

# Logics on words

Regular expressions give operational descriptions of regular languages.

Often the natural description of a language is denotational:

- even number of a's and even number of b's  
 $(aa+bb+(ab+ba)(aa+bb)^*(ba+ab))^*$
- words not containing "hello"

Goal: find a denotational language able to express all the regular languages, and only the regular languages.

# Logics on words

Idea: use a logic that has an interpretation on words

A formula expresses a property that each word may satisfy or not, like

" the word contains only a's "

" the word has even length "

" between every occurrence of an 'a' and a 'b' there is an occurrence of a 'c' "

Every formula (indirectly) defines a language: the language of all the words (over the given fixed alphabet) that satisfy it.

# First-order logic on words

Atomic formulas: for each letter  $a$  we introduce the formula  $Q_a(x)$ , with intuitive meaning: the letter at position  $x$  (starting at 0) is an "a".

Formulas:

**Definition 7.1** *Let  $V = \{x, y, z, \dots\}$  be an infinite sets of variables, and let  $\Sigma = \{a, b, c, \dots\}$  be a finite alphabet. The set  $FO(\Sigma)$  of first-order formulas over  $\Sigma$  is the set of expressions generated by the grammar:*

$$\varphi := Q_a(x) \mid x < y \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \exists x\varphi$$

where  $a \in \Sigma$ .

Observe: the logic is parametrized by the alphabet.

# Abbreviations

- $\forall x\varphi := \neg\exists x\neg\varphi$
- $\varphi_1 \wedge \varphi_2 := \neg(\neg\varphi_1 \vee \neg\varphi_2)$
- $\varphi_1 \rightarrow \varphi_2 := \neg\varphi_1 \vee \varphi_2$

first( $x$ ) :=

last( $x$ ) :=

$y = x + 1$  :=

$y = x + 2$  :=

$y = x + (k + 1)$  :=

# Examples (without formal semantics yet)

- “The last letter is a b and before it there are only a’s.”

$$\exists Q_b(x) \wedge \forall x (\text{last}(x) \rightarrow Q_b(x) \wedge \neg \text{last}(x) \rightarrow Q_a(x))$$

- “Every a is immediately followed by a b.”

$$\forall x (Q_a(x) \rightarrow \exists y (y = x + 1 \wedge Q_b(y)))$$

- “Every a is immediately followed by a b, unless it is the last letter.”

$$\forall x (Q_a(x) \rightarrow \forall y (y = x + 1 \rightarrow Q_b(y)))$$

- “Between every a and every later b there is a c.”

$$\forall x \forall y (Q_a(x) \wedge Q_b(y) \wedge x < y \rightarrow \exists z (x < z \wedge z < y \wedge Q_c(z)))$$

# Semantics

**Definition 9.3** *The satisfaction relation  $(w, \mathcal{I}) \models \varphi$  between a formula  $\varphi$  of  $FO(\Sigma)$  and an interpretation  $(w, \mathcal{I})$  of  $\varphi$  is defined by:*

$$(w, \mathcal{I}) \models Q_a(x) \quad \text{iff} \quad w[\mathcal{I}(x)] = a$$

$$(w, \mathcal{I}) \models x < y \quad \text{iff} \quad \mathcal{I}(x) < \mathcal{I}(y)$$

$$(w, \mathcal{I}) \models \exists x \varphi \quad \text{iff} \quad |w| \geq 1 \text{ and some } i \in \{1, \dots, |w|\} \text{ satisfies } (w, \mathcal{I}[i/x]) \models \varphi$$

where  $w[i]$  is the letter of  $w$  at position  $i$ , and  $\mathcal{I}[i/x]$  is the interpretation that assigns  $i$  to  $x$  and otherwise coincides with  $\mathcal{I}$ . (Notice that  $\mathcal{I}$  may not assign any value to  $x$ .) If  $(w, \mathcal{I}) \models \varphi$  we say that  $(w, \mathcal{I})$  is a model of  $\varphi$ . Two formulas are equivalent if they have the same models.

# The empty word ...

... is as usual a pain :-)

It satisfies all universally quantified formulas, and no existentially quantified formula.

# **Can we only express regular languages?**

## **Can we express all regular languages?**

We consider one-letter alphabets.

Proposition: a language over a one-letter alphabet is expressible in FO logic iff it is finite or co-finite (its complement is finite)

Consequence: we can only express regular languages, but not all, not even the language of words of even length.

So we extend the logic ...

# Monadic second-order logic

First-order variables: interpreted on positions

Monadic second-order variables: interpreted on sets of positions.

Diadic second-order variables: interpreted on relations over positions

Monadic third-order variables: interpreted on sets of sets of positions

New atomic formulas:  $x \in X$

# Expressing that a word has even length

- Idea: express "there is a set  $X$  of positions such that
- $X$  contains exactly the even positions, and
  - the last position belongs to  $X$ "

**Definition 9.10** Let  $X_1 = \{x, y, z, \dots\}$  and  $X_2 = \{X, Y, Z, \dots\}$  be two infinite sets of first-order and second-order variables. Let  $\Sigma = \{a, b, c, \dots\}$  be a finite alphabet. The set  $MSO(\Sigma)$  of monadic second-order formulas over  $\Sigma$  is the set of expressions generated by the grammar:

$$\varphi := Q_a(x) \mid x < y \mid x \in X \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists x \varphi \mid \exists X \varphi$$

An interpretation of a formula  $\varphi$  is a pair  $(w, \mathcal{I})$  where  $w \in \Sigma^*$ , and  $\mathcal{I}$  is a mapping that assigns every free first-order variable  $x$  a position  $\mathcal{I}(x) \in \{1, \dots, |w|\}$  and every free second-order variable  $X$  a set of positions  $\mathcal{I}(X) \subseteq \{1, \dots, |w|\}$ . (The mapping may also assign positions to other variables.)

The satisfaction relation  $(w, \mathcal{I}) \models \varphi$  between a formula  $\varphi$  of  $\text{MSO}(\Sigma)$  and an interpretation  $(w, \mathcal{I})$  of  $\varphi$  is defined as for  $\text{FO}(\Sigma)$ , with the following additions:

$$\begin{aligned} (w, \mathcal{I}) \models x \in X & \quad \text{iff} \quad \mathcal{I}(x) \in \mathcal{I}(X) \\ (w, \mathcal{I}) \models \exists X \varphi & \quad \text{iff} \quad |w| > 0 \text{ and some } S \subseteq \{0, \dots, |w| - 1\} \\ & \quad \text{satisfies } (w, \mathcal{I}[S/X]) \models \varphi \end{aligned}$$

where  $\mathcal{I}[S/X]$  is the interpretation that assigns  $S$  to  $X$  and otherwise coincides with  $\mathcal{I}$  — whether  $\mathcal{I}$  is defined for  $X$  or not. If  $(w, \mathcal{I}) \models \varphi$  we say that  $(w, \mathcal{I})$  is a model of  $\varphi$ . Two formulas are equivalent if they have the same models. The language  $L(\varphi)$  of a sentence  $\varphi \in \text{MSO}(\Sigma)$  is the set  $L(\varphi) = \{w \in \Sigma^* \mid w \models \varphi\}$ . A language  $L \subseteq \Sigma^*$  is MSO-definable if  $L = L(\varphi)$  for some formula  $\varphi \in \text{MSO}(\Sigma)$ .

## Expressing $c^*(ab)^*d^*$

There is a block of consecutive positions  $X$  such that: before  $X$  there are only  $c$ 's; after  $X$  there are only  $d$ 's; in  $X$   $b$ 's and  $a$ 's alternate; the first letter in  $X$  is an  $a$  and the last is a  $b$ .

$$\exists X( \text{Cons}(X) \wedge \text{Before\_only\_c}(X) \wedge \text{After\_only\_d}(X) \wedge \\ \text{Alternate}(X) \wedge \text{First\_a}(X) \wedge \text{Last\_b}(X) )$$

- “ $X$  is a block of consecutive positions.”

- “Before  $X$  there are only c’s.”
- “After  $X$  there are only d’s.”

- “a’s and b’s alternate in  $X$ .”
- the first letter in  $X$  is an a and the last is a b.

# Every regular language is expressible in MSO logic

Goal: given an arbitrary regular language  $L$ , construct an MSO sentence having  $L$  as language

We use: if  $L$  is regular, then there is a DFA  $A$  recognizing  $L$ .

Idea: construct a formula expressing  
"the run of  $A$  on this word is accepting"

# Every regular language is expressible in MSO logic

**Proposition 9.12** *If  $L \subseteq \Sigma^*$  is regular, then  $L$  is expressible in  $\text{MSO}(\Sigma)$ .*

**Proof:** Let  $A = (Q, \Sigma, \delta, q_0, F)$  be a DFA with  $Q = \{q_0, \dots, q_n\}$  and  $L(A) = L$ . We construct a formula  $\varphi_A$  such that for every  $w \neq \epsilon$ ,  $w \models \varphi_A$  iff  $w \in L(A)$ . If  $\epsilon \in L(A)$ , then we can extend the formula to  $\varphi_A \vee \varphi'_A$ , where  $\varphi'_A$  is only satisfied by the empty word (e.g.  $\varphi'_A = \forall x \, x < x$ ).

We start with some notations. Let  $w = a_1 \dots a_m$  be a word over  $\Sigma$ , and let

$$P_q = \{i \in \{1, \dots, m\} \mid \hat{\delta}(q_0, a_0 \dots a_i) = q\} .$$

In words,  $i \in P_q$  iff  $A$  is in state  $q$  immediately *after* reading the letter  $a_i$ . Then  $A$  accepts  $w$  iff  $m \in \bigcup_{q \in F} P_q$ .

Assume we were able to construct formula  $\text{Visits}(X_0, \dots X_n)$  with free variables  $X_0, \dots X_n$  such that  $\mathcal{I}(X_i) = P_{q_i}$  holds for *every* model  $(w, \mathcal{I})$  and for every  $0 \leq i \leq n$ . In words,  $\text{Visits}(X_0, \dots X_n)$  is only true when  $X_i$  takes the value  $P_{q_i}$  for every  $0 \leq i \leq n$ . Then  $(w, \mathcal{I})$  would be a model of

$$\psi_A := \exists X_0 \dots \exists X_n \text{Visits}(X_0, \dots X_n) \wedge \exists x \left( \text{last}(x) \wedge \bigvee_{q_i \in F} x \in X_i \right)$$

iff  $w$  has a last letter, and  $w \in L$ . So we could take

$$\varphi_A := \begin{cases} \psi_A & \text{if } q_0 \notin F \\ \psi_A \vee \forall x \, x < x & \text{if } q_0 \in F \end{cases}$$

Let us now construct the formula  $\text{Visits}(X_0, \dots X_n)$ . The sets  $P_q$  are the unique sets satisfying the following properties:

- (a)  $1 \in P_{\delta(q_0, a_1)}$ , i.e., after reading the letter at position 1 the DFA is in state  $\delta(q_0, a_1)$ ;
- (b) every position  $i$  belongs to exactly one  $P_q$ , i.e., the  $P_q$ 's build a partition of the set positions; and
- (c) if  $i \in P_q$  and  $\delta(q, a_{i+1}) = q'$  then  $i + 1 \in P_{q'}$ , i.e., the  $P_q$ 's “respect” the transition function  $\delta$ .

We express these properties through formulas. For every  $a \in \Sigma$ , let  $q_{i_a} = \delta(q_0, a)$ . The formula for (a) is:

$$\text{Init}(X_0, \dots, X_n) = \exists x \left( \text{first}(x) \wedge \left( \bigvee_{a \in \Sigma} (Q_a(x) \wedge x \in X_{i_a}) \right) \right)$$

(in words: if the letter at position 1 is  $a$ , then the position belongs to  $X_{i_a}$ ).

Formula for (b):

$$\text{Partition}(X_0, \dots, X_n) = \forall x \left( \bigvee_{i=0}^n x \in X_i \wedge \bigwedge_{\substack{i, j=0 \\ i \neq j}}^n (x \in X_i \rightarrow x \notin X_j) \right)$$

Formula for (c):

$$\text{Respect}(X_0, \dots, X_n) = \forall x \forall y \left( y = x + 1 \rightarrow \bigvee_{\substack{a \in \Sigma \\ i, j \in \{0, \dots, n\} \\ \delta(q_i, a) = q_j}} (x \in X_i \wedge Q_a(x) \wedge y \in X_j) \right)$$

Altogether we get

$$\text{Visits}(X_0, \dots, X_n) := \text{Init}(X_0, \dots, X_n) \wedge \text{Partition}(X_0, \dots, X_n) \wedge \text{Respect}(X_0, \dots, X_n)$$

# Every language expressible in MSO logic is regular

An interpretation of a formula is a pair consisting of a word and assignments to the free first and second order variables (and perhaps to others).

$$\left( aab, \begin{array}{l} x \mapsto 0 \\ y \mapsto 2 \\ X \mapsto \{1, 2\} \\ Y \mapsto \{0, 1\} \end{array} \right) \quad \left( ba, \begin{array}{l} x \mapsto 1 \\ y \mapsto 0 \\ X \mapsto \emptyset \\ Y \mapsto \{0\} \end{array} \right)$$

We encode interpretations of the free variables as words:

$$\left( aab, \begin{array}{l} x \mapsto 0 \\ y \mapsto 2 \\ X \mapsto \{1, 2\} \\ Y \mapsto \{0, 1\} \end{array} \right) \quad \left( ba, \begin{array}{l} x \mapsto 1 \\ y \mapsto 0 \\ X \mapsto \emptyset \\ Y \mapsto \{0\} \end{array} \right)$$

	<i>a</i>	<i>a</i>	<i>b</i>
<i>x</i>	1	0	0
<i>y</i>	0	0	1
<i>X</i>	0	1	1
<i>Y</i>	1	1	0



	<i>b</i>	<i>a</i>
<i>x</i>	0	1
<i>y</i>	1	0
<i>X</i>	0	0
<i>Y</i>	1	0

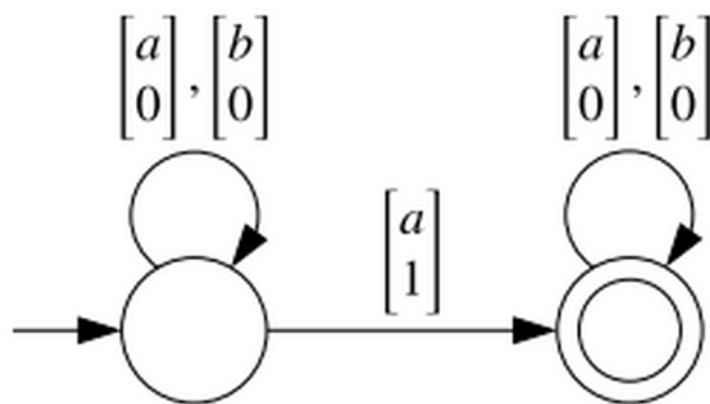
**Definition 7.13** *Let  $\varphi$  be a formula with  $n$  free variables, and let  $(w, \mathcal{I})$  be an interpretation of  $\varphi$ . We denote by  $\text{enc}(w, \mathcal{I})$  the word over the alphabet  $\Sigma \times \{0, 1\}^n$  described above. The language of  $\varphi$  is  $\mathcal{L}(\varphi) = \{\text{enc}(w, \mathcal{I}) \mid (w, \mathcal{I}) \models \varphi\}$ .*

we prove by induction on the structure of  $\varphi$  that  $\mathcal{L}(\varphi)$  is regular.

- $\varphi = Q_a(x)$ . Then  $free(\varphi) = x$ , and the interpretations of  $\varphi$  are encoded as words over  $\Sigma \times \{0, 1\}$ . The language  $\mathcal{L}(\varphi)$  is given by

$$\mathcal{L}(\varphi) = \left\{ \begin{bmatrix} a_1 \\ b_1 \end{bmatrix} \cdots \begin{bmatrix} a_k \\ b_k \end{bmatrix} \mid \begin{array}{l} k \geq 0, \\ a_i \in \Sigma \text{ and } b_i \in \{0, 1\} \text{ for every } i \in \{1, \dots, k\}, \text{ and} \\ b_i = 1 \text{ for exactly one index } i \in \{1, \dots, k\} \end{array} \right\}$$

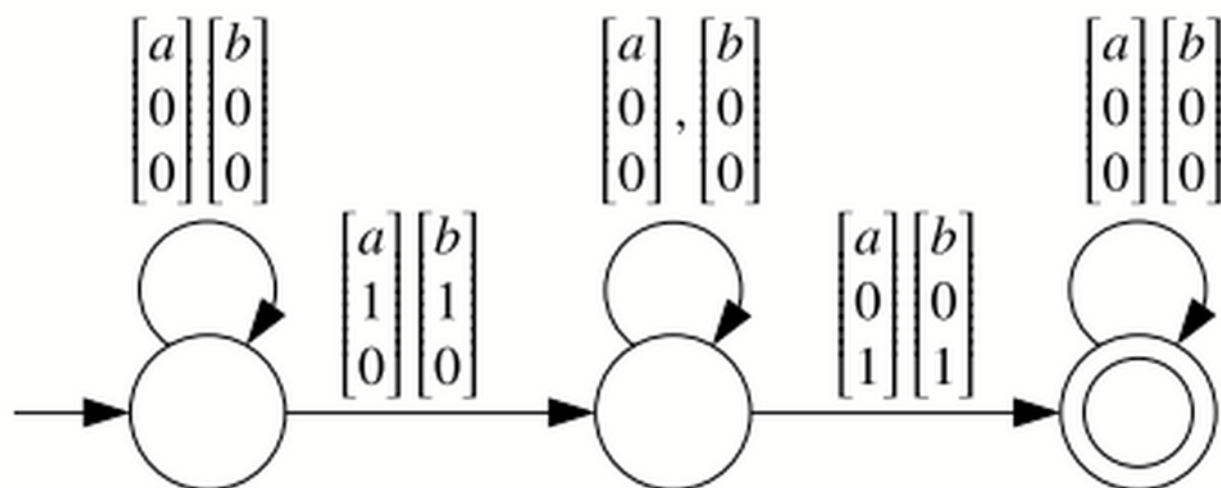
and is recognized by



- $\varphi = x < y$ . Then  $free(\varphi) = \{x, y\}$ , and interpretations are encoded as words over  $\Sigma \times \{0, 1\}^2$ . The language  $\mathcal{L}(\varphi)$  is given by

$$\mathcal{L}(\varphi) = \left\{ \begin{bmatrix} a_1 \\ b_1 \\ c_1 \end{bmatrix} \cdots \begin{bmatrix} a_k \\ b_k \\ c_k \end{bmatrix} \mid \begin{array}{l} k \geq 0, \\ a_i \in \Sigma \text{ and } b_i, c_i \in \{0, 1\} \text{ for every } i \in \{1, \dots, k\}, \\ b_i = 1 \text{ for exactly one index } i \in \{1, \dots, k\}, \\ c_j = 1 \text{ for exactly one index } j \in \{1, \dots, k\}, \text{ and} \\ i < j \end{array} \right\}$$

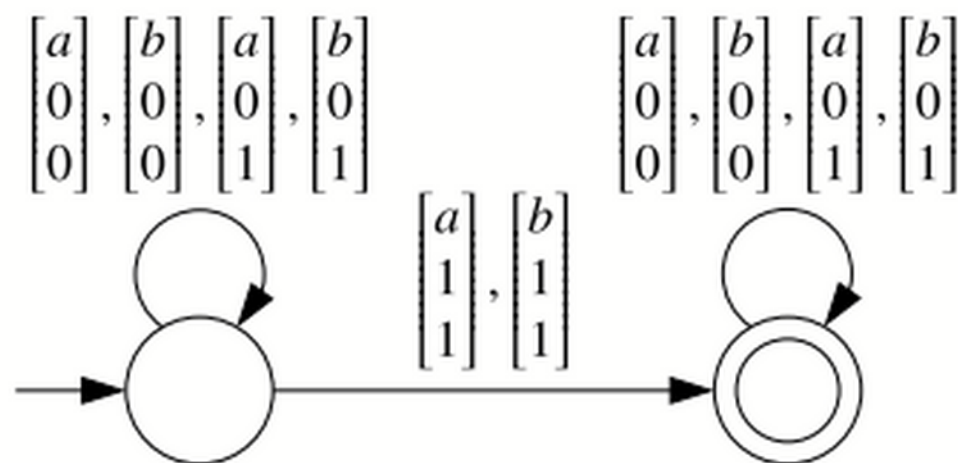
and is recognized by



- $\varphi = x \in X$ . Then  $free(\varphi) = \{x, X\}$ , and interpretations are encoded as words over  $\Sigma \times \{0, 1\}^2$ . The language  $\mathcal{L}(\varphi)$  is given by

$$\mathcal{L}(\varphi) = \left\{ \begin{bmatrix} a_1 \\ b_1 \\ c_1 \end{bmatrix} \cdots \begin{bmatrix} a_k \\ b_k \\ c_k \end{bmatrix} \mid \begin{array}{l} k \geq 0, \\ a_i \in \Sigma \text{ and } b_i, c_i \in \{0, 1\} \text{ for every } i \in \{1, \dots, k\}, \\ b_i = 1 \text{ for exactly one index } i \in \{1, \dots, k\}, \text{ and} \\ \text{for every } i \in \{1, \dots, k\}, \text{ if } b_i = 1 \text{ then } c_i = 1 \end{array} \right\}$$

and is recognized by



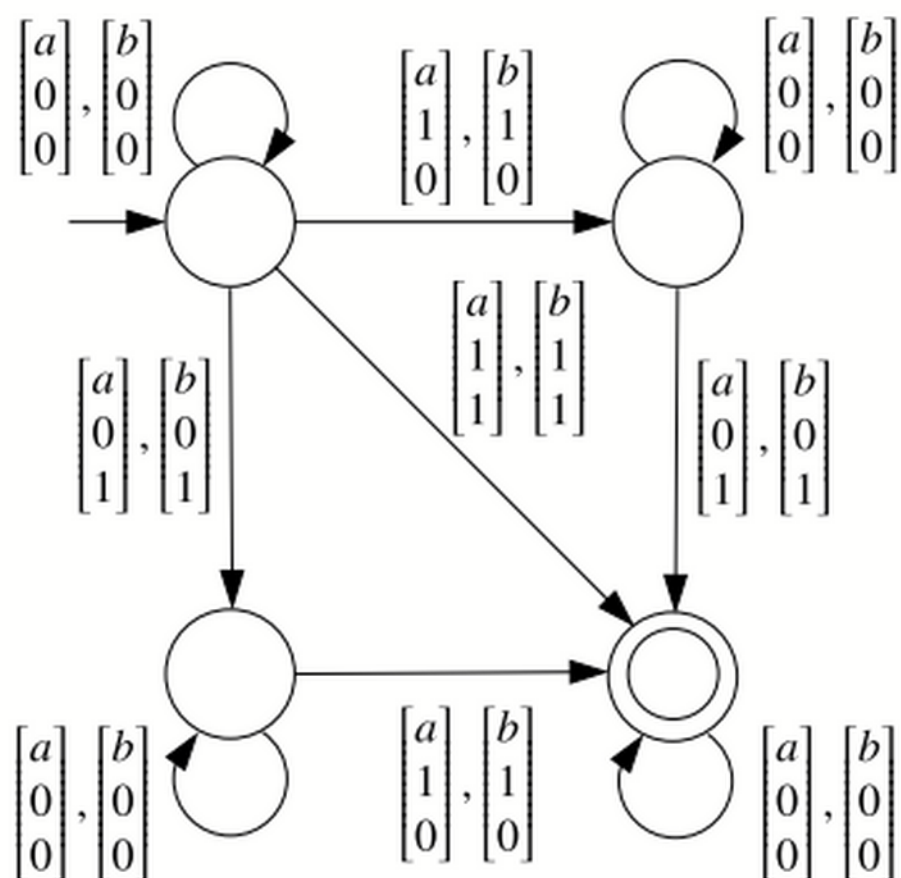
- $\varphi = \neg\psi$ . Then  $free(\varphi) = free(\psi)$ , and by induction hypothesis there exists an automaton  $A_\psi$  s.t.  $\mathcal{L}(A_\psi) = \mathcal{L}(\psi)$ .

Observe that  $\mathcal{L}(\varphi)$  is *not* in general equal to  $\overline{\mathcal{L}(\psi)}$ . To see why, consider for example the case  $\psi = Q_a(x)$  and  $\varphi = \neg Q_a(x)$ . The word

$$\begin{bmatrix} a \\ 1 \end{bmatrix} \begin{bmatrix} a \\ 1 \end{bmatrix} \begin{bmatrix} a \\ 1 \end{bmatrix}$$

belongs neither to  $\mathcal{L}(\psi)$  nor  $\mathcal{L}(\varphi)$ , because it is not the encoding of any interpretation: the bitstring for  $x$  contains more than one 1. What holds is  $\mathcal{L}(\varphi) = \overline{\mathcal{L}(\psi) \cap Enc(\psi)}$ , where  $Enc(\psi)$  is the language of the encodings of all the interpretations of  $\psi$  (whether they are models of  $\psi$  or not). We construct an automaton  $A_\psi^{enc}$  recognizing  $Enc(\psi)$ , and so we can take  $A_\varphi = A_\psi \times_{\cap} A_\psi^{enc}$ .

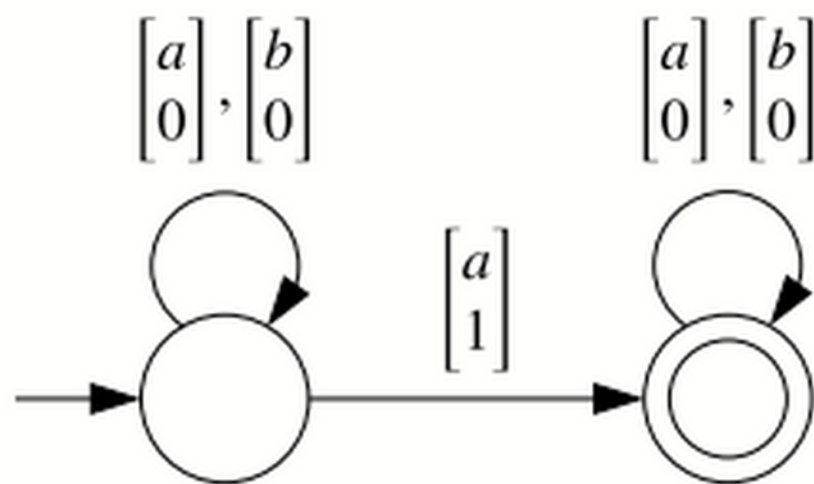
Assume  $\psi$  has  $k$  first-order variables. Then a word belongs to  $Enc(\psi)$  iff each of its projections onto the 2nd, 3rd,  $\dots$ ,  $(k+1)$ -th component is a bitstring containing exactly one 1. As states of  $A_{\psi}^{enc}$  we take all the strings  $\{0, 1\}^k$ . The intended meaning of a state, say state 101 for the case  $k=3$ , is “the automaton has already read the 1’s in the bitstrings of the first and third variables, but not yet read the 1 in the second.” The initial and final states are  $0^k$  and  $1^k$ , respectively. The tran-



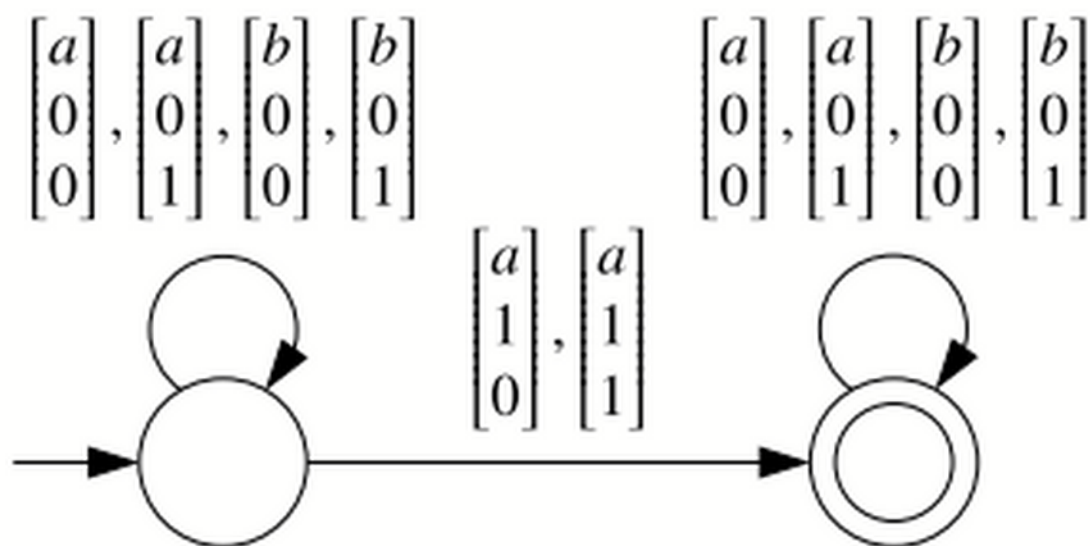
- $\varphi = \varphi_1 \wedge \varphi_2$ . Then  $free(\varphi) = free(\varphi_1) \cup free(\varphi_2)$ , and by induction hypothesis there are automata  $A_{\varphi_1}, A_{\varphi_2}$  such that  $\mathcal{L}(A_{\varphi_1}) = \mathcal{L}(\varphi_1)$  and  $\mathcal{L}(A_{\varphi_2}) = \mathcal{L}(\varphi_2)$ .

If  $free(\varphi_1) = free(\varphi_2)$ , then we can take  $A_\varphi = A_{\varphi_1} \times_{\cap} A_{\varphi_2}$ . But this need not be the case. If  $free(\varphi_1) \neq free(\varphi_2)$ , then  $\mathcal{L}(\varphi_1)$  and  $\mathcal{L}(\varphi_2)$  are languages over different alphabets  $\Sigma_1, \Sigma_2$ , or over the same alphabet, but with different intended meaning, and we cannot just compute their intersection. For example, if  $\varphi_1 = Q_a(x)$  and  $\varphi_2 = Q_b(y)$ , then both  $\mathcal{L}(\varphi_1)$  and  $\mathcal{L}(\varphi_2)$  are languages over  $\Sigma \times \{0, 1\}$ , but the second component indicates in the first case the value of  $x$ , in the second the value of  $y$ .

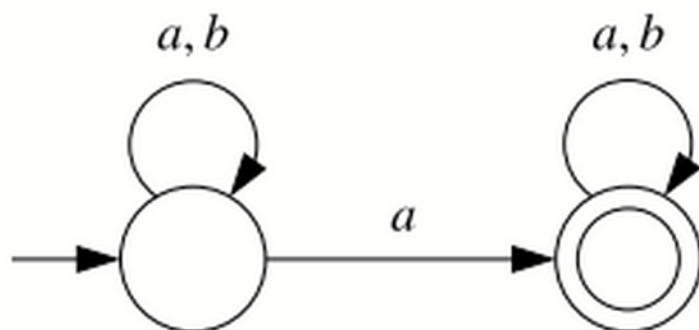
This problem is solved by extending  $\mathcal{L}(\varphi_1)$  and  $\mathcal{L}(A_{\varphi_2})$  to languages  $L_1$  and  $L_2$  over  $\Sigma \times \{0, 1\}^2$ . In our example, the language  $L_1$  contains the encodings of all interpretations  $(w, \{x \mapsto n_1, y \mapsto n_2\})$  such that the projection  $(w, \{x \mapsto n_1\})$  belongs to  $\mathcal{L}(Q_a(x))$ , while  $L_2$  contains the interpretations such that  $(w, \{y \mapsto n_2\})$  belongs to  $\mathcal{L}(Q_b(y))$ . Now, given the automaton  $A_{Q_a(x)}$  recognizing  $\mathcal{L}(Q_a(x))$



we transform it into an automaton  $A_1$  recognizing  $L_1$



- $\varphi = \exists x \psi$ . Then  $free(\varphi) = free(\psi) \setminus \{x\}$ , and by induction hypothesis there is an automaton  $A_\psi$  s.t.  $\mathcal{L}(A_\psi) = \mathcal{L}(\psi)$ . Define  $A_{\exists x \psi}$  as the result of the projection operation, where we project onto all variables but  $x$ . The operation simply corresponds to removing in each letter of each transition of  $A_\sigma$  the component for variable  $x$ . For example, the automaton  $A_{\exists x Q_a(x)}$  is obtained by removing the second components in the automaton for  $A_{Q_a(x)}$  shown above, yielding



Observe that the automaton for  $\exists x \psi$  can be nondeterministic even if the one for  $\psi$  is deterministic, since the projection operation may map different letters into the same one.

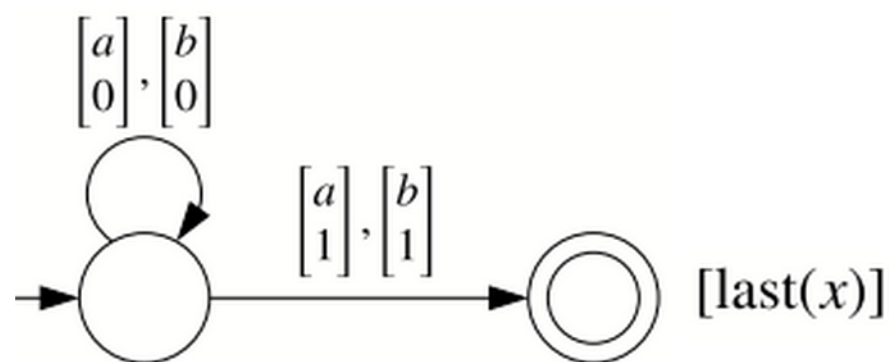
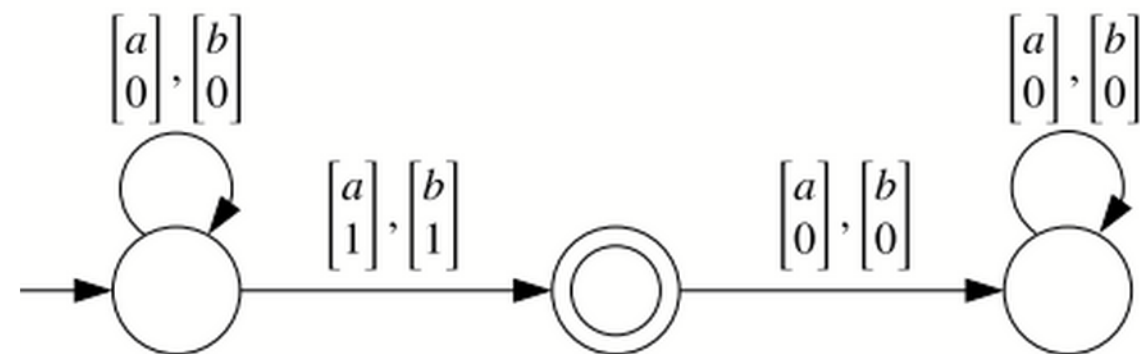
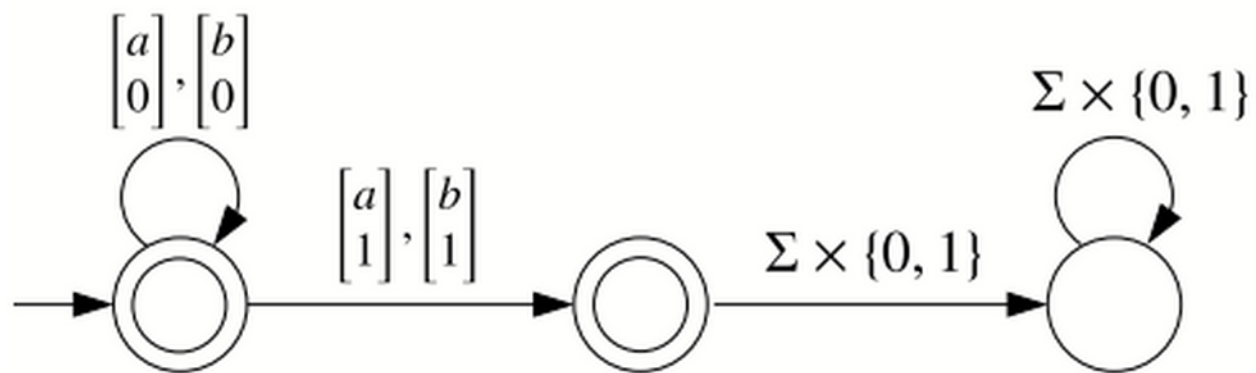
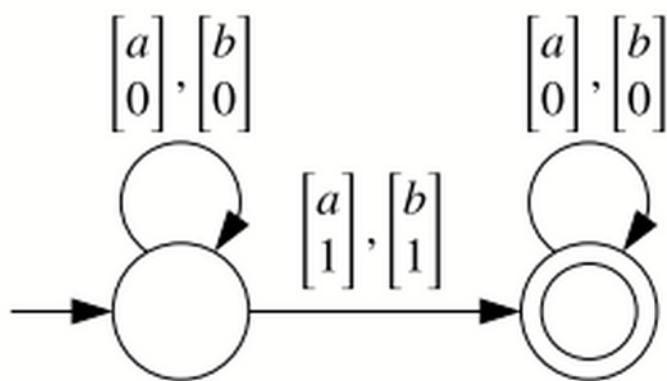
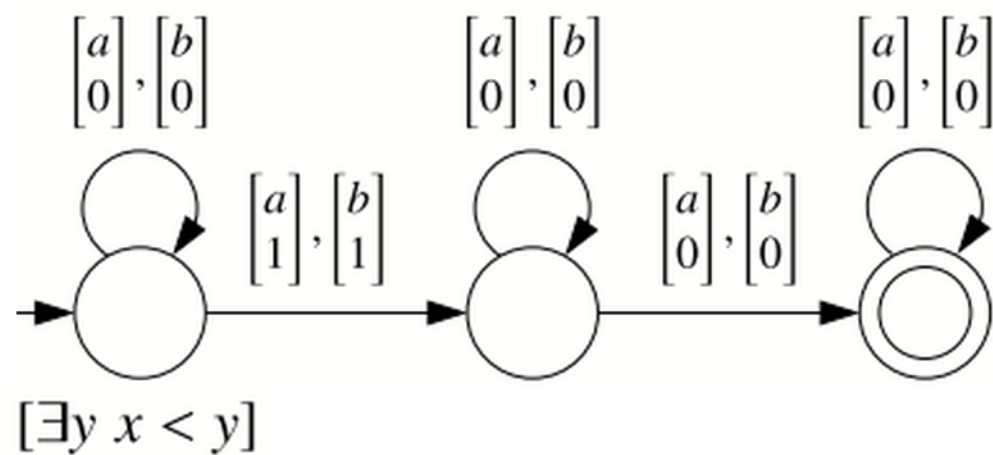
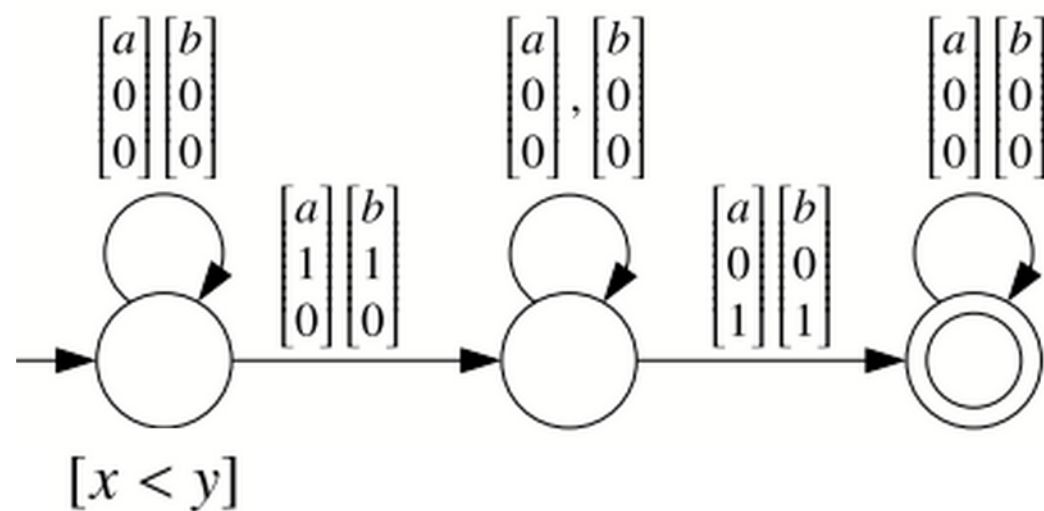
- $\varphi = \exists X \varphi$ . We proceed as in the previous case.

## The mega-example

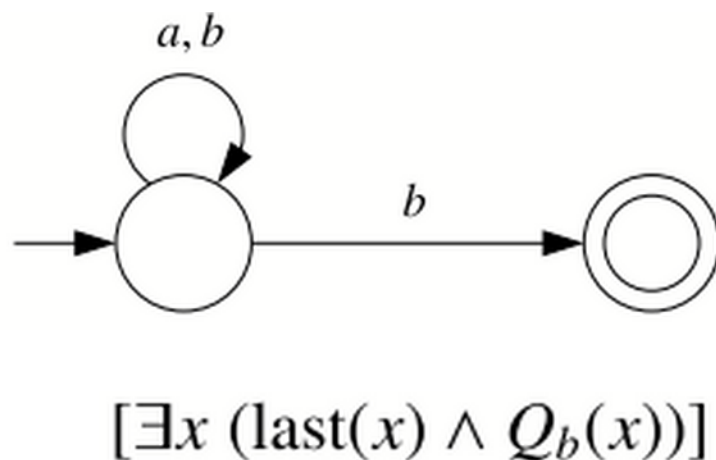
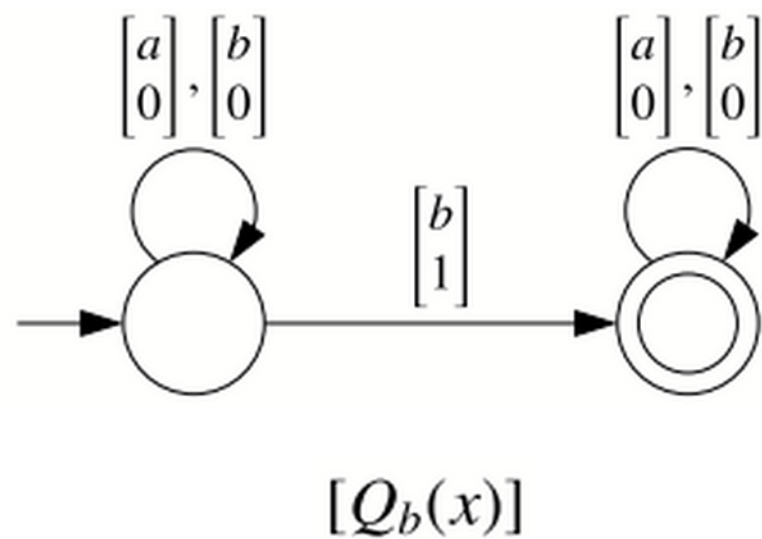
$$\varphi = \exists x (\text{last}(x) \wedge Q_b(x)) \wedge \forall x (\neg \text{last}(x) \rightarrow Q_a(x))$$

$$\psi = \exists x (\text{last}(x) \wedge Q_b(x)) \wedge \neg \exists x (\neg \text{last}(x) \wedge \neg Q_a(x))$$

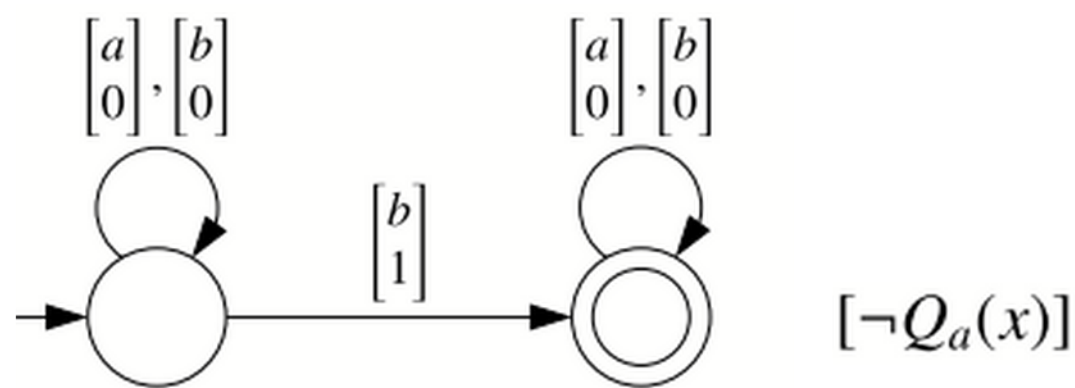
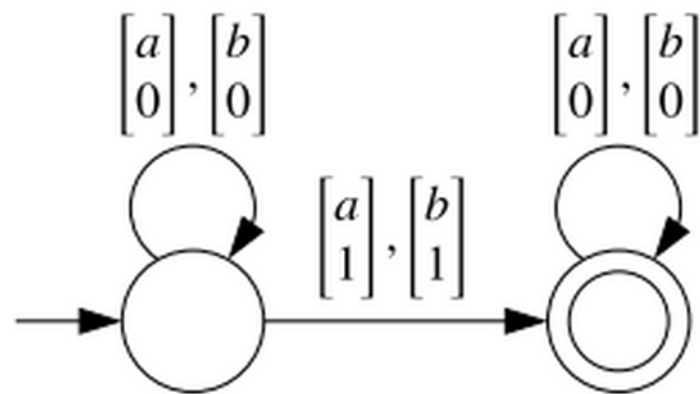
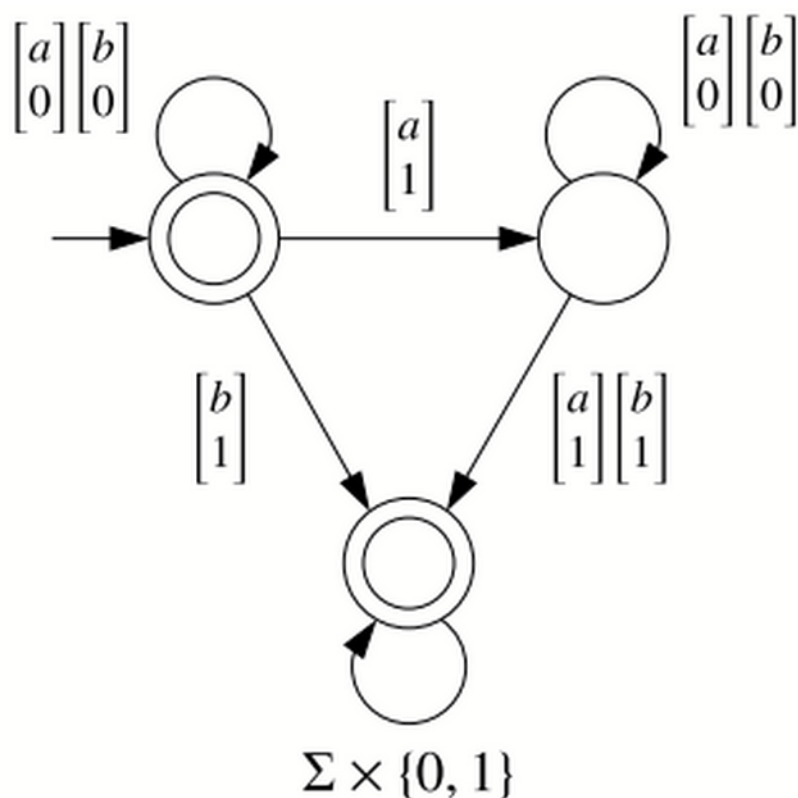
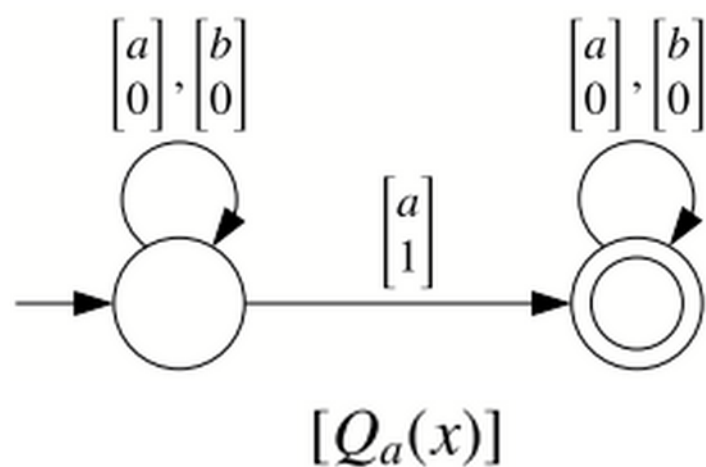
First, we compute an automaton for  $\text{last}(x) = \neg \exists y \, x < y$ .

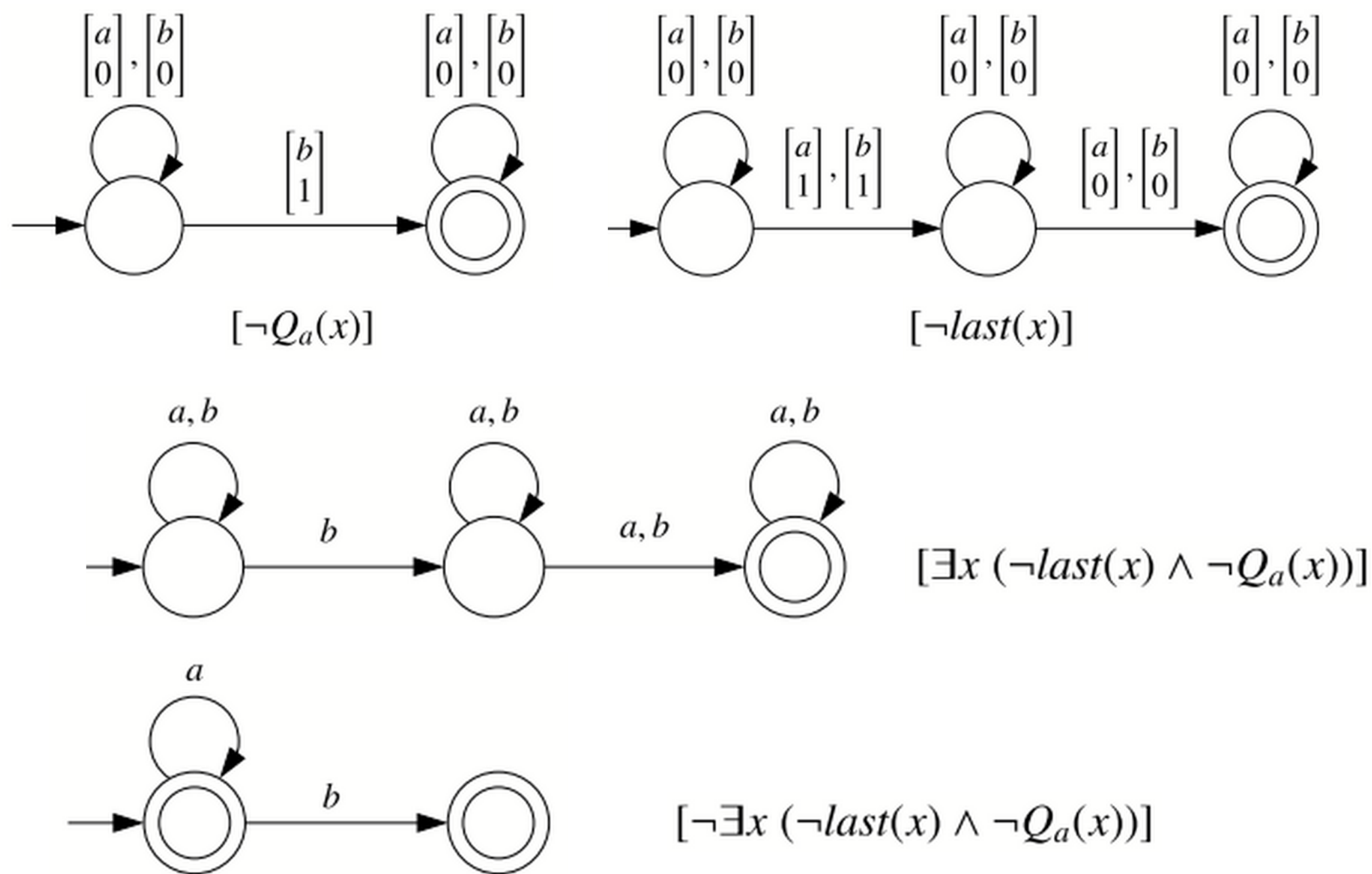


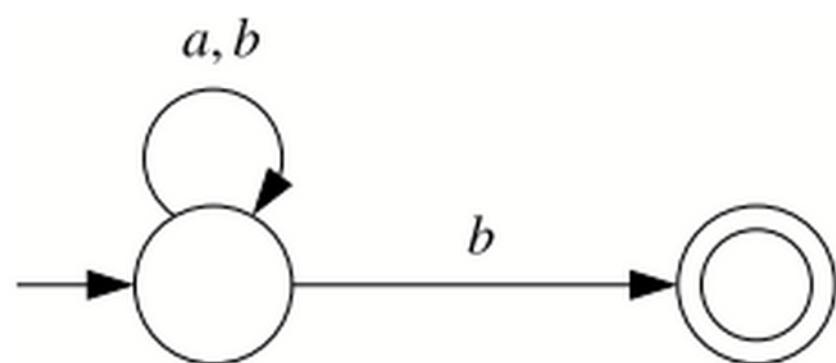
Next we compute an automaton for  $\exists x (\text{last}(x) \wedge Q_b(x))$



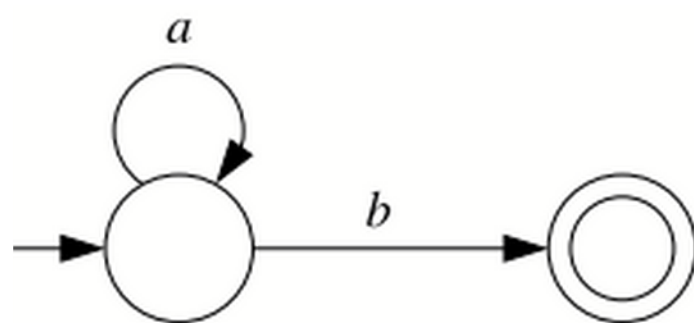
Now we compute an automaton for  $\neg\exists x (\neg\text{last}(x) \wedge \neg Q_a(x))$ ,



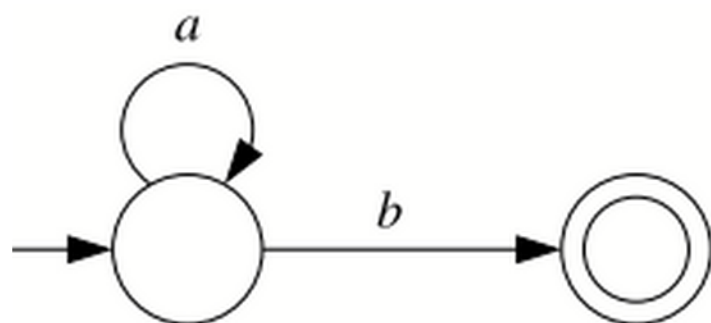




$$[\exists x (\text{last}(x) \wedge Q_b(x))]$$



$$[\neg \exists x (\neg \text{last}(x) \wedge \neg Q_a(x))]$$



$$[\exists x (\text{last}(x) \wedge Q_b(x)) \wedge \neg \exists x (\neg \text{last}(x) \wedge \neg Q_a(x))]$$