Solutions of Exam Languages and Machines, 18 June 2015

Duration 3 hours. Closed book. You are allowed to use theorems from the Lecture Notes, provided you phrase them correctly. Give clear and crisp arguments for all your assertions.

Exercise 1 (10 %). Consider a language L over the alphabet Σ. Fill in the dots (...) with a property of machines defined in the course.
(a) L is context-free $\equiv$ $\exists\, M : L = L(M)$ and M is a ...
(b) L is decidable $\equiv$ $\exists\, M : L = L(M)$ and M is a ...
(c) L is semi-decidable $\equiv$ $\exists\, M : L = L(M)$ and M is a ...
(d) L is regular $\equiv$ $\exists\, M : L = L(M)$ and M is a ...
(e) Give all valid implications between these four assertions about L.

Solution.
(a) L is context-free $\equiv$ $\exists\, M : L = L(M)$ and M is a PDM
(b) L is decidable $\equiv$ $\exists\, M : L = L(M)$ and M is an always terminating TM
(c) L is semi-decidable $\equiv$ $\exists\, M : L = L(M)$ and M is a TM
(d) L is regular $\equiv$ $\exists\, M : L = L(M)$ and M is a DFSM
(e) L is regular $\Rightarrow$ L is context-free $\Rightarrow$ L is decidable $\Rightarrow$ L is semi-decidable.

Exercise 2 (12 %). Let $G = (V, \Sigma, P, S)$ be a context-free grammar.
(a) When is G *essentially noncontracting*? When is G *productive*? Give the two definitions.
(b) Let the context-free grammar G be given by $\Sigma = \{a, b, c\}$, $V = \{S, D, E\}$, and the production rules:

$$\begin{array}{lcl} S & \to & cE \mid aDb \\ D & \to & Sc \mid \varepsilon \mid aE \\ E & \to & bE \mid DD\ . \end{array}$$

Use the standard algorithm to determine an equivalent *productive* grammar. Give and prove all intermediate results.

Solution. (a: 3 %) G is essentially noncontracting iff its start symbol S is nonrecursive and G has no production rules of the form $A \to \varepsilon$ with $A \neq S$. It is productive if its start symbol S is nonrecursive and every production rule $A \to v$ with $A \neq S$ satisfies $v \in \Sigma$ or $|v| > 1$.

(b: 9 %) In view of the rule $D \to Sc$, we make the start symbol nonrecursive by adding a new start symbol T:

$$\begin{array}{lcl} T & \to & S \\ S & \to & cE \mid aDb \\ D & \to & Sc \mid \varepsilon \mid aE \\ E & \to & bE \mid DD\ . \end{array}$$

Next we determine the nullable nonterminals: D is directly nullable; therefore E is also nullable; no more nullables. We then extend the grammar by nulling all nullables:

$$\begin{array}{lcl} T & \to & S \\ S & \to & cE \mid aDb \mid c \mid ab \\ D & \to & Sc \mid \varepsilon \mid aE \mid a \\ E & \to & bE \mid DD \mid b \mid D \mid \varepsilon\ . \end{array}$$

We then remove all forbidden epsilon productions.

$$
\begin{array}{lcl}
T & \to & S \\
S & \to & cE \mid aDb \mid c \mid ab \\
D & \to & Sc \mid aE \mid a \\
E & \to & bE \mid DD \mid b \mid D\ .
\end{array}
$$

We now see the chain rules: $T \to S$ and $E \to D$. Subsequently, the grammar is extended by pushing forward along the chain rules:

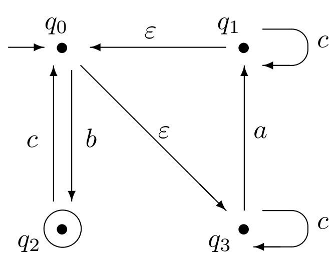
$$
\begin{array}{lcl}
T & \to & S \\
S & \to & cE \mid aDb \mid c \mid ab \mid cD \\
D & \to & Sc \mid aE \mid a \mid aD \\
E & \to & bE \mid DD \mid b \mid D \mid bD\ .
\end{array}
$$

Finally, all forbidden chain rules are removed:

$$
\begin{array}{lcl}
T & \to & S \\
S & \to & cE \mid aDb \mid c \mid ab \mid cD \\
D & \to & Sc \mid aE \mid a \mid aD \\
E & \to & bE \mid DD \mid b \mid bD\ .
\end{array}
$$

This grammar is indeed productive.

Exercise 3 (10 %). Consider the alphabet $\Sigma = \{a, b, c\}$ and the nondeterministic finite state machine M with ε-transitions, with the state diagram:



Use the standard algorithm to determine the transition table of an equivalent deterministic finite state machine. Indicate the start state and the accepting states.

Solution (writing `i` for q_i)

```
delta           a          b         c
-> {0,3}     {0,1,3}     {2}       {3}
   {0,1,3}   {0,1,3}     {2}     {0,1,3}
 * {2}          {}       {}       {0,3}
   {3}       {0,1,3}     {}        {3}
   {}           {}       {}        {}
```

Exercise 4 (12 %). (a) Phrase the Pumping Lemma for *regular* languages.
(b) Given is the language $L_4 = \{ww \mid w \in \Sigma^*\}$ over the alphabet $\Sigma = \{a, b\}$. Prove that this language is not regular.

Solution (a: 3%) Let L be a regular language. Then there is a number k, such that every string $z \in L$ with $|z| \geq k$ can be split into three substrings $z = uvw$ such that $|uv| \leq k$ and $v \neq \varepsilon$, and $uv^iw \in L$ for every $i \geq 0$.

(b: 9%) Proof by contradiction. Assume that L_4 is regular. Then there is a number k as in the lemma.

Consider the string $z = a^kba^kb$. It is clear that $z \in L_4$ and that $|z| = 2k+2 \geq k$. The lemma therefore implies that z has a splitting $z = uvw$ such that $|uv| \leq k$ and $v \neq \varepsilon$, and $uv^iw \in L$ for every $i \geq 0$. As $a^kba^kb = uvw$ and $|uv| \leq k$, the substring

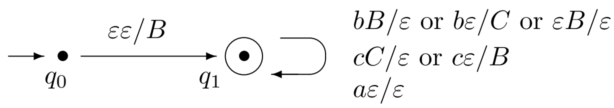
uv is contained in the prefix a^k. As $v \neq \varepsilon$, it follows that $v = a^m$ for some number $m > 0$. Taking $i := 2$, we get $z_2 = uv^2w \in L$. This implies $z_2 = a^{k+m}ba^kb = xx$ for some string $x \in \Sigma^*$. As $n_b(z_2) = 2$ and $z_2 = xx$, the string x contains only one symbol b. As z ends with b, string x ends with b. This implies that $a^{k+m}b = x = a^kb$, and hence $m = 0$, a contradiction. Therefore, L_4 is not regular.

Exercise 5 (11 %). Consider the language L_5 over $\Sigma = \{a, b, c\}$ given by

$$L_5 = \{w \in \Sigma^* \mid n_b(w) \leq 1 + n_c(w)\} \ .$$

Construct a simple pushdown machine M_5 that accepts the language L_5. Give the state diagram, and give convincing arguments that the language accepted by M_5 indeed equals L_5.

Solution While scanning the input string, we need to keep track of the number of additional symbols b or c that have yet to be read to have equality $n_b(w) = 1+n_c(w)$.



The machine first pushes a symbol B onto the stack. In state q_1, it preserves the invariant $n_b(w) + n_B(\gamma) \leq 1 + n_c(w) + n_C(\gamma)$, where w is the input read, and γ is the current stack. When the machine accepts the input w, the invariant with empty stack implies that $w \in L_5$.

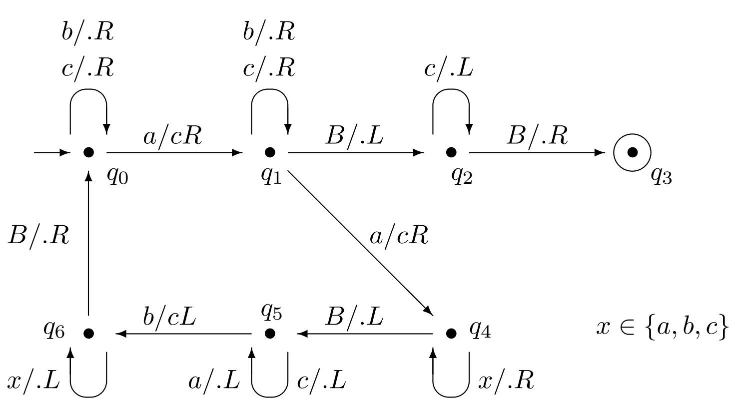
Conversely, if the input is $w \in L_5$, the nondeterminism can preserve the invariants $n_b(w) + n_B(\gamma) = 1 + n_c(w) + n_C(\gamma)$ and $\gamma \in B^* \cup C^*$, until the input has been scanned completely. At that point the stack is of the form $\gamma \in B^*$, and the string can be accepted after these Bs have been popped.

Exercise 6 (11 %). Consider the alphabet $\Sigma = \{a, b, c\}$ and the language

$$L_6 = \{w \in \Sigma^* \mid n_a(w) = 1 + 2 \cdot n_b(w)\} \ .$$

Construct a simple always terminating Turing machine M with $L(M) = L_6$. Give the complete state diagram. Indicate in which states the computation can terminate when the input does not belong to L_6, why the machine always terminates, and why it accepts the language L_6.

Solution Recall that a *simple* Turing machine is deterministic and has a single tape. In the input string, we count symbols a and b by replacing them by the tape symbol c. We first need to replace one a, and subsequently, for every two symbols a replaced, we replace one b, until the input is of the form c^*.



In the state diagram, each self-loop terminates because it has a fixed direction: L or R. In the cycle q_0, q_1, q_4, q_5, q_6, two symbols a and one symbol b are replaced by c. Therefore the machine terminates, and the difference $n_a(w) - 2n_b(w)$ remains constant. In the end, only one additional a needs to be replaced. Therefore, the machine allows to exit the loop at q_1.

In the states q_0 and q_1, the head moves to the right, and ensures that the string to the left of the head contains no symbols a. It therefore only enters q_2 when all symbols a have been replaced. In q_2, it then verifies that the input is of the form c^*. If the input contains too many symbols b, execution ends in q_2 with a b on the tape.

In the lower row, the head first moves to the righthand end of the input string, then moves left, replaces one symbol b, and subsequently moves to the lefthand end of the input string. If there are too few symbols b, execution ends in q_5 with a blank on the tape.

Exercise 7 (12 %). Let L be a language over alphabet Σ, and let x and y be strings over Σ.
(a) Assume L is decidable. Give the definition of this.
Prove that $L' = \{w \in \Sigma^* \mid xw \in L \wedge wy \notin L\}$ is decidable.
(b) Assume L is semi-decidable. Give the definition of this.
Prove that $L'' = \{w \in \Sigma^* \mid xw \in L \vee wy \in L\}$ is semi-decidable.

Solution (a) Decidability of L means that there is an always terminating simple Turing machine M that accepts L. We construct an always terminating TM M' for L'. Machine M' is a 2-tape TM. It first copies the input w to the second tape. Subsequently, it writes string x before w in tape 1 and string y after w on tape 2. It places the tape heads on the first symbols of xw and wy, respectively. Subsequently it executes machine M on both tapes, say one after the other. It accepts w if and only if M accepts xw and rejects wy. As M always terminates, M' always terminates. It is clear that M' accepts L'. This proves that L' is decidable.

(b) Semi-decidability of L means that there is a simple Turing machine M that accepts L by termination only. We construct a TM M'' that accepts L'' by termination only. Machine M'' is a 2-tape TM. It first copies the input w to the second tape. Subsequently, it writes string x before w in tape 1 and string y after w on tape 2. It places the tape heads on the first symbols of xw and wy, respectively. Subsequently it executes copies of machine M on both tapes on lock-step. Machine M'' terminates when either machine M terminates on its own tape. Therefore the language accepted by M'' is L''. This proves that L'' is semi-decidable.

Exercise 8 (10 %). The Lecture Notes describe how to encode a Turing machine $M \in \mathit{TM0}$ by means of a string $R(M)$, and they describe a universal Turing machine that can simulate any Turing machine M thus encoded.
(a) Describe the class *TM0* of the machines that can be encoded in this way, and describe the encoding $R(M)$ for an arbitrary machine $M \in \mathit{TM0}$.
(b) Describe the language L_U accepted by this universal Turing machine in <u>words</u> and in <u>set notation</u>.
(c) Is the language L_U decidable? Is it semi-decidable? Justify your answers.

Solution (a: 4%) *TM0* consists of the simple TMs over $\mathbb{B}$ that accept by termination only (i.e., without a set of accepting states). For such a machine $M = (Q, \Sigma, \Gamma, \delta, q_0)$, the encoding $R(M)$ is a bit string, defined as follows. First, the states of Q are numbered from $n(q_0) = 1$, etc. ; next the tape symbols in Γ are numbered with $n(0) = 1$, $n(1) = 2$, $n(B) = 3$, etc. The directions are numbered $n(L) = 1$ and $n(R) = 2$. Now every transition $\delta(q, X) = [r, y, d]$ is encoded $1^{n(q)}01^{n(X)}01^{n(r)}01^{n(Y)}01^{n(d)}00$. The encoding of M is obtained by concatenating the encodings of the transitions of M, prefixing this with 00, and postfixing it with a final 0. This has the effect that $R(M)$ contains precisely one substring 000, and this substring is at the end of the bit string.

(b: 4%) $L_U = \{R(M)w \mid M \in \mathit{TM0}, w \in \mathbb{B}^* : w \in L(M)\}$.
L_U consists of the bit strings $R(M)w$ that consist of an encoding of some Turing machine, say M, followed by an input string w such that M terminates on w.

(c: 2%) As L_U is accepted by a TM, it is semi-decidable. Turing's Halting Theorem states that L_U is not decidable.

Exercise 9 (12 %) Consider the language

$$L_9 = \{R(M) \mid M \in \mathit{TM0} : 1001 \in L(M)\}.$$

(a) Prove that the language L_9 is not decidable.
(b) Is the language L_9 semi-decidable? Justify your answer.

Solution (a: 9%) Proof by reduction to the Halting Theorem. Assume that L_9 is decidable. Then there is a simple always terminating TM M_9 that accepts L_9. We use M_9 to construct an always terminating Turing machine K that accepts the Halting language L_U of the previous exercise.

For an input string u, machine K first verifies that $u = R(M)w$ for some machine M and some bitstring w, just like the UTM. Otherwise K rejects u. Subsequently, K uses the encoding $R(M)$ and string w to construct the encoding $R(M')$ of a machine $M' \in \mathit{TM0}$ that does the following. Given input v, it first erases its input v on the tape, then writes w on the tape, and executes M of w. After the construction of the bitstring $R(M')$, machine K applies M_9 with input $R(M')$. As M_9 always terminates, K always terminates.

$$\begin{array}{ll}
 & K \text{ accepts } R(M)w \\
\equiv & M_9 \text{ accepts } R(M') \\
\equiv & 1001 \in L(M') \\
\equiv & M' \text{ terminates on } 1001 \\
\equiv & M \text{ terminates on } w.
\end{array}$$

Therefore, K solves the Halting problem. This contradicts Turing's Theorem. Therefore L_9 is not decidable.

(b: 3%) L_9 is semi-decidable, because it is accepted by the following TM by termination: the machine first verifies that its input is of the form $R(M)$ for some $M \in \mathit{TM0}$, then postfixes its input with the string 1001, and then applies the universal Turing machine to the string. This accepts by termination if and only if the input is in L_9.