Technische Universität München **Winter term 2010/11**
I7
Prof. J. Esparza / J. Křetínský **16.12.2010**

Automata and Formal Languages – Homework 8

Due 13.1.2011.

Exercise 8.1 **

Characterize the languages described by the following formulae and give a corresponind automata:

(a) $\exists x\ zero(x)$

(b) $\forall x\ zero(x)$

(c) $\big(\neg\exists x\exists y(x < y \wedge Q_a(x) \wedge Q_b(y))\big) \wedge \big(\forall x(Q_b(x) \rightarrow \exists y(x < y \wedge Q_a(y)))\big) \wedge \big(\exists x(\neg\exists y x < y \wedge Q_a(x))\big)$

Exercise 8.2 **

For the following languages over $\{a, b\}$, write down their defining MSO formula, automaton and regular expression.

- The set of words of even length and containing only a's or only b's.
- The set of words, where between each two b's with no other b in between there is a block of an odd number of letters a.
- The set of words with odd length and an odd number of occurrences of a.

Exercise 8.3 *

We can show that a given formula of MSO holds on some word, i.e., is satisfiable, as follows. Firstly, we transform the given formula into the equivalent automaton. Secondly, we decide whether it has an accepting run (note that the automaton can be input-free and no letters are actually read). The formula is *satisfiable* iff there is such an accepting run.

Moreover, if the given formula is a sentence, i.e., with no free variables, it is a *tautology* iff there is an accepting run. Thus we have a method to prove (and disprove) MSO formulae.

Prove that every nonempty (finite) subset Z of natural numbers has its minimal element, i.e. show that

$$\forall Z\exists x\forall y(y \in Z \rightarrow (x \leq y \wedge x \in Z))$$

where $x \leq y$ is a shortcut for $\neg y < x$ is true on all words.

Exercise 8.4

Using the algorithms discussed in the lecture, construct a finite automaton for the Presburger formula

$$\exists y : x = 3y.$$

Exercise 8.5 **

Give formulae expressing the following macros:

(a) $\mathrm{Sing}(X)$ meaning that the set X is a singleton,

(b) $X \subseteq Y$ meaning subset inclusion,

(c) $X \subseteq Q_a$ meaning all elements of X are labelled by a, for $a \in \Sigma$,

(d) $X < Y$ that is true for singletons $X = \{x\}, Y = \{y\}$ satisfying $x < y$.

Exercise 8.6

We interpret the monadic second order logic over finite words with the standard interpretation of $<$ as less than relation.

Let $MSO'(S)$ be a modification of the standard monadic second-order logic given by the following syntax. Assume a set of second-order logical variables ranged over by X, Y, Z. Let Σ be an alphabet. An $MSO'(<)$ formula over Σ is defined by the following BNF, where $a \in \Sigma$:

$$\varphi ::= X \subseteq Q_a \mid X < Y \mid \mathrm{Sing}(X) \mid X \subseteq Y \mid \neg\varphi \mid (\varphi \vee \varphi) \mid \exists X \varphi$$

Although we quantify over set variables only, we want this logic to be equally "powerful" as the original $MSO(<)$. As there are no first-order variables, the first-order predicates $<$ will be replaced by the second-order predicates, so new atomic formulas are introduced: $\mathrm{Sing}(X)$ (meaning singleton), $X \subseteq Y$ (meaning subset inclusion), $X \subseteq Q_a$ for every $a \in \Sigma$ (meaning all elements of X are labelled by a), and $X < Y$ (true for singletons $X = \{x\}, Y = \{y\}$ satisfying $x < y$).

(a) Show that $MSO(<)$ and $MSO'(<)$ are equally expressive, i.e., a language is definable in $MSO(<)$ iff it is definable in $MSO'(<)$.

Hint: Express the newly defined predicates in the original MSO and vice versa.

Remark: This logic can be used to create a different (a bit easier) procedure to translate formulae into automata: the problem of incorrect encodings does not arise.

(b) Translate the formula

$$\exists Z \forall x (Q_a(x) \to \exists y (x < y \wedge y \in Z))$$

into an equivalent one of $MSO'(<)$.

Exercise 8.7

Express the addition using $MSO(<)$. More precisely, find a formula $\mathrm{Plus}(X, Y, Z) \in MSO(<)$ that is true iff $x + y = z$, where x, y, z are numbers encoded by the sets X, Y, Z, respectively, in the binary lsbf. You are allowed to use the successor macro S.

Remarks:

- Thus, we can translate any first-order formula with signature $+$, i.e. any formula of Presburger arithmetic, to an equivalent $MSO(<)$ formula: the first-order quantification is replaced by the subset quantification and $+$ by Plus. The same holds for sentences. Since $MSO(<)$ has already been proved to be decidable (see the translation into automata in the lecture and Exercise 10.2), Presburger arithmetic decidable, i.e. we have an algothm to determine whether a formula is true for every valuation.
- There is also another approach to prove the decidability of Presburger arithmetic: we find the respective automaton directly (see the lecture).