Automata and Formal Languages — Exercise Sheet 8

Exercise 8.1
(a) Let $0 \leq m < n$. Give an MSO formula $\text{Mod}^{m,n}$ such that $\text{Mod}^{m,n}(i,j)$ holds whenever $|w_iw_{i+1}\cdots w_j| \equiv m \pmod{n}$, i.e. whenever $j - i + 1 \equiv m \pmod{n}$.

(b) Let $0 \leq m < n$. Give an MSO sentence for $a^m(a^n)^*$.

(c) Give an MSO sentence for the language of words such that every two $b$'s with no other $b$ in between are separated by a block of $a$'s of odd length.

Exercise 8.2
Consider the logic PureMSO($\Sigma$) with syntax
\[
\varphi := X \subseteq Q_a \mid X < Y \mid X \subseteq Y \mid \neg \varphi \mid \varphi \lor \varphi \mid \exists X. \varphi
\]
Notice that formulas of PureMSO($\Sigma$) do not contain first-order variables. The satisfaction relation of PureMSO($\Sigma$) is given by:
\[
(w, J) \models X \subseteq Q_a \iff w[p] = a \text{ for every } p \in J(X)
\]
\[
(w, J) \models X < Y \iff p < p' \text{ for every } p \in J(X), p' \in J(Y)
\]
\[
(w, J) \models X \subseteq Y \iff p \in J(Y) \text{ for every } p \in J(X)
\]
with the rest as for MSO($\Sigma$).

Prove that MSO($\Sigma$) and PureMSO($\Sigma$) have the same expressive power for sentences. That is, show that for every sentence $\varphi$ of MSO($\Sigma$) there is an equivalent sentence $\psi$ of PureMSO($\Sigma$), and vice versa.

Exercise 8.3
1. Given a sentence $\varphi$ of MSO($\Sigma$) and a second order variable $X$ not occurring in $\varphi$, show how to construct a formula $\varphi^X$ with $X$ as free variable expressing “the projection of the word onto the positions of $X$ satisfies $\varphi$”. Formally, $\varphi^X$ must satisfy the following property: for every interpretation $J$ of $\varphi^X$, we have $(w, J) \models \varphi^X$ iff $(w|_{J(X)}, J) \models \varphi$, where $w|_{J(X)}$ denotes the result of deleting from $w$ the letters at all positions that do not belong to $J(X)$.

2. Given two sentences $\varphi_1$ and $\varphi_2$ of MSO($\Sigma$), construct a sentence Conc($\varphi_1,\varphi_2$) satisfying $L(\text{Conc}(\varphi_1,\varphi_2)) = L(\varphi_1) \cdot L(\varphi_2)$.

3. Given a sentence $\varphi$ of MSO($\Sigma$), construct a sentence Star($\varphi$) satisfying $L(\text{Star}(\varphi)) = L(\varphi)^*$.

4. Give an algorithm $\text{RegtoMSO}$ that accepts a regular expression $r$ as input and directly constructs a sentence $\varphi$ of MSO($\Sigma$) such that $L(\varphi) = L(r)$, without first constructing an automaton for the formula.

Exercise 8.4
Construct a finite automaton for the Presburger formula $\exists y. x = 2y$ using the algorithms of the chapter.
Solution 8.1
(a) We want to express \( j - i + 1 \equiv m \pmod{n} \), i.e., there exists \( l \geq 0 \) such that \( j = i + m - 1 + l \cdot n \).

\[
\text{Mod}^{m,n}(i, j) = \exists x \ (x = i + m - 1) \land \text{Mult}^n(x, j)
\]

where

\[
\text{Mult}^n(x, j) = \exists X \ (j \in X) \land (\forall z \in X \ [(z = x) \lor \exists y \in X \ (z = y + n)])
\]

Intuitively, \( x \) is the smallest option for \( j \), the one corresponding to \( l = 0 \). Set \( X \) to be the positions that are a multiple of \( n \) away from this \( x \). The subformula \( x = i + m - 1 \) is syntactic sugar for "\( x \) is the \((i + m - 1)\)-th position in the word" (since \( i, m \) are given, \( i + m - 1 \) is a constant). For example \( x = 3 \) is short for \( \exists y \ \text{first}(y) \land \exists z \ z = y + 1 \land x = z + 1 \), where \( \text{first}(y) \) and \( z = y + 1 \) are classic abbreviations you can find in the class notes.

(b) \( \left[(m = 0) \land \neg \exists x \ \text{first}(x) \right] \lor \left[\forall x \ Q_a(x) \land \exists x, y \ \text{first}(x) \land \text{last}(y) \land \text{Mod}^{m,n}(x, y) \right] \).

(c)

\[
\begin{align*}
&\forall x, y \ [(x < y) \land Q_a(x) \land Q_b(y) \land \forall z (x < z < y) \rightarrow \neg Q_a(z)] \rightarrow \\
&
\left[(\forall z (x < z < y) \rightarrow Q_a(z)) \land (\exists x', y' (x' = x + 1) \land (y = y' + 1) \land \text{Mod}^{1,2}(x', y')) \right] .
\end{align*}
\]

As remarked in the tutorial, the subformula \( \exists x', y' (x' = x + 1) \land (y = y' + 1) \land \text{Mod}^{1,2}(x', y') \) can be simplified to \( \text{Mod}^{1,2}(x, y) \).

Solution 8.2
Given a sentence \( \psi \) of PureMSO(\( \Sigma \)), let \( \phi \) be the sentence of MSO(\( \Sigma \)) obtained by replacing every subformula of \( \psi \) of the form

\[
\begin{align*}
X & \subseteq Y & & \text{by} & \forall x \ (x \in X \rightarrow x \in Y) \\
X & \subseteq Q_a & & \text{by} & \forall x \ (x \in X \rightarrow Q_a(x)) \\
X & < Y & & \text{by} & \forall x \forall y \ (x \in X \land y \in Y) \rightarrow x < y
\end{align*}
\]

Clearly, \( \phi \) and \( \psi \) are equivalent. For the other direction, let

\[
\text{empty}(X) := \forall Y \ X \subseteq Y
\]

and

\[
\text{sing}(X) := \neg \text{empty}(X) \land \forall Y \ (Y \subseteq X \land \neg \text{empty}(Y)) \rightarrow X = Y.
\]

Let \( \phi \) be a sentence of MSO(\( \Sigma \)). Assume without loss of generality that for every first-order variable \( x \) the second-order variable \( X \) does not appear in \( \phi \) (if necessary, rename second-order variables appropriately). Let \( \psi \) be the sentence of PureMSO(\( \Sigma \)) obtained by replacing every subformula of \( \phi \) of the form

\[
\begin{align*}
&\exists x \ \psi' & & \text{by} & \exists X \ (\text{sing}(X) \land \psi'[X/x]) \\
&Q_a(x) & & \text{by} & X \subseteq Q_a \\
x & < y & & \text{by} & X < Y \\
x & \in Y & & \text{by} & X \subseteq Y
\end{align*}
\]

Clearly, \( \phi \) and \( \psi \) are equivalent.

Solution 8.3
1. We build \( \varphi_X \) using the following inductive rules:

- If \( \varphi = Q_a(x), x < y, x \in X, \neg \varphi_1, \varphi_1 \lor \varphi_2 \), then \( \varphi_X = \varphi \)
- If \( \varphi = \neg \varphi_1 \) (resp. \( \varphi_1 \lor \varphi_2 \)), then \( \varphi_X = \neg \varphi_1^X \) (resp. \( \varphi_1^X \lor \varphi_2^X \)).
• If $\varphi = \exists x \psi$, then $\varphi^X = \exists x \left( x \in X \land \psi^X \right)$.
• If $\varphi = \exists Y \psi$, then $\varphi^X = \exists Y \left( \forall x \in Y \rightarrow x \in X \right) \land \psi^X$.

2. We take the formula

$$\text{Conc}(\varphi_1, \varphi_2) := \exists X \exists Y \forall x \left( x \in X \lor y \in Y \right) \land \forall x \forall y \left( \left( x \in X \land y \in Y \right) \rightarrow x < y \right) \land \varphi_1^X \land \varphi_2^Y \lor \forall x \text{false} \land \varphi_1 \land \varphi_2$$

We add the last line because although sets of positions like $X$ and $Y$ can be empty, a word $w$ satisfying a sentence of the form $\exists X \psi$ must be of length $|w| > 0$ so the empty word is not accounted for.

3. We first express that $Y$ is a set of consecutive positions between two consecutive positions of $X$. Intuitively our $X$ is the set of positions at which starts each subword verifying $\varphi$.

$$\text{Block}(Y, X) := \exists x \in X \exists z \left( \text{Next}(x, z, X) \land \forall y \left( y \in Y \leftrightarrow (x \leq y \land y < z) \right) \right)$$

where $\text{Next}(x, z, X) = z \in X \land \exists i \in X \ x < i \land i < z$ denotes that $z$ comes just after $x$ in $X$. The last line of $\text{Block}(Y, X)$ is for the case where we are considering the block from the last position of $X$ to the end of the word.

Now we express that there exists a set $X$ of positions such that every subword between any two consecutive positions of $X$ satisfies $\varphi$.

$$\text{Star}(\varphi) := \exists X \forall x \left( \text{first}(x) \rightarrow x \in X \right) \land \forall Y \left( \text{Block}(Y, X) \rightarrow \varphi^Y \right) \lor \forall z \text{false}$$

4. $\text{REtoMSO}(r)$

Input: Regular expression $r$

Output: Sentence $\varphi$ such that $L(\varphi) = L(r)$.

- $r = \emptyset \rightarrow \exists x \ x < x$
- $r = \varepsilon \rightarrow \forall x \ x < x$
- $r = a \rightarrow \exists x \left( \text{first}(x) \land \text{last}(x) \land Q_a(x) \right)$
- $r = r_1 + r_2 \rightarrow \text{REtoMSO}(r_1) \lor \text{REtoMSO}(r_2)$
- $r = r_1 r_2 \rightarrow \text{Conc}(\text{REtoMSO}(r_1), \text{REtoMSO}(r_2))$
- $r = r_1^* \rightarrow \text{Star}(\text{REtoMSO}(r_1))$

Solution 8.4

We can rewrite the formula as $\exists y. x - 2y = 0$.

To build an automaton recognizing the $\text{lsbf}$ encodings of the $x$ that are solution of this formula, we can first construct automata for the atomic formulas $x - 2y \leq 0$ and $-x + 2y \leq 0$, then intersect them and then project on the $x$ component. Here we will use $\text{EqtoDFA}$ (section 10.2.1 of the lecture notes) to directly get an automaton for $x - 2y = 0$ after which we just need to project on $x$.

We first use $\text{EqtoDFA}$ to obtain an automaton for $x - 2y = 0$: 
It remains to project the automaton on $x$, i.e. on the first component of the letters. We obtain:

which says that all encodings starting with a 0 are solutions.