Tomáš Höger, 2i2

Úloha 1
Zostrojte zásobníkový automat A pre jazyk:

$$L = \{ucv \mid u, v \in 1\{0,1\}^* \text{ a zároveň } u, v \text{ nie sú binárne zápisy po sebe idúcich čísel takých, že } u \leq v \text{ a "c" je terminálny symbol}\}$$

Dokážte správnosť konštrukcie (t.j. $L(\mathrm{A}) = L$).
Pozn.: $1\{0,1\}^*$ reprezentuje množinu binárnych reťazcov, začínajúcich bitom 1 (teda nemajú žiadne leading zeros).

Čo máme zostrojiť?
Potrebujeme zostrojiť nedeterministický zásobníkový automat (push down automat, PDA), čo je 7-tica $A = (K, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, kde K je konečná množina stavov, Σ je abeceda vstupných symbolov, Γ ja abeceda zásobníkových symbolov, $q_0 \in K$ je počiatočný stav, $Z_0 \in \Gamma$ je počiatočný symbol v zásobníku, $F \subseteq K$ je množina akceptačných stavov a

$$\delta : K \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{K \times \Gamma^*}_{KON}$$

Zásobníkové automaty môžu akceptovať buď akceptačným stavom alebo prázdnou pamäťou, pričom oba tieto spôsoby su ekvivalentné a teda generujú ten istý jazyk. V tomto prípade budem konštruovať automat akceptujúci akceptačným stavom. Aby sme definovali akceptovaný jazyk, musíme definovať este konfiguráciu a krok výpočtu. Konfigurácia je trojica $(q, av, \gamma Z)$ z $K \times \Sigma^* \times \Gamma^*$, automat je v stave q, zvyšok vstupu je av (číta a) a obsah zásobníka je γZ (Z je vrch zásobníka). Krok výpočtu zásobníkového automatu je relácia $\vdash$ na konfiguráciách, $(q, av, \gamma Z) \vdash (p, v, \gamma\beta)$ ak $(p, \beta) \in \delta(q, a, Z)$, kde $a \in \Sigma \cup \{\varepsilon\}$ a $Z \in \Gamma$. Jazyk akceptovaný koncovým stavom nedeterministického automatu A je

$$L(A) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \gamma), \gamma \in \Gamma^*, q \in F\}$$

Slovná charakterizácia jazyka L
Jazyk L obsahuje všetky slová tvaru ucv, kde $u, v \in 1\{0,1\}^*$, teda sú to binárne reťazce začínajúce bitom 1, pričom musí platiť, že u, v nie sú po sebe idúce čísla také, že $u \leq v$. To znamená, že pre u, v daného tvaru, kde navyše $u + 1 = v$ slovo ucv nepatrí do jazyka L ($ucv \notin L$), ale slovo vcu do jazyka L patrí ($vcu \in L$). Teda jazyk L obsahuje všetky slová, ktoré sú požadovaného tvaru okrem tých, v ktorých je u predchodcom (predecessorom) slova v (u môže byť nasledníkom (successorom) v).

Prečo zlyhajú štandardné metódy riešenia?
Úlohy podobného typu sa pomocou zásobníkových automatov najčastejšie riešia tak, že sa prečíta prvá časť vstupného slova (v tomto prípade by to bolo slovo u čítané až po oddeľovač c), pri tom sa zásobník naplní nejakými symbolmi, ktorých typ a počet má väčšinou veľmi úzku väzbu na dĺžku a štruktúru prečítaného slova (môže to byť aj priamo slovo u). Následne sa číta druhá časť vstupu spolu so symbolmi zo zásobníka a po jej dočítaní obyčajne vieme rozhodnúť, či celé slovo do daného jazyka patrí alebo nie. V tomto prípade však takýto postup zlyháva. Po prečítaní slova u máme totiž na vrchu zásobníka symboly súvisiace s koncovými symbolmi slova u, no na vstupe máme začiatok slova v. Keďže máme určiť, či sú to po sebe idúce čísla, potrebovali by sme porovnať konce slov u a v, pretože po sebe idúce čísla sa určite odlišujú na poslednej pozícii a možno na niekoľkých predchádzajúcich, na prvých pozíciách

sa môžu zhodovať. Potrebovali by sme teda zásobník otočiť, čo ale konečným automatom urobiť nedokážeme, pretože počet symbolov v zásobníku môže byť nekonečný. Prípadne by sme túto úlohu vedeli ľahko vyriešiť, ak by sme na vstupe mali namiesto slova v jeho reverz, teda v^R. Pozrime sa teba bližšie na to, aké vlastnosti majú binárne zápisy po sebe idúcich čísel.

Vlastnosti binárneho zápisu po sebe idúcich čísel

Pokúsme sa čo najpresnejšie popísať, ako vyzerajú binárne zápisy po sebe idúcich čísel. Dve po sebe idúce čísla majú určite rozdielnu paritu, jedno je párne a druhé nepárne. Táto vlastnosť sa na ich binárnom zápise prejaví tak, že sa určite budú odlišovať v poslednom, najmenej významnom bite. Ďalšou dôležitou vlastnosťou je, že dve po sebe idúce čísla majú binárny zápis približne rovnakej dĺžky, táto dĺžka sa môže líšiť najviac o 1. Ak teda dve čísla majú rozdiel dĺžok svojich binárnych zápisov v absolútnej hodnote väčší alebo rovný 2, tak určite nemôžu byť po sebe idúce. Venujme sa ďalej len prípadu popisovanému v zadaní, teda že $u \leq v$. Ak posledný bit slova u je 0 a ak v je nasledovník u, potom $|u| = |v|$, posledný bit v je 1 a všetky ostané bity sa zhodujú. Napr. $u = 1001011010$ a $v = 1001011011$. Ak je posledný bit u rovný 1, v ja nasleník u, potom buď $|u| = |v|$ alebo $|u|+1 = |v|$ a posledný bit v musí byť 0. Zároveň však dochádza k prenosu, to znamená, že sa menia aj vyššie bity. Ak sa posúvame od posledného bitu k prvému, kým sú v slove u bity 1, tak sú na danej pozícii v slove v bity 0. Na pozícii prvého nulového bitu v slove u od konca je na zodpovedajúcom bite vo v číslo 0. Teda ak u je predchodca v s jednotkou na konci, tak určite má sufix tvaru $0\underbrace{1\ldots1}_{i}$ a v má potom sufix $1\underbrace{0\ldots0}_{i}$, prípadne $u = \underbrace{1\ldots1}_{n}$ a $v = 1\underbrace{0\ldots0}_{n}$, ak u neobsahuje žiaden nulový bit, teda dochádza k prenosu až na najvyšší bit. V žiadnom inom prípade nemôže byť u predchodcom v.

Konštrukcia

Táto konštrukcia pomerne dosť presne simuluje vyššie popísané vlastnosti binárneho zápisu po sebe idúcich čísel. Prvý krok pri spracovávaní slova automatom A bude nedeterministické rozhodnutie, aký bude vzťah dĺžky u a v. Podľa toho si vyberieme jednu z možných vetví spracovania a v nej overíme, či bol tip správny, prípadne skontrolujeme, či u nie je predchodca v. Výpočet sa teda nedeterministicky rozdelí na tieto vetvy:

- $|u| + 2 \leq |v|$ - u nemôže byť predchodca v, teda ak slovo ucv spĺňa aj všetky ostatné požiadavky (t.j. $u, v \in 1\{0,1\}^*$, táto vlastnosť musí samozrejme platiť aj v ostatných prípadoch, kvôli stručnosti ju už ďalej nebudem explicitne uvádzať), tak ho akceptujeme

- $|u| + 1 = |v|$ - v tomto prípade u môže byť predchodcom v len vtedy, ak $u = 1^n$ a $v = 10^n$, v ostatných prípadoch môžeme slovo akceptovať

- $|u| = |v|$ - rozlišujeme ďalšie dva podprípady v závislosti od parity čísel u, v:

  - u a v majú rovnakú paritu, potom určite u nie je predchodcom v, slovo teda akceptujeme
  - u a v majú rôznu paritu, overíme, či u je predchodca v, ak nie je, tak slovo akceptujeme, pri tomto overovaní postupujeme tak, že hľadáme pozíciu, v ktorej sa slová u, v odlišujú, pričom to musí byť taká pozícia, ktorá nie je súčasťou sufixu, v ktorom sa odlišujú dve po sebe idúce čísla (teda sa musia nezhodovať aj na nejakom inom mieste ako len v sufixe, ktorý má ale nejaký špeciálny tvar)

- $|u| > |v|$ - v tomto prípade je určite $u > v$ (vďaka tomu, že u, v nemajú leading zeros), teda u určite nie je predchodcom v a celé slovo môžeme akceptovať

Konštruovaný automat má nasledujúci tvar:

$$A = (\{q_0, \ldots, q_{21}, p_1, \ldots, p_5, s_1, \ldots, s_8, r_1, \ldots, r_4, t_1, \ldots, t_{10}\}, \\ \{0, 1, c\}, \{Z_0, \alpha\}, \delta, q_0, Z_0, \{p_5, s_8, r_4, t_{10}, q_{21}\})$$

Stavy automatu sú označené rôznymi písmenami kvôli prehľadnosti, stavy označené rovnakými písmenami patria do tej istej vetvy spracovania. Jedinou výnimkou je stav q_0, ktorý je spoločný pre všetky vetvy, je počiatočným stavom a v ňom sa rozhoduje, v ktorej vetve bude výpočet následne pokračovať. Písmeno q na označenie je použité aj pre stavy, ktoré vykonávejú tú "najzaujímavejšiu" časť výpočtu, t.j. tie, v ktorých sa pre slová u, v rovnakej dĺžky a rôznej parity určuje, či u je predchodcom v alebo nie. Nakoľko je popis δ-funkcie dosť obšírny, tak ju budem uvádzať po častiach, ktoré spolu logicky súvisia.
Najprv prebehne nedeterministické rozhodnutie, do ktorej vetvy výpočtu sa vyberieme.

$$\delta(q_0, \varepsilon, Z_0) = \{(x, Z_0) \mid x \in \{q_1, p_1, s_1, r_1, t_1\}\} \quad (1)$$

Vo vetve so stavmi p_i rozoberáme možnosť, že $|u| + 2 \leq |v|$, vo vetve so stavmi s_i možnosť, že $|u| + 1 = |v|$, v stavoch t_i možnosť, že $|u| = |v|$ a zároveň u a v majú rovnakú paritu, v stavoch q_i opäť možnosť $|u| = |v|$, no tentoraz pri rôznej parite u, v a nakoniec vo vetve so stavmi r_i rozoberáme možnosť $|u| > |v|$.
Venujme sa prípadu $|u| + 2 \leq |v|$:

$$\begin{array}{lll}
\delta(p_1, 1, Z_0) = \{(p_2, Z_0\alpha\alpha)\} & & (2) \\
\delta(p_2, a, \alpha) = \{(p_2, \alpha\alpha)\} & a \in \{0,1\} & (3) \\
\delta(p_2, c, \alpha) = \{(p_3, \alpha)\} & & (4) \\
\delta(p_3, 1, \alpha) = \{(p_4, \varepsilon)\} & & (5) \\
\delta(p_4, a, \alpha) = \{(p_4, \varepsilon)\} & a \in \{0,1\} & (6) \\
\delta(p_4, a, Z_0) = \{(p_5, Z_0)\} & a \in \{0,1\} & (7) \\
\delta(p_5, a, Z_0) = \{(p_5, Z_0)\} & a \in \{0,1\} & (8)
\end{array}$$

Práca tejto vetvy automatu prebieha tak, že sa najprv skontroluje, či je prvý bit u rovný 1, pozri (2). Aj nie je, tak sa automat zasekne hneď na začiatku, teda také slovo určite neakceptuje. Ak prvý bit je 1, tak sa dostávame do stavu do stavu p_2 a zároveň do zásobníka vložíme symboly $\alpha\alpha$. V stave p_2 načítavame ostatné bity slova u a po prečítaní každého vložíme do zásobníka jeden symbol α (3). Teda pomocou symbolov α v zásobníku si počítame počet znakov slova u. Po prečítaní symbolu c sa presunieme do spracovávania druhej časti slova (4). Keď sme v stave p_3, tak máme v zásobníku $n+1$ symbolov α, kde $n = |u|$. Ten jeden znak navyše sme tam vložili v kroku (2). Na vstupe máme začiatok slova v. Opäť musíme skontrolovať, či v nemá leading zeros (5). Následne v stave p_4 čítame znaky zo vstupu a za každý prečítaný znak vyberieme zo zásobníka jednu α-u. Ak sa dostaneme do stavu, že sme vybrali všetky a na vstupe máme ešte stále nejaké znaky, znemá to, že $|v| > n + 1$, to je presne to, čo sme potrebovali a môžeme prejsť do akceptačného stavu (7), v ktorom už len dočítame zvyšok vstupu a dané slovo akceptujeme (8). Ak v nie je dostatočne dlhé slovo, tak sa výpočet zasekne. Teda touto vetvou akceptujeme jazyk $L_1 = \{ucv \mid u, v \in 1\{0,1\}^*$, kde $|u| + 2 \leq |v|\}$.
Rozoberme si ďalšiu vetvu, t.j. prípad $|u| + 1 = |v|$:

$$
\begin{array}{lll}
\delta(s_1, 1, Z_0) = \{(s_2, Z_0\alpha)\} & & (9)\\
\delta(s_2, 1, \alpha) = \{(s_2, \alpha\alpha)\} & & (10)\\
\delta(s_2, c, \alpha) = \{(s_3, \alpha)\} & & (11)\\
\delta(s_2, 0, \alpha) = \{(s_4, \alpha\alpha)\} & & (12)\\
\delta(s_4, a, \alpha) = \{(s_4, \alpha\alpha)\} & a \in \{0, 1\} & (13)\\
\delta(s_4, c, \alpha) = \{(s_5, \alpha)\} & & (14)\\
\delta(s_3, 1, \alpha) = \{(s_6, \varepsilon)\} & & (15)\\
\delta(s_5, 1, \alpha) = \{(s_7, \varepsilon)\} & & (16)\\
\delta(s_6, 0, \alpha) = \{(s_6, \varepsilon)\} & & (17)\\
\delta(s_6, 1, \alpha) = \{(s_7, \varepsilon)\} & & (18)\\
\delta(s_6, 1, Z_0) = \{(s_8, Z_0)\} & & (19)\\
\delta(s_7, a, \alpha) = \{(s_7, \varepsilon)\} & a \in \{0, 1\} & (20)\\
\delta(s_7, a, Z_0) = \{(s_8, Z_0)\} & a \in \{0, 1\} & (21)
\end{array}
$$

Opäť musíme skontrolovať, či u, v nemajú leading zeros, čo sa robí v (9) pre u a v (15) a (16) pre v. Aj v tejto vetve si podobne ako v prechádzajúcej musíme skontrolovať správnu dĺžku slov u, v, čo opäť robíme pomocou symbolu α v zásobníku. Navyše musíme sa vyhnúť akceptácii slova ucv, kde $u = 1^n$ a $v = 10^n$. To ale nie je veľmi komplikovaný problém, pretože stačí nájsť buď nulový bit v u alebo jednotkový vo v na inej ako prvej pozícii. Najprv hľadáme nulový bit v u a keď ho nájdeme, presunieme sa do podvetvy automatu, ktorá už len skontroluje dĺžku a slovo akceptuje (12). Ak sa nenašiel, pokúšame sa nájsť jednotkový bit vo v. Jeho nájdenie opäť znamená presun do akceptačnej podvetvy (18). Ak sa žiaden nenájde, tak sa automat zasekne. Akceptačná vetva overuje správnu dĺžku podobným spôsobom ako v predchádzajúcom prípade. Akonáhle sa na vrchu zásobníka objaví Z_0, prečíta sa zo vstupu ešte jeden znak a presunieme sa do akceptačného stavu (21). Ak by bol na vstupe ešte viac ako jeden symbol, automat by sa zasekol v akceptačnom stave, čo ale podľa definície možno trochu kontraintuitívne znamená, že dané slovo nebude akceptované. Teda táto vetva určite neakceptuje slová, v ktorých $|u| + 1 < |v|$. Teda táto časť automatu akceptuje jazyk $L_2 = \{ucv \mid u, v \in 1\{0, 1\}^*$, kde $|u| + 1 = |v|$ a zároveň u nie je predchodca $v\}$.
Rozoberme teraz prípad $|u| = |v|$, kde u a v majú rovnakú paritu:

$$
\begin{array}{lll}
\delta(t_1, 1, Z_0) = \{(t_2, Z_0\alpha)\} & & (22)\\
\delta(t_2, a, \alpha) = \{(t_2, \alpha\alpha)\} & a \in \{0, 1\} & (23)\\
\delta(t_2, 0, \alpha) = \{(t_3, \alpha\alpha)\} & & (24)\\
\delta(t_2, 1, \alpha) = \{(t_4, \alpha\alpha)\} & & (25)\\
\delta(t_3, c, \alpha) = \{(t_5, \alpha)\} & & (26)\\
\delta(t_4, c, \alpha) = \{(t_6, \alpha)\} & & (27)\\
\delta(t_5, 1, \alpha) = \{(t_7, \varepsilon)\} & & (28)\\
\delta(t_6, 1, \alpha) = \{(t_8, \varepsilon)\} & & (29)\\
\delta(t_7, a, \alpha) = \{(t_7, \varepsilon)\} & a \in \{0, 1\} & (30)\\
\delta(t_8, a, \alpha) = \{(t_8, \varepsilon)\} & a \in \{0, 1\} & (31)\\
\delta(t_7, 0, \alpha) = \{(t_9, \varepsilon)\} & & (32)\\
\delta(t_8, 1, \alpha) = \{(t_9, \varepsilon)\} & & (33)\\
\delta(t_9, \varepsilon, Z_0) = \{(t_{10}, Z_0)\} & & (34)
\end{array}
$$

Spracovanie v tejto vetve prebieha tak, že sa skontroluje prvý bit u (22), potom sa načítavajú ostatné bity a zároveň sa do zásobníka pomocou α zaznamenáva ich počet. Potom sa nedeterministicky rozhodne, aký bude posledný bit a podľa toho sa zvolí jedna z podvetiev výpočtu (24), (25). Toto rozhodnutie musí nastať v poslednom bite, inak sa automat zasekne. Potom sa cez c presunie výpočet do druhej polovice slova (26), (27), opät sa skontroluje prvý bit

(28), (29) a načítava sa slovo v za súčasného vyberania znakov α zo zásobníka. Nakoniec sa tipne posledný bit, ktorý ale musí byť rovnaký ako posledný bit slova u. To sa kontroluje v (32) a (33). V (34) sa prejde do akceptačného stavu, ak to bol naozaj správny bit a ak v malo rovnakú dĺžku ako u. Na vstupe už nesmie byť žiaden symbol, lebo by sa automat zasekol v akceptačnom stave. Jazyk definovaný touto vetvou je $L_3 = \{ucv \mid u, v \in 1\{0,1\}^*$, kde $|u| = |v|$ a zároveň u a v majú rovnakú paritu$\}$.
Venujme sa teraz najzložitejšiemu prípadu, t.j. $|u| = |v|$ a zároveň u a v nemajú rovnakú paritu. V tejto množine sa totiž nachádza relatívne veľa takých slov, ktoré nespĺňajú podmienku, že u nesmie byť predchodcom v.

$$\begin{array}{llr}
\delta(q_1, 1, Z_0) = \{(q_2, Z_0)\} & & (35) \\
\delta(q_2, a, Z_0) = \{(q_2, Z_0)\} & a \in \{0,1\} & (36) \\
\delta(q_2, 1, Z_0) = \{(q_3, Z_0\alpha)\} & & (37) \\
\delta(q_3, 1, \alpha) = \{(q_3, \alpha\alpha)\} & & (38) \\
\delta(q_3, 0, \alpha) = \{(q_4, \alpha\alpha)\} & & (39) \\
\delta(q_4, a, \alpha) = \{(q_4, \alpha\alpha)\} & a \in \{0,1\} & (40) \\
\delta(q_4, c, \alpha) = \{(q_5, \alpha)\} & & (41) \\
\delta(q_5, 1, \alpha) = \{(q_6, \alpha)\} & & (42) \\
\delta(q_6, a, \alpha) = \{(q_6, \alpha)\} & a \in \{0,1\} & (43) \\
\delta(q_6, 0, \alpha) = \{(q_7, \varepsilon)\} & & (44) \\
\delta(q_7, a, \alpha) = \{(q_7, \varepsilon)\} & a \in \{0,1\} & (45) \\
\delta(q_7, \varepsilon, Z_0) = \{(q_{21}, Z_0)\} & & (46) \\
\delta(q_3, c, \alpha) = \{(q_8, \alpha)\} & & (47) \\
\delta(q_8, 1, \alpha) = \{(q_9, \alpha)\} & & (48) \\
\delta(q_9, a, \alpha) = \{(q_9, \alpha)\} & a \in \{0,1\} & (49) \\
\delta(q_9, 0, \alpha) = \{(q_{10}, \varepsilon)\} & & (50) \\
\delta(q_{10}, 0, \alpha) = \{(q_{10}, \varepsilon)\} & & (51) \\
\delta(q_{10}, 1, \alpha) = \{(q_{11}, \varepsilon)\} & & (52) \\
\delta(q_{11}, a, \alpha) = \{(q_{11}, \varepsilon)\} & a \in \{0,1\} & (53) \\
\delta(q_{11}, \varepsilon, Z_0) = \{(q_{21}, Z_0)\} & & (54) \\
\delta(q_2, 0, Z_0) = \{(q_{12}, Z_0\alpha)\} & & (55) \\
\delta(q_{12}, 1, \alpha) = \{(q_{12}, \alpha\alpha)\} & & (56) \\
\delta(q_{12}, c, \alpha) = \{(q_{13}, \alpha)\} & & (57) \\
\delta(q_{13}, 1, \alpha) = \{(q_{14}, \alpha)\} & & (58) \\
\delta(q_{14}, a, \alpha) = \{(q_{14}, \alpha)\} & a \in \{0,1\} & (59) \\
\delta(q_{14}, 1, \alpha) = \{(q_{15}, \varepsilon)\} & & (60) \\
\delta(q_{15}, 0, \alpha) = \{(q_{15}, \varepsilon)\} & & (61) \\
\delta(q_{15}, 1, \alpha) = \{(q_{16}, \varepsilon)\} & & (62) \\
\delta(q_{16}, a, \alpha) = \{(q_{16}, \varepsilon)\} & a \in \{0,1\} & (63) \\
\delta(q_{16}, \varepsilon, Z_0) = \{(q_{21}, Z_0)\} & & (64) \\
\delta(q_{12}, 0, \alpha) = \{(q_{17}, \alpha\alpha)\} & & (65) \\
\delta(q_{17}, a, \alpha) = \{(q_{17}, \alpha\alpha)\} & a \in \{0,1\} & (66) \\
\delta(q_{17}, c, \alpha) = \{(q_{18}, \alpha)\} & & (67) \\
\delta(q_{18}, 1, \alpha) = \{(q_{19}, \alpha)\} & & (68) \\
\delta(q_{19}, a, \alpha) = \{(q_{19}, \alpha)\} & a \in \{0,1\} & (69) \\
\delta(q_{19}, 1, \alpha) = \{(q_{20}, \varepsilon)\} & & (70) \\
\delta(q_{20}, a, \alpha) = \{(q_{20}, \varepsilon)\} & a \in \{0,1\} & (71) \\
\delta(q_{20}, \varepsilon, Z_0) = \{(q_{21}, Z_0)\} & & (72)
\end{array}$$

Táto časť sa dosť výrazne odlišuje od ostatných častí automatu A. Jednak preto, že neprevádza kontrolu dĺžky slova a tiež aj tým, že akceptuje o niečo viac, ako nutne potrebujeme. Výpočet v tejto časti prebieha tak, že hľadáme bit, resp. pozíciu, v ktorej sa slová u a v odlišujú a zároveň to nie je posledná pozícia alebo pozícia, ktorá je ovplyvnená prenosom, ktorý môže vzniknúť pri pričítavaní 1 k nejakému číslu. Slovo spracúvame tak, že načítavame znaky zo vstupu, kým sa nedostaneme na pozíciu, o ktorej si nedeterministicky tipneme, že je tou pozíciou, v ktorej sa dané dve slová odlišujú (37), (55). Ak je na danej pozícii v slove u 1, tak na to, aby sme zaručili, že u nie je predchodca v nám stačí ukázať, že na zodpovedajúcej pozícii od konca je v slove v 0 a ak táto pozícia je n-tá od konca, tak sa v slove u nachádza na posledných $n-1$ pozíciách aspoň jedna 0 alebo v slove v sa na jednej z posledných $n-1$ pozícií nachádza aspoň jedna 1. Ak je teda na tipnutej pozícii v slove u znak 1, tak kontrolujeme jeho zvyšok, či obsahuje samé jednotky (38). Ak na niektorej zo zvyšných pozícií nájdeme aspoň jeden nulový bit (39), tak už len overíme, či je v slove v na n-tom mieste od konca 0 (44) a ak je, tak celé slovo môžeme akceptovať. Toto overovanie robíme za pomoci symbolov α v zásobníku, ktorých počet v tomto prípade, na rozdiel od predchádzajúcich, neznamená dĺžku slova u, ale pozíciu overovaného bitu od konca. Ak sú všetky zvyšné bity jednotkové, prejdeme na spracovanie slova v. Opäť odignorujeme počiatočné znaky až po ten, o ktorom si tipneme, že je n-tý od konca. Znovu pomocou vyberania znaku α zo zásobníka overíme, či bol tip správny. Na tipnutej pozícii sa musí nachádzať 0 (50), inak by sa slová u, v v tomto bite nelíšili. Potom vo v na posledných $n-1$ pozíciách hľadáme nejaký jednotkový bit. Ak ho nájdeme (52), stačí dočítať vstup (53) (teda overiť správnosť nedeterministického tipu) a môžeme slovo akceptovať. Ako bolo vyššie spomenuté, v tejto vetve neoverujeme, či slová u, v majú naozaj rovnakú dĺžku, symbol α v zásobníku má inú funkciu. Je teda riziko, že by sme mohli akceptovať nejaké slovo, v ktorom $|u| \neq |v|$ a zároveň u je predchodca v. To ale môže nastať len v prípade, keď $u = 1^n$ a $v = 10^n$. To ale znamená, že nech si v slove u tipneme ľubovoľný bit, tak vždy bude mať hodnotu 1. Za ním ale nasledujú už len samé jednotkové bity. Na zodpovedajúcej pozícii v slove v je síce 0, ale nulové sú aj všetky nasledujúce bity, teda takéto slovo táto časť automatu neakceptuje.

Analogickým spôsobom postupujeme aj vtedy, keď tipnutý bit v u je nulový. Potom v môže byť následníkom u len vtedy, keď má v na n-tej pozícii od konca znak 1, posledných $n-1$ bitov v u je jednotkových a posledných $n-1$ bitov vo v je nulových. Či nastal tento prípad overujeme rovnako ako v predchádzajúcom prípade. Tiež nemôže dôjsť k tomu, že by sme akceptovali nejaké po sebe slovo ucv, kde $|u| \neq |v|$ a zároveň $u+1=v$, pretože jediný takýto prípad je ten, ktorý bol vyššie popísaný a v tomto prípade u neobsahuje žiadne nulový bit, teda opäť sa nemôže stať, že by sme akceptovali do jazyka nepatriace slovo.

Jazyk definovaný touto časťou automatu je $L_4 = \{ucv \mid u, v \in 1\{0,1\}^*$ a zároveň u, v sa odlišujú aspoň v jednom bite, ktorý ale nie je súčasťou charakteristického sufixu po sebe idúcich čísel$\}$.

Nakoniec rozoberme poslednú možnosť, t.j. $|u| > |v|$:

$$\begin{aligned}
&\delta(r_1, 1, Z_0) = \{(r_2, Z_0)\} && && (73)\\
&\delta(r_2, a, x) = \{(r_2, x\alpha)\} && a \in \{0,1\}, x \in \Gamma && (74)\\
&\delta(r_2, c, \alpha) = \{(r_3, \alpha)\} && && (75)\\
&\delta(r_3, 1, \alpha) = \{(r_4, \varepsilon)\} && && (76)\\
&\delta(r_4, a, \alpha) = \{(r_4, \varepsilon)\} && a \in \{0,1\} && (77)
\end{aligned}$$

Výpočet tejto časti automatu je podobný výpočtu pre $|u| + 2 \leq |v|$, len do zásobníka vložíme len $|u| - 1$ symbolov α a ak ich pri spracovaní vyberieme všetky, tak sa zaseknem v akceptačnom stave, lebo nemôžeme dočítať vstup, teda ani nemôžeme akceptovať. Ak úspešne dočítame vstup, tak sme v akceptačnom stave a na zásobníku sa ešte môžu nachádzať symboly α, ale to akceptácii nebráni. Jazyk definovaný touto časťou automatu je $L_5 = \{ucv \mid u, v \in 1\{0,1\}^*$, kde $|u| > |v|\}$.

Dôkaz správnosti konštrukcie

V dôkaze budeme postupovať tak, že ukážeme rovnosť $L = L_1 \cup L_2 \cup L_3 \cup L_4 \cup L_5$. Z definície týchto jazykov je zrejmé, že $L_i \subseteq L$ a teda $L_1 \cup L_2 \cup L_3 \cup L_4 \cup L_5 \subseteq L$. Označme L_4' je taká podmnožina L, že v nej platí $|u| = |v|$. Potom jazyky L_1, L_2, L_3, L_4', L_5 zrejme tvoria rozklad jazyka L, t.j. $L_1 \cap L_2 \cap L_3 \cap L_4' \cap L_5 = \emptyset$ a $L_1 \cup L_2 \cup L_3 \cup L_4' \cup L_5 = L$. Keďže $L_4' \subseteq L_4$, potom $L_1 \cup L_2 \cup L_3 \cup L_4 \cup L_5 \supseteq L$, teda naozaj $L = L_1 \cup L_2 \cup L_3 \cup L_4 \cup L_5$, teda nám stačí dokázať, že jednotlivé časti automatu naozaj generujú im priradený jazyk.

Urobme dôkaz, že L_1 je rovný jazyku, ktorý generuje prislúchajúca časť automatu.
$\subseteq$:
Dôkaz indukciou na dĺžku slova u.
I. Ak $|u| = 1$, tak $|v| \geq 3$. Po prečítaní slova u sa do zásobníka vložia dva symboly α, tie sa vyberú po prečítaní prvých dvoch znakov z v, prečítaním tretieho znaku sa presunieme do akceptačného stavu a slovo akceptujeme.
II. Predpokladajme platnosť pre $|u| = n$, ukážme platnosť pre $|u| = n + 1$. Ak $|u| = n + 1$, potom $|v| = m \geq n + 3$. Odoberme posledný znak z u i v, takto vzniknuté slová označme u', v'. Slovo zložené z týchto častí patrí podľa predpokladu do L_1. Ak k u' pridáme na koniec ľubovoľný znak, tak sa to pri jeho spracovaní automatom prejaví tak, že sa do zásobníka vloží o jeden symbol α viac. Ak vezmeme slovo v' predĺžené o ľuboboľný znak z $\{0,1\}$, tak tento znak spôsobí vybratie nadbytočného znaku α zo zásobníka, teda slovo zložené z u', v' predĺžených o jeden znak bude tiež automatom akceptované. Formálne to môžeme zapísať takto: $(p_1, u'xcv'y, Z_0) \vdash^* (p_2, xcv'y, Z_0\alpha^{|u'|+1}) \vdash (p_2, cv'y, Z_0\alpha^{|u'|+2}) \vdash^* (p_4, y, Z_0) \vdash (p_5, \varepsilon, Z_0)$, kde $x, y \in \{0,1\}$.
$\supseteq$:
Indukciou na počet použití predpisu (3) δ-funkcie:
I. Ak $n = 0$, tak pred použitím pravidla (4) použijeme len raz pravidlo (2). V zásobníku teda máme dva znaky α. Pri spracovní v jeden z nich vyberieme po prečítaní prvého bitu, ďalší po prečítaní nasledujúceho znaku a na prechod do akceptačného stavu potrebujeme ešte aspoň jeden znak na vstupe. Teda slovo v musí mať aspoň 3 znaky.
II. Ak sme použili predpis (3) $n + 1$ krát, máme v zásobníku o jeden znak α viac ako pri n-násobnom použití. Pri n-násobnom použití sme potrebovali na ich odstránenie slovo v dĺžky $n + 3$, pričom dĺžka u je v tomto prípade $n + 1$. Ak sme slovo u predĺžili o jeden znak, tak na vyprázdnenie zásobníka a následný prechod do akceptačného stavu potrebujeme slovo v dĺžky aspoň $n + 4$.
Teda L_1 je naozaj jazyk, ktorý generuje jemu priradená časť auotomatu.

Urobme dôkaz, že L_2 je rovný jazyku, ktorý generuje prislúchajúca časť automatu.
$\subseteq$:
Indukciou na dĺžku slova $|u| = n = |v| - 1$.
I. Ak $n = 1$ tak, tak $u = 1$. Jeho spracovanie prebieha tak, že sa po jeho prečítaní vloží do zásobníka jeden znak α. Ten sa vyberie po prečítaní prvého znaku v. Druhý znak v môže byť 0 alebo 1. Ak je to 0, tak $v = 10$, čo je ale následník u a automat ho skutočne neakceptuje, ale sa zasekne v stave s_6. Ak je to 1, teda $v = 11$, tak sa zo zásobníka vyberú všetky znaky α hneď pri čítaní prvého znaku, následne sa pomocou predpisu (19) dostaneme do akceptačného stavu a slovo akceptujeme.

II. Ak automat akceptuje slovo $ucv \in L_2$ také, že $|u| = n$, ukážme, že akceptuje aj slová, kde $|u| = n+1$. Vezmime slovo $u'cv'$, kde $|u'| = n$. Ak u' aj v' predĺžime o ľubovoľný znak z $\{0, 1\}$, tak takto vzniknuté slovo ucv bude automatom akceptované, pretože buď má u na niektorej pozícii okrem prvej nulový bit, alebo má v na niektorej pozícii okrem prvej jednotkový bit, pretože túto vlastnosť majú podľa indukčného predpokladu aj ich podslová u', v', teda slovo ucv automat akceptuje. Ak slová u', v' majú tvar 1^n, resp. 10^n, teda nepatria do jazyka L_2, tak ich predĺženia patria do jazyka L_2 len vtedy, ak u' predĺžime o nulový bit alebo slovo v' o jednotkový bit. Ak spracúvavame slovo ucv vzniknuté predĺžením u', v', tak po spracovaní u' sme v stave s_2. Ak sa prečíta ako posledný znak u 0, tak sa dostaneme do akceptačnej vetvy kde overenie dĺžky bude z indukčného predpokladu úspešné. Ak je posledný bit 1, prejdeme na spracovanie v. Po prečítaní v' máme na zásobníku už len symbol Z_0 a podľa posledného bitu sa rozhodne, či bude slovo akceptované. Akceptované je len vtedy, keď je jednotkový. Teda naozaj slová $ucv \in L_2$, kde $|u| = n + 1$ sú akceptované automatom.
$\supseteq$:
Automat má tú vlastnosť, že najprv do zásobníka symboly α vkladá a potom ich odtiaľ vyberá, pričom po vybratí prvého sa už žiadne ďalšie nevkladajú. To využijeme v tejto časti dôkazu, ktorý urobíme indukciou na počet symbolov α vložených do zásobníka.
I. Ak $n = 1$, potom $u = 1$ a slovo ucv akceptujeme len vtedy, keď $v = 11$, viď (15) a (19). Toto slovo patrí do L_2.
II. Ukážme pre $n + 1$. Ak do zásobníka vložíme o jeden symbol α viac, znamená to, že slovo u sa predĺžilo o jeden znak. Na jeho opätovné vybratie budeme potrebovať predĺžiť aj slovo v o jeden znak, predĺženie o viac znakov by spôsobilu neakceptáciu slova. Je jedno, o aký znak slová u, v predĺžime, ak ucv akceptoval automat, pretože buď u musel obsahovať nejaký nulový bit alebo v musel obsahovvať jednotkový bit aj na inej ako prvej pozícii. Tento bit "umožňujúci akceptáciu" sa ale nachádza aj v predĺžených slovách a teda nám garantuje príslušnosť daného slova v jazyku L_2. Automat akceptuje aj slová zložené z $u = 1^n0, v = 10^{n+1}$, resp. $u = 1^{n+1}, v = 10^n1$, hoci slová zložené z u, v bez posledného bitu nie sú akceptované automatom a nepatria ani do jazyka L_2. Práve ich posledný bit umožní akceptáciu, čo je korektné, leto tieto slová naozaj patria do L_2. Podrobnejšie by sa to dalo rozobrať podobne ako v predchádzajúcej inklúzii. Automat správne neakceptuje slová 1^mc10^m, ktoré do jazyka L_2 nepatria.

Dôkaz pre L_3 si dovolím vynechať, pretože sa dá urobiť veľmi podobne ako predchádzajúci.

Rozoberme najťažší prípad, t.j. časť automatu, ktorá akceptuje jazyk L_4. Najprv si ale zopakujme, čo presne je myslené formuláciou "odližujú sa aspoň v jednom bite, ktorý nie je súčasťou charakteristického sufixu dvoch po sebe idúcich čísel". Ak u, v sú po sebe idúce čísla, tak ich sufixy sú 01^m a 10^m, kde $m \geq 0$. Automat hľadá, resp. nedeterministicky tipuje bit, v ktorom sa slová odlišujú, pričom za týmto bitom sa v slove u musí nachádzať aspoň jeden nulový bit, alebo sa v slove v musí za príslušným bitom nachádzať aspoň jeden jednotkový bit. Každé dve po sebe nenasledujúce čísla rôznej parity majú túto vlastnosť (podmienka na rôznu paritu je tam kvôli prípadu $u = v$, pretože tieto dve slová nie sú po sebe idúce, no nelíšia sa v žiadnom bite).
$\subseteq$:
Tak ako v predchádzajúcich prípadoch by sme mohli skúsiť dôkaz indukciou, no ten by bol dosť komplikovaný. Pokúsme sa radšej o nejaký jednoduchší priamy dôkaz. Nech slovo ucv patrí do L_4, potom z toho, čo vieme o vlastnostiach binárneho zápisu po sebe idúcich čísel vieme povedať, že slovo u určite obasahuje bit, ktorého hodnota je rôzna od hodnoty zod-

povedajúceho bitu v slove v. Slovo zodpovedajúci znamená ten istý v poradí počítajúc od konca. Ešte raz zdôraznime, že dĺžka u a v nemusí byť nutne rovnaká. Výpočet v automate prebieha nasleduúcim spôsobom. Po kontrole prvého bitu kvôli overeniu, že u neobsahuje leading zeros sa odignoruje začiatok slova (36) až po bit, v ktorom nastane nedeterministické rozhodnutie. Nemôže to ale byť prvý bit, inak by jedno zo slov obsahovalo na začiatku nulu. Za týmto odlišným bitom kontrolujeme ostatok slova u, či obsahuje len jednotkové bity a zároveň si pomocou symbolu α počítame pozíciu tohto rozdielneho bitu od konca. Akonáhle nájdeme nulový bit, presunieme sa do stavu q_4, resp. q_{17}, podľa toho, akú hodnotu mal tipnutý bit, teda do inej podvetvy výpočtu, ktorá už len skutočne dočíta vstup a v slove v overí či je na n-tej pozícii od konca bit opačnej hodnoty ako v slove u, kde n je samozrejme pozícia tipnutého bitu od konca slova. Teda slová, ktoré majú v slove u za tipnutým bitom nejakú nulu sú akceptované. Ak sa žiadna nula vo zvyšku slova u nenájde, tak sa prejde na spracovanie slova v, skontroluje sa prvý bit, odignoruje sa začiatok slova až po tipnutý bit. Ten musí mať opačnú hodnotu ako zodpovedajúci bit v u. Potom sa za ním hľadá nejaký jednotkový bit. Ak sa tam už žiaden nenachádza, tak výpočet ostane zaseknutý v stave q_{10}, resp. q_{15}. Jediná cesta, ktorá zabezpečí cestu do akceptačného stavu je nájdenie znaku 1 (52), (62). Potom sa overí ešte správna poloha tipovaného bitu a slovo sa zakceptuje. Automat teda akceptuje všetky slová z jazyka L_4.
$\supseteq$:
Opät urobme kvôli jednoduchosti priamy dôkaz. Konštrukcia tejto časti automatu zaručuje, že akceptované budú len slová, ktoré obsahujú bit, v ktorom sa podslová u, v líšia. Zároveň sa musí za rozdielnym bitom v u nachádzať aspoň jedna jednotka alebo vo v za rozdielnym bitom aspoň jedna nula, inak by ucv automatom nebolo akceptované. No táto vlastnosť nám už priamo implikuje, že $ucv \in L_4$ (podrobné vysvetlenie by bolo podobné tomu z opačnej implikácie a kvôli tejto analogickosti ho nebudem uvádzať).

Dôkaz pre L_5 by sme mohli urobiť v oboch inklúziách analogickou indukciou ako pre L_1, prípadne aj jednoduchým priamym dôkazom.

Tým, že sme ukázali, že jednotlivé časti automatu skutočne generujú im priradený jazyk L_i sme vlastne dokázali správnosť celej konštrukcie, teda rovnosť $L = L(A)$.