PART II AUTOMATA AND FORMAL LANGUAGES
MICHAELMAS 2017-18
SOLUTIONS TO EXAMPLE SHEET 3

$*$ denotes a harder problem.

(1) Construct ϵ-NFA's and regular expressions for the following regular languages:

(a) All words $w \in \{0,1\}^*$ consisting of either the string 01 repeated some number of times (possibly none), or the string 010 repeated some number of times (possibly none).

(b) All words $w \in \{a,b,c\}^*$ consisting of some number of a's (possibly none), followed by some number of b's (at least one), followed by some number of c's (possibly none).

(c) All words $w \in \{0,1\}^*$ which contain a 1 somewhere in the last 4 positions. If $|w| < 4$, then w must contain a 1 somewhere.

A: These are all fairly obvious. Try and encourage the students to use ϵ-transitions to make the constructions easier.
For the regular expressions, use the following:
1a: $(\mathbf{01})^* + (\mathbf{010})^*$
1b: $\mathbf{a}^*\mathbf{b}\mathbf{b}^*\mathbf{c}^*$
1c: $(\mathbf{0}+\mathbf{1})^*(\mathbf{1}+\mathbf{10}+\mathbf{100}+\mathbf{1000})$. (I assume some students will write out ALL strings of length 4 with at least one occurence of 1; this is not necessary).

(2) Convert each of the following regular expressions to ϵ-NFA's:

(a) $(\mathbf{0}+\mathbf{1})(\mathbf{01})$

(b) $(\mathbf{a}+\mathbf{bb})^*(\mathbf{ba}^*+\epsilon)$

(c) $((\mathbf{aa}^*)^*\mathbf{b})^*+\mathbf{c}$

A: Apply the algorithm from Theorem 2.30 in the printed notes. These are very straightforward.

(3) Prove that $\{w \in \{0,1\}^* \mid w$ contains no more than 5 consecutive 0's$\}$ is regular.
A: It is easiest to show that the complement is regular, and then use the fact that the complement of a regular language is regular. So, we need only show $\{w \in \{0,1\}^* \mid w$ contains 000000$\}$ is regular. But this is precisely the language given by the regular expression $(\mathbf{0}+\mathbf{1})^*(\mathbf{00000})(\mathbf{0}+\mathbf{1})^*$, and so is regular.

Maurice Chiodo, mcc56@cam.ac.uk, November 10, 2017.

Try to encourage the students to look for shortcuts wherever possible; we spent a lot of time in the lectures showing that DFA's, NFA's, ϵ-NFA's, and regular expressions all define the same set of languages, so the students should learn how to use the most appropriate one for the given problem.

(4) Let R, S, T be regular expressions. For each of the following statements, either prove that it is true, or find a specific counterexample.

(a) $\mathcal{L}(R(S+T)) = \mathcal{L}(RS) + \mathcal{L}(RT)$
A: True (proof is elementary)

(b) $\mathcal{L}((R^*)^*) = \mathcal{L}(R^*)$
A: True (proof is elementary)

(c) $\mathcal{L}((RS)^*) = \mathcal{L}(R^*S^*)$
A: False: $(\mathbf{01})^* \neq \mathbf{0}^*\mathbf{1}^*$

(d) $\mathcal{L}((R+S)^*) = \mathcal{L}(R^*) + \mathcal{L}(S^*)$
A: False: $(\mathbf{0}+\mathbf{1})^* \neq \mathbf{0}^* + \mathbf{1}^*$

(e) $\mathcal{L}((R^*S^*)^*) = \mathcal{L}((R+S)^*)$
A: True (proof is elementary)

(5) Use the pumping lemma to show that none of the following languages are regular:

(a) $\{a^n b^n \mid n \geq 0\}$
A: Let N be as in the pumping lemma. Pump the word $a^N b^N$ up to $a^{N+l}b^N$ for some $l > 0$. This example was done explicitly in lectures.

(b) $\{a^{2n} b^{2n} \mid n \geq 0\}$
A: Let N be as in the pumping lemma. Pump the word $a^{2N} b^{2N}$ up to $a^{2N+l}b^{2N}$ for some $l > 0$.

(c) $\{ww \mid w \in \{0,1\}^*\}$
A: Let N be as in the pumping lemma. Pump the word $0^N 1^N 0^N 1^N$ up to $0^{N+l}1^N 0^N 1^N$ for some $l > 0$.

(6) For each of the following languages, determine whether or not they are regular. Justify your answers.

(a) $\{a^n b^{2n} \mid n \geq 0\}$
A: No. Let N be as in the pumping lemma. Pump the word $a^N b^{2N}$ up to $a^{N+l}b^{2N}$ for some $l > 0$.

(b) $\{a^n b^m \mid n \neq m\}$
A: No. It is related to the complement of the language in 5a, which is not regular. More specifically: let K be the language in 5a, and L the language above. Then $K = (\{a,b\}^* \setminus L) \cap \mathcal{L}(\mathbf{a}^*\mathbf{b}^*)$; complements and intersections of regular languages are regular, so

if L were regular then K would be regular also.
Emphasise to the students that this sort of argument is sufficient (rather than reinventing the wheel).

(c) $\{xcx \mid x \in \{a,b\}^*\}$
A: No. Let N be as in the pumping lemma. Pump the word $a^N ca^N$ up to $a^{N+l}ca^N$ for some $l > 0$.

(d) $\{xcy \mid x, y \in \{a,b\}^*\}$
A: Yes. This is precisely $\mathcal{L}((\mathbf{a}+\mathbf{b})^*\mathbf{c}(\mathbf{a}+\mathbf{b})^*)$

(e) $\{a^n b^m \mid n > m\}$
A: No. Let N be as in the pumping lemma. Pump the word $a^{N+1}b^N$ down to $a^{N-l}b^N$ for some $l \geq 0$. This is an important example of when we need to pump *down*, not up.

(f) $\{a^n b^m \mid n \geq m \text{ and } m \leq 1000\}$
A: Yes. We can informally describe an NFA which accepts this language as follows: Start with state q_0. Take a set of states $q_1, \ldots, q_{1000}$. Have a transition labelled a from q_i to q_{i+1} for each $0 \leq i < 1000$. Then have a transition labelled a from q_{1000} to itself. Now, for each q_i, we introduce a family of states $q_{i,1}, \ldots, q_{i,i}$. Then we have transitions labelled b from q_1 to $q_{i,1}$, and from $q_{i,j}$ to $q_{i,j+1}$ for each $1 \leq j \leq i-1$. There are no other states or transitions, and each state above should be made an accepting state.
Another way to look at this set X is to observe:

$$X = \cap_{i=0}^{1000}\{a^n b^i \mid n \geq i\} = \cap_{i=0}^{1000}\mathcal{L}(\mathbf{a}^*\mathbf{ab})$$

(g) $\{a^n b^m \mid n \geq m \text{ and } m \geq 1000\}$
A: No. Let N be as in the pumping lemma. Pump the word $a^{N+1000}b^{N+1000}$ down to $a^{N+1000-l}b^{N+1000}$ for some $l > 0$.

(7) Prove that no infinite subset of $\{0^n 1^n \mid n \geq 0\}$ is a regular language.
A: Let X be an infinite subset of $\{0^n 1^n \mid n \geq 0\}$. Let N be as in the pumping lemma. Take some $M \geq N$ with $a^M b^M \in X$; such an M must exist as X is infinite. Now pump the word $a^M b^M$ up to $a^{M+l}b^M$ for some $l > 0$.

(8) Find minimal DFA's for each of the following languages. In each case, *prove* that your DFA is minimal.

(a) $\{a^n \mid n \geq 0,\ n \neq 3\}$

(b) $\{a^m b^n \mid m \geq 2,\ n \geq 3\}$

(c) $\{a^m b \mid m \geq 0\} \cup \{b^n a \mid n \geq 0\}$

A: I'm not going to write these out here. The point is that the students need to show, for each DFA that they construct, that no pair of states are equivalent (as then the minimised DFA will have the same number of states as the original DFA). A full-blown table filling algorithm is not completely necessary, but I suspect many will do it this way.

Also, remind students that for a given pair of states p, q, if one is accepting and the other isn't, then they are distinguishable (by ϵ). Thus, they only need to find distinguishing elements for pairs of states which are either both accepting or both non-accepting.

(9) If $D_1 = (Q, \Sigma, \delta, q_0, F)$ is a minimal DFA, and $D_2 = (Q, \Sigma, \delta, q_0, Q \setminus F)$ is a DFA for $\Sigma^* \setminus \mathcal{L}(D_1)$, then is D_2 necessarily a minimal DFA? Prove your answer.
A: Prove by contradiction. Suppose D_2 was not minimal. Take a minimal DFA D_3 for $\Sigma^* \setminus \mathcal{L}(D_1)$; then D_3 has fewer states than D_2 (and thus fewer than D_1). But then swapping the accept / non-accept states in D_3 gives a DFA for $\Sigma^* \setminus (\Sigma^* \setminus \mathcal{L}(D_1)) = \mathcal{L}(D_1)$ with FEWER states than D_1; a contradiction as D_1 was a minimal DFA.

(10) Let L, M be languages over Σ, Γ respectively. We define the *difference* $L - M$ to be the words that are in L but not M. That is, $L - M := (L \cup M) \setminus M$. Show that if L, M are both regular languages, then $L - M$ is a regular language over Σ.
A: Observe that, as M is regular, then so is its complement $\overline{M}$ (taking the complement in Γ^*). So $L \cap \overline{M}$ is also regular, over $\Sigma \cup \Gamma$. But $L - M = L \cap \overline{M}$ (in $(\Sigma \cup \Gamma)^*$). Finally, it is clear that $L - M$ must lie in Σ^*, by definition.

(11) Find a regular expression for $\{w \in \{0,1\}^* \mid w \text{ is a multiple of 3 when interpreted in binary }\}$.
Hint: Find a suitable DFA, and then convert it to a regular expression.
A: Construct a DFA D with:
- Alphabet $\{0, 1\}$.
- States $\{1, 2, 3\}$ (corresponding to remainders 0 mod 3, 1 mod 3, and 2 mod 3 respectively).
- Start state 1.
- Accept state 1.
- Transition function: $(1, 0) \mapsto 1$, $(1, 1) \mapsto 2$, $(2, 0) \mapsto 3$, $(2, 1) \mapsto 1$, $(3, 0) \mapsto 2$, $(3, 1) \mapsto 3$.
Note that I am accepting 'leading 0's'; i.e., 000101 is a binary representation for 5.

Now, a regular expression for this language will be $R^{(3)}_{1,1}$. Recall we have the recursive definition

$$R^{(k)}_{i,j} := R^{(k-1)}_{i,j} + R^{(k-1)}_{i,k}(R^{(k-1)}_{k,k})^* R^{(k-1)}_{k,j}$$

So thus we are looking for

$$R^{(3)}_{1,1} := R^{(2)}_{1,1} + R^{(2)}_{1,3}(R^{(2)}_{3,3})^* R^{(2)}_{3,1}$$

We build this up indictively:
$R^{(0)}_{1,1} = \epsilon + \mathbf{0}$
$R^{(0)}_{1,2} = \mathbf{1}$
$R^{(0)}_{1,3} = \emptyset$
$R^{(0)}_{2,1} = \mathbf{1}$
$R^{(0)}_{2,2} = \epsilon$

$R^{(0)}_{2,3} = \mathbf{0}$
$R^{(0)}_{3,1} = \emptyset$
$R^{(0)}_{3,2} = \mathbf{0}$
$R^{(0)}_{3,3} = \epsilon + \mathbf{1}$

Now for the next step:

$R^{(1)}_{1,1} = R^{(0)}_{1,1} + R^{(0)}_{1,1}(R^{(0)}_{1,1})^* R^{(0)}_{1,1} = \ldots = \mathbf{0}^*$
$R^{(1)}_{1,2} = R^{(0)}_{1,2} + R^{(0)}_{1,1}(R^{(0)}_{1,1})^* R^{(0)}_{1,2} = \ldots = \mathbf{1} + \mathbf{0}^*\mathbf{1}$
$R^{(1)}_{1,3} = R^{(0)}_{1,3} + R^{(0)}_{1,1}(R^{(0)}_{1,1})^* R^{(0)}_{1,3} = \ldots = \emptyset$
$R^{(1)}_{2,1} = R^{(0)}_{2,1} + R^{(0)}_{2,1}(R^{(0)}_{1,1})^* R^{(0)}_{1,1} = \ldots = \mathbf{1} + \mathbf{1}(\mathbf{0})^*$
$R^{(1)}_{2,2} = R^{(0)}_{2,2} + R^{(0)}_{2,1}(R^{(0)}_{1,1})^* R^{(0)}_{1,2} = \ldots = \epsilon + \mathbf{1}(\mathbf{0})^*\mathbf{1}$
$R^{(1)}_{2,3} = R^{(0)}_{2,3} + R^{(0)}_{2,1}(R^{(0)}_{1,1})^* R^{(0)}_{1,3} = \ldots = \mathbf{0}$
$R^{(1)}_{3,1} = R^{(0)}_{3,1} + R^{(0)}_{3,1}(R^{(0)}_{1,1})^* R^{(0)}_{1,1} = \ldots = \emptyset$
$R^{(1)}_{3,2} = R^{(0)}_{3,2} + R^{(0)}_{3,1}(R^{(0)}_{1,1})^* R^{(0)}_{1,2} = \ldots = \mathbf{0}$
$R^{(1)}_{3,3} = R^{(0)}_{3,3} + R^{(0)}_{3,1}(R^{(0)}_{1,1})^* R^{(0)}_{1,3} = \ldots = \epsilon + \mathbf{1}$

Recall that we need only build $R^{(2)}_{1,1}$, $R^{(2)}_{1,3}$, $R^{(2)}_{3,3}$ and $R^{(2)}_{3,1}$ in the final stage. So:

$R^{(2)}_{1,1} = R^{(1)}_{1,1} + R^{(1)}_{1,2}(R^{(1)}_{2,2})^* R^{(1)}_{2,1} = \ldots = \mathbf{0} + (\mathbf{0}^*\mathbf{1})(\mathbf{1}(\mathbf{0})^*\mathbf{1})^*(\mathbf{1}(\mathbf{0})^*)$
$R^{(2)}_{1,3} = R^{(1)}_{1,3} + R^{(1)}_{1,2}(R^{(1)}_{2,2})^* R^{(1)}_{2,3} = \ldots = (\mathbf{0}^*\mathbf{1})(\mathbf{1}(\mathbf{0})^*\mathbf{1})^*\mathbf{0}$
$R^{(2)}_{3,3} = R^{(1)}_{3,3} + R^{(1)}_{3,2}(R^{(1)}_{2,2})^* R^{(1)}_{2,3} = \ldots = \epsilon + \mathbf{1} + \mathbf{0}(\mathbf{1}(\mathbf{0})^*\mathbf{1})^*\mathbf{0}$
$R^{(2)}_{3,1} = R^{(1)}_{3,1} + R^{(1)}_{3,2}(R^{(1)}_{2,2})^* R^{(1)}_{2,1} = \ldots = \mathbf{0}(\mathbf{1}(\mathbf{0})^*\mathbf{1})^*(\mathbf{1} + (\mathbf{0})^*)$

Finally, we get

$$R^{(3)}_{1,1} = \big(\mathbf{0}+(\mathbf{0}^*\mathbf{1})(\mathbf{1}(\mathbf{0})^*\mathbf{1})^*(\mathbf{1}(\mathbf{0})^*)\big)+\big((\mathbf{0}^*\mathbf{1})(\mathbf{1}(\mathbf{0})^*\mathbf{1})^*\mathbf{0}\big)\big(\mathbf{1}+\mathbf{0}(\mathbf{1}(\mathbf{0})^*\mathbf{1})^*\mathbf{0}\big)^*\big(\mathbf{0}(\mathbf{1}(\mathbf{0})^*\mathbf{1})^*(\mathbf{1}+(\mathbf{0})^*)\big)$$

which is the desired regular expression (though you may wish to double-check my working here....)

There is an easier way to do this, via 'elimination of states' (See Hopcroft, Section 3.2.2.) Doing this yields:

$$\big(\mathbf{0} + \mathbf{11} + \mathbf{10}(\mathbf{1} + \mathbf{00})^*\mathbf{01}\big)^*$$

or, even shorter still,

$$(\mathbf{0} + \mathbf{1}(\mathbf{01}^*\mathbf{0})^*\mathbf{1})^*$$

However, this was not taught in the lecures.

(12) Let D be a DFA with N states. Prove the following:

(a) If D accepts at least one word, then D accepts a word of length less than N.
A: If $|w| \geq N$ is an accepted word, then by the pumping lemma (using N) we can pump w down to a shorter accepted word. We keep repeating this process until we reach an accepted word v whose length is less than N.

(b) If D accepts at least one word of length $\geq N$, then D accepts infinitely many words.

A: If $|w| \geq N$ is an accepted word, then by the pumping lemma (using N) we can pump w up to an arbitrarily long accepted word. Thus D accepts infinitely many words.

(13) Is the following language regular:

$$\{1^p \mid p \text{ is a prime } \}$$

Justify your answer.
A: No. Let N be as in the pumping lemma. Take a prime $q > N$. Look at the word 1^q. By the pumping lemma, there is a subword $y = 1^k$ which can be pumped, where $1^q = xyz$. Now pump y to y^{q+1}. Then $xy^{q+1}z = xyzy^q = 1^q 1^{kq} = 1^{q(k+1)}$, whose length is $q(k+1)$ and thus not prime.

(14) Give an algorithm that, on input of a DFA D, decides if $\mathcal{L}(D) = \emptyset$ or not.
A: Take any DFA A on the same alphabet as D, but with no accept states. Then $\mathcal{L}(A) = \emptyset$. Now run the algorithm which compares whether A, D define the same regular language or not.

(15*) Give an algorithm that, on input of DFA's D_1, D_2, decides if $\mathcal{L}(D_1) \subseteq \mathcal{L}(D_2)$ or not.
(You may appeal to results from the lectures.)
A: The first point to note is that $\mathcal{L}(D_1) \subseteq \mathcal{L}(D_2)$ iff $\mathcal{L}(D_1) \cap \mathcal{L}(D_2) = \mathcal{L}(D_1)$. So we need only compare the regular languages $\mathcal{L}(D_1)$ and $\mathcal{L}(D_1) \cap \mathcal{L}(D_2)$. This requires constructing, from D_1 and D_2, a DFA for $\mathcal{L}(D_1) \cap \mathcal{L}(D_2)$. Set $L := \mathcal{L}(D_1)$, $M := \mathcal{L}(D_2)$, and suppose that D_1, D_2 have alphabets Σ, Γ respectively.
First, construct DFA's A_1, A_2 for $(\Sigma \cup \Gamma)^* \backslash L$ and $(\Sigma \cup \Gamma)^* \backslash M$ respectively by adding extra alphabet and extra dummy state / dummy transitions to D_1 and D_2, and then interchanging their accept and reject states. Then convert these to regular expressions R_1, R_2 respectively. So $R_1 + R_2$ is a regular expression for the language $\mathcal{L}(R_1 + R_2) = ((\Sigma \cup \Gamma)^* \setminus L) \cup ((\Sigma \cup \Gamma)^* \setminus M)$. Now convert $R_1 + R_2$ to a DFA B, which will have alphabet $\Sigma \cup \Gamma$ (adding dummy states/transitions, if necessary). Now interchange the accept and reject states of B to get a DFA C for $(\Sigma \cup \Gamma)^* \setminus \mathcal{L}(B)$. So in summary, we have:

$$\begin{aligned}
\mathcal{L}(C) &= (\Sigma \cup \Gamma)^* \setminus \mathcal{L}(B) \\
&= (\Sigma \cup \Gamma)^* \setminus (\mathcal{L}(R_1) \cup \mathcal{L}(R_2)) \\
&= \big((\Sigma \cup \Gamma)^* \setminus \mathcal{L}(R_1)\big) \cap \big((\Sigma \cup \Gamma)^* \setminus \mathcal{L}(R_2)\big) \quad \text{(by De Morgan's laws)} \\
&= \big((\Sigma \cup \Gamma)^* \setminus \mathcal{L}(A_1)\big) \cap \big((\Sigma \cup \Gamma)^* \setminus \mathcal{L}(A_2)\big) \\
&= \big((\Sigma \cup \Gamma)^* \setminus ((\Sigma \cup \Gamma)^* \setminus L)\big) \cap \big((\Sigma \cup \Gamma)^* \setminus ((\Sigma \cup \Gamma)^* \setminus M)\big) \\
&= L \cap M \quad \text{(by De Morgan's laws)}
\end{aligned}$$

So we have constructed a DFA C for the regular language $L \cap M = \mathcal{L}(D_1) \cap \mathcal{L}(D_2)$. Now simply compare if C and D_1 define the same regular language, as done in the lectures (Theorem 2.55).

(16*) For any $X \subseteq \{1\}^*$, show that X^* is a regular language.

A: If $X = \emptyset$ or $\{\epsilon\}$ then $X^* = \{\epsilon\}$ which is a regular language. If X contains 1, then $X^* = \{1\}^*$ which is a regular language (given by the regular expression $\mathbf{1}^*$). So now suppose that X contains words of length at least 2, but no shorter words. Let m be the gcd of the set of lengths of words in X (such a number *exists*, even if we can't compute it explicitly). Then there exists some k (see below) for which, beyond km, X^* is precisely all words of the form 1^{ln} for some l. That is, $X^* \cap \{1^t \mid t \geq km\} = \{1^{sm} \mid s \geq k\}$. But this language is given by the regular expression $\mathbf{1}^{km}(\mathbf{1}^m)^*$, and so is a regular language (call this language L_1). Moreover, the remaining part of X^*, consisting of its words of length less than km, is finite, and thus a regular language (call this language L_2). Thus, $X^* = L_1 \cup L_2$, the union of two regular languages. So X^* is a regular language.

Emphasise to the students that this is a non-constructive proof; there is a DFA for each such X, but we may not be able to algorithmically construct D from X as we can't construct k. The existence of such a k relates to something known as the *coin problem* (also known as *Schur's theorem* in combinatorics).