Jméno: **UČO:**

0007 *list* 1 *učo* *body*

Oblast strojově snímaných informací. Své učo a číslo listu vyplňte zleva dle vzoru číslic. Jinak do této oblasti nezasahujte. 0 1 2 3 4 5 6 7 8 9

2. [2 body] Nechť L je regulární jazyk nad abecedou $\Sigma = \{a, b, c\}$.
Mějme operaci *changeWhileNotA*() takovou, že jazyk $R = \mathit{changeWhileNotA}(L)$ obsahuje pro každé slovo $w \in L$ modifikované slovo w' vzniklé tak, že v začátku slova w, před prvním výskytem písmene a, se všechna písmena b nahradí za písmena c a všechna písmena c se nahradí za písmena b. Pokud se ve slově w žádné písmeno a nenachází, pak dojde k těmto náhradám v celém slově.

Například:

$$\mathit{changeWhileNotA}(\{\varepsilon, a, bcba, bacab\}) = \{\varepsilon, a, cbca, cacab\}$$
$$\mathit{changeWhileNotA}(\{b, cbc, aba, bcabac\}) = \{c, bcb, aba, cbabac\}$$
$$\mathit{changeWhileNotA}(\{bca\}^+) = \{cba\} \cdot \{bca\}^*$$
$$\mathit{changeWhileNotA}(\{bc\}^+) = \{cb\}^+$$
$$\mathit{changeWhileNotA}(\emptyset) = \emptyset$$

Vaším úkolem je rozhodnout, zda pro každý regulární jazyk L je jazyk *changeWhileNotA*(L) taktéž regulární. Tedy zda je třída regulárních jazyků uzavřená na operaci *changeWhileNotA*(). Vaši odpověď dokažte, a to tak, že:

- Pokud rozhodnete, že není, najděte regulární jazyk L takový, že jazyk *changeWhileNotA*(L) regulární není.
- Pokud rozhodnete, že je, dokažte tvrzení například s pomocí známých uzávěrových vlastností třídy regulárních jazyků prezentovaných na přednášce, nebo konstruktivně popsáním algoritmu na transformaci nějakého formalizmu pro popis regulárních jazyků.

Třída regulárních jazyků je uzavřená na operaci *changeWhileNotA*(). To můžeme dokázat popisem algoritmu na konstrukci konečného automatu $\mathcal{A}$ rozpoznávajícího jazyk *changeWhileNotA*(L) pro libovolný regulární jazyk L.

Základní myšlenka konstrukce

Budeme konstruovat konečný automat pro výsledný jazyk. Vyjdeme z DFA s totální přechodovou funkcí pro jazyk L (ten existuje, protože L je regulární) a jeho kopie. V původním automatu všechny přechody pod b nahradíme přechody pod c a všechny přechody pod c nahradíme přechody pod b, tím zaručíme záměny písmen b a c (před prvním načtením písmene a). Přechody pod a z původního automatu přesměrujeme do odpovídajících stavů kopie automatu, abychom zaručili načtení zbytku slova v nezměněné kopii. Iniciálním stavem výsledku bude iniciální stav původního automatu, akceptující stavy budou všechny akceptující stavy originálu i kopie.

Myšlenka konstrukce konečného automatu – podrobněji

Mějme deterministický konečný automat $\mathcal{A}_L$ s totální přechodovou funkcí pro jazyk L a jeho kopii $\mathcal{A}'_L$ (stavy automatu $\mathcal{A}'_L$ mají stejné označení, jako odpovídající stavy v $\mathcal{A}_L$, ale navíc jsou označeny $'$, tedy například q je stav automatu $\mathcal{A}_L$, q' je odpovídající stav $\mathcal{A}'_L$).

Nyní spojíme uvedené automaty do jednoho automatu $\mathcal{A}$. Každý přechod v automatu $\mathcal{A}_L$ ze stavu q do stavu r pod písmenem b nahradíme přechodem pod písmenem c a každý přechod pod písmenem

c nahradíme přechodem pod písmenem b. Každý přechod ze stavu q do stavu r v automatu $\mathcal{A}_L$ pod písmenem a smažeme a místo něj vytvoříme přechod ze stavu q pod písmenem a do stavu r' automatu $\mathcal{A}'_L$. Stav r' odpovídá kopii stavu r.

Tímto jsme spojili automaty a dále je potřeba zvolit počáteční stav a akceptující stavy $\mathcal{A}$. Jako počáteční stav $\mathcal{A}$ vybereme stav, který byl počátečním stavem automatu $\mathcal{A}_L$. Koncové stavy budou sjednocením všech koncových stavů automatů $\mathcal{A}'_L$ a $\mathcal{A}_L$, tím zabezpečíme, že výsledný automat bude akceptovat jak slova, která obsahovala znak a a tedy k dočtení slov došlo v kopii automatu, tak i slova, která znak a neobsahovala a tudíž došlo k záměně znaků v celém slově.

Formální zápis konstrukce konečného automatu

Nechť totální DFA pro jazyk L je dán pěticí $\mathcal{A}_L = (Q_L, \Sigma, \delta_L, q_{L0}, F_L)$. Pro jazyk $changeWhileNotA(L)$ vytvoříme DFA $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$, kde

- $Q = \{1, 2\} \times Q_L$,
- $q_0 = (1, q_{L0})$,
- $F = \{1, 2\} \times F_L$,
- $\delta((1, q), b) = (1, \delta_L(q, c))$, pro všechna $q \in Q_L$,
- $\delta((1, q), c) = (1, \delta_L(q, b))$, pro všechna $q \in Q_L$,
- $\delta((1, q), a) = (2, \delta_L(q, a))$, pro všechna $q \in Q_L$,
- $\delta((2, q), x) = (2, \delta_L(q, x))$, pro všechna $q \in Q_L$ a $x \in \Sigma$.

Poznámka: Stavy automatu $\mathcal{A}$ jsou označeny uspořádanými dvojicemi, kde první složka označuje ve které části automatu se nacházíme. Dvojice tvaru $(1, q)$ patří části automatu, která byla vytvořena z původního automatu a obsahuje zaměněné přechody pod b a c a pak přechody pod a do druhé části. Dvojice $(2, q)$ patří části, která byla vytvořena z kopie původního automatu, ta obsahuje stejné přechody, jaké obsahoval původní automat $\mathcal{A}_L$.

Dokázali jsme, že pro libovolný regulární jazyk L dokážeme zkonstruovat DFA akceptující jazyk $changeWhileNotA(L)$, tedy třída regulárních jazyků je uzavřená na operaci $changeWhileNotA$.