Middle East Technical University Department of Computer Engineering

CENG 280

Formal Languages and Abstract Machines

Spring 2023-2024

Homework 1

Question 1

a) The set of all strings on the alphabet $\Sigma = \{a, b, c\}$ which are starting with ab, ending with bc:

$L_1 = L(abc \cup ab(a \cup b \cup c)^*bc)$

The set of all strings on the same alphabet ($\Sigma = \{a, b, c\}$) which include the substring "$aabbcc$":

$L_2 = L(((a \cup b \cup c)^*aabbcc(a \cup b \cup c)^*)^+)$

L_1 and L_2 are regular since they can be generated by the given regular expressions. Then, $L_1 \cap L_2$, the set given in the question, is also regular since the class of regular languages is closed under intersection (by theorems 2.3.1 and 2.3.2 of the textbook).

Further, a regular language is a countable set. Let us follow such an enumeration method for any regular language: Starting with strings of length 0 and increasing the string length by 1 at each iteration, enumerate strings of the same length, say of length n, according to their lexicographical order then continue enumeration with strings of length $n+1$. For any length $n \in \mathbb{N}$, there are at most $|\Sigma|^n$ enumerations, which concludes that enumeration for any length n will eventually end, and the overall enumeration will continue with strings of length $n+1$ (This is due to the fact that alphabets are finite sets of symbols (i.e. Σ is finite)). Therefore, this is a valid enumeration method.

Consequently, the given language is countable since it is a regular language, and a regular language is a countable set.

Moreover, the given language is infinite. For any string σ in that language, say of length $n \in \mathbb{N}$, it is possible to produce another string of the language by appending σ with "bc". That is, $\sigma_1 = \sigma \circ$ "bc" (of length $n+2$) is in the language. This operation of appending with "bc" may continue recursively and infinitely, at each time creating a new string of the language (i.e. $\sigma_2 = \sigma \circ$ "bc" $\circ$ "bc" $\sigma_3 = \sigma \circ$ "bc" $\circ$ "bc" $\circ$ "bc" $\cdots$

Hence, the given language is an infinite language. To sum up, the language given in the question is countably infinite.

b) For any regular language there is a DFA that accepts the language. Therefore, the set of all regular languages on the alphabet $\Sigma = \{0, 1\}$ is countably infinite since there are countably infinite number of DFAs on alphabet $\Sigma = \{0, 1\}$.

Let us define an arbitrary DFA $M = (K, \Sigma, \Delta, s_0, F)$ with a single restriction: $\Sigma = \{0, 1\}$. Firstly, we need states (i.e. the set K). For an arbitrary DFA M , $|K|$ can take any value $n \in \mathbb{N}$ (i.e. K is finite). Then, the set of all possible $|K|$'s is a bijection to $\mathbb{N}$, thus, countably infinite.

Moreover, for any K, there can be $|K|$ different s_0 selections and can be $2^{|K|}$ different F selections, both of which are finite since any distinct $|K|$ is a natural number. Thus, for any distinct K, there are finite number of possible (K, s_0, F) triples. On the other hand, since the number of possible $(K, s0, F)$'s is a finite factor of $|K|$, the set of all possible (K, s_0, F) triples is also countably infinite (still forms a bijection to $\mathbb{N}$).

Lastly, we need a transition function, Δ. For an arbitrary DFA M with the set of states K, number of possible Δ's $|K|^{|\Sigma|\times|K|}$ which is finite since finite exponent of a finite number is finite. In other words, $|\Delta|$ is a function from (K, Σ) to K and for any distinct K, there is finite number of number of different Δ's since K and Σ are finite. Then, since for any distinct K, number of possible s_0's, F's and Δ's are independent of each other and each are finite factors of K, adding Δ to the last triple, number of all possible (K, Δ, s_0, F) quadruples continues to form a bijection to $\mathbb{N}$. Thus its cardinality remains same: countably infinite. Further, since Σ is constant, cardinality of the set of all possible $(K, \Sigma, \Delta, s_0, F)$'s also the same. To sum up, on the given alphabet Σ, countably infinite number of DFAs can be defined.

(TL;DR: Intuitively, there are countably infinite different possible K's and cardinalities of all other elements of the definition of the arbitrary DFA is finite factors of the K. Therefore, number of possible different DFAs is dominated by number of possible K's, thus, is countably infinite).

c) First, consider the set of all strings on the binary alphabet $\Sigma = \{0, 1\}$ which is a regular language ($\{0, 1\}^*$). As shown in the answer above (Answer 1.1), a regular language is a countable set. Similarly, the set generated by $\{0, 1\}^*$ is an infinite set since it is possible to produce another string of the language by appending any element of the alphabet to any string in the language and this operation may continue recursively and infinitely, at each time creating a new string of the language. That is, the set of all strings on the binary alphabet $\Sigma = \{0, 1\}$ is countably infinite. Hence, the set of all languages on the binary alphabet $\Sigma = \{0, 1\}$ $2^{\{0,1\}^*}$ is an uncountably infinite set since powerset of a countably infinite set is uncountably infinite.

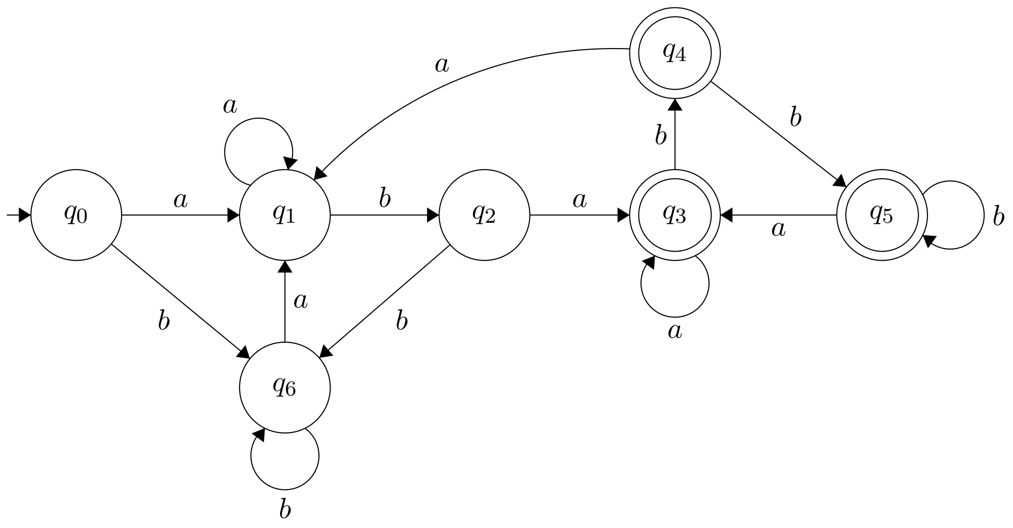
(Remark: The fact that the powerset of a countably infinite set is uncountably infinite can be proven using Cantor's Diagonal Argument.)

Question 2

$\bullet L_{01} = \{\omega \in \{a, b\}^* |\ \omega \text{ includes an odd number of } aba \text{ substrings}\}$

a) $(a \cup b^+a)(a \cup bb^+a \cup ba^+b(b^+a^+b)^*a)^*(ba^+ \cup ba^+b(b^+a^+b)^*(\epsilon \cup b^+ \cup b^+a))$

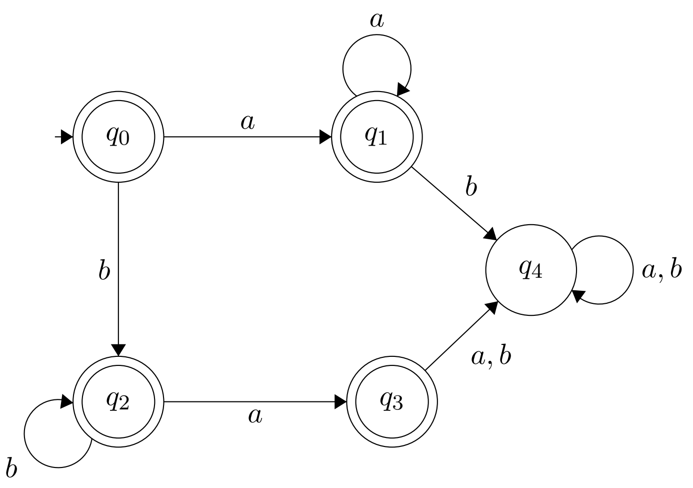
b)



$\bullet L_{02} = \{\omega \in \{a,b\}^* |$ neither baa nor ab is a substring of $\omega\}$

a) $a^* \cup b^* \cup b^+ a$

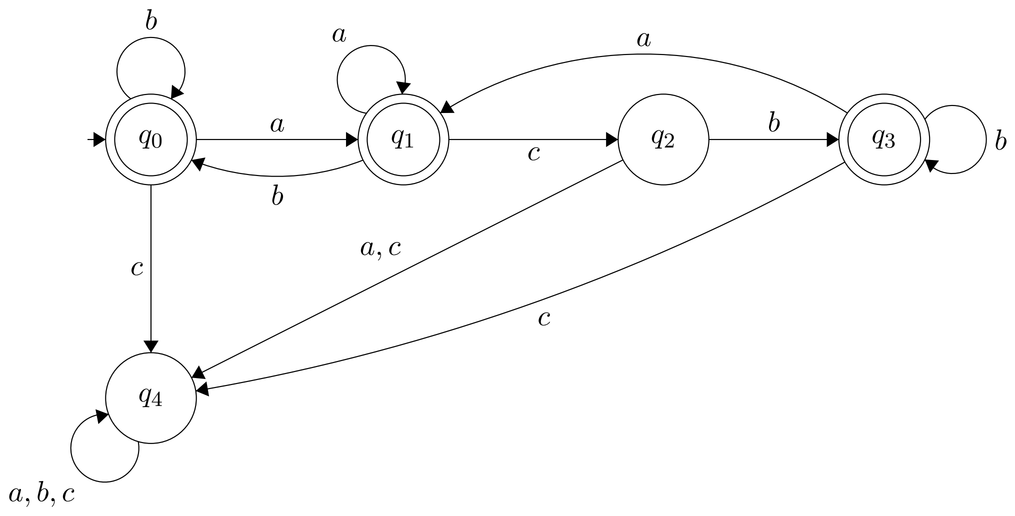
b)



$\bullet L_{03} = \{\omega \in \{a,b,c\}^* |$ every c is directly preceded by a and followed by $b\}$

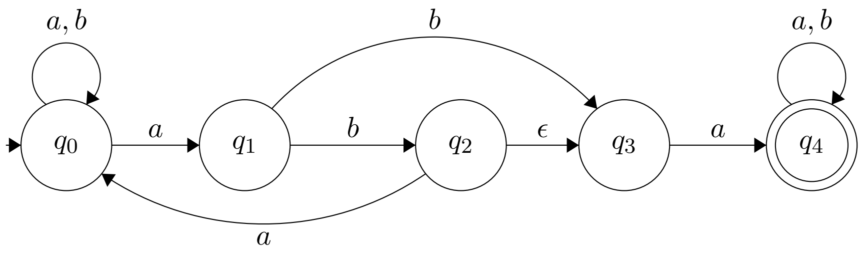
a) $(a \cup b \cup acb)^*$

b)



Question 3

N_1:



a) Construction of the equivalent DFA:

Closure of States:
$E_{q_0} = \{q_0\}$, $E_{q_1} = \{q_1\}$, $E_{q_2} = \{q_2, q_3\}$,$E_{q_3} = \{q_3\}$, $E_{q_4} = \{q_4\}$

Produced **reachable** transitions at first iteration:

$(E_{q_0}, a, E_{q_0} \cup E_{q_1}), (E_{q_0}, b, E_{q_0})$

Newly introduced state: $E_{q_0} \cup E_{q_1}$

Produced transitions at second iteration (for the newly introduced state):

$(E_{q_0} \cup E_{q_1}, a, E_{q_0} \cup E_{q_1}), (E_{q_0} \cup E_{q_1}, b, E_{q_0} \cup E_{q_2} \cup E_{q_3})$

Newly introduced state: $E_{q_0} \cup E_{q_2} \cup E_{q_3}$ $(= E_{q_0} \cup E_{q_2})$

Produced transitions at third iteration (for the newly introduced state):

$(E_{q_0} \cup E_{q_2}, a, E_{q_0} \cup E_{q_1} \cup E_{q_4}), (E_{q_0} \cup E_{q_2}, b, E_{q_0})$

Newly introduced state: $E_{q_0} \cup E_{q_1} \cup E_{q_4}$

Produced transitions at fourth iteration (for the newly introduced state):

$(E_{q_0} \cup E_{q_1} \cup E_{q_4}, a, E_{q_0} \cup E_{q_1} \cup E_{q_4}), (E_{q_0} \cup E_{q_1} \cup E_{q_4}, b, E_{q_0} \cup E_{q_2} \cup E_{q_3} \cup E_{q_4})$

Newly introduced state: $E_{q_0} \cup E_{q_2} \cup E_{q_3} \cup E_{q_4}$ $(= E_{q_0} \cup E_{q_2} \cup E_{q_4})$

Produced transitions at fifth iteration (for the newly introduced state):

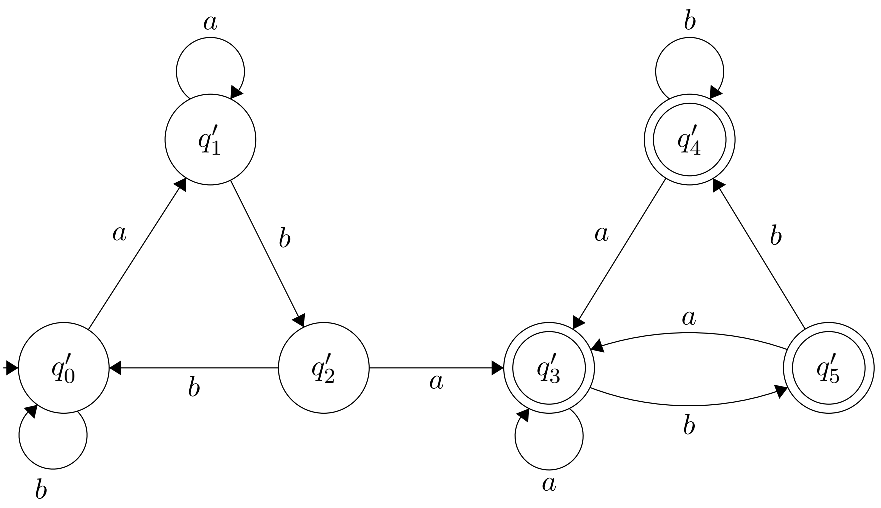
$(E_{q_0} \cup E_{q_2} \cup E_{q_2}, a, E_{q_0} \cup E_{q_1} \cup E_{q_4}), (E_{q_0} \cup E_{q_2} \cup E_{q_4}, b, E_{q_0} \cup E_{q_4})$

Newly introduced state: $E_{q_0} \cup E_{q_4}$

Produced transitions at sixth iteration (for the newly introduced state):

$(E_{q_0} \cup E_{q_4}, a, E_{q_0} \cup E_{q_1} \cup E_{q_4}), (E_{q_0} \cup E_{q_4}, b, E_{q_0} \cup E_{q_4})$

No new states are introduced at sixth iteration. Computation ends here. Resulting equivalent DFA is:



where $q_0' = E_{q_0}$, $q_1' = E_{q_0} \cup E_{q_1}$, $q_2' = E_{q_0} \cup E_{q_2}$, $q_3' = E_{q_0} \cup E_{q_1} \cup E_{q_4}$, $q_4' = E_{q_0} \cup E_{q_4}$, $q_5' = E_{q_0} \cup E_{q_2} \cup E_{q_4}$

b) A possible trace of "aba" on the given NFA as such: $(q_0, aba) \vdash (q1, ba) \vdash (q2, a) \vdash (q3, a) \vdash (q4, \epsilon)$

Since the computation ends with the final state q4, the given NFA accepts "aba".

"aba" traces on the constructed equivalent DFA as such: $(q_0', aba) \vdash (q_1', ba) \vdash (q_2', a) \vdash (q_3', a) \vdash (q_4', \epsilon)$. Since the computation ends with the final state q_4', the equivalent DFA accepts "aba", as expected.

A possible trace of "babb" on the given NFA as such: $(q_0, babb) \vdash (q_0, abb) \vdash (q_1, bb) \vdash (q_2, b) \vdash (q_3, b)$ which gets stuck in q3 without consuming the whole input.

Similarly an other possible trace of "babb" on the given NFA gets stuck at the same point:

$(q_0, babb) \vdash (q_0, abb) \vdash (q_1, bb) \vdash (q_3, b)$ which gets stuck in q_3 without consuming the whole input.

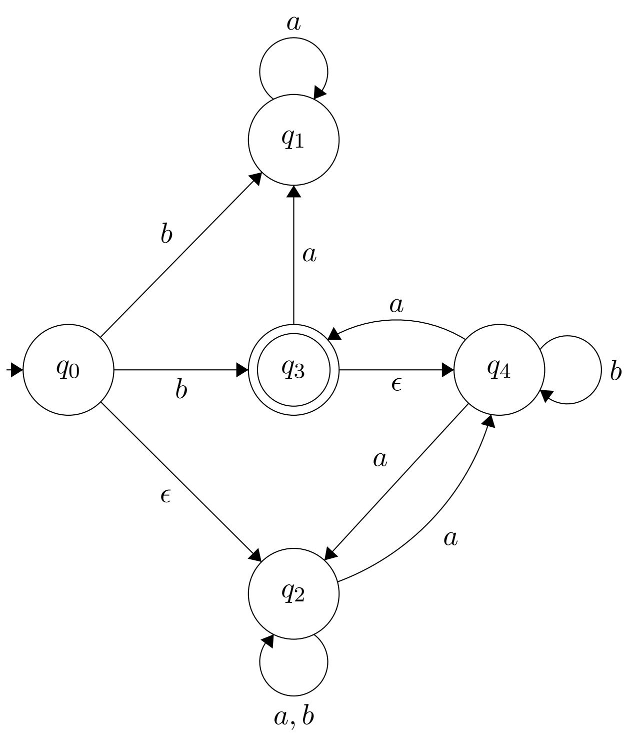
$(q_0, babb) \vdash (q_0, abb) \vdash (q_0, bb) \vdash (q_0, b) \vdash (q_0, \epsilon)$ is the last possible trace, which and at a nonfinal state.

Since the computation gets either stuck without consuming the whole input or ends in a nonfinal state, the given NFA does not accept "babb".

"babb" traces on the constructed equivalent DFA as such: $(q_0', babb) \vdash (q_0', abb) \vdash (q_1', bb) \vdash (q_2', b) \vdash (q_0', \epsilon)$

Since the computation does not end with a final state, the equivalent DFA does not accept "babb", as expected.

$N2:$



a) Construction of the equivalent DFA:

Closure of States:

$E_{q_0} = \{q_0, q_2\}, E_{q_1} = \{q_1\}, E_{q_2} = \{q_2\}, E_{q_3} = \{q_3, q_4\}, E_{q_4} = \{q_4\}$

Produced **reachable** transitions at first iteration:

$(E_{q_0}, a, E_{q_2} \cup E_{q_4}), (E_{q_0}, b, E_{q_1} \cup E_{q_2} \cup E_{q_3})$

Newly introduced states: $E_{q_2} \cup E_{q_4}$ and $E_{q_1} \cup E_{q_2} \cup E_{q_3}$

Produced transitions at second iteration (for newly introduced states):

$(E_{q_2} \cup E_{q_4}, a, E_{q_2} \cup E_{q_3} \cup E_{q_4}), (E_{q_2} \cup E_{q_4}, b, E_{q_2} \cup E_{q_4})$,

$(E_{q_1} \cup E_{q_2} \cup E_{q_3}, a, E_{q_1} \cup E_{q_2} \cup E_{q_3} \cup E_{q_4}), (E_{q_1} \cup E_{q_2} \cup E_{q_3}, b, E_{q_2} \cup E_{q_4})$

Closures of states:

Newly introduced state: $E_{q_2} \cup E_{q_3} \cup E_{q_4} (= E_{q_2} \cup E_{q_3})$

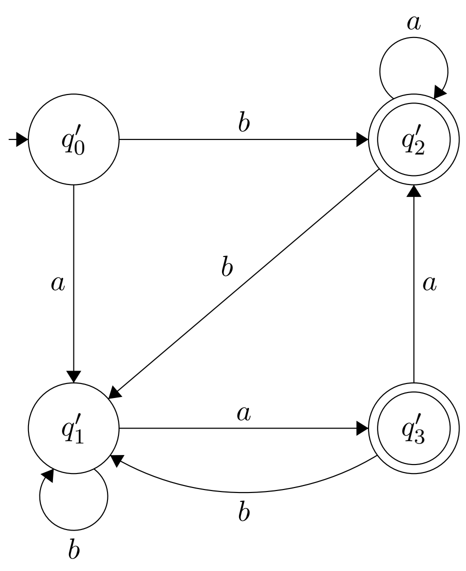
$(E_{q_1} \cup E_{q_2} \cup E_{q_3} \cup E_{q_4} = E_{q_1} \cup E_{q_2} \cup E_{q_3}$, thus not a new state.)

Produced transitions at third iteration (for newly introduced states):

$(E_{q_2} \cup E_{q_3}, a, E_{q_1} \cup E_{q_2} \cup E_{q_3} \cup E_{q_4}), (E_{q_2} \cup E_{q_3}, b, E_{q_2} \cup E_{q_4}),$

No new states are introduced at third iteration. Computation ends here.

Resulting equivalent DFA is:



where $q_0' = E_{q_0}$, $q_1' = E_{q_2} \cup E_{q_4}$, $q_2' = E_{q_1} \cup E_{q_2} \cup E_{q_3} = E_{q_1} \cup E_{q_2} \cup E_{q_3} \cup E_{q_4}$, $q_3' = E_{q_2} \cup E_{q_3} = E_{q_2} \cup E_{q_3} \cup E_{q_4}$

b) A possible trace of "aba" on the given NFA as such:

$(q_0, aba) \vdash (q_2, aba) \vdash (q_4, ba) \vdash (q_4, a) \vdash (q3, \epsilon)$

Since the computation ends with the final state q_3, the given NFA accepts "aba".

"aba" traces on the constructed equivalent DFA as such: $(q_0', aba) \vdash (q_1', ba) \vdash (q_1', a) \vdash (q_3', \epsilon)$

Since the computation ends with a final state, q_3', the equivalent DFA accepts "aba", as expected.

Possible traces of "babb" on the given NFA are: $(q_0, babb) \vdash (q_1, abb) \vdash (q_1, bb)$
$(q_0, babb) \vdash (q_3, abb) \vdash (q_1, bb)$
$(q_0, babb) \vdash (q_3, abb) \vdash (q_4, abb) \vdash (q_2, bb) \vdash (q_2, b) \vdash (q_2, \epsilon)$
$(q_0, babb) \vdash (q_2, babb) \vdash (q_2, abb) \vdash (q_2, bb) \vdash (q_2, b) \vdash (q_2, \epsilon)$
$(q_0, babb) \vdash (q_2, babb) \vdash (q_2, abb) \vdash (q_4, bb) \vdash (q_4, b) \vdash (q_4, \epsilon)$

each of which either which gets stuck before consuming the whole input or ends in a nonfinal state. Therefore, the given NFA does not accept "babb".

"babb" traces on the constructed equivalent DFA as such:

$(q_0', babb) \vdash (q_2', abb) \vdash (q_2', bb) \vdash (q_1', b) \vdash (q_1', \epsilon)$

Since the computation does not end with a final state, the equivalent DFA does not accept "babb".

Question 4

a)

$K = \{q_0, q_1, q_2, q_3\}$

$\Sigma = \{a, b\}$

$\delta = \{(q_0, a, q_1), (q_0, b, q_0), (q_1, a, q_1), (q_1, b, q_2), (q_2, a, q_3), (q_2, b, q_0), (q_3, a, q_3), (q_3, b, q_3)\}$

$s = \{q_0\}$

$F = \{q_0, q_1, q_2\}$

b)

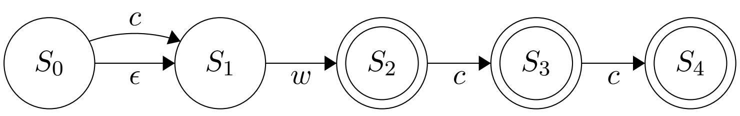
$(q_0, abbaabab) \vdash$
$(q_1, bbaabab) \vdash$
$(q_2, baabab) \vdash$
$(q_0, aabab) \vdash$
$(q_1, abab) \vdash$
$(q_1, bab) \vdash$
$(q_2, ab) \vdash$
$(q_3, b) \vdash$
(q_3, ϵ)

Since the computation does not end with a final state, the DFA does not accept the input *abbaabab*.

Question 5

a) One possible NFA that recognizes the language of strings which are valid syllables :

(For brevity, all possible vowels are denoted as w.)



Note that the NFA above is not complete.

b) $(((c \cup \epsilon)(a \cup ı \cup o \cup u)(c \cup \epsilon)(c \cup \epsilon))((c \cup \epsilon)(a \cup ı \cup u)(c \cup \epsilon)(c \cup \epsilon))^*) \cup$
$(((c \cup \epsilon)(e \cup i \cup ö \cup ü)(c \cup \epsilon)(c \cup \epsilon))((c \cup \epsilon)(e \cup i \cup ü)(c \cup \epsilon)(c \cup \epsilon))^*)$