Technische Universität München **Winter term 2013/2014**
I7
M. Kufleitner / A. Durand-Gasselin **16.12.2013**

Automata and Formal Languages – Exercise sheet 6

Exercise 6.1

Given a formula of the form $\sum_i a_i x_i \equiv c \bmod k$ with $\gcd(2a_i, k) = 1$ for all i.

1. Show that the minimal deterministic automaton accepting solutions (represented in base 2) of this formula has exactly k states.
2. Show that there does not exist any smaller nondeterministic automaton accepting that language.

Exercise 6.2

Give an MSO sentence defining the language $\{ab, ba\}^*$ over the alphabet $\{a, b\}$.

Exercise 6.3

Construct an automaton for the following MSO sentence

$$\exists X\, \forall x\, \forall y\colon (\lambda(x) = a \wedge x \notin X) \vee \lambda(y) = b \vee (x < y \wedge y \in X)$$

over $\{a, b\}^*$.

Solution:

The first step consists in rewriting the formula as:

$$\exists X\, \neg \exists x\, \exists y\ (\lambda(x) \neq a \vee x \in X) \wedge (\lambda(y) \neq b) \wedge (x \geq y \vee y \notin X)$$

Then we give an automaton for each of the atomic subformulas:

- $\varphi_1(X, x, y) : \lambda(x) \neq a$,
  The automaton A_1 will be over alphabet $\Sigma \times \{0,1\} \times \{0,1\} \times \{0,1\}$.
  The second component of a letter indicates whether the position is in X,
  the third (resp. fourth) whether the position is that of x (resp. y).
  Notice that this automaton should accept only words who exactly have a single letter whose third component is 1 as x is a first-order variable. (The same holds for the fourth component.)
  Because of this restriction we know that we can split each language in three pairwise disjoint languages: the language of solutions when $x > y$, when $x < y$ and when $x = y$. As these languages are disjoint, we can perform the boolean operations over each of these 3 classes of languages independantly.

- $x < y$, $A_{1,<} : (\Sigma \times \{0,1\} \times \{0\} \times \{0\})^*((b,1,1,0)|(b,0,1,0))(\Sigma \times \{0,1\} \times \{0\} \times \{0\})^*((a,0,0,1)|(b,0,0,1)|(a,1,0,1)|(b,1,0,1))(\Sigma \times \{0,1\} \times \{0\} \times \{0\})^*$

  We can simplify this notation by omitting the $\times$ and writing 2 as a shorthand for $\{0,1\}$, 1 for $\{1\}$ and 0 for $\{0\}$.

  We thus get:

  $x < y$, $A_{1,<} : (\Sigma 200)^*(b210)(\Sigma 200)^*(\Sigma 201)(\Sigma 200)^*$
- $x > y$, $A_{1,>} : (\Sigma 200)^*(\Sigma 201)(\Sigma 200)^*(b210)(\Sigma 200)^*$
- $x = y$, $A_{1,=} : (\Sigma 200)^*(b211)(\Sigma 200)^*$

* $\varphi_2(X,x,y) : x \in X$
  $A_{2,<} : (\Sigma 200)^*(\Sigma 110)(\Sigma 200)^*(\Sigma 201)(\Sigma 200)^*$
  $A_{2,>} : (\Sigma 200)^*(\Sigma 201)(\Sigma 200)^*(\Sigma 110)(\Sigma 200)^*$
  $A_{2,=} : (\Sigma 200)^*(\Sigma 111)(\Sigma 200)^*$

* $\varphi_3(X,x,y) : \lambda(y) \neq b$
  $A_{3,<} : (\Sigma 200)^*(\Sigma 210)(\Sigma 200)^*(a201)(\Sigma 200)^*$
  $A_{3,>} : (\Sigma 200)^*(a201)(\Sigma 200)^*(\Sigma 210)(\Sigma 200)^*$
  $A_{3,=} : (\Sigma 200)^*(a211)(\Sigma 200)^*$

* $\varphi_4(X,x,y) : x \geq y$
  $A_{4,<} : \emptyset$
  $A_{4,>} : (\Sigma 200)^*(\Sigma 201)(\Sigma 200)^*(\Sigma 210)(\Sigma 200)^*$
  $A_{4,=} : (\Sigma 200)^*(\Sigma 211)(\Sigma 200)^*$

* $\varphi_5(X,x,y) : y \notin X$
  $A_{5,<} : (\Sigma 200)^*(\Sigma 210)(\Sigma 200)^*(\Sigma 001)(\Sigma 200)^*$
  $A_{5,>} : (\Sigma 200)^*(\Sigma 001)(\Sigma 200)^*(\Sigma 210)(\Sigma 200)^*$
  $A_{5,=} : (\Sigma 200)^*(\Sigma 011)(\Sigma 200)^*$

We will now build inductively an automaton accepting the whole quantifier free formula:

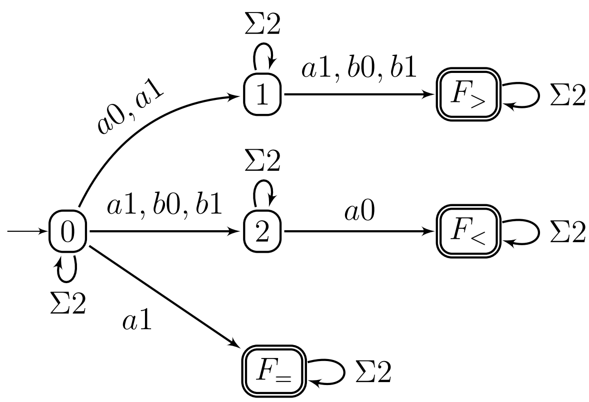
First remark that $A_{1,<} \cup A_{2,<}$ (which we will denote $A_{12,<}$) is
$(\Sigma 200)^*((b210) \cup (\Sigma 110))(\Sigma 200)^*((\Sigma 201) \cup (\Sigma 201))(\Sigma 200)^*$
similarly $A_{1,=} \cup A_{2,=}$ (denoted $A_{12,=}$) is
$(\Sigma 200)^*((b211) \cup (\Sigma 111))(\Sigma 200)^*$
Also remark that (because the pairwise disjointness)
$(A_{i,<} \cup A_{i,>} \cup A_{i,=}) \cap (A_{j,<} \cup A_{j,>} \cup A_{j,=}) = (A_{i,<} \cap A_{j,<}) \cup (A_{i,>} \cap A_{j,>}) \cup (A_{i,=} \cap A_{j,=})$
and that $A_{12,<} \cap A_{3,<} =$
$(\Sigma 200)^*(((b210) \cup (\Sigma 110)) \cap (\Sigma 210))(\Sigma 200)^*)(((\Sigma 201) \cup (\Sigma 201)) \cap (a201))(\Sigma 200)^*$
Applying these remarks allow us to compute the following (threefold) regular expression for the quantifier-free formula:

$A = (\Sigma 200)^*\gamma_>(\Sigma 200)^*\sigma_>(\Sigma 200)^* \cup (\Sigma 200)^*\gamma_<(\Sigma 200)^*\sigma_<(\Sigma 200)^* \cup (\Sigma 200)^*\gamma_=(\Sigma 200)^*$
where:
$\gamma_> = (\Sigma 201 \cup \Sigma 201) \cap (a201) \cap (\Sigma 201 \cup \Sigma 001) = \{a001, a101\}$
$\sigma_> = \{a110, b110, b010\}$, $\gamma_< = \{a110, b110, b010\}$, $\sigma_< = \{a001\}$, $\gamma_= = \{a111\}$.

We now perform the language projection corresponding to the existential quantification of the first-order variables x and y. Notice that a word should belong to the projection, if it can be obtained by erasing the third an fourth components of each letter of a word in the language A. It corresponds exactly to erasing each third an fourth component in the regular expression (as this regular expression does not contain any complement operation).

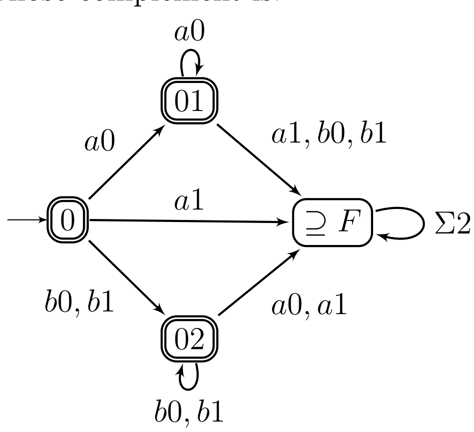
$A' = (\Sigma 2)^*\{a0, a1\}(\Sigma 2)^*\{a1, b0, b1\}(\Sigma 2)^* \cup (\Sigma 2)^*\{a1, b0, b1\}(\Sigma 2)^* a0 (\Sigma 2)^* \cup (\Sigma 2)^* a1 (\Sigma 2)^*$

We now need to perform a language complementation of A' before we project away the second component. We give the automaton $\mathcal{A}'$ that accepts that regular expression:



	$a0$	$a1$	$b0$	$b1$
0	01	$012F_=$	02	02
01	01	$012F_>F_=$	$012F_>$	$012F_>$
02	$012F_<$	$012F_=$	02	02
$\supseteq F$	$\supseteq F$	$\supseteq F$	$\supseteq F$	$\supseteq F$

To easily determinize this automaton (using the standard subset construction), we rely on the fact that we can merge all states that contain a final states as from any of those states we are guaranteed to accept $(\Sigma 2)^*$. This produces the a 4-state deterministic automaton, whose complement is:



Discarding the second component leads us to an automaton that accepts the language $L = a^* \cup b^*$ which is the set of words that satisfy the MSO sentence.

A careful analysis of the formula could have spared us this long construction: That sentence is satisfied by any word of a^*: take $X = \emptyset$ the first disjunct is always true, similarly, it is always satisfied by a word of the form b^* as the second disjunct will always be true. If it is satisfied by a word w that contains an a (at position i), let us show that w does not contain any b: assume it contains a b at position j, then when $x = y = i$, we have that $i \notin X$. When $y = i, x = j$, only the third disjunct might be true, thus $j < i \wedge i \in X$ so $i \in X$ and $i \notin X$ which implies a contradiction.

Exercise 6.4

Apply Angluin's L^*-algorithm for learning the language $L = a(ba)^*$ over the alphabet $\{a, b, c\}$.

Exercise 6.5

Give a Büchi automaton for the language L of all words $\alpha \in \{a, b, c\}^\omega$ such that α contains infinitely many a's, finitely many c's, and between any two a's there is an even number of b's or c's.