Logic
Logics on words

• Regular expressions give operational descriptions of regular languages.
• Often the natural description of a language is declarative:
  – even number of a's and even number of b's vs. 
    \[(aa + bb + (ab + ba)(aa + bb)^*(ba + ab))^*\]
  – words not containing ‘hello’
• Goal: find a declarative 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.
Atomic formulas: for each letter \( a \) we introduce the formula \( Q_a(x) \), with intuitive meaning: the letter at position \( x \) is an \( a \).
First-order logic on words: Syntax

• Formulas constructed out of atomic formulas by means of standard “logic machinery”:
  – Alphabet \( \Sigma = \{a, b, \ldots\} \) and position variables \( V = \{x, y, \ldots\} \)
  – \( Q_a(x) \) is a formula for every \( a \in \Sigma \) and \( x \in V \).
  – \( x < y \) is a formula for every \( x, y \in V \)
  – If \( \varphi, \varphi_1, \varphi_2 \) are formulas then so are \( \neg \varphi \) and \( \varphi_1 \lor \varphi_2 \)
  – If \( \varphi \) is a formula then so is \( \exists x \varphi \) for every \( x \in V \)
Abbreviations

- $\varphi_1 \land \varphi_2 := \neg (\neg \varphi_1 \lor \neg \varphi_2)$
- $\varphi_1 \rightarrow \varphi_2 := \neg \varphi_1 \lor \varphi_2$
- $\varphi_1 \leftrightarrow \varphi_2 := (\varphi_1 \land \varphi_2) \lor (\neg \varphi_1 \land \neg \varphi_2)$
- $\forall x \varphi := \neg \exists x \neg \varphi$
Abbreviations

- \( \text{first}(x) := \neg \exists y \ y < x \quad \text{last}(x) := \neg \exists y \ x < y \)
- \( y = x + 1 := x < y \land \neg \exists z \ (x < z \land z < y) \)
- \( y = x + 2 := \exists z \ (z = x + 1 \land y = z + 1) \)
  ...
- \( y = x + k := \exists z \ (z = x + 1 \land y = z + (k - 1)) \)

\[ x < k := \forall y \forall z \ (\text{first}(y) \land z = y + k) \rightarrow x < z \]
- \( \text{last} < k := \forall x \ (\text{last}(x) \rightarrow x < k) \)
Examples (without semantics yet)

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

- “Every \(a\) is immediately followed by a \(b\).”

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

- “Between every \(a\) and every later \(b\) there is a \(c\).”
Examples (without semantics yet)

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

\[ \exists x \ Q_b(x) \land \forall x \ (\text{last}(x) \rightarrow Q_b(x) \land \neg \text{last}(x) \rightarrow Q_a(x)) \]

• “Every $a$ is immediately followed by a $b$.”

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

• “Between every $a$ and every later $b$ there is a $c$.”
Examples (without semantics yet)

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

\[ \exists x \ Q_b(x) \land \forall x (\text{last}(x) \rightarrow Q_b(x) \land \neg \text{last}(x) \rightarrow Q_a(x)) \]

• “Every $a$ is immediately followed by a $b$.”

\[ \forall x (Q_a(x) \rightarrow \exists y (y = x + 1 \land Q_b(y))) \]

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

• “Between every $a$ and every later $b$ there is a $c$.”
Examples (without semantics yet)

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

  \[
  \exists x \ Q_b(x) \land \forall x \ (\text{last}(x) \rightarrow Q_b(x)) \land \neg \text{last}(x) \rightarrow Q_a(x)
  \]

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

  \[
  \forall x \ (Q_a(x) \rightarrow \exists y \ (y = x + 1 \land Q_b(y)))
  \]

- “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 $c$.”
Examples (without semantics yet)

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

\[ \exists x \ Q_b(x) \land \forall x (\text{last}(x) \rightarrow Q_b(x) \land \neg \text{last}(x) \rightarrow Q_a(x)) \]

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

\[ \forall x (Q_a(x) \rightarrow \exists y (y = x + 1 \land Q_b(y))) \]

- “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.”

\[ \forall x \forall y (Q_a(x) \land Q_b(y) \land x < y \rightarrow \exists z (x < z \land z < y \land Q_c(z))) \]
Formulas are interpreted on pairs \((w, J)\) called interpretations, where

- \(w\) is a word, and
- \(J\) assigns positions to the free variables of the formula (and maybe to others too—who cares)

It does not make sense to say a formula is true or false: it can only be true or false for a given interpretation.

If the formula has no free variables (if it is a sentence), then for each word it is either true or false.
• Satisfaction relation:

\[(w, J) \models Q_a(x) \iff w[J(x)] = a\]
\[(w, J) \models x < y \iff J(x) < J(y)\]
\[(w, J) \models \neg \varphi \iff (w, J) \not\models \varphi\]
\[(w, J) \models \varphi_1 \lor \varphi_2 \iff (w, J) \models \varphi_1 \text{ or } (w, J) \models \varphi_2\]
\[(w, J) \models \exists x \varphi \iff |w| \geq 1 \text{ and some } i \in \{1, \ldots, |w|\} \text{ satisfies } (w, J[i/x]) \models \varphi\]

• More logic jargon:
  – A formula is **valid** if it is true for all its interpretations
  – A formula is **satisfiable** if it is true for at least one of its interpretations
The empty word ...

• ... satisfies all universally quantified formulas, and no existentially quantified formula.
Can FOL express non-regular languages? Can FOL express all regular languages?

- The language $L(\varphi)$ of a sentence $\varphi$ is the set of words that satisfy $\varphi$.
- A language $L$ is expressible in first-order logic or FO-definable if some sentence $\varphi$ satisfies $L(\varphi) = L$.
- **Proposition**: a language over a one-letter alphabet is expressible in first-order 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.
1. If $L$ is finite, then it is FO-definable

2. If $L$ is co-finite, then it is FO-definable.
Proof sketch

3. If \( L \) is FO-definable (over a one-letter alphabet), then it is finite or co-finite.

1) We define a new logic QF (quantifier-free fragment)

2) We show that a language is QF-definable iff it is finite or co-finite

3) We show that a language is QF-definable iff it is FO-definable.
1) The logic QF

- $x < k$, $x > k$
- $x < y + k$, $x > y + k$
- $k < \text{last}$, $k > \text{last}$

are formulas for every variable $x$, $y$ and every $k \geq 0$.

- If $f_1$, $f_2$ are formulas, then so are $f_1 \lor f_2$ and $f_1 \land f_2$
2) $L$ is QF-definable iff it is finite or co-finite

$(\rightarrow)$ Let $f$ be a sentence of QF.

Then $f$ is a positive boolean combination of formulas $k < \text{last}$ and $k > \text{last}$.

$L(k < \text{last}) = \{k + 1, k + 2, \ldots\}$ is co-finite (we identify words and numbers)

$L(k > \text{last}) = \{0, 1, \ldots, k\}$ is finite

$L(f_1 \lor f_2) = L(f_1) \cup L(f_2)$ and so if $L(f_1)$ and $L(f_2)$
finite or co-finite then $L$ is finite or co-finite.

$L(f_1 \land f_2) = L(f_1) \cap L(f_2)$ and so if $L(f_1)$ and $L(f_2)$
finite or co-finite then $L$ is finite or co-finite.
2) \(L\) is QF-definable iff it is finite or co-finite

(\(\iff\)) If \(L = \{k_1, \ldots, k_n\}\) is finite, then

\[
(k_1 - 1 < \text{last} \land \text{last} < k_1 + 1) \lor \cdots \lor
(k_n - 1 < \text{last} \land \text{last} < k_n + 1)
\]

expresses \(L\).

If \(L\) is co-finite, then its complement is finite, and so expressed by some formula. We show that for every \(f\) some formula \(\neg(f)\) expresses \(\overline{L(f)}\)

- \(\neg(k < \text{last}) = (k - 1 < \text{last} \land \text{last} < k + 1) \lor \text{last} < k\)
- \(\neg(f_1 \lor f_2) = \neg(f_1) \land \neg(f_2)\)
- \(\neg(f_1 \land f_2) = \neg(f_1) \lor \neg(f_2)\)
3) Every first-order formula $\varphi$ has an equivalent QF-formula $QF(\varphi)$

- $QF(x < y) = x < y + 0$
- $QF(\neg \varphi) = \text{neg}(QF(\varphi))$
- $QF(\varphi_1 \lor \varphi_2) = QF(\varphi_1) \lor QF(\varphi_2)$
- $QF(\varphi_1 \land \varphi_2) = QF(\varphi_1) \land QF(\varphi_2)$
- $QF(\exists x \ \varphi) =$
  - Put $QF(\varphi)$ in disjunctive normal form. Assume $QF(\varphi) = (\varphi_1 \lor \ldots \lor \varphi_n)$, where each $\varphi_i$ is a conjunction of atomic formulas.
  - Since $\exists x (\varphi_1 \lor \ldots \lor \varphi_n) \equiv \exists x \varphi_1 \lor \ldots \lor \exists x \varphi_n$, it suffices to define $QF(\exists x \ \varphi)$ for the case in which $\varphi$ is a conjunction of atomic formulas of QF.
  - For this case, see example in the next slide.
• Consider the formula
  \[ \exists x \quad x < y + 3 \quad \land \]
  \[ z < x + 4 \quad \land \]
  \[ z < y + 2 \quad \land \]
  \[ y < x + 1 \]

• The equivalent QF-formula is
  \[ z < y + 8 \quad \land \quad y < y + 5 \quad \land \quad z < y + 2 \]
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 „even length“

• Express
  There is a set $X$ of positions such that
  – $X$ contains exactly the even positions, and
  – the last position belongs to $X$.

• Express
  \[ X \text{ contains exactly the even positions} \]
  as
  A position is in $X$ iff it is the second position or the second successor of another position of $X$.
Syntax and semantics of MSO

• New set \( \{X, Y, Z, \ldots \} \) of second-order variables
• New syntax: \( x \in X \) and \( \exists X \varphi \)
• New semantics:
  – Interpretations now also assign sets of positions to the free second-order variables.
  – Satisfaction defined as expected.
Expressing „even length“

- \( \text{second}(x) = \exists y \ (\text{first}(y) \land x = y + 1) \)

- \( \text{Even}(X) = \forall y \left( x \in X \iff \left( \exists y \ (x = y + 2 \land y \in X) \right) \right) \)

- \( \text{Evenlength} = \exists X \left( \text{Even}(X) \land \forall x \ (\text{last}(x) \rightarrow x \in X) \right) \)
Expressing $c^*(ab)^*d^*$

• Express:

There is a block $X$ of consecutive positions such that
  – 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 $b$.

• Then we can take the formula

$$\exists X \left( Cons(X) \land Boc(X) \land Aod(X) \land Alt(X) \land Fa(X) \land Lb(X) \right)$$
• $X$ is a block of consecutive positions

• Before $X$ there are only $c$‘s

• In $X$ $a$‘s and $b$‘s alternate
• *X* is a block of consecutive positions

\[
\text{Cons}(X) := \forall x \in X \ \forall y \in X \ (x < y \rightarrow (\forall z \ (x < z \land z < y) \rightarrow z \in X))
\]

• Before *X* there are only *c*′s

• In *X* *a*′s and *b*′s alternate
• *X* is a block of consecutive positions

\[ \text{Cons}(X) := \forall x \in X \; \forall y \in X \; (x < y \rightarrow (\forall z \; (x < z \land z < y) \rightarrow z \in X)) \]

• **Before** *X* **there are only** *c*′s

\[ \text{Before}(x, X) := \forall y \in X \; x < y \]

\[ \text{Before\_only\_c}(X) := \forall x \; \text{Before}(x, X) \rightarrow Q_c(x) \]

• **In** *X* *a*′s and *b*′s alternate
• **$X$ is a block of consecutive positions**

\[
\text{Cons}(X) := \forall x \in X \forall y \in X \ (x < y \rightarrow (\forall z \ (x < z \land z < y) \rightarrow z \in X))
\]

• **Before $X$ there are only $c$'s**

\[
\text{Before}(x, X) := \forall y \in X \ x < y \\
\text{Before\_only\_c}(X) := \forall x \ \text{Before}(x, X) \rightarrow Q_c(x)
\]

• **In $X$ $a$'s and $b$'s alternate**

\[
\text{Alternate}(X) := \forall x \in X \ (Q_a(x) \rightarrow \forall y \in X \ (y = x + 1 \rightarrow Q_b(y))) \\
\wedge \\
Q_b(x) \rightarrow \forall y \in X \ (y = x + 1 \rightarrow Q_a(y))
\]
Every regular language is expressible in MSO logic

- **Goal:** given an arbitrary regular language \( L \), construct an MSO sentence \( \varphi \) s.t. \( L = L(\varphi) \).
- It suffices to construct \( \varphi \) s.t. \( w \in L \) iff \( w \in L(\varphi) \) for every nonempty word \( w \). (Avoid the corner-case of the empty word.)
- 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
• Fix a regular language $L$.
• Fix a DFA $A$ with states $q_0, \ldots, q_n$ recognizing $L$.
• Fix a nonempty word $w = a_1 a_2 \ldots a_m$.
• Let $R(q)$ be the set of positions $i$ such that after reading $a_1 a_2 \ldots a_i$ the automaton $A$ is in state $q$.
• We have:

\[A \text{ accepts } w \text{ iff } m \in P_q \text{ for some final state } q.\]
Run: \[ q_0 \xrightarrow{a} q_1 \xrightarrow{a} q_1 \xrightarrow{b} q_2 \xrightarrow{b} q_0 \xrightarrow{b} q_2 \]

Position: \[ 1 \quad 2 \quad 3 \quad 4 \quad 5 \]

\[ R_w(q_0) = \{4\} \]
\[ R_w(q_1) = \{1, 2\} \]
\[ R_w(q_2) = \{3, 5\} \]
• Assume we can construct a formula

\[ \text{Visits}(X_0, \ldots, X_n) \]

which is true for \((w, J)\) iff

\[ J(X_0) = R(q_0), \ldots, J(X_n) = R(q_n) \]

• Then \((w, J)\) satisfies the formula

\[
\forall X_0 \ldots \forall X_n \forall x \left( (\text{Visits}(X_0, \ldots, X_n) \land \text{last}(x)) \rightarrow \bigvee_{q_i \in F} x \in X_i \right)
\]

iff the state after the last position is accepting, and we easily get a formula expressing \(L\).
• To construct $\textbf{Visits}(X_0,\ldots,X_n)$ we observe that the sets $R(q)$ are the unique sets satisfying
  a) $1 \in R(\delta(q_0,a_1))$ i.e., after reading the first letter the DFA is in state $\delta(q_0,a_1)$.
  b) The sets $R(q)$ build a partition of the set of positions, i.e., the DFA is always in exactly one state.
  c) If $i \in R(q)$ and $\delta(q,a_{i+1}) = q'$ then $i + 1 \in R(q')$, i.e., the sets „match“ $\delta$.
• We give formulas for a), b), and c)
\[
\text{Init}(X_0, \ldots, X_n) = \exists x \left( \text{first}(x) \land \bigvee_{a \in \Sigma} (Q_a(x) \land x \in X_{i_a}) \right)
\]

\[
\text{Partition}(X_0, \ldots, X_n) = \forall x \left( \bigvee_{i=0}^{n} x \in X_i \land \bigwedge_{i, j = 0 \atop i \neq j}^{n} (x \in X_i \rightarrow x \notin X_j) \right)
\]
• Formula for c)

\[
\text{Respect}(X_0, \ldots, X_n) = \left\{ \forall x \forall y \left[ y = x + 1 \rightarrow \bigvee_{a \in \Sigma} (x \in X_i \land Q_a(x) \land y \in X_j) \right] \right. \\
\left. i, j \in \{0, \ldots, n\} \right. \\
\delta(q_i, a) = q_j
\]

• Together:

\[
\text{Visits}(X_0, \ldots, X_n) := \text{Init}(X_0, \ldots, X_n) \land \\
\text{Partition}(X_0, \ldots, X_n) \land \\
\text{Respect}(X_0, \ldots, X_n)
\]
Every language expressible in MSO logic is regular

- Recall: an interpretation of a formula is a pair \((w, \mathcal{I})\) consisting of a word \(w\) and assignments \(\mathcal{I}\) to the free first and second order variables (and perhaps to others).

\[
\begin{align*}
\left( \begin{array}{l}
aab, \\
x \mapsto 1 \\
y \mapsto 3 \\
X \mapsto \{2, 3\} \\
Y \mapsto \{1, 2\}
\end{array} \right) & \quad \left( \begin{array}{l}
ba, \\
x \mapsto 2 \\
y \mapsto 1 \\
X \mapsto \emptyset \\
Y \mapsto \{1\}
\end{array} \right)
\end{align*}
\]
• We encode interpretations as words.

\[
\begin{align*}
\begin{pmatrix}
 x & \mapsto & 1 \\
 y & \mapsto & 3 \\
 X & \mapsto & \{2, 3\} \\
 Y & \mapsto & \{1, 2\}
\end{pmatrix}
\quad \begin{pmatrix}
 x & \mapsto & 2 \\
 y & \mapsto & 1 \\
 X & \mapsto & \emptyset \\
 Y & \mapsto & \{1\}
\end{pmatrix}
\end{align*}
\]

\[
\begin{array}{cccc}
 a & a & b \\
 x & 1 & 0 & 0 \\
 y & 0 & 0 & 1 \\
 X & 0 & 1 & 1 \\
 Y & 1 & 1 & 0 \\
\end{array}
\quad
\begin{array}{cccc}
 b & a \\
 x & 0 & 1 \\
 y & 1 & 0 \\
 X & 0 & 0 \\
 Y & 1 & 0 \\
\end{array}
\]
• Given a formula with \( n \) free variables, we encode an interpretation \((w, \mathcal{I})\) as a word \(\text{enc}(w, \mathcal{I})\) over the alphabet \(\Sigma \times \{0,1\}^n\).

• The language of the formula \(\varphi\), denoted by \(L(\varphi)\), is given by

\[
L(\varphi) := \{\text{enc}(w, \mathcal{I}) | (w, \mathcal{I}) \vDash \varphi\}
\]

• We prove by induction on the structure of \(\varphi\) that \(L(\varphi)\) is regular (and explicitly construct an automaton for it).
Case $\varphi = Q_a(x)$

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

$$L(\varphi) = \left\{ \begin{bmatrix} a_1 \\ b_1 \end{bmatrix} \ldots \begin{bmatrix} a_k \\ b_k \end{bmatrix} \mid k \geq 0, \quad a_i \in \Sigma \text{ and } b_i \in \{0, 1\} \text{ for every } i \in \{1, \ldots, k\}, \text{ and } b_i = 1 \text{ for exactly one index } i \in \{1, \ldots, k\} \text{ such that } a_i = a \right\}$$

and is recognized by

![Diagram](image)
Case $\varphi = x < y$

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

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

and is recognized by

[Diagram showing the structure of the automaton recognizing $L(\varphi)$]
Case $\varphi = x \in X$

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

$$L(\varphi) = \left\{ \begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ \vdots \\ a_k \\ b_k \\ c_k \end{bmatrix} \middle| \begin{array}{l} k \geq 0, \\
\forall i \in \{1, \ldots, k\}, \quad a_i \in \Sigma \text{ and } b_i, c_i \in \{0, 1\} \\
\exists i \in \{1, \ldots, k\}, \quad b_i = 1 \text{ and for every } i \in \{1, \ldots, k\}, \text{ if } b_i = 1 \text{ then } c_i = 1 \end{array} \right\}$$

and is recognized by

$$\begin{bmatrix} a_1 \\ b_1 \\ c_1 \\ \vdots \\ a_k \\ b_k \\ c_k \end{bmatrix}$$
Case $\varphi = \neg \psi$

- Then $\text{free}(\varphi) = \text{free}(\psi)$. By i.h. $L(\psi)$ is regular.
- $L(\varphi)$ is equal to $L(\psi)$ minus the words that do not encode any implementation ("the garbage").
- Equivalently, $L(\varphi)$ is equal to the intersection of $L(\psi)$ and the encodings of all interpretations of $\psi$.
- We show that the set of these encodings is regular.
  - Condition for encoding: Let $x$ be a free first-order variable of $\psi$. The projection of an encoding onto $x$ must belong to $0^*10^*$ (because it represents one position).
  - So we just need an automaton for the words satisfying this condition for every free first-order variable.
Example: $\text{free}(\varphi) = \{x, y\}$
Case $\varphi = \varphi_1 \lor \varphi_2$

- Then $\text{free}(\varphi) = \text{free}(\varphi_1) \cup \text{free}(\varphi_2)$. By i.h. $L(\varphi_1)$ and $L(\varphi_2)$ are regular.

- If $\text{free}(\varphi_1) = \text{free}(\varphi_2)$ then $L(\varphi) = L(\varphi_1) \cup L(\varphi_2)$ and so $L(\varphi)$ is regular.

- If $\text{free}(\varphi_1) \neq \text{free}(\varphi_2)$ then we extend $L(\varphi_1)$ to $L_1$ encoding all interpretations of $\text{free}(\varphi_1) \cup \text{free}(\varphi_2)$ whose projection onto $\text{free}(\varphi_1)$ belongs to $L(\varphi_1)$. Similarly we extend $L(\varphi_2)$ to $L_2$. We have
  
  - $L_1$ and $L_2$ are regular.
  
  - $L(\varphi) = L_1 \cup L_2$. 
Example: $\varphi = Q_a(x) \lor Q_b(y)$

- $L_1$ contains the encodings of all interpretations $(w, \{x \mapsto n_1, y \mapsto n_2\})$ such that the encoding of $(w, \{x \mapsto n_1\})$ belongs to $L(Q_a(x))$.

- Automata for $L(Q_a(x))$ and $L_1$: 
Cases $\varphi = \exists x \psi$ and $\varphi = \exists X \psi$

• Then $\text{free}(\varphi) = \text{free}(\psi) \setminus \{x\}$ or $\text{free}(\varphi) = \text{free}(\psi) \setminus \{X\}$

• By i.h. $L(\psi)$ is regular.

• $L(\varphi)$ is the result of projecting $L(\psi)$ onto the components for $\text{free}(\psi) \setminus \{x\}$ or for $\text{free}(\psi) \setminus \{X\}$. 
Example: $\varphi = Q_a(x)$

- Automata for $Q_a(x)$ and $\exists x Q_a(x)$
The mega-example

• We compute an automaton for
  \[ \exists x \left( \text{last}(x) \land Q_b(x) \right) \land \forall x \left( \neg \text{last}(x) \rightarrow Q_a(x) \right) \]

• First we rewrite it into
  \[ \exists x \left( \text{last}(x) \land Q_b(x) \right) \land \neg \exists x \left( \neg \text{last}(x) \land \neg Q_a(x) \right) \]

• In the next slides we
  1. compute a DFA for \text{last}(x)
  2. compute DFAs for \[ \exists x \left( \text{last}(x) \land Q_b(x) \right) \] and \[ \neg \exists x \left( \neg \text{last}(x) \land \neg Q_a(x) \right) \]
  3. compute a DFA for the complete formula.

• We denote the DFA for a formula \( \psi \) by \([\psi]\).
$[\text{last}(x)]$

$x < y$

\[
\begin{bmatrix}
0 & 0 \\
0 & 0
\end{bmatrix}
\rightarrow
\begin{bmatrix}
1 & 1 \\
0 & 0
\end{bmatrix}
\rightarrow
\begin{bmatrix}
0 & 0 \\
0 & 1
\end{bmatrix}
\rightarrow
\begin{bmatrix}
0 & 0 \\
0 & 0
\end{bmatrix}
\]
\[ \text{last}(x) \]

\[ x < y \]

\[ \begin{bmatrix} a \\ b \\ 0 \\ 0 \end{bmatrix} \]

\[ \begin{bmatrix} a \\ b \\ 0 \\ 0 \end{bmatrix} \]

\[ \begin{bmatrix} a \\ b \\ 0 \\ 0 \end{bmatrix} \]

\[ \begin{bmatrix} a \\ b \\ 0 \\ 0 \end{bmatrix} \]

\[ \begin{bmatrix} a \\ b \\ 0 \\ 0 \end{bmatrix} \]

\[ \begin{bmatrix} a \\ b \\ 0 \\ 0 \end{bmatrix} \]

\[ \begin{bmatrix} a \\ b \\ 0 \\ 0 \end{bmatrix} \]

\[ \exists y \ x < y \]
\[ \text{Enc}(\exists y \ x < y) \]
\[ \text{last}(x) \]

\[ [x < y] \]

\[ [\exists y \ x < y] \]

\[ \Sigma \times \{0, 1\} \]

\[ \text{Enc}(\exists y \ x < y) \]

\[ [\exists y \ x < y] \]
\[ \exists x \ (\text{last}(x) \land Q_b(x)) \]
$\neg Q_a(x)$

Enc($Q_a(x)$)

$\Sigma \times \{0, 1\}$

$\neg Q_a(x)$
\[ \neg \exists x \left( \neg \text{last}(x) \land \neg \text{Q}_a(x) \right) \]
\[ \exists x \left( \text{last}(x) \land Q_b(x) \right) \land \neg \exists x \left( \neg \text{last}(x) \land \neg Q_a(x) \right) \]