1 Determinization (30)

Background
We have seen that nondeterministic machines are often much easier to construct for given languages than their deterministic counterparts. Here are two such examples.

\[ A = \{w, w'\} \]
\[ B = \{x \in \{a, b\}^* \mid x_{-k} = a\} \]

Here \(w\) and \(w'\) are arbitrary distinct words over \(\{a, b\}\).

Task

A. Explain what the natural NFA (no \(\varepsilon\)-moves) for \(A\) looks like. What is the state complexity?

B. Determine the DFA \(M\) obtained from this machine by applying the power automaton construction (accessible part only). Give a bound on the state complexity.

C. Should one expect the machine \(M\) to be minimal? Explain.

D. Repeat for the language \(B\).

2 Blow-Up (40)

Background
Also write \(A_n\) for the (boring) automaton on \(n\) states whose diagram is the circulant with \(n\) nodes and strides 1 and 2. The edges with stride 1 are labeled \(a\) and the edges with stride 2 are labeled \(b\). For example, the following picture shows \(A_6\). We assume \(I = F = Q\).
Let $B_n$ be the NFA obtained from $A_n$ by switching one of the $b$ labels to an $a$ label; write $K_n$ for the acceptance language of $B_n$.

**Task**

A. Show that determinization of $B_n$ produces an accessible automaton $B'_n$ of $2^n$ states.

B. Argue that $B'_n$ is already reduced and conclude that $K_n$ has state complexity $2^n$.

**Comment**

The language $K_n$ has no particular significance (as far as I know). Thinking about pebble automata might help with the argument.

By the way, if you switch an $a$ to a $b$ there is still full blow-up for odd $n$, but for even $n$ the power automaton has only size $2^n - 1$. Extra credit.

---

### 3 Forward State Merging (30)

**Background**

Below is the transition matrix for a (somewhat random) 13-state DFA $M$ over the alphabet $\{a, b\}$. Initial state is 1 and the final states are $\{4, 6, 8, 9, 10, 11\}$.

\[
\begin{array}{c|cccccccccccc}
  & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 \\
  a & 2 & 4 & 6 & 8 & 12 & 9 & 12 & 13 & 12 & 13 & 12 & 13 & 12 \\
  b & 3 & 5 & 7 & 12 & 10 & 13 & 11 & 12 & 13 & 12 & 13 & 12 & 12 \\
\end{array}
\]

This machine turns out to be non-minimal.

**Task**

A. Use Moore’s forward state merging algorithm to compute the minimal DFA $M_0$ for this machine.
B. Describe the language $L$ accepted by the machine.

C. Construct the minimal DFA $M_1$ for $L$ directly by hand, using whatever method you prefer.

D. Show that $M_0$ and $M_1$ are isomorphic.

**Comment** Make sure to build a nice table for the state merging process, don’t just write down the final result. If you like, you can write a program to do this (or use the code on the web).

Part C. will be obvious once you have done part B.