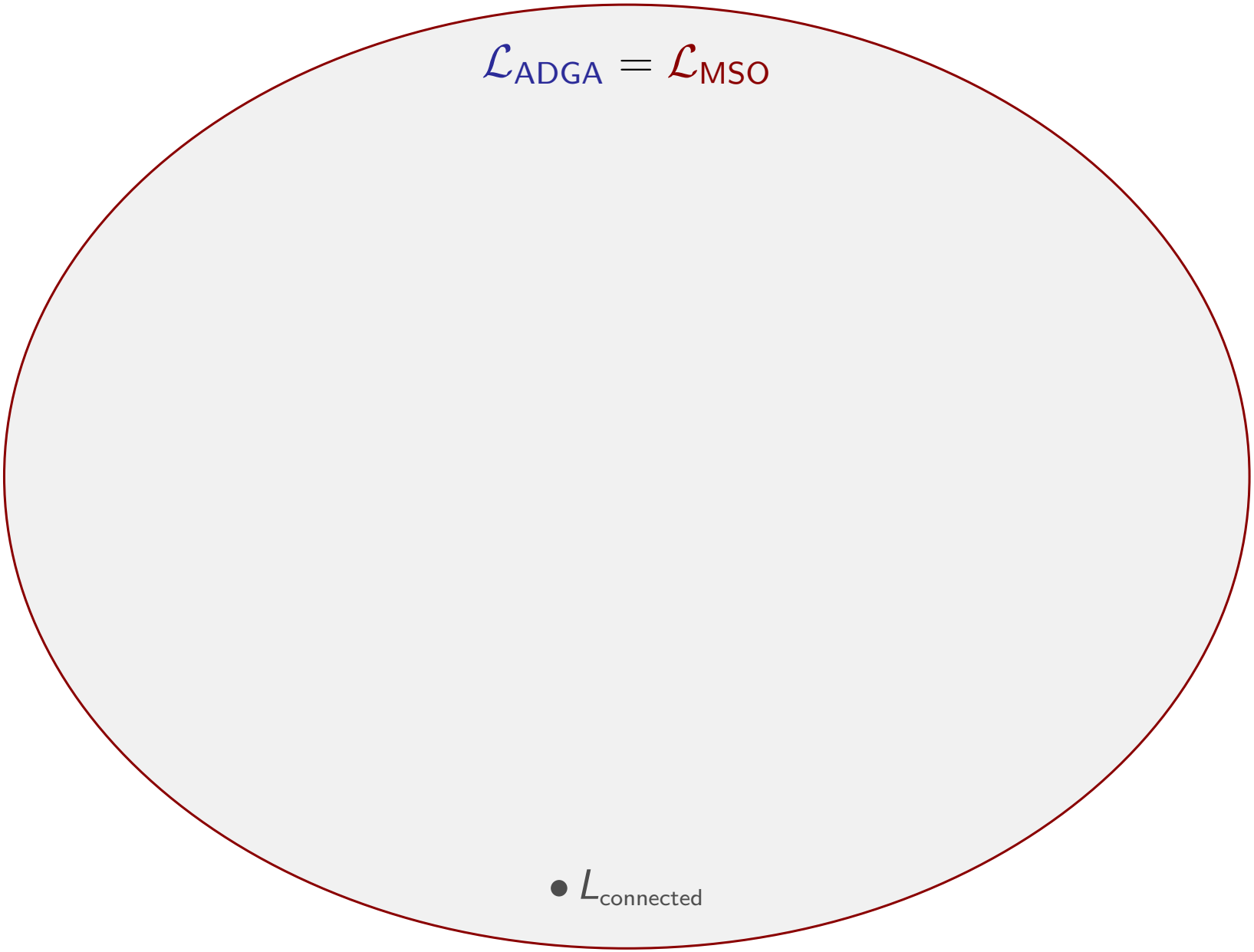
ADGA: Alternating

NDGA: Nondeterministic

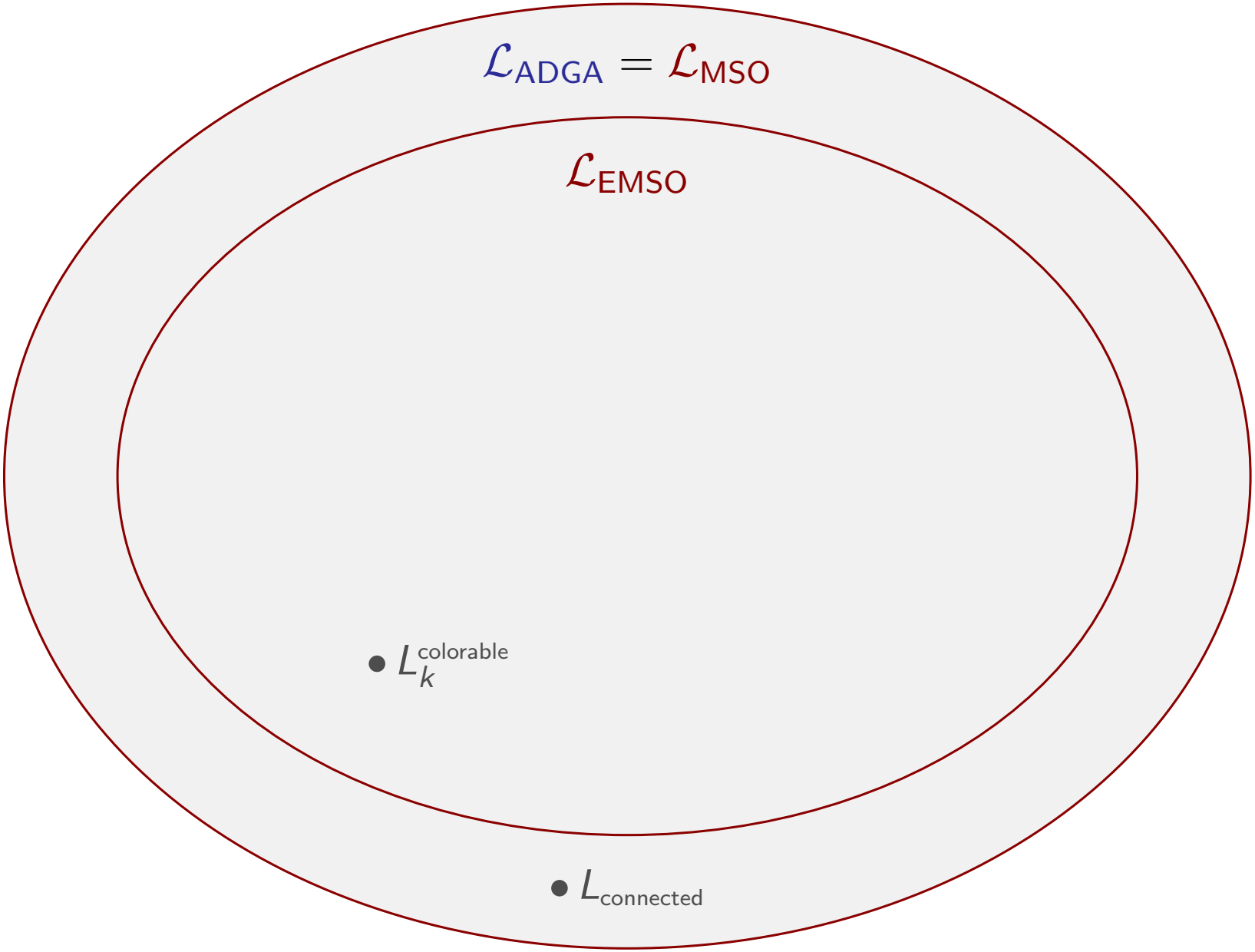
DDGA: Deterministic

} Emptiness decidable

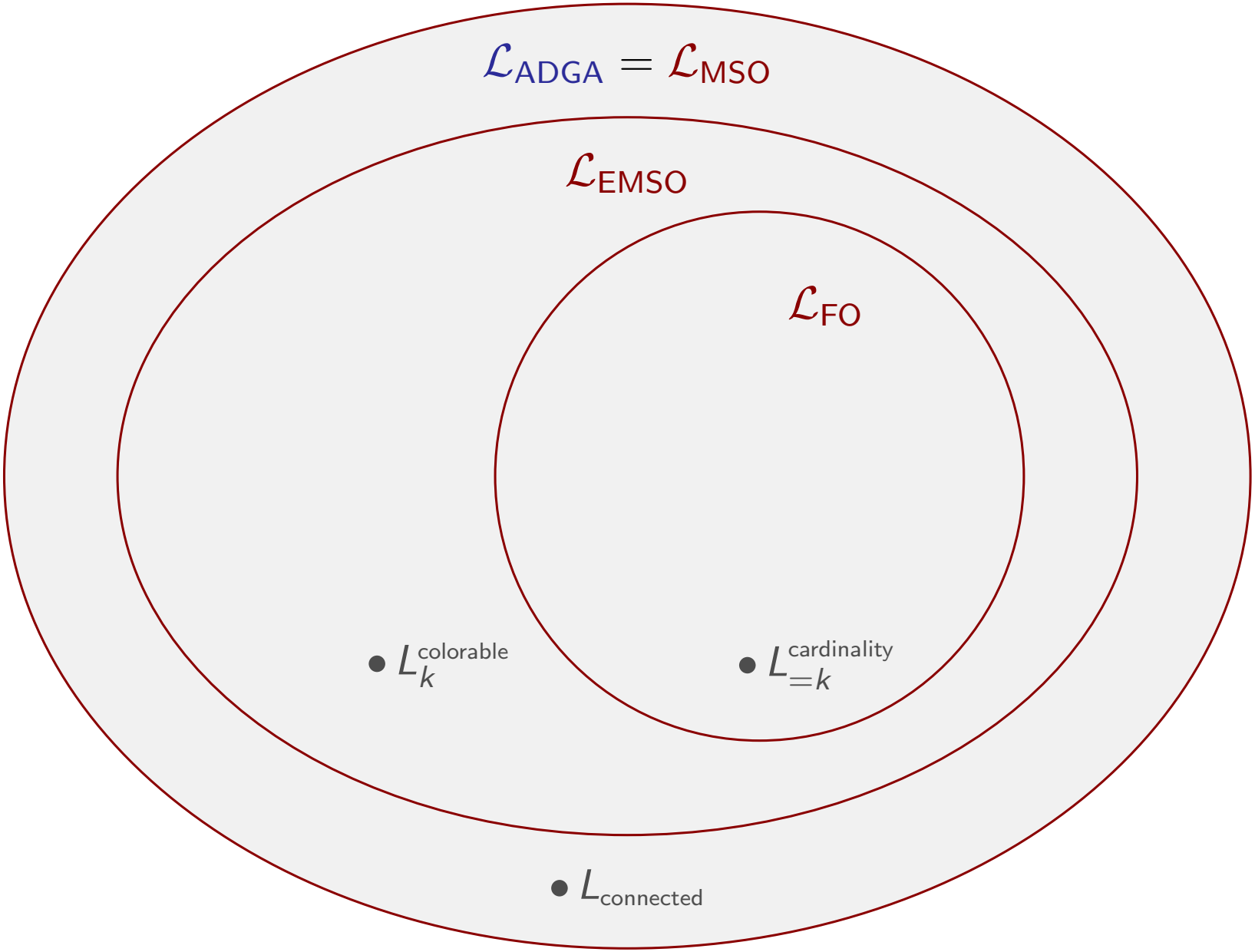
Big Picture



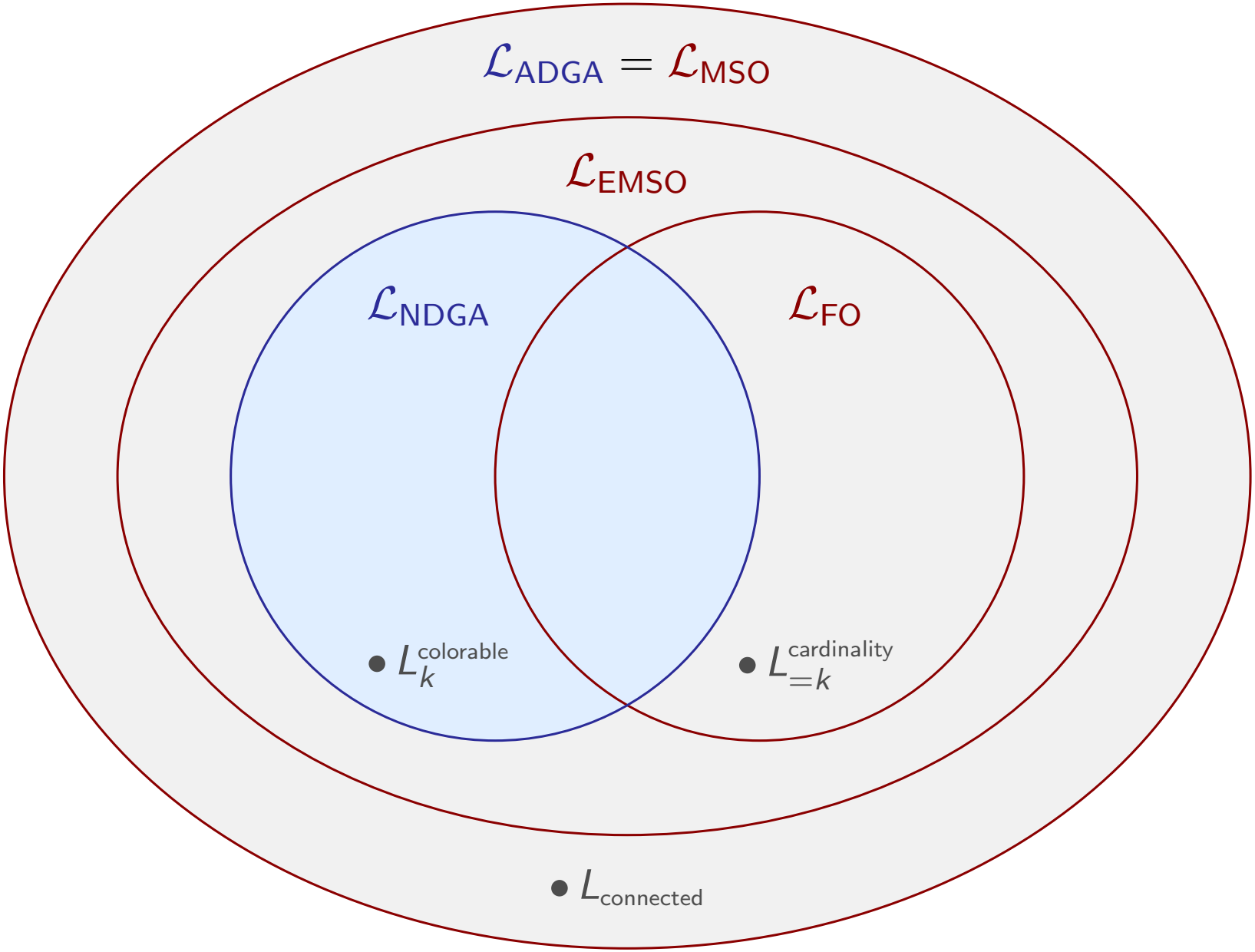
Big Picture


$\mathcal{L}_{\text{ADGA}} = \mathcal{L}_{\text{MSO}}$
$\mathcal{L}_{\text{EMSO}}$
$L_k^{\text{colorable}}$
$L_{\text{connected}}$

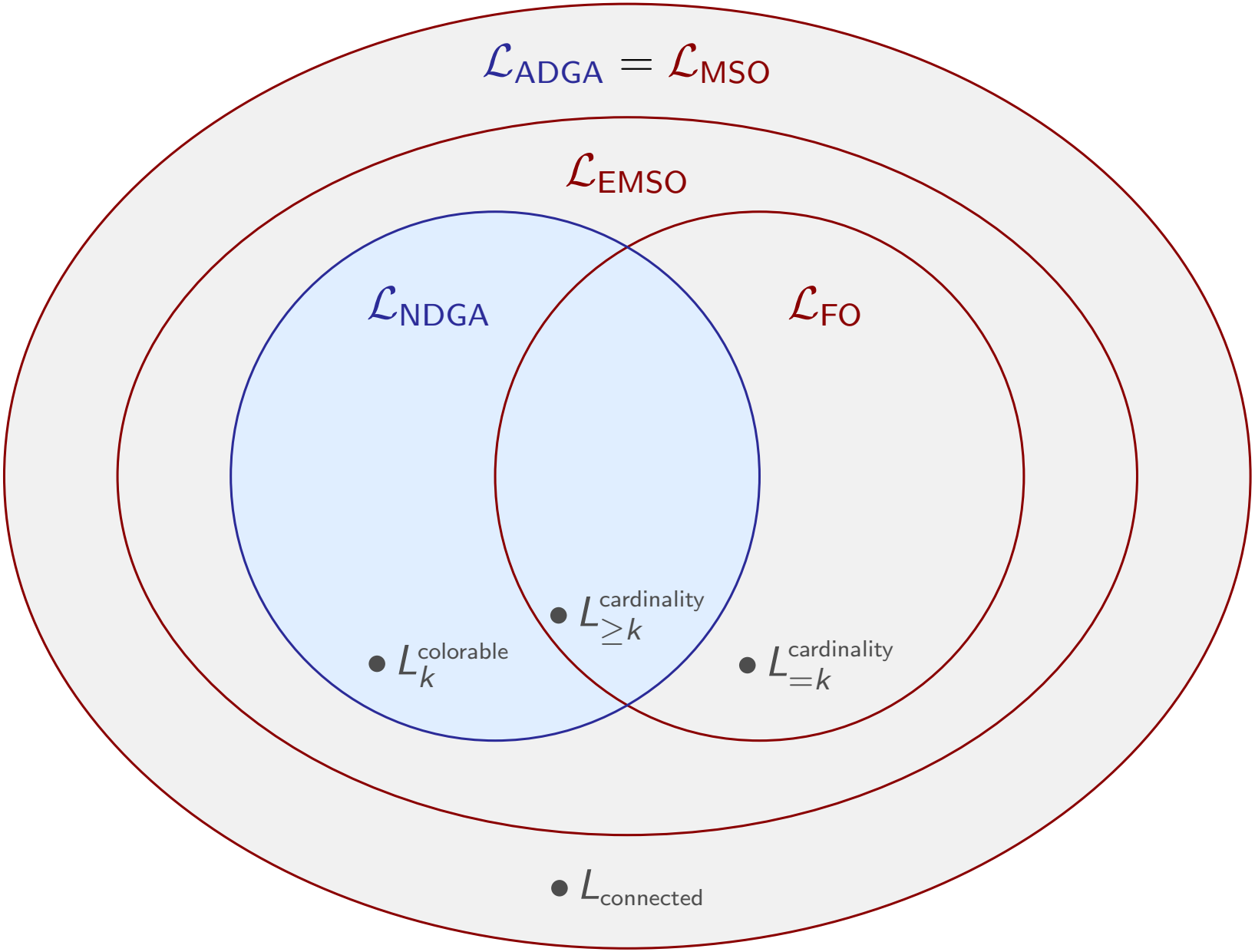
Big Picture


$\mathcal{L}_{\text{ADGA}} = \mathcal{L}_{\text{MSO}}$
$\mathcal{L}_{\text{EMSO}}$
$\mathcal{L}_{\text{FO}}$
$L_k^{\text{colorable}}$
$L_{=k}^{\text{cardinality}}$
$L_{\text{connected}}$

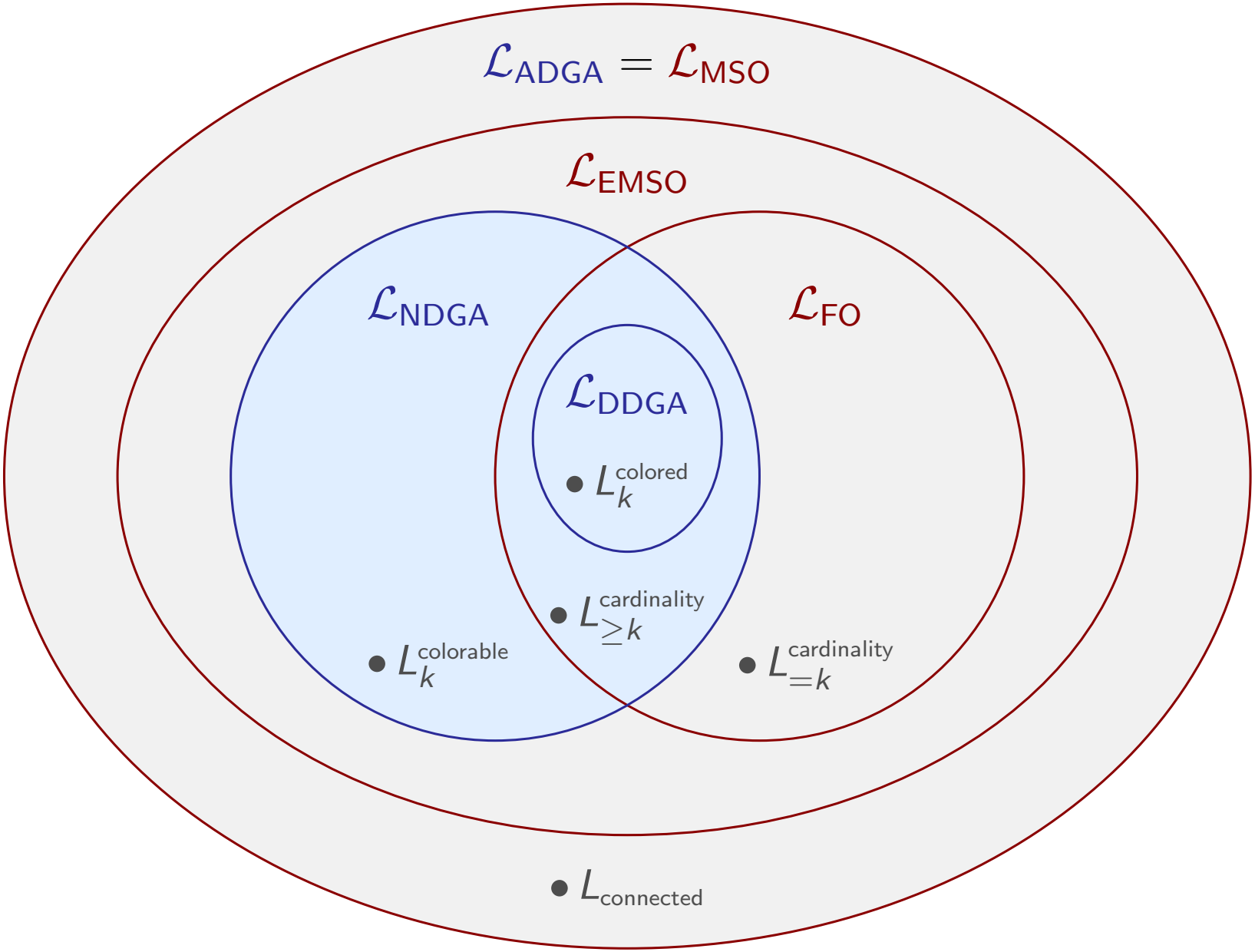
Big Picture


$\mathcal{L}_{\text{ADGA}} = \mathcal{L}_{\text{MSO}}$
$\mathcal{L}_{\text{EMSO}}$
$\mathcal{L}_{\text{NDGA}}$
$\mathcal{L}_{\text{FO}}$
$L_k^{\text{colorable}}$
$L_{=k}^{\text{cardinality}}$
$L_{\text{connected}}$

Big Picture


$\mathcal{L}_{\text{ADGA}} = \mathcal{L}_{\text{MSO}}$
$\mathcal{L}_{\text{EMSO}}$
$\mathcal{L}_{\text{NDGA}}$
$\mathcal{L}_{\text{FO}}$
$L_{\geq k}^{\text{cardinality}}$
$L_k^{\text{colorable}}$
$L_{=k}^{\text{cardinality}}$
$L_{\text{connected}}$

Big Picture


$\mathcal{L}_{\text{ADGA}} = \mathcal{L}_{\text{MSO}}$
$\mathcal{L}_{\text{EMSO}}$
$\mathcal{L}_{\text{NDGA}}$
$\mathcal{L}_{\text{FO}}$
$\mathcal{L}_{\text{DDGA}}$
L_k^{colored}
$L_{\geq k}^{\text{cardinality}}$
$L_k^{\text{colorable}}$
$L_{=k}^{\text{cardinality}}$
$L_{\text{connected}}$

Current Research

Current Research

- Characterization of ADGAs via modal logic

Current Research

- Characterization of ADGAs via modal logic ✓

Current Research

- Characterization of ADGAs via modal logic ✓
- ADGA quantifier alternation hierarchy strict

Current Research

- Characterization of ADGAs via modal logic ✓
- ADGA quantifier alternation hierarchy strict ✓

Current Research

- Characterization of ADGAs via modal logic ✓
- ADGA quantifier alternation hierarchy strict ✓
- Level of this hierarchy that fully covers $\mathcal{L}_{\mathsf{FO}}$

Current Research

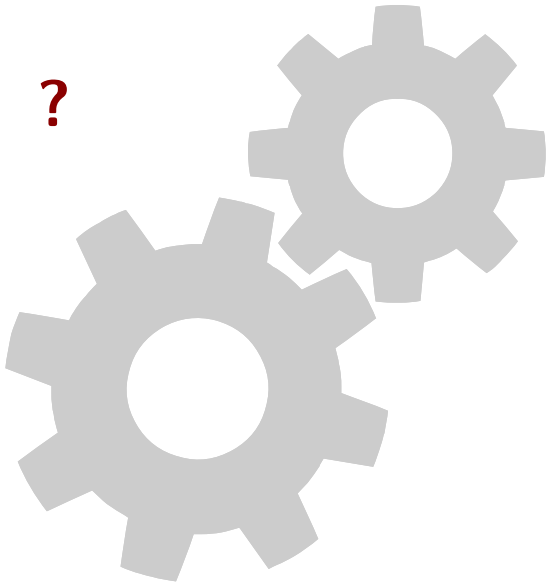
- Characterization of ADGAs via modal logic ✓
- ADGA quantifier alternation hierarchy strict ✓
- Level of this hierarchy that fully covers $\mathcal{L}_{\mathsf{FO}}$?

Current Research

- Characterization of ADGAs via modal logic ✓
- ADGA quantifier alternation hierarchy strict ✓
- Level of this hierarchy that fully covers $\mathcal{L}_{\mathsf{FO}}$?
- Decidability of other levels than Σ_1 (NDGA)

Current Research

- Characterization of ADGAs via modal logic ✓
- ADGA quantifier alternation hierarchy strict ✓
- Level of this hierarchy that fully covers $\mathcal{L}_{\mathsf{FO}}$?
- Decidability of other levels than Σ_1 (NDGA) ?



Thank you!

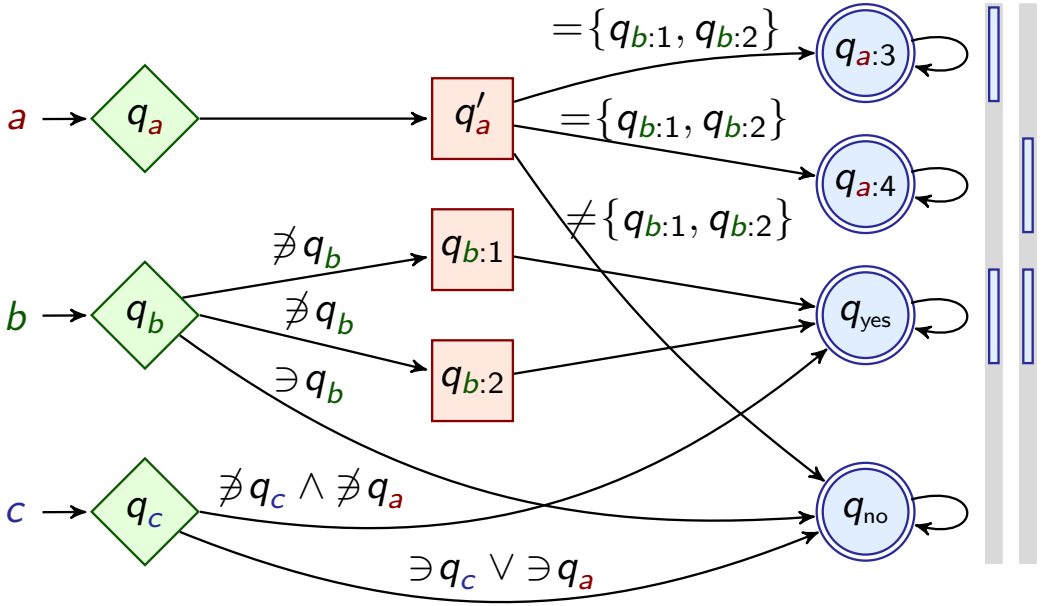
Extra: Monadic Second-Order Logic

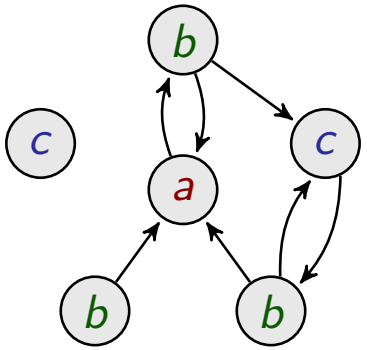
MSO-formulas on graphs over alphabet Σ:

$$\varphi ::= \underbrace{a(u)}_{a \in \Sigma} \mid u \rightharpoonup v \mid u = v \mid u \in U \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists u(\varphi) \mid \exists U(\varphi)$$

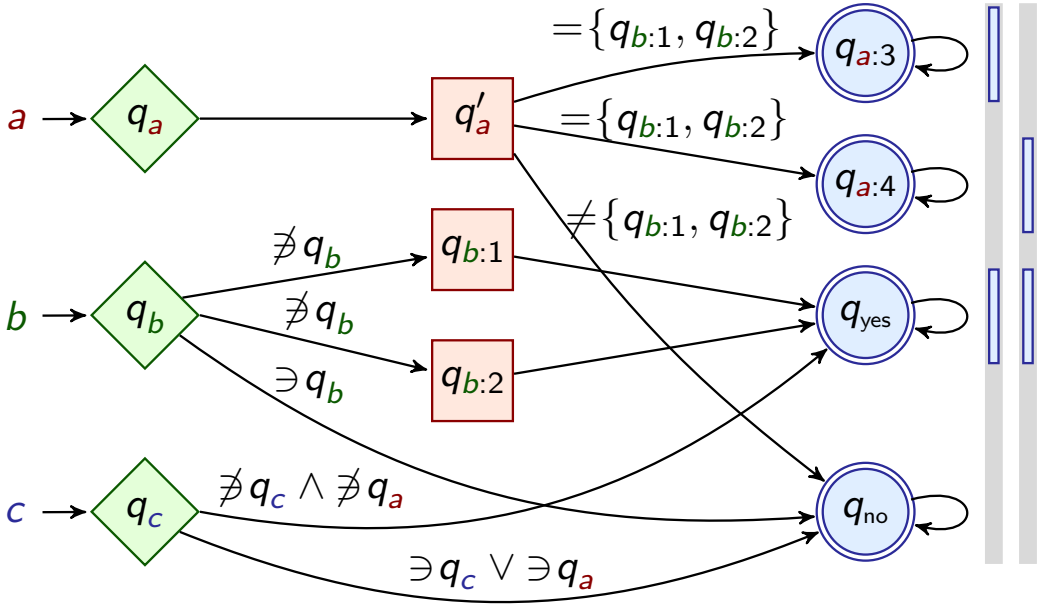
- $u, v, \ldots$: node variables
- $U, V, \ldots$: set variables

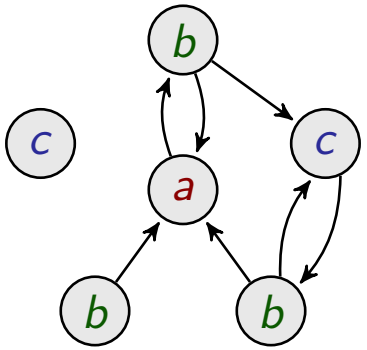
Extra: ADGA – Recognized Graph Language





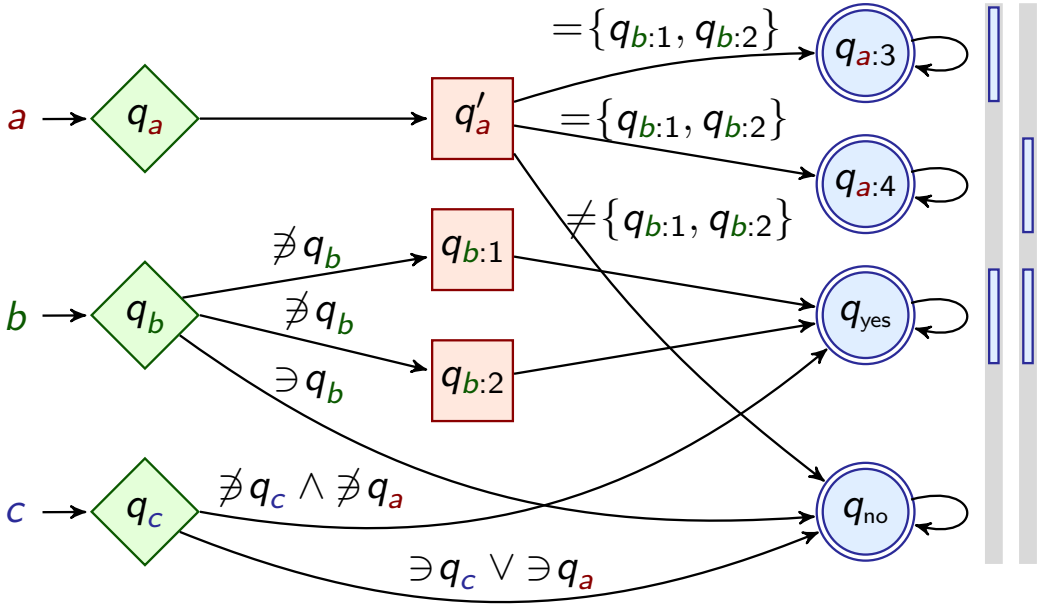
Extra: ADGA – Recognized Graph Language

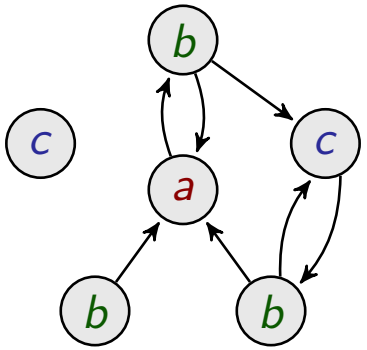




- Precisely one a-labeled node v_a.

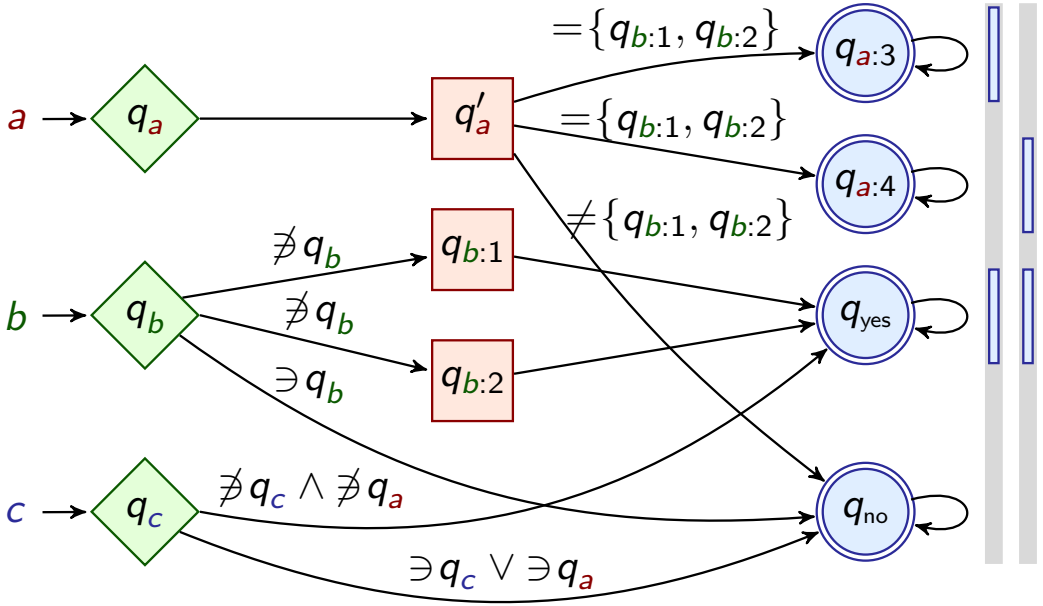
Extra: ADGA – Recognized Graph Language

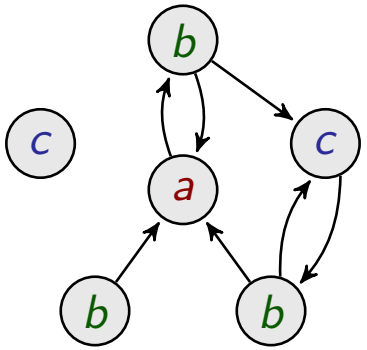




- Precisely one a-labeled node v_a.
- Node labeling constitutes a valid 3-coloring.

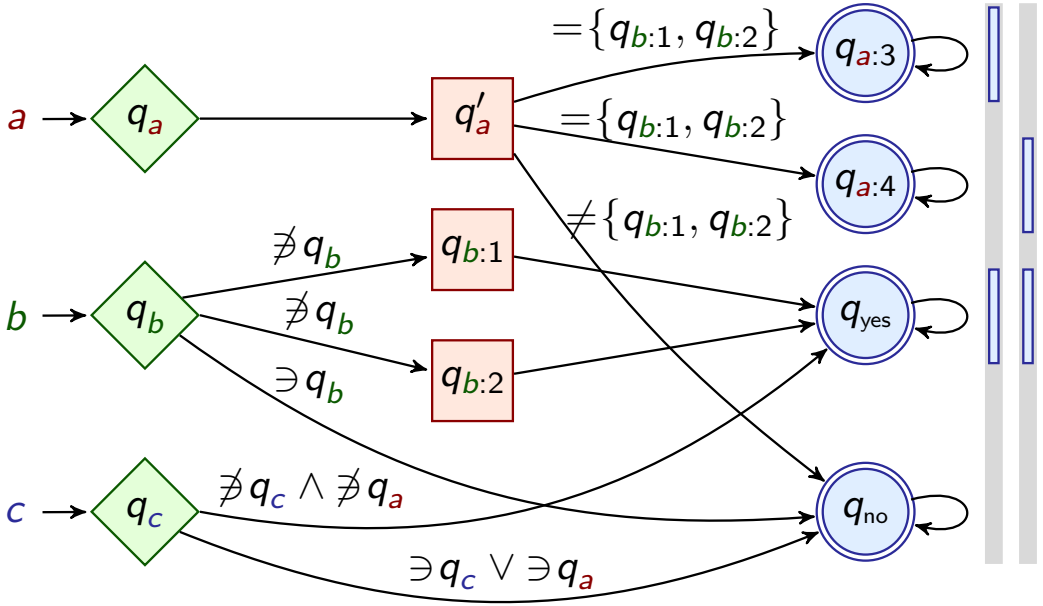
Extra: ADGA – Recognized Graph Language

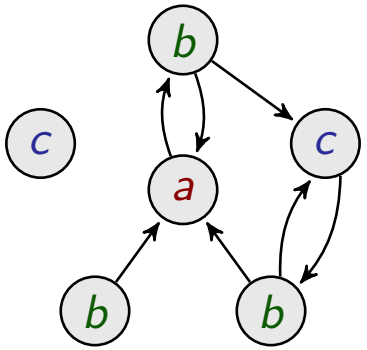




- Precisely one a-labeled node v_a.
- Node labeling constitutes a valid 3-coloring.
- v_a has $\begin{cases} \text{only } b\text{-labeled neighbors,} \\ \\ \end{cases}$

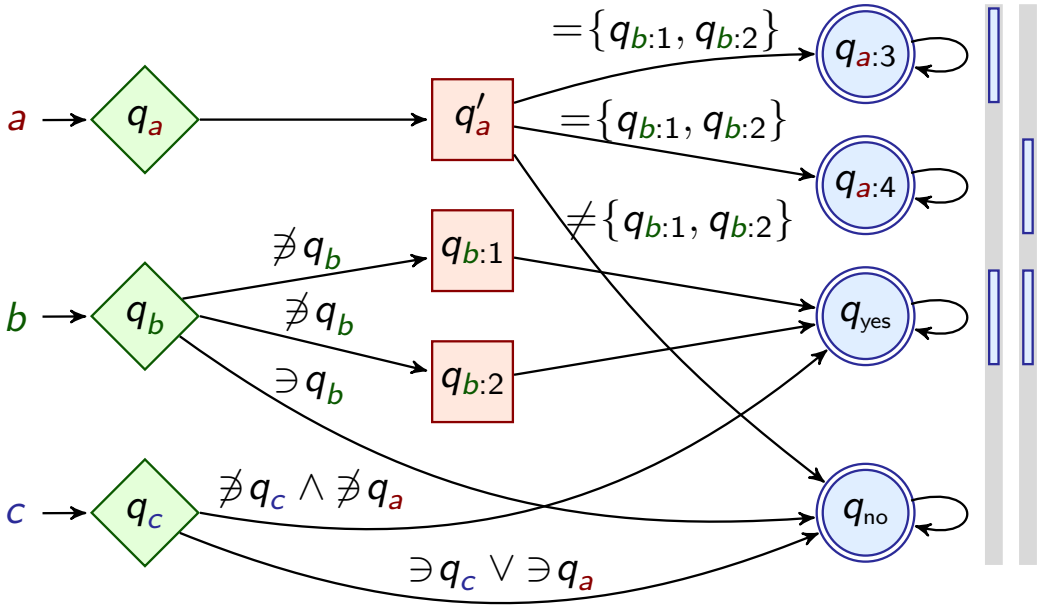
Extra: ADGA – Recognized Graph Language

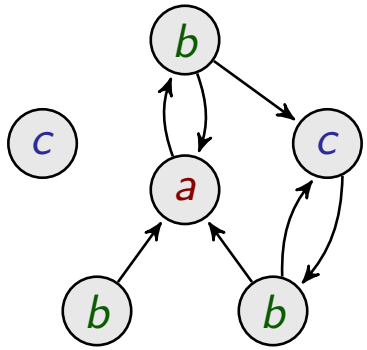




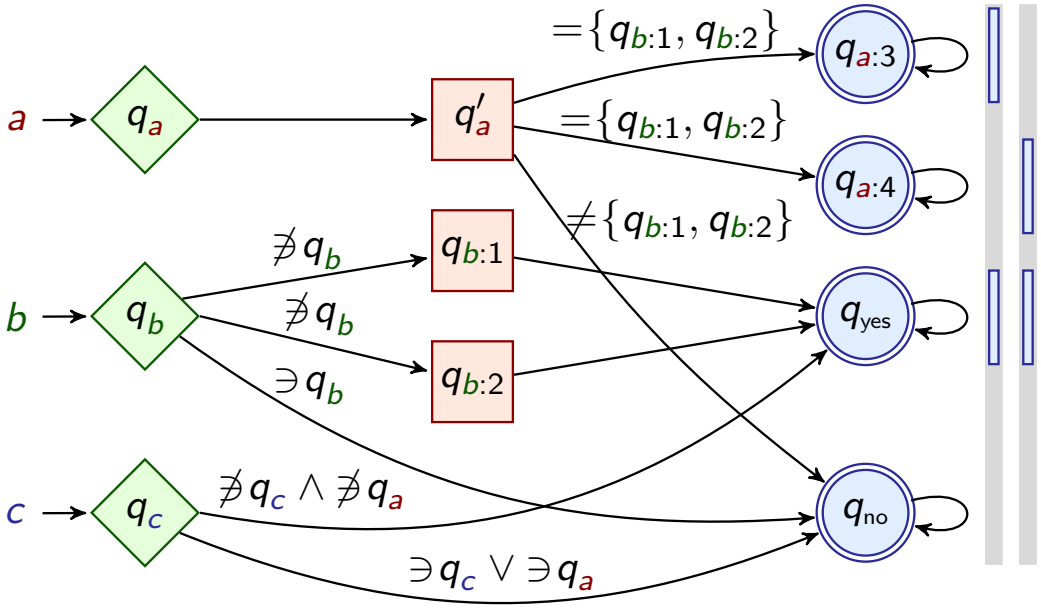
- Precisely one a-labeled node v_a.
- Node labeling constitutes a valid 3-coloring.
- v_a has $\begin{cases} \text{only } b\text{-labeled neighbors,} \\ \text{at least two incoming neighbors.} \end{cases}$

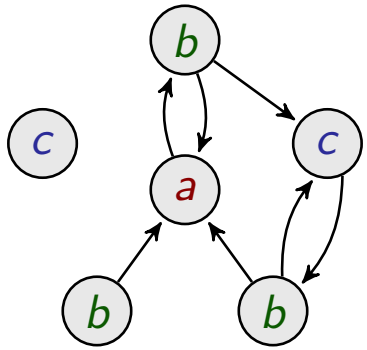
Extra: Example Automaton in MSO-Logic





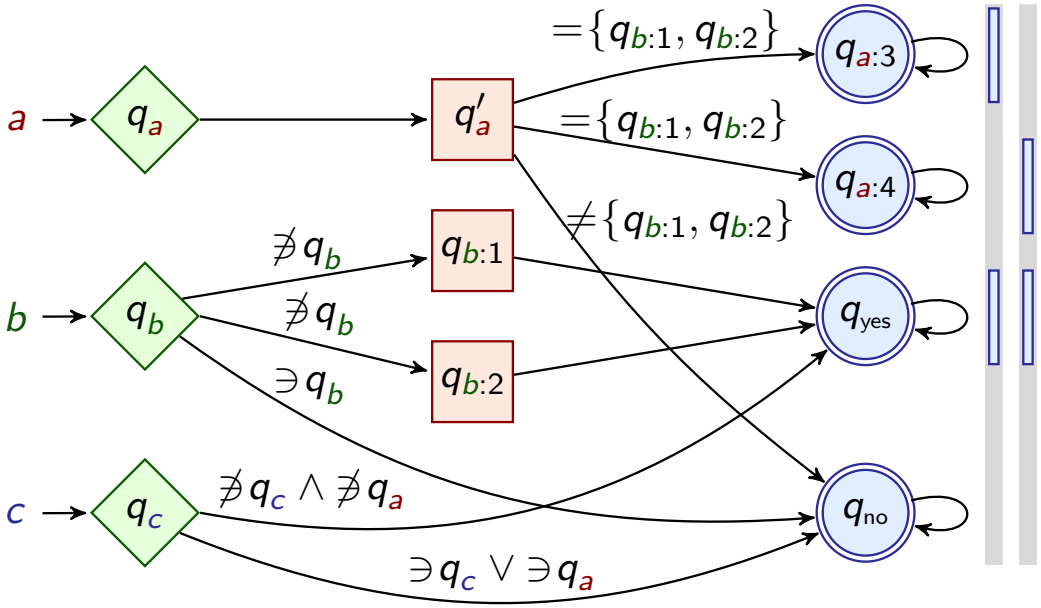
Extra: Example Automaton in MSO-Logic

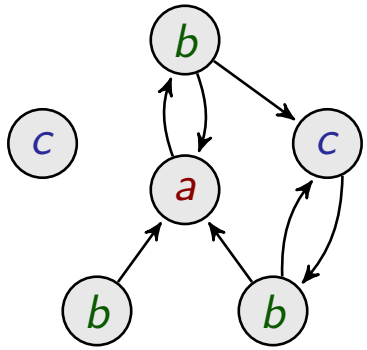




$$\forall u, v \Big(u \rightharpoonup v \;\Rightarrow\; \neg\big(b(u) \wedge b(v)\big) \wedge \neg\big(c(u) \wedge c(v)\big) \Big)$$

Extra: Example Automaton in MSO-Logic

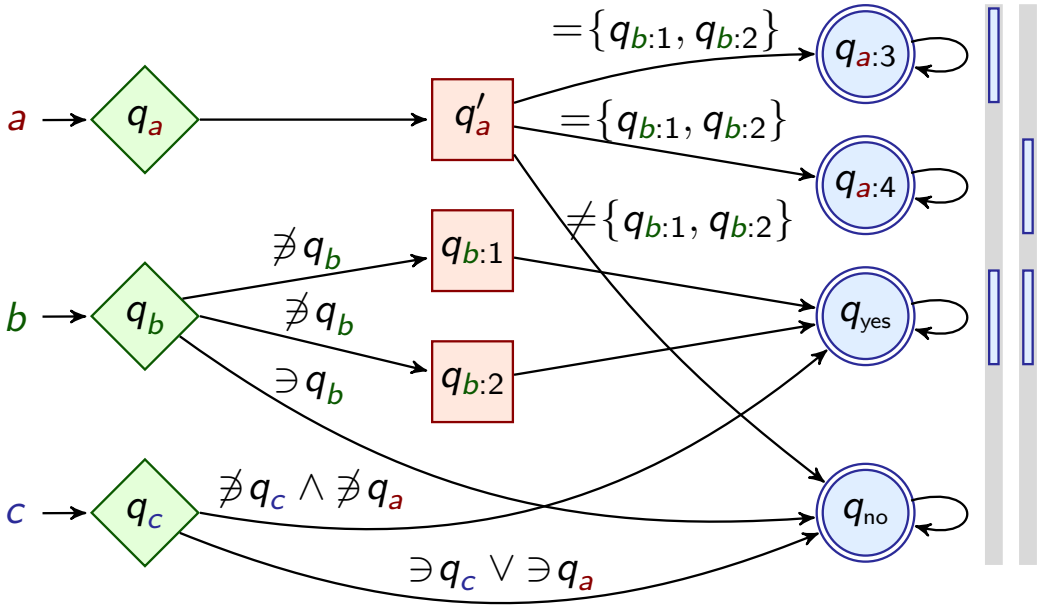


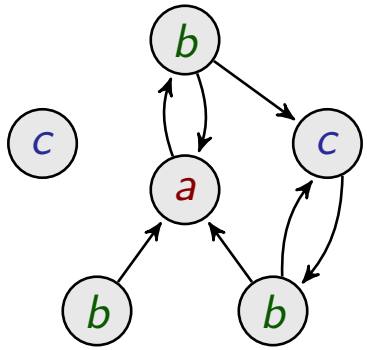


$$\forall u, v \Big(u \rightharpoonup v \;\Rightarrow\; \neg\big(b(u) \wedge b(v)\big) \wedge \neg\big(c(u) \wedge c(v)\big) \Big) \;\wedge$$

$$\exists v_a \Big(\forall u \Big(\big(a(u) \Leftrightarrow u = v_a\big) \Big) \Big)$$

Extra: Example Automaton in MSO-Logic



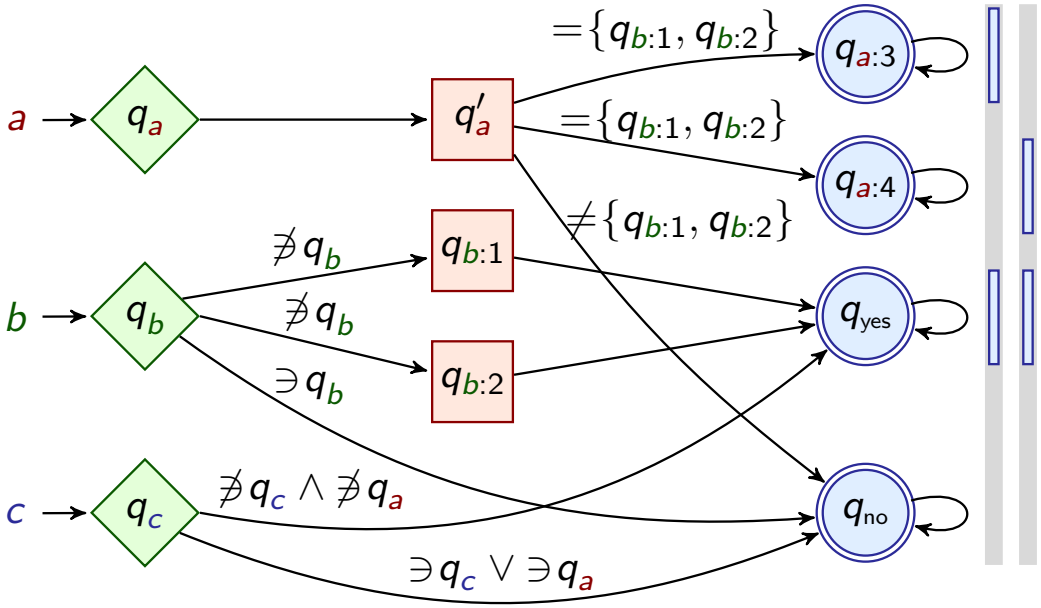


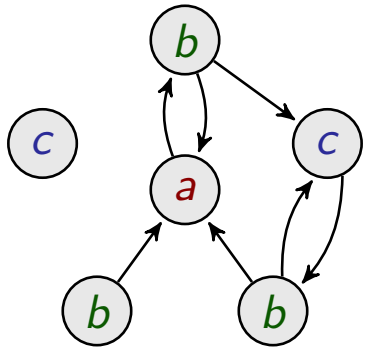
$$\forall u, v \Big(u \rightharpoonup v \;\Rightarrow\; \neg\big(b(u) \wedge b(v)\big) \wedge \neg\big(c(u) \wedge c(v)\big) \Big) \;\wedge$$

$$\exists v_a \Big(\forall u \Big(\big(a(u) \Leftrightarrow u = v_a\big) \;\wedge\; \big(u \rightharpoonup v_a \vee v_a \rightharpoonup u \;\Rightarrow\; b(u)\big) \Big)$$

$$\Big)$$

Extra: Example Automaton in MSO-Logic



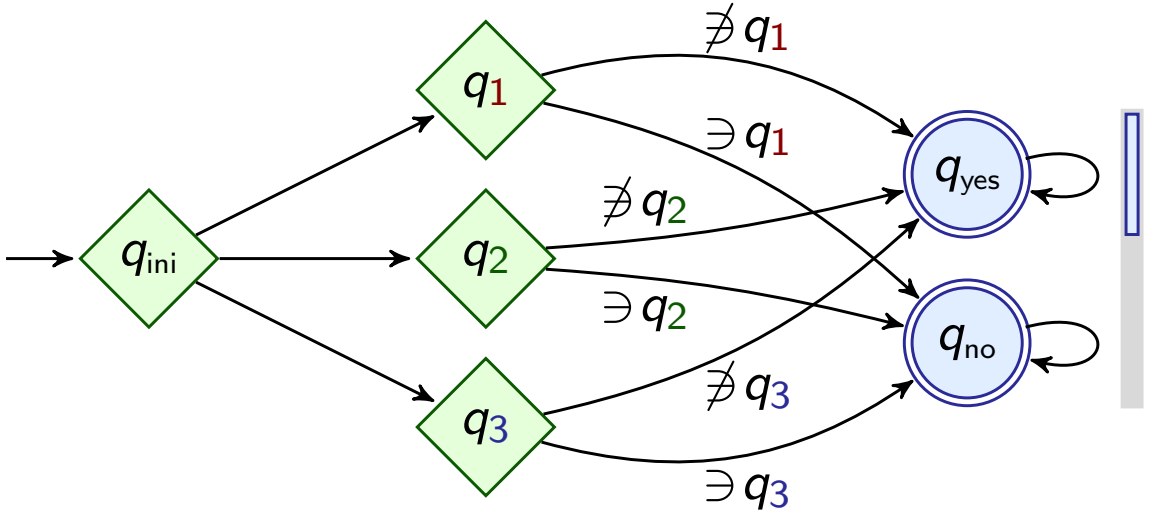


$$\forall u, v \Big(u \rightharpoonup v \;\Rightarrow\; \neg\big(b(u) \wedge b(v)\big) \wedge \neg\big(c(u) \wedge c(v)\big) \Big) \;\wedge$$

$$\exists v_a \Big(\forall u \Big(\big(a(u) \Leftrightarrow u = v_a\big) \;\wedge\; \big(u \rightharpoonup v_a \vee v_a \rightharpoonup u \;\Rightarrow\; b(u)\big) \Big) \;\wedge$$

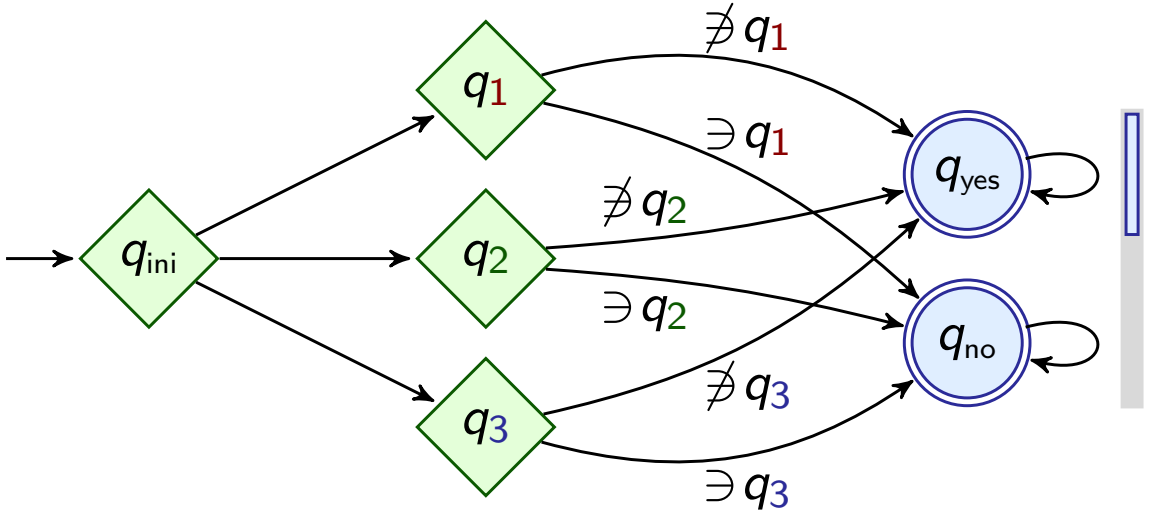
$$\exists u_1, u_2 \Big(u_1 \rightharpoonup v_a \;\wedge\; u_2 \rightharpoonup v_a \;\wedge\; \neg\, u_1 = u_2 \Big) \Big)$$

Extra: 3-Colorability in MSO-Logic



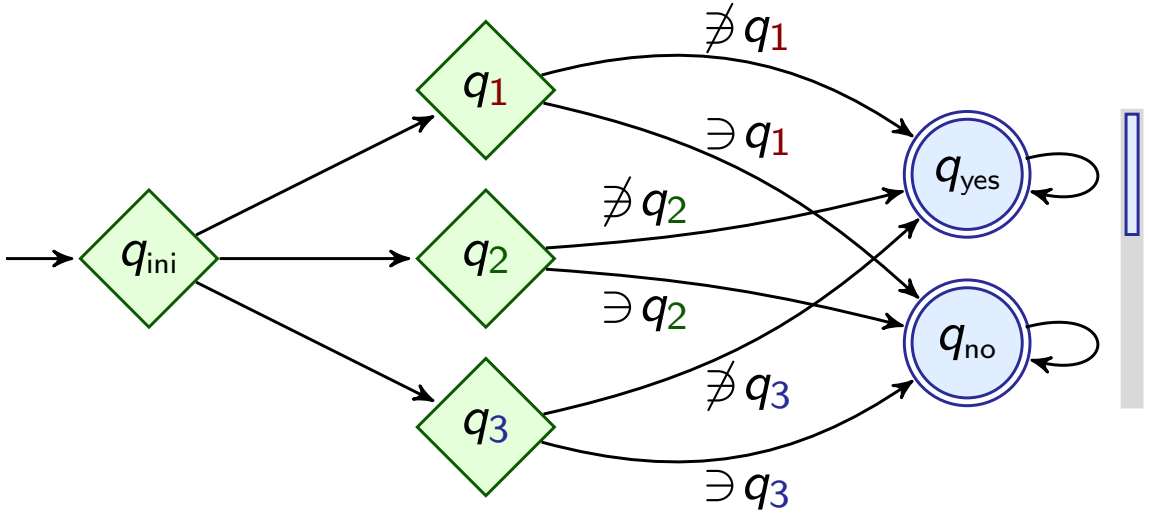
$$\exists U_1, U_2, U_3 \Big(\qquad \Big)$$

Extra: 3-Colorability in MSO-Logic



$$\exists U_1, U_2, U_3 \Big(\forall u \Big(\big(u \in U_1 \vee u \in U_2 \vee u \in U_3\big) \wedge \neg\big(u \in U_1 \wedge u \in U_2\big) \wedge \neg\big(u \in U_1 \wedge u \in U_3\big) \wedge \neg\big(u \in U_2 \wedge u \in U_3\big) \Big) \Big)$$

Extra: 3-Colorability in MSO-Logic



$$\exists U_1, U_2, U_3 \Big(\forall u \Big(\big(u \in U_1 \vee u \in U_2 \vee u \in U_3\big) \wedge \neg\big(u \in U_1 \wedge u \in U_2\big) \wedge \neg\big(u \in U_1 \wedge u \in U_3\big) \wedge \neg\big(u \in U_2 \wedge u \in U_3\big) \Big) \wedge \forall u, v \Big(u \rightharpoonup v \Rightarrow \neg\big(u \in U_1 \wedge v \in U_1\big) \wedge \neg\big(u \in U_2 \wedge v \in U_2\big) \wedge \neg\big(u \in U_3 \wedge v \in U_3\big) \Big) \Big)$$