CS 4810: Homework 2

due 09/12 midnight

Matvey Soloviev ms2837

Collaborators: Would be kind of nice...

Each problem is worth 20 points.

Problem 1

Show that a Boolean function can be represented by a straight-line program of length at most $\ell$ if and only if it can be represented by a Boolean circuit of size at most $\ell$. (See the lecture notes for the definition of a straight-line program.)

Problem 2

Show that every finite subset of $\{0, 1\}^*$ is the language accepted by some DFA.

Solution

Suppose $S$ is the subset in question. Let $n$ be the length of a longest string in $S$; as $S$ is of finite cardinality and each of its elements is of finite length, $n$ is also well-defined and finite.

Construct a full rooted binary tree of states of depth $n + 1$, with the two transitions from each node to its children labelled 0 and 1 respectively; due to the acyclicity of trees, each state in the tree is then reached by reading in exactly one string of length up to $n$, and each such string lands us in a well-defined state.

We can now make every state which is reached after reading in a string in the language accepting, resulting in those strings (and no others) being accepted by the automaton; finally, to ensure that there is a transition for every character being observed at every state, we add one additional non-accepting state with 0- and 1-labelled transitions to itself and 0- and 1- labelled transitions from
each leaf node to absorb any excessively long strings that might be read by the
automaton.

Formally, we can define the DFA as follows:

\[
Q = \{ w \in \{0, 1\}^* \mid |w| \leq \max\{|x| \mid x \in S\} \} \cup \{q\};
\]

\[
\delta(w, c) = \begin{cases} 
wc & \text{if } w \in \{0, 1\}^* \text{ and } |w| < \max\{|x| \mid x \in S\} \\
q & \text{otherwise}
\end{cases}
\]

\[
q_0 = \varepsilon;
\]

\[
F = S.
\]

Problem 3

For reasons unknown, a man is travelling with a wolf, a goat and a very large
cabbage. He is faced with unforeseen difficulties when he finds his path blocked
by a roaring river, which may only be crossed by using a small boat tied to the
shore. Indeed, the boat is so small that he can only possibly take one of his
possessions (the cabbage is not to be taken lightly) on it at a time. Furthermore,
his animal companions are proving less than cooperative and it is evident that
neither the wolf and the goat nor the goat and the cabbage can be left ashore
without bipedal supervision, lest an undesirable consumption event
occur.

Give a DFA that characterises the problem, with an appropriately chosen al-
phabet of possible actions for the traveller to take. By looking at your DFA or
otherwise, find a sequence of actions that solves the conundrum.

Solution

We can associate each state with the set of entities (out of \(E = \{\text{man}, \text{wolf}, \text{goat}, \text{cabbage}\}\)) that have successfully made it across the river, and create
an additional state to represent the consumption having occurred (which will
happen whenever \(\{w, g\}\) or \(\{g, c\}\) are in the state or its complement under \(E\)
and \(m\) is not). The alphabet of actions will be chosen to to be \(E\) as well, with
each letter representing the action of the man taking that entity across the river
with him; choosing himself or an entity which is already on the other side shall
be treated as equivalent to going alone.
\[ Q = \{ \emptyset, \{w\}, \{g\}, \{m, g\}, \{w, c\}, \{m, w, g\}, \{m, w, c\}, \{m, g, c\}, \{m, w, g, c\}, C \} \]

\[ \delta(S, x) = \begin{cases} 
C & \text{if } S = C \\
S \cup \{m, x\} & \text{if } m \notin S \text{ and } \{w, g\}, \{g, c\} \notin E \setminus (S \cup \{m, x\}) \\
S \setminus \{m, x\} & \text{if } m \in S \text{ and } \{w, g\}, \{g, c\} \notin S \setminus \{m, x\} \\
C & \text{otherwise} 
\end{cases} \]

\[ q_0 = \emptyset \]

\[ F = \{E\}. \]

By drawing this automaton in a slightly more visually appreciable format and staring at it hard enough, or just by systematic guessing, we may find that the string \textit{gmwgcmg} leads us to the accepting state.

**Problem 4**

Construct a DFA that accepts exactly the following language:

\[ L_{\text{odd}} = \{ x \in \{0, 1\}^* \mid x \text{ contains an odd number of 0's and an odd number of 1's} \}. \]

**Solution**

We can build a DFA to keep track of the parity of the number of each digit observed so far, requiring \(2 \times 2\) states. Each newly observed digit will flip the parity in its corresponding coordinate in the 2-tuple, and we can accept if we finish reading the string in the state representing both parities being odd.

The DFA can be defined as follows:

\[ Q = \{(0, 0), (0, 1), (1, 0), (1, 1)\}; \]

\[ \delta((a, b), c) = \begin{cases} 
(\neg a, b) & \text{if } c = 0 \\
(a, \neg b) & \text{if } c = 1 
\end{cases} \]

\[ q_0 = (0, 0); \]

\[ F = \{(1, 1)\}. \]
Problem 5

(a) Construct a DFA for the following language:

\[ L_7 = \{ x \in \{0, 1\}^* \mid x \text{ is the binary encoding of a multiple of } 7 \} \].

(b) Sketch a proof that, in fact, for any \( m, k \in \mathbb{N}, k > 1 \), the language

\[ L_{m,k} = \{ x \in \{0, 1, \ldots, k-1\}^* \mid x \text{ is the base-}k \text{ encoding of a multiple of } m \} \]

has a DFA.

Solution

(a) We can build an automaton to keep track of the remainder (modulo 7) of the number represented by the initial segment of the binary encoding observed so far; if the remainder is 0 when we are done reading the string, we accept, otherwise we reject. To see that this can indeed be done, we observe that due to the definition of the positional representation, if \( w \) is the binary representation of some number \( n \), then \( w0 \) represents \( 2n + 0 \) and \( w1 \) represents \( 2n + 1 \); accordingly, we can keep track of the remainder modulo 7 by encoding the previous remainder in the current state and transitioning to the next state according to the preceding expression based on the digit we read in.

We may define the DFA codifying the preceding reasoning as follows:

\[
Q = \{0, 1, \ldots, 6\}; \\
\delta(n, c) = (2n + c) \mod 7; \\
q_0 = 0; \\
F = \{0\}.
\]

(b) The existence of a DFA which accepts the language in question constitutes proof of it being regular. We may therefore simply generalise the construction (and the reasoning for its correctness) from part (a) to arbitrary bases and moduli (in each case, if \( 0 \leq c < k \) is a single trailing digit and \( w \) is the base-\( k \) encoding of \( n \), \( wc \) represents \( kn + c \); this can be proven e.g. by writing out the definition of positional notation). The definition will be the same as in (a), except that \( Q \) must be adapted to reflect the correct number of remainder classes, i.e. \( Q = \{0, 1, \ldots, k - 1\} \).