CS21004 Formal Languages and Automata Theory, Spring 2011–12

End-Semester Test

Maximum marks: 60 | Date: 20–Apr–2012 | Duration: Three hours

---

Roll no: ________________ **Name:** ____

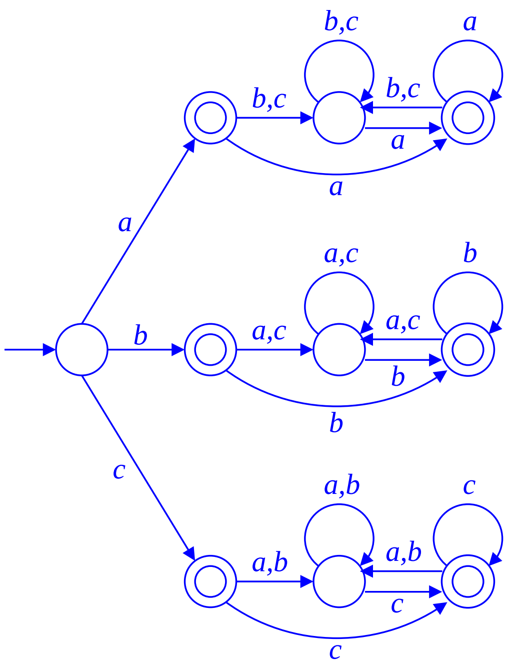
Write your answers in the question paper itself. Be brief and precise. Answer all questions. Use Pages 8–10 of this question paper to write answers or their parts that do not fit in the spaces provided. Do rough work in supplementary sheets. Any solutions (or parts) written in supplementary sheet(s), attached or loose, will not be evaluated. Please do not accept answer books from invigilators.

1. Design a DFA (*deterministic* finite automaton) to accept the language

$$L_1 = \left\{ \alpha \in \{a, b, c\}^* \mid \alpha \text{ starts and ends with the same symbol} \right\}.$$

Only draw the transition diagram, and clearly indicate the start state and the final state(s). **(10)**

Solution



2. Let L_2 denote the context-free language $\{\alpha\alpha^R \mid \alpha \in \{a, b\}^*\}$, where α^R stands for the reverse of the string α. Prove or disprove: The complement of L_2 (that is, $\sim L_2 = \{a, b\}^* \setminus L_2$) is context-free. (**Note:** Either construct a CFG/PDA to accept $\sim L_2$, or supply a proof based on the pumping lemma. Intuitive arguments will deserve no credit.) **(10)**

Solution $\sim L_2$ is context-free. It consists of strings of the following two types:

(a) All strings in $\{a, b\}^*$ of odd lengths.

(b) All strings $\alpha \in \{a, b\}^*$ of even lengths such that for some i, the i-th and the i-th last symbols in α are different.

Strings of type (a) can be generated by the following grammar with start symbol S_1:

$$S_1 \rightarrow aaS_1 \mid abS_1 \mid baS_1 \mid bbS_1 \mid a \mid b\,,$$

whereas strings of type (b) can be generated by the following grammar with start symbol S_2:

$$\begin{aligned} S_2 &\rightarrow aS_2a \mid bS_2b \mid T_2\,, \\ T_2 &\rightarrow aU_2b \mid bU_2a\,, \\ U_2 &\rightarrow aU_2a \mid aU_2b \mid bU_2a \mid bU_2b \mid \epsilon\,. \end{aligned}$$

Therefore, $\sim L_2$ is generated by the start symbol S with the added production

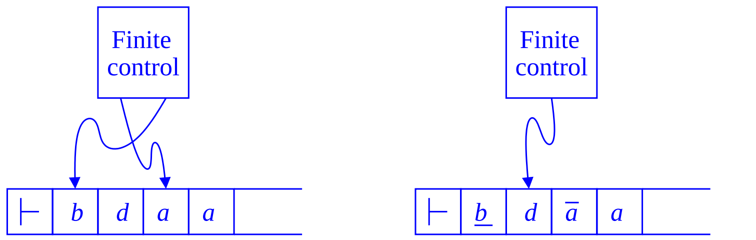
$$S \rightarrow S_1 \mid S_2\,.$$

Roll no: ____________ **Name:** ______________________________

3. Let M be a Turing machine with one semi-infinite tape and *two* read/write heads. Each transition of M is determined by the current state p of the finite control, and the two symbols a and b scanned by the two heads. A transition of M is of the form $\delta(p, a, b) = (q, c, d, D_1, D_2)$ implying that the finite control goes to state q, the symbol a at the cell pointed by the first head is replaced by c, and the symbol b at the cell pointed by the second head is replaced by d. If both the heads point to the same tape cell ($a = b$ in this case), then the symbol at this cell is replaced by c (not by d unless $c = d$). Finally, the first head moves by one cell in direction D_1 (left or right), and the second head moves by one cell in direction D_2.

Argue that this two-head Turing machine M can be simulated by a standard Turing machine N with one semi-infinite tape and with only one read/write head. **(10)**

Solution Let Γ be the tape alphabet of M. For each $a \in \Gamma$, introduce three new symbols $\overline{a}$, $\underline{a}$ and $\overline{\underline{a}}$. The tape alphabet of N consists of Γ and the three new symbols introduced for each $a \in \Gamma$. The symbol $\overline{a}$ in a cell indicates that the first head of M points to this cell which contains the symbol a, $\underline{a}$ indicates that the second head of M points to this cell, whereas $\overline{\underline{a}}$ indicates that both the heads point to this cell. In order to simulate a single move of M, N first locates the two markers ¯ and _, and remembers the corresponding symbols $a, b \in \Gamma$ in its finite control. N now consults the transition function of M, replaces a, b by appropriate symbols c, d, and moves the markers ¯ and _ as dictated by $\delta(p, a, b)$, where the state p of N is remembered in the finite control of M.



(a) A two−head machine (b) Simulation by a one−head machine

4. (a) Write an unrestricted grammar to accept the language

$$L_4 = \left\{ a^i b^j c^k d^l \mid i = k \text{ and } j = l \right\}.$$

Mention the start symbol of your grammar. Use upper-case Roman letters for non-terminal symbols. **(5)**

Solution The following grammar with start symbol S generates L_4:

$$\begin{array}{rcll} S & \rightarrow & aSC \mid T & \text{[Generate } a^i T C^k \text{ with } i = k] \\ T & \rightarrow & bTd \mid R & \text{[Generate } a^i b^j R d^l C^k \text{ with } i = k \text{ and } j = l] \\ dC & \rightarrow & Cd & \text{[Allow } C \text{ to move past } d] \\ RC & \rightarrow & cR & [C \text{ is converted to } c \text{ after it reaches the correct place]} \\ R & \rightarrow & \epsilon & \text{[After } R \text{ converts all } C\text{'s to } c\text{'s, it vanishes]} \end{array}$$

(b) Show a derivation of the string $a^2b^3c^2d^3$ according to your grammar. **(5)**

Solution The derivation of $a^2b^3c^2d^3$ and the rules used in the derivation process are given below.

$$\begin{array}{ll} S \Rightarrow aSC \Rightarrow aaSCC & [S \rightarrow aSC] \\ \quad \Rightarrow aaTCC & [S \rightarrow T] \\ \quad \Rightarrow aabTdCC \Rightarrow aabbTddCC \Rightarrow aabbbTdddCC & [T \rightarrow bTd] \\ \quad \Rightarrow aabbbRdddCC & [T \rightarrow R] \\ \quad \Rightarrow aabbbRddCdC \Rightarrow aabbbRdCddC \Rightarrow aabbbRCdddC & [dC \rightarrow Cd] \\ \quad \Rightarrow aabbbcRdddC & [RC \rightarrow cR] \\ \quad \Rightarrow aabbbcRddCd \Rightarrow aabbbcRdCdd \Rightarrow aabbbcRCddd & [dC \rightarrow Cd] \\ \quad \Rightarrow aabbbccRddd & [RC \rightarrow cR] \\ \quad \Rightarrow aabbbccddd & [R \rightarrow \epsilon] \end{array}$$

Roll no: ____________________ **Name:** ____

5. A *shuffle* of two strings α and β is a string γ of length $|\alpha| + |\beta|$, in which α and β are non-overlapping subsequences (not necessarily substrings). For example, all shuffles of ab and cd are $abcd$, $cabd$, $cdab$, $acbd$, $acdb$, and $cadb$. For two languages A and B, we define $\mathrm{shuffle}(A, B)$ as the language consisting of all shuffles of all $\alpha \in A$ and all $\beta \in B$. Prove that recursively enumerable languages are closed under the shuffle operation, that is, if A and B are r.e. languages, then so also is the language

$$\mathrm{shuffle}(A, B) = \{\gamma \mid \gamma \text{ is a shuffle of some } \alpha \in A \text{ and } \beta \in B\}.$$ **(10)**

Solution Let A and B be accepted by Turing machines M_1 and M_2, respectively. We design a Turing machine M for $\mathrm{shuffle}(A, B)$.

One possibility is to design M as a non-deterministic Turing machine with three tapes. Let $\gamma = c_1 c_2 \ldots c_n$ be an input for M (provided in its first tape). For each $i = 1, 2, \ldots, n$, M non-deterministically chooses whether c_i comes from α or from β. In the first case, M writes c_i to the second tape, and in the second case, M writes c_i to the third tape. After all c_i's are copied, M simulates M_1 on Tape 2 and M_2 on Tape 3 in parallel (in a round-robin fashion). If both the simulations accept, M accepts γ and halts. If any of the two simulations rejects, M also rejects γ and halts. If both the simulations loop, M continues the simulations for ever.

If one wants to design M as a deterministic Turing machine, one may construct a Turing machine with a two-dimensional tape, semi-infinite in both the directions. There are 2^n ways of writing an input $\gamma = c_1 c_2 \ldots c_n$ for M as the shuffle of α and β. M first writes all these possibilities (call them $(\alpha_1, \beta_1), (\alpha_2, \beta_2), \ldots, (\alpha_{2^n}, \beta_{2^n})$), in 2^{n+1} rows of its tape. Subsequently, M simulates M_1 or M_2 appropriately in each of these rows. All these simulations run in parallel (in a round-robin fashion). If two corresponding simulations (on inputs α_i, β_i for the same i) accept and halt, M too accepts and halts. If all pairs of simulations reject (either α_i is rejected by M_1, or β_i by M_2, or both), then M rejects and halts. Otherwise, M keeps on looping for ever.

6. Assume that Turing machines are encoded by strings over some alphabet Σ, and that $\# \notin \Sigma$. Consider the following language over the alphabet $\Sigma \cup \{\#\}$:

$$L_6 = \{M_1\#M_2\#M_3 \mid \ M_1, M_2, M_3 \text{ are Turing machines with } \mathcal{L}(M_1) \cap \mathcal{L}(M_2) = \mathcal{L}(M_3)\}\,.$$

(a) Prove that L_6 is not recursively enumerable. (**Note:** You must supply a complete reduction proof. No intuitive justification will be given any credit. Same for Part (b).) **(5)**

Solution Reduce ~HP to L_6, that is, given an input $M\#\alpha$ for HP, we plan to generate an input $M_1\#M_2\#M_3$ for L_6 such that $\mathcal{L}(M_1) \cap \mathcal{L}(M_2) = \mathcal{L}(M_3)$ if and only if M does not halt on α.

M_1, upon input β_1, accepts and halts.

M_2, upon input β_2, accepts and halts.

M_3, upon input β_3, simulates M on α for $|\beta_3|$ steps. If the simulation halts in $|\beta_3|$ steps, M_3 rejects and halts. Otherwise, M_3 accepts and halts.

We have $\mathcal{L}(M_1) = \mathcal{L}(M_2) = \Sigma^*$, so $\mathcal{L}(M_1) \cap \mathcal{L}(M_2) = \Sigma^*$. On the other hand, $\mathcal{L}(M_3) = \Sigma^*$ if M does not halt on α. Finally, if M halts on α in n steps, then $\mathcal{L}(M_3) = \{\beta_3 \in \Sigma^* \mid |\beta_3| < n\} \neq \Sigma^*$.

(b) Prove that $\sim L_6$ (that is, the complement of L_6 in $(\Sigma \cup \{\#\})^*$) is not recursively enumerable. **(5)**

Solution Reduce $\sim$HP to $\sim L_6$, or equivalently, HP to L_6, that is, given an input $M\#\alpha$ for HP, we plan to generate an input $M_1\#M_2\#M_3$ for L_6 such that $\mathcal{L}(M_1) \cap \mathcal{L}(M_2) = \mathcal{L}(M_3)$ if and only if M halts on α.

M_1, upon input β_1, accepts and halts.

M_2, upon input β_2, accepts and halts.

M_3, upon input β_3, erases β_3 and simulates M on α. If the simulation halts, M_3 accepts and halts.

We have $\mathcal{L}(M_1) = \mathcal{L}(M_2) = \Sigma^*$, so $\mathcal{L}(M_1) \cap \mathcal{L}(M_2) = \Sigma^*$. On the other hand, $\mathcal{L}(M_3) = \Sigma^*$ if M halts on α, and $\mathcal{L}(M_3) = \emptyset$ if M does not halt on α.

Write left-over answers here with appropriate pointers earlier.