1 Filter for an Input Stream [exam problem]

We would like to construct an automaton that recognizes substrings from an input stream. The input stream consists of symbols \{a, b\} and the substrings that the automaton should detect are of the form bab\(^*\). In other words, the input of the automaton is a series of a’s and b’s. The automaton should go into an accepting state whenever the most recently received symbols form a string of the form bab\(^*\). For example, in the input stream \(b a b b a a b a b a a a b a\), the automaton should be in an accepting state exactly after the reception of an underlined symbol. Construct a deterministic finite automaton that precisely fulfills the above specification.

2 Nondeterministic Finite Automata

a) Consider the alphabet \{a, b\}. Construct an NFA that accepts all strings containing the substring \(abb\) at least twice. (This means that words containing \(abbaba\) as a substring should also be accepted!)

b) Construct an NFA which accepts the following regular expression: \((00 \cup (0(0 \cup 1)^*)^*)^*\).

c) Construct an NFA accepting \(1^*0^*1^+\) with as few states as possible. (cf. Exercise 1.1.a)

d) Consider a machine \(M := (Q, \Sigma, \delta, q_0, Q)\). Is it possible to make a statement about the strings being accepted by \(M\)? Does it make a difference whether \(M\) is deterministic or not?

3 De-randomization

a) Give a regular expression for the following NFA and construct an equivalent NFA without \(\varepsilon\)-transitions.

```
0          1          2
\varepsilon \varepsilon \varepsilon
```

b) Finally, transform the machine into a deterministic automaton.
4 States Minimization

Simplify the following automaton. Explain why your changes are allowed. Finally, give the corresponding regular expression.

5 “Regular” Operations in UNIX

In this exercise you are asked to provide a UNIX command to output all lines in a file ending with “password” or “passwort”, followed by an unknown number (potentially zero) of vowels.