CoCo - Exam 2021

Johannes Agerskov

Dated: April 15, 2021

Some of the problem describtions below have been copied from the exam sheet.

Question 1

Consider the following languages over the alphabet $\Sigma = \{a, b, c\}$:

$$
\begin{aligned}
L_1 &= \{w \in \Sigma^* : w \text{ contains an odd number of occurrences of the letter } a\} \\
L_2 &= \{a^m b^n c^{m+n} : m, n \in \mathbb{N}\} \\
L_3 &= \{a^m c^{m+n} b^n : m, n \in \mathbb{N}\} \\
L_4 &= \left\{a^m c^{m^2 n^2} b^n : m, n \in \mathbb{N}\right\}
\end{aligned}
$$

Part 1.1

We show that L_1 is regular, and it thus follows that it also is context-free by Corollary 2.32 in M. Sipser.

Proof. The DFA in Figure 1 recognizes L_1, it then follows from definition 1.16 that L_1 is regular. □

Part 1.2

We show that L_2 is not regular, but it is context-free.

Proof. That is L_2 is not regular can be seen by the following proof by contradiction: Assume L_2 is regular, and let p denote the pumping length as given by the pumping lemma for regular languages Theorem 1.70. Consider now the string $w = a^p b^p c^{2p} \in L_2$, which clearly satisfies $|w| > p$. Thus by the pumping lemma we can split $w = xyz$ with $|xy| \leq p$ and $|y| > 0$. Clearly we then have $xy = a^k$ for some $0 < k \leq p$, and therefore $y = a^m$ for some $0 < m \leq p$. But then clearly $xy^i z = a^{p+(i-1)m} b^p c^{2p}$ which for $i \neq 1$ is not in L_2, contradicting the pumping lemma. Thus we conclude that L_2 is not regular.

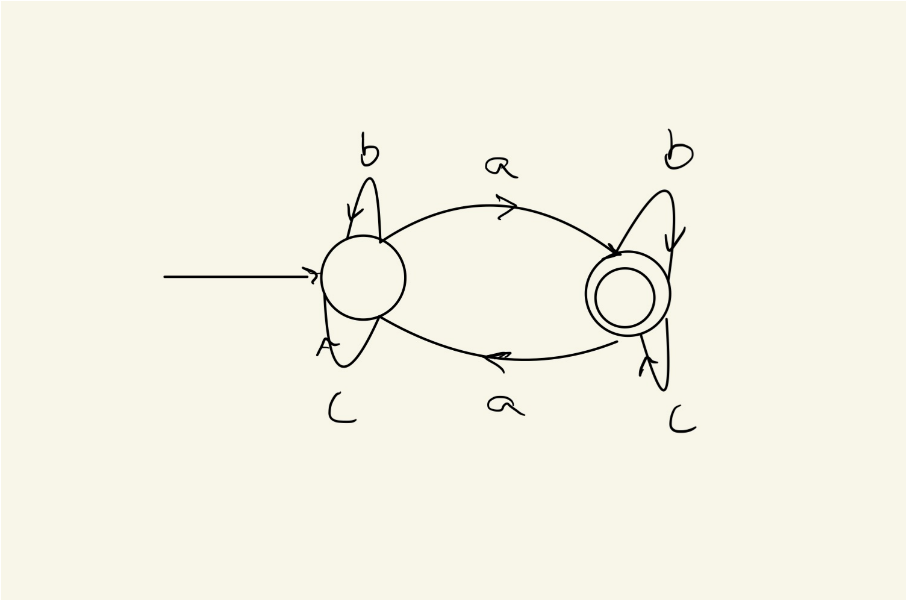

Figure 1: DFA that reckognizes L_1

That L_2 is context-free can be seen by the fact that it is generated by the CFG

$$S \to aAc,$$
$$A \to aAc \mid bBc,$$
$$B \to bBc \mid \epsilon.$$

Notice that it is not clear wether $\mathbb{N}$ contains 0 or not. Since the concention in M. Sipser is $\mathbb{N} = \{1, 2, 3, ...\}$ that is what is used here. But even if $\mathbb{N} = \{0, 1, 2, ...\}$ L_2 is still context-free since then if is generated by CFG

$$S \to aAc \mid \epsilon,$$
$$A \to aAc \mid B,$$
$$B \to bBc \mid \epsilon.$$

□

Part 1.3

We show that L_3 is not regular, but it is context-free.

Proof. The proof essentially goes as that for L_2.
That is L_3 is not regular can be seen by the following proof by contradiction: Assume L_3 is regular, and let p denote the pumping length as given by the pumping lemma for regular languages Theorem 1.70. Consider now the string $w = a^p b^{2p} c^p \in L_3$, which clearly satisfies $|w| > p$. Thus by the pumping lemma we can split $w = xyz$ with $|xy| \leq p$ and $|y| > 0$. Clearly we then have $xy = a^k$ for some $0 < k \leq p$, and therefore $y = a^m$ for some $0 < m \leq p$. But then clearly $xy^i z = a^{p+(i-1)m} b^{2p} c^p$ which for $i \neq 1$ is not in L_3, contradicting the pumping lemma.

Thus we conclude that L_3 is not regular.
That L_3 is context-free can be seen by the fact that it is generated by the CFG

$$S \to aAc^2Cc,$$
$$A \to aAc \mid \epsilon,$$
$$C \to cCb \mid \epsilon.$$

if $\mathbb{N} = \{1, 2, 3, ...\}$ and by CFG

$$S \to AC,$$
$$A \to aAc \mid \epsilon,$$
$$C \to cCb \mid \epsilon.$$

if $\mathbb{N} = \{0, 1, 2, 3, ...\}$ □

Part 1.4

We show that L_4 is not context-free, and it then clearly follows from Corollary 2.32 that L_4 is not regular.

Proof. Assume that L_4 is context-free, and let p be the pumping length as given by the pumping lemma for context-free languages Theorem 2.34. Consider then the string $w = a^p c^{p^2+1} b$. By the pumping lemma we may split $w = uvxyz$ where $|vy| > 0$ and $|vxy| \leq p$. Thus either $vxy = a^k$ for some $0 < k \leq p$ in which case $vy = a^m$ for some $0 < m \leq p$, but then $uv^ixy^iz = a^{p+(i-1)m}c^{p^2+1}b$ which is not in L_4 for $i \neq 1$, contradicting the pumping lemma. Or $vxy = a^k c^l$ for some $0 < k + l \leq p$, but then $vy = a^s c^t$ for some $0 < k + l \leq p$ and $uv^ixy^iz = a^{p+(i-1)s}c^{p^2+1+(i-1)t}b$, but clearly there exist for any s, t such that $s > 0$ an $i \in \{0, 1, 2, ...\}$ such that $(p + (i-1)s)^2 > p^2 + 1 + (i-1)t$, and thus for such an i, uv^ixy^iz is not in L_4 contradicting the pumping lemma and if $s = 0$ we obviously have $uv^ixy^iz = a^{p+(i-1)s}c^{p^2+1+(i-1)t}b \notin L_4$ also contradicting the pumping lemma. Or we may have $vxy = c^k$ for some $0 < k \leq p$, in which case $vy = c^m$ for some $0 < m \leq p$, but then $uv^ixy^iz = a^p c^{p^2+1+(i-1)m}b$ which is not in L_4 for $i \neq 1$ contradicting the pumping lemma. Finally we may have $vxy = c^k b$ in which case the contradicting is obtained by the same method as for $vxy = a^k c^l$. Thus a contradiction is unavoidable, and we conclude that L_4 is not context-free. □

Part 1.5

We show now that L_4 belongs to L.

Proof. We assume that $\mathbb{N} = \{1, 2, 3, ...\}$, but the proof works for $\mathbb{N} = \{0, 1, 2, ...\}$ with small modifications. Consider the log-space TM, $M =$"On input w

1. Scan the input, and compare neighboring letters (by storing and overwriting one letter on the worktape all the way). If ever substring ab, ca, ba or bc is found, *reject*.

2. Count the number of as, bs and cs with three counters, i, j, k respectively, on the worktape.

3. Check that $i^2 + j^2 = k$. If yes *accept*, if no, *reject*."

Evidently this M decides L_4. Furthermore, it is log-space, as the first step requires only one slot on the worktape, step 2 requires only tree counters (in binary) which take up only logarithmic space. And finally we may multiply $i \cdot i$ by adding i, i times, which can be done by having an extra counter keeping track of how many times we have added i. Of course we also need the obviuous seperator symbols, which can clearly be included in log space. Thus we see that M run in logarithmic space, and we conclude that L_4 is in L. $\square$

Question 2

For any string w we let $rev(w)$ denote the reverse of w.

Part 2.1

We consider in the following the language

$$\text{REV}_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ is a TM such that at some point during the execution of } M \text{ on } w \\ \text{the tape of } M \text{ consists of } rev(w) \text{ followed by only blank symbols }\}$$

Part 2.2

We show that REV_{TM} is Turing recognizable.

Proof. The following TM, recognizes REV_{TM}, M' ="On input $\langle M, w\rangle$

1. Simulate M on w

2. While M is running:

3. keep a counter, i, of how many computation steps of M have been simulated.

4. In each computation step of M, compare the first $\min(i, |w|)$ entries of the tape to $rev(w)$, if at some point the tape content mathces $rev(w)$ *accept*. If M halts without $rev(w)$ appearing on the tape, *reject*.

Clearly M' accept if and only if there is some point where the tape content of M running on w mathces $rev(w)$. Notice that the counter i ensures that the checking in each step can be done in finite time, and also that the tape content of M clearly cannot be anything but blank beyond i, since M have not have time to alter these tape cells yet. $\square$

Part 2.3

We show that $\mathrm{REV}_{\mathrm{TM}}$ is not decidable.

Proof. This follows by simply noting that we can reduce A_{TM} to REV_{TM} by the following reduction: For any $\langle M, w\rangle$ construct $\langle \tilde{M}, w\rangle$, where $\tilde{M}$ is equivalent to M but it starts by writing a special character, which is not in the input alphabet, on the tape, say $\aleph$, which does not interfere with the computation, and it erases the tape (including $\aleph$) and writes $rev(w)$ on the tape before accepting. Then clearly $\langle \tilde{M}, w\rangle \in \mathrm{REV}_{\mathrm{TM}}$ if and only if $\langle M, w\rangle \in A_{TM}$. Furthermore, this reduction is clearly done by a computatble function. Thus we conclude that $A_{TM} \leq \mathrm{REV}_{\mathrm{TM}}$ and it follows by Theorem 4.11 and Corollary 5.23 that $\mathrm{REV}_{\mathrm{TM}}$ is undecidable. $\square$

Part 2.4

Let $f : \Sigma^* \to \Sigma^*$ be a computable function. We show that there is a TM M with the property that $f(\langle M\rangle)$ is a description of a TM M' such that for each $w \in \Sigma^*$,

1. M halts on w if and only if M' halts on w, and
2. if M' halts on w with a string s_w on its tape then M halts on w with the string $s_w s_w$ on its tape.

Proof. Consider the following TM, $M =$"On input w

1. Obtain own describtion via recursion theorem (Thm 6.3).
2. Compute $f(M)$, and let G be the TM such that $\langle G\rangle = f(M)$.
3. Simulate G on w. If G halts, erase everything on the tape exept the tape content of G and duplicate the tape content. *Accept* if G accepted and *reject* if G rejected.

Clearly, if $f(M)$ is a describtion of the TM M' we see that M halts if and only if M' halts, since M simulates M' in its describtion. Also we see that by design, M will have exactly $s_w s_w$ on the tape when halting on w, if M' have s_w on the tape when halting on w. Thus M is exactly a describtion of the desired TM. $\square$

Question 3

For a directed graph $G = (V, E)$ and a subset V' of V, the induced subgraph $G[V']$ is the subgraph of G with vertex set V' and edge set consisting of those edges of E with both endpoints in V'.

When we refer to a cycle in the following, we refer to a directed graph consisting of distinct vertices $v_1, v_2, \ldots, v_m$ and edges (v_i, v_{i+1}) for $i = 1, 2, \ldots, m-1$, and an edge (v_m, v_1); m is the size of the cycle. Let $s \geq 2$ be a given integer and consider the following decision problem:

Question 4