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 $|w_i w_{i+1} \cdots w_j| \equiv m \,(\text{mod } n)$, i.e. whenever $j - i + 1 \equiv m \,(\text{mod } n)$.

(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(Σ) with syntax

$$\varphi := X \subseteq Q_a \mid X < Y \mid X \subseteq Y \mid \neg\varphi \mid \varphi \vee \varphi \mid \exists X.\ \varphi$$

Notice that formulas of PureMSO(Σ) do not contain first-order variables. The satisfaction relation of PureMSO(Σ) is given by:

$$\begin{array}{llll} (w, \mathcal{J}) & \models & X \subseteq Q_a & \text{iff} \quad w[p] = a \text{ for every } p \in \mathcal{J}(X) \\ (w, \mathcal{J}) & \models & X < Y & \text{iff} \quad p < p' \text{ for every } p \in \mathcal{J}(X),\ p' \in \mathcal{J}(Y) \\ (w, \mathcal{J}) & \models & X \subseteq Y & \text{iff} \quad p \in \mathcal{J}(Y) \text{ for every } p \in \mathcal{J}(X) \end{array}$$

with the rest as for MSO(Σ).

Prove that MSO(Σ) and PureMSO(Σ) have the same expressive power for sentences. That is, show that for every sentence ϕ of MSO(Σ) there is an equivalent sentence ψ of PureMSO(Σ), and vice versa.

Exercise 8.3

1. Given a sentence φ of MSO(Σ) and a second order variable X not occurring in φ, show how to construct a formula φ^X with X as free variable expressing "the projection of the word onto the positions of X satisfies φ". Formally, φ^X must satisfy the following property: for every interpretation $\mathcal{J}$ of φ^X, we have $(w, \mathcal{J}) \models \varphi^X$ iff $(w|_{\mathcal{J}(X)}, \mathcal{J}) \models \varphi$, where $w|_{\mathcal{J}(X)}$ denotes the result of deleting from w the letters at all positions that do not belong to $\mathcal{J}(X)$.
2. Given two sentences φ_1 and φ_2 of MSO(Σ), construct a sentence $\text{Conc}(\varphi_1, \varphi_2)$ satisfying $L(\text{Conc}(\varphi_1, \varphi_2)) = L(\varphi_1) \cdot L(\varphi_2)$.
3. Given a sentence φ of MSO(Σ), construct a sentence $\text{Star}(\varphi)$ satisfying $L(\text{Star}(\varphi)) = L(\varphi)^*$.
4. Give an algorithm *RegtoMSO* that accepts a regular expression r as input and directly constructs a sentence φ of MSO(Σ) 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 $j = i + m - 1 + l \cdot n$.

$$\mathrm{Mod}^{m,n}(i,j) = \exists x \ (x = i + m - 1) \wedge \mathrm{Mult}^n(x,j)$$

where

$$\mathrm{Mult}^n(x,j) = \exists X \ (j \in X) \wedge (\forall z \in X \ [(z = x) \vee \exists y \in X \ (z = y + n)])$$

Intuitively x is the smallest option for j, the one corresponding to $l = 0$. Set X is 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 $\exists y \ first(y) \wedge \exists z \ z = y + 1 \wedge x = z + 1$, where $first(y)$ and $z = y + 1$ are classic abbreviations you can find in the class notes.

(b) $[(m = 0) \wedge (\neg \exists x \ \mathrm{first}(x))] \vee [\forall x \ Q_a(x) \wedge \exists x, y \ \mathrm{first}(x) \wedge \mathrm{last}(y) \wedge \mathrm{Mod}^{m,n}(x,y)]$.

(c)

$$\begin{aligned} &\forall x, y \ [(x < y) \wedge Q_b(x) \wedge Q_b(y) \wedge \forall z (x < z < y \rightarrow \neg Q_b(z))] \rightarrow \\ &\qquad [(\forall z \ (x < z < y) \rightarrow Q_a(z)) \wedge (\exists x', y' \ (x' = x + 1) \wedge (y = y' + 1) \wedge \mathrm{Mod}^{1,2}(x', y'))] \ . \end{aligned}$$

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

Solution 8.2

Given a sentence ψ of PureMSO(Σ), let ϕ be the sentence of MSO(Σ) obtained by replacing every subformula of ψ of the form

$$\begin{array}{lll} X \subseteq Y & \text{by} & \forall x \ (x \in X \rightarrow x \in Y) \\ X \subseteq Q_a & \text{by} & \forall x \ (x \in X \rightarrow Q_a(x)) \\ X < Y & \text{by} & \forall x \, \forall y \ (x \in X \wedge y \in Y) \rightarrow x < y \end{array}$$

Clearly, ϕ and ψ are equivalent. For the other direction, let

$$\mathrm{empty}(X) := \forall Y \ X \subseteq Y$$

and

$$\mathrm{sing}(X) := \neg \mathrm{empty}(X) \wedge \forall Y \ (Y \subseteq X \wedge \neg \mathrm{empty}(Y)) \rightarrow X = Y.$$

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

$$\begin{array}{lll} \exists x \ \psi' & \text{by} & \exists X \ (\mathrm{sing}(X) \wedge \psi'[X/x]) \\ & & \text{where } \psi'[X/x] \text{ is the result of substituting } X \text{ for } x \text{ in } \psi' \\ Q_a(x) & \text{by} & X \subseteq Q_a \\ x < y & \text{by} & X < Y \\ x \in Y & \text{by} & X \subseteq Y \end{array}$$

Clearly, ϕ and ψ are equivalent.

Solution 8.3

1. We build φ^X using the following inductive rules:
   - if $\varphi = Q_a(x), x < y, x \in X, \neg\varphi_1, \varphi_1 \vee \varphi_2$, then $\varphi^X = \varphi$
   - If $\varphi = \neg\varphi_1$ (resp. $\varphi_1 \vee \varphi_2$), then $\varphi^X = \neg\varphi_1^X$ (resp. $\varphi_1^X \vee \varphi_2^X$).

- If $\varphi = \exists x\ \psi$, then $\varphi^X = \exists x\ (x \in X \wedge \psi^X)$.
- If $\varphi = \exists Y\ \psi$, then $\varphi^X = \exists Y\ \Big(\forall x\ x \in Y \rightarrow x \in X\Big) \wedge \psi^X$.

2. We take the formula

$$\begin{array}{rlll} \text{Conc}(\varphi_1, \varphi_2) & := \exists X\ \exists Y & & \forall x\ (x \in X \vee y \in Y) \\ & & \wedge & \forall x \forall y\ \Big((x \in X \wedge y \in Y) \rightarrow x < y)\Big) \\ & & \wedge & \varphi_1^X \wedge \varphi_2^Y \\ & & \vee & \forall x\ false \wedge \varphi_1 \wedge \varphi_2 \end{array}$$

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 φ.

$$\begin{array}{rlll} \text{Block}(Y, X) & := \exists x\ x \in X & & \exists z\ \Big(\text{Next}(x, z, X) \wedge \forall y\ (y \in Y \leftrightarrow (x \leq y \wedge y < z))\Big) \\ & & \vee & \text{Last}(x, X) \wedge \forall y\ (y \in Y \leftrightarrow x \leq y) \end{array}$$

where $\text{Next}(x, z, X) = z \in X \wedge \neg \exists i \in X\ x < i \wedge i < z$ 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 φ.

$$\begin{array}{rlll} \text{Star}(\varphi) & := \exists X & & \forall x\ \big(\text{first}(x) \rightarrow x \in X\big) \wedge \forall Y\ (\text{Block}(Y, X) \rightarrow \varphi^Y) \\ & & \vee & \forall z\ false \end{array}$$

4. *REtoMSO*(r)

Input: Regular expression r

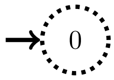
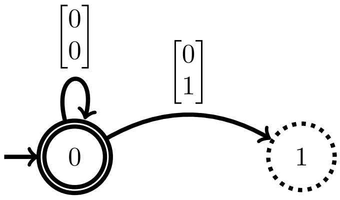
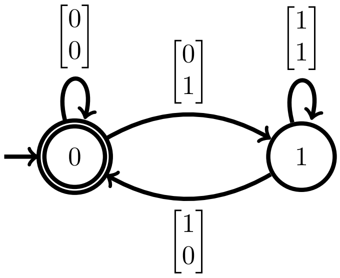
Output: Sentence φ such that $L(\varphi) = L(r)$.

$r = \emptyset\ \rightarrow\ \exists x\ x < x$
$r = \varepsilon\ \rightarrow\ \forall x\ x < x$
$r = a\ \rightarrow\ \exists x\ (\text{first}(x) \wedge \text{last}(x) \wedge Q_a(x))$
$r = r_1 + r_2\ \rightarrow\ \textit{REtoMSO}(r_1) \vee \textit{REtoMSO}(r_2)$
$r = r_1 r_2\ \rightarrow\ \text{Conc}(\textit{REtoMSO}(r_1), \textit{REtoMSO}(r_2))$
$r = r_1^*\ \rightarrow\ \text{Star}(\textit{REtoMSO}(r_1))$

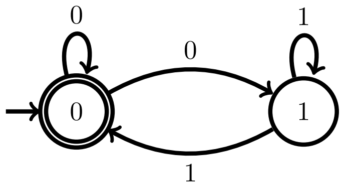
Solution 8.4
We can rewrite the formula as $\exists y.\ x - 2y = 0$.

To build an automaton recognizing the *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 *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 *EqtoDFA* to obtain an automaton for $x - 2y = 0$:

Iter.	**Current automaton**	W
0		$\{0\}$
1		$\{1\}$
2		$\emptyset$

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.