MATH 2301

* Admin : Monday 02 Oct public holiday. make-up sessions Details on Wattle; Lecture on Tuesday 3/10, 2-3 pm Robertson

Final exam Wed 8 Nov 2023 → 19:00 – 21:15
in-person (Venue to be announced)

* Warnings (regex syntax)

① The shorthand Σ to mean $a_1 | a_2 | \cdots | a_n$ where $a_1, \cdots, a_n$ are the letters of Σ, may be used within a regex

② The shorthand a^k to mean $a \cdots a$ (k times) may not be used within a regex.

* Last time : Deterministic Finite Automata (DFAs)

Given a DFA M, the language of M, denoted L(M), is the set of all strings accepted by M.

* Today : DFAs vs regular expressions

Q1: Given a DFA M, can we find a regex r such that $L(M) = L(r)$?

Q2: Given a regex r, can we find a DFA M such that $L(M) = L(r)$?

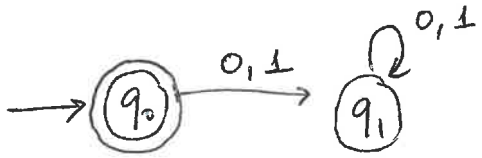
Let us try to answer Q2

Warm-up

① Consider $r = \varepsilon$, $L(r) = \{\varepsilon\}$

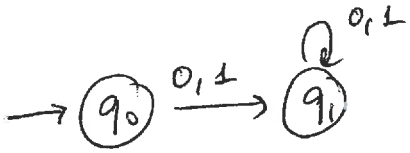
Try to build DFA M such that $L(M) = \{\varepsilon\}$

Attempt:



← M and $L(M) = \{\varepsilon\}$

② Consider $r = \phi$, $L(r) = \phi$

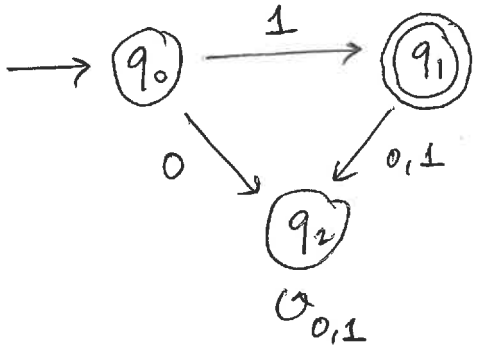


← M such that $L(M) = \phi$

[There are other options for M!], namely, any M where $A = \phi$]

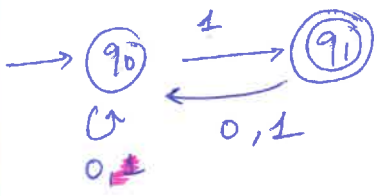
③ Consider $r = a$ for some $a \in \Sigma$

Concretely, say $r = 1$. $L(r) = \{1\}$



← M such that $L(M) = \{1\}$.

The following doesn't work:



(it accepts extra stuff, e.g 101)

④ Consider $r = r_1 | r_2$, $L(r) = L(r_1) \cup L(r_2)$

Suppose we know DFAs M_1 and M_2, such that $L(M_1) = L(r_1)$, and $L(M_2) = L(r_2)$

Then we want to use these to construct a DFA M such that $L(M) = L(M_1) \cup L(M_2)$

Schematic:

M_1

M_2

Problem: We have to run strings through both M_1 & M_2, to decide if one of them accepts. But that is not technically allowed. How do we fix this ??? (Come back to this ...)

⑤ $r = r_1 r_2$, $L(r) = L(r_1) \circ L(r_2)$

Given M_1, M_2 such that $L(M_1) = L(r_1)$ and $L(M_2) = L(r_2)$,

build M such that $L(M) = L(M_1) \circ L(M_2)$

If $w = w_1 w_2$ such that w_1 matches r_1, w_2 matches r_2,

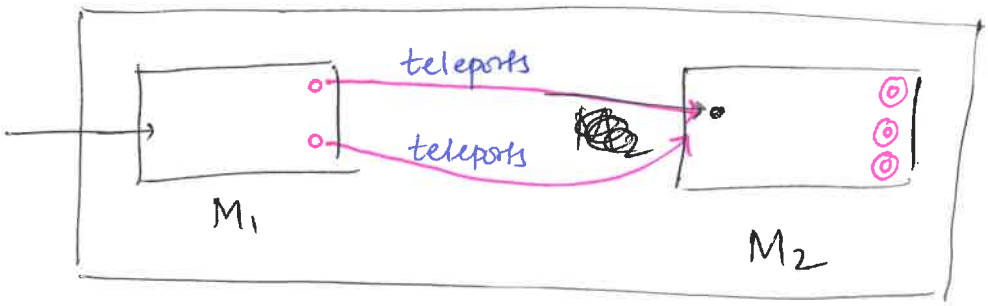
then $w \in L(r)$.

Given

M_1

M_2

Attempt: (Failed attempt)

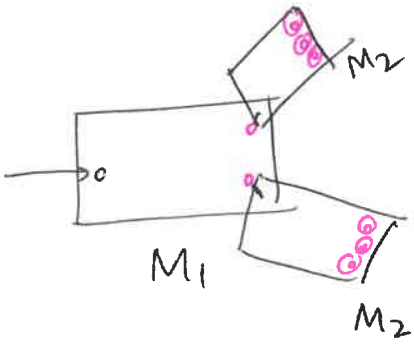


M

Such a machine M would have the language $L(M_1) \circ L(M_2)$

Such a machine is not a DFA so this is not a valid solution to the problem...

Attempt 2:



From each accept state of M_1, spin off a copy of M_2.

(Not a valid attempt!!)