

---

# **PAL: 9. cvičení**

Tomáš Sieger

19. 11. 2020

---

# Opakování z minula

## Př. 8/1: skládání automatů 2

---

Nad abecedou  $\{0, 1\}$ , jsou dány dva jazyky  $L_1$  a  $L_2$ . Slova  $L_1$  jsou popsána výrazem  $0 * 1 * 0 * 1 * 0^*$ , slova  $L_2$  jsou popsána výrazem  $(01 + 10)^*$ .

Sestrojte konečné automaty rozpoznávající jazyk:

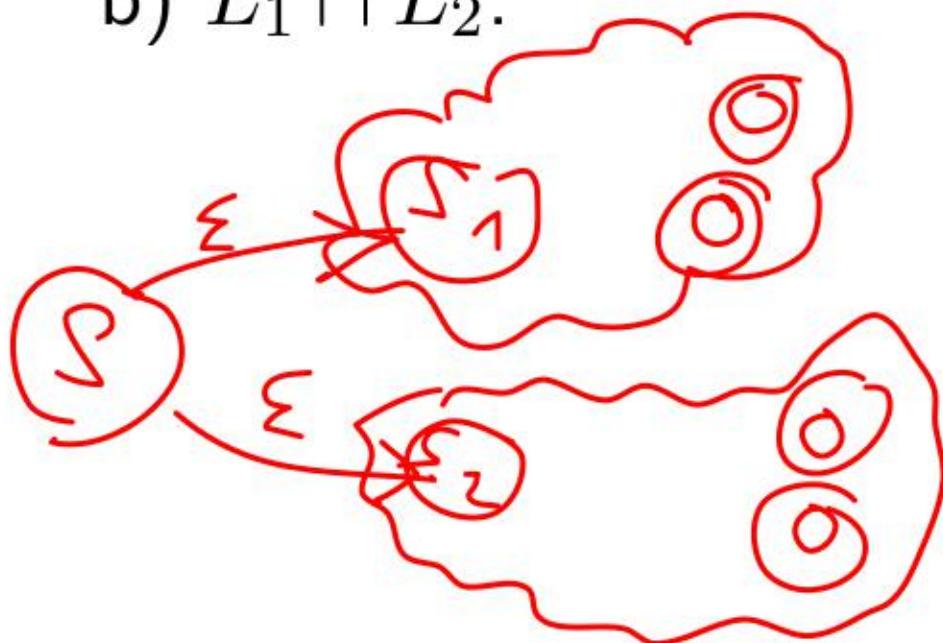
- a)  $L_1 \cup L_2$ ,
- b)  $L_1 \cap L_2$ .

## Př. 8/1: skládání automatů 2

Nad abecedou  $\{0, 1\}$ , jsou dány dva jazyky  $L_1$  a  $L_2$ . Slova  $L_1$  jsou popsána výrazem  $0 * 1 * 0 * 1 * 0^*$ , slova  $L_2$  jsou popsána výrazem  $(01 + 10)^*$ .

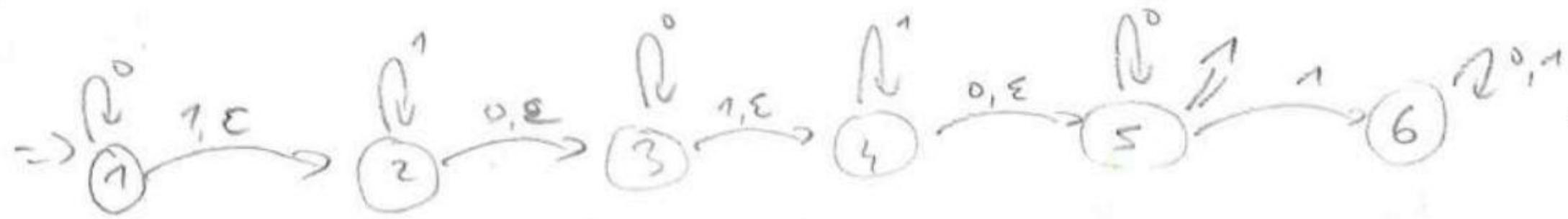
Sestrojte konečné automaty rozpoznávající jazyk:

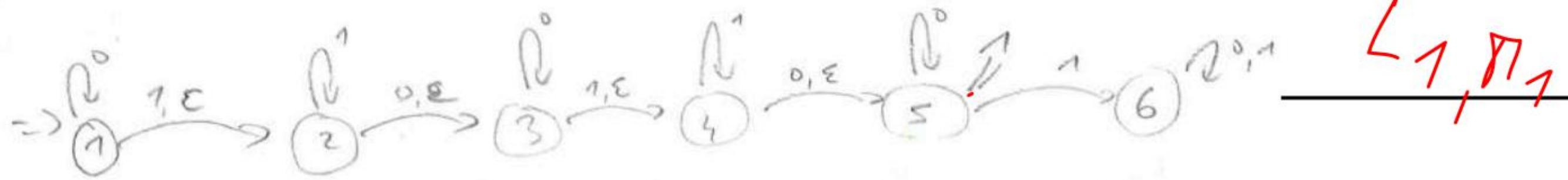
- a)  $L_1 \cup L_2$ ,
- b)  $L_1 \cap L_2$ .



$$(00) \times$$
  
$$(000) \times$$

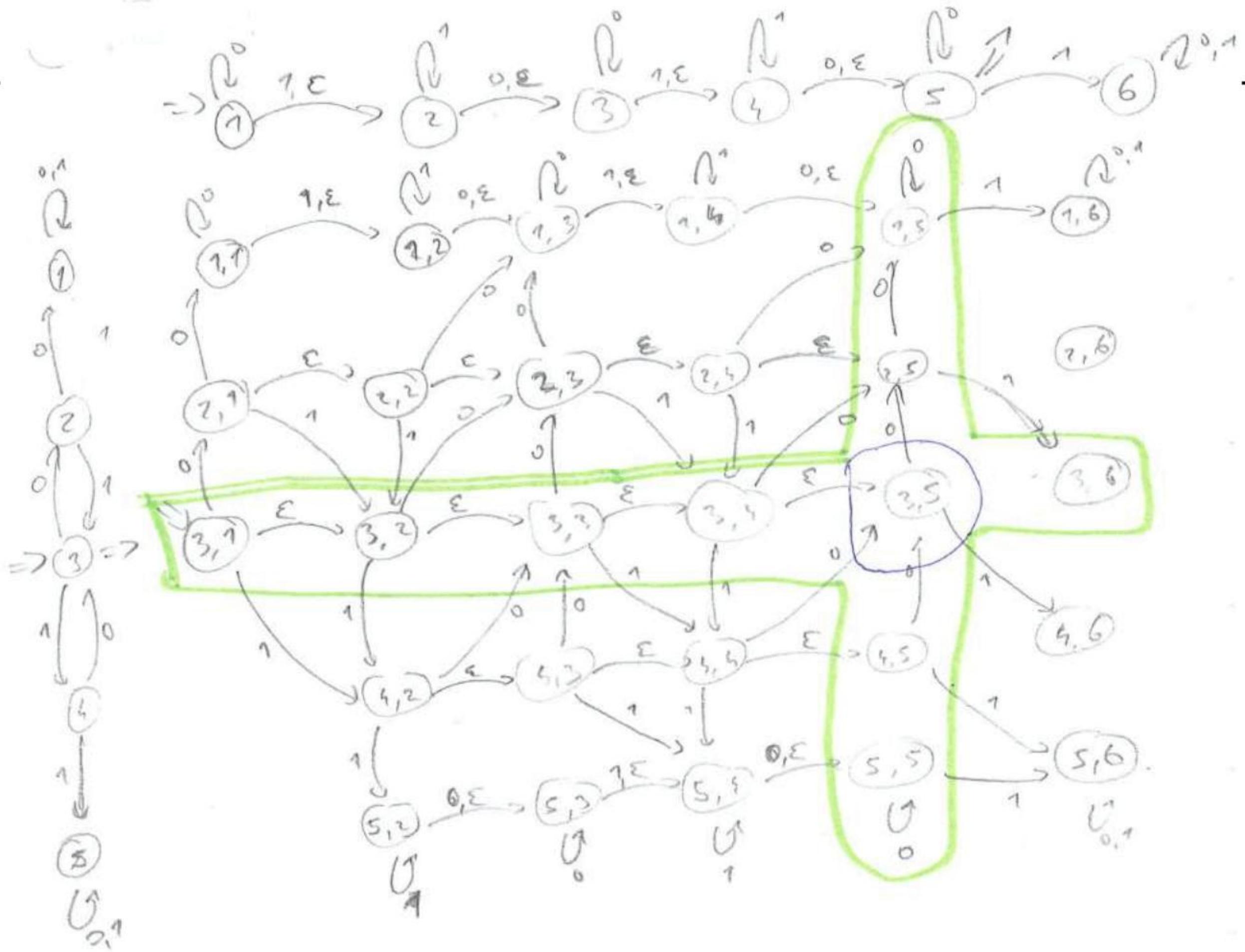






$L_1, \pi_1$



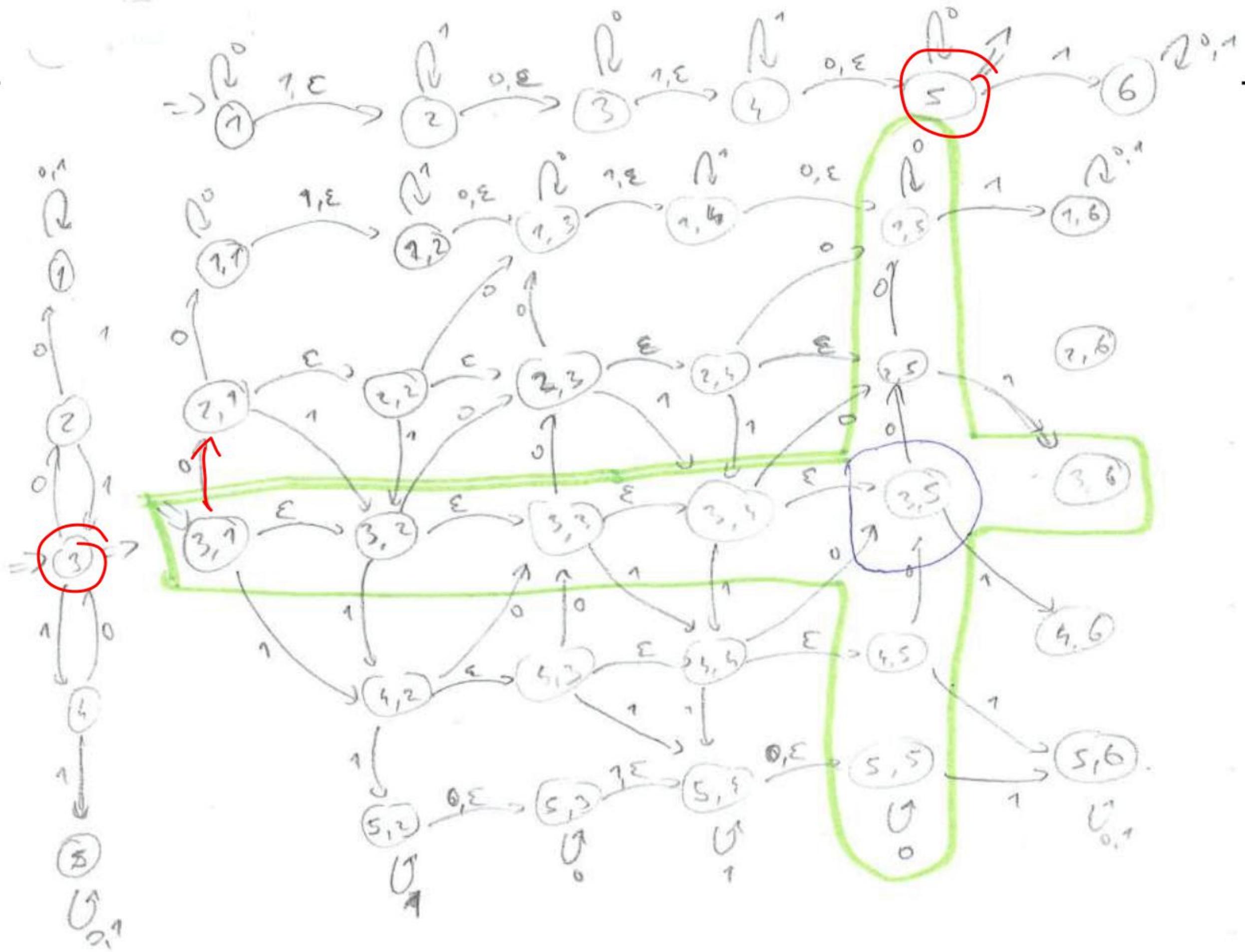


b)

$L_1 \cap L_2$



$L_1 \cup L_2$



b)



$L_1 \cup L_2$

Yes

9

100%

No 0

50% 48



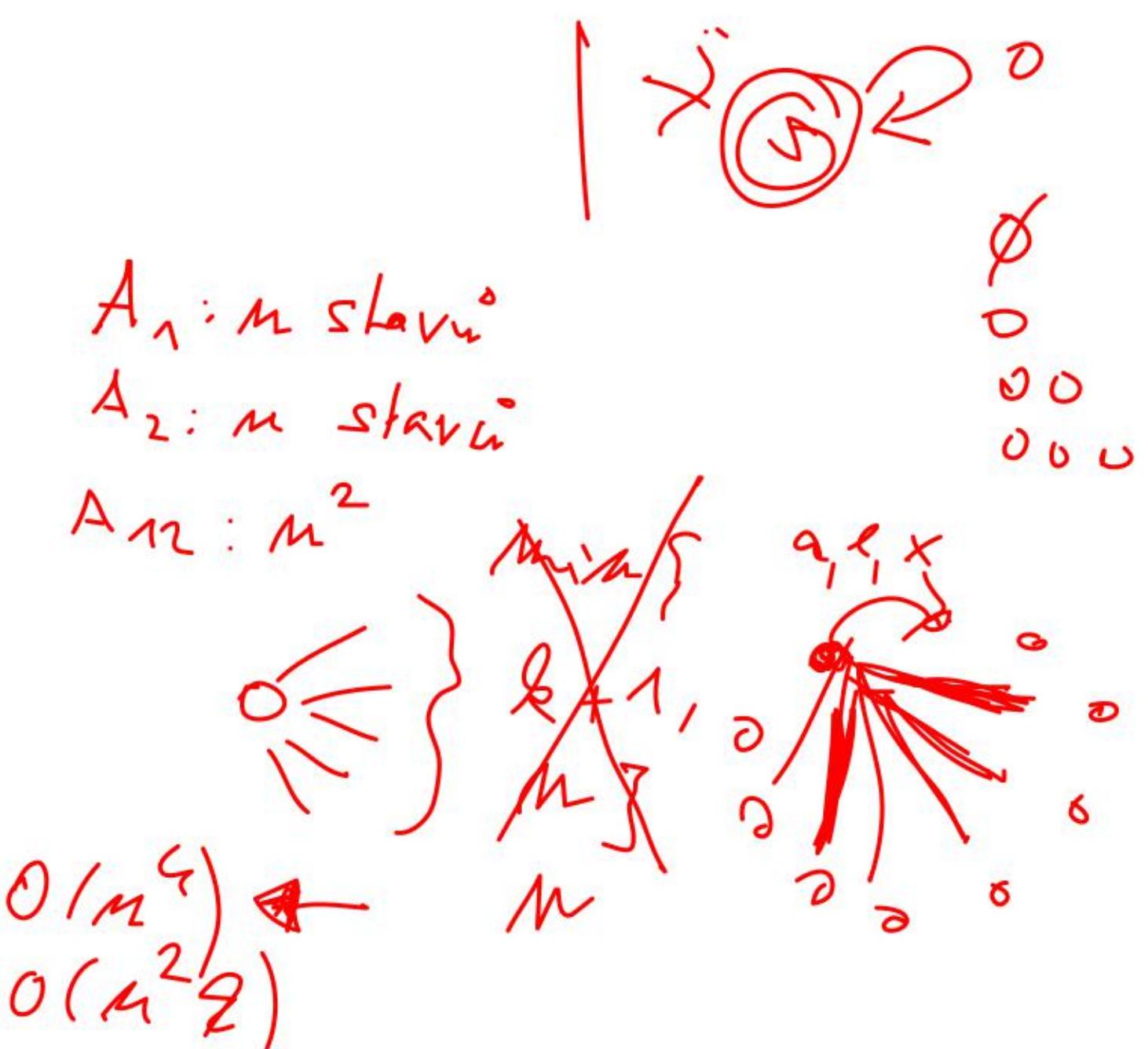
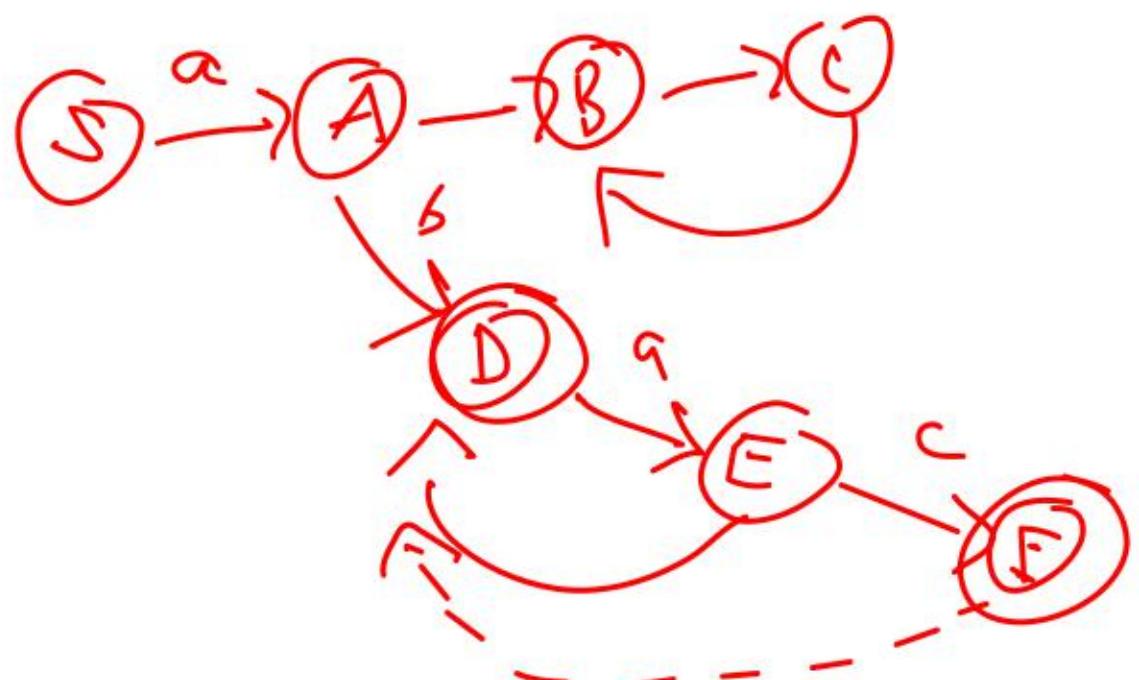
## Př. 8/2: konečný průnik

---

Automat  $A_1$  rozpoznává jazyk  $L_1$ , automat  $A_2$  rozpoznává jazyk  $L_2$ . Oba automaty mají  $n$  stavů. Abeceda pro oba jazyky je shodná a má  $k$  znaků. Jaká je asymptotická složitost algoritmu, který efektivně určí, zda jazyk  $L_1 \cap L_2$  je konečný?

## Př. 8/2: konečný průnik

Automat  $A_1$  rozpoznává jazyk  $L_1$ , automat  $A_2$  rozpoznává jazyk  $L_2$ . Oba automaty mají  $n$  stavů. Abeceda pro oba jazyky je shodná a má  $k$  znaků. Jaká je asymptotická složitost algoritmu, který efektivně určí, zda jazyk  $L_1 \cap L_2$  je konečný?





## Př. 8/3: hledání Hammingovsky pozměněného slova

---

V textu nad abecedou  $\{a, b, c, d\}$  máme určit všechny výskyty takových podřetězců, které začínají i končí znakem  $b$  a zároveň mají od daného vzorku  $abbbcdabcdab$  Hammingovu vzdálenost větší než 2. Navrhněte konečný nedeterministický automat pro řešení této úlohy.

## Př. 8/3: hledání Hammingovsky pozměněného slova

---

V textu nad abecedou  $\{a, b, c, d\}$  máme určit všechny výskytu takových podřetězců, které začínají i končí znakem  $b$  a zároveň mají od daného vzorku  $abbbcdabcdab$  Hammingovu vzdálenost ~~větší~~ než 2. Navrhněte konečný ~~nedeterministický~~ automat pro řešení této úlohy.



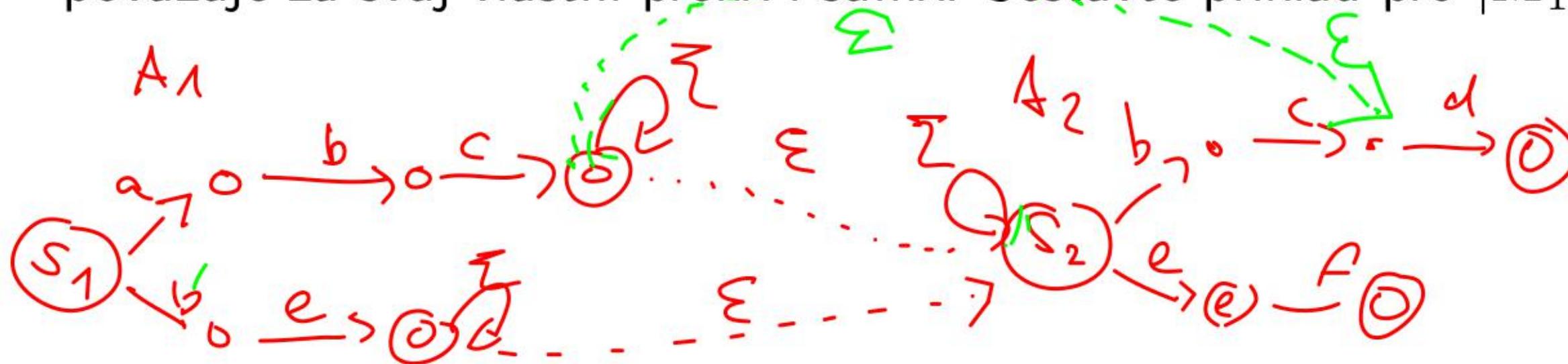
## Př. 8/12: předpona a přípona

---

Nad abecedou  $A$  jsou dány dvě konečné množiny řetězců  $M_1$  a  $M_2$ . Popište, jak sestavíte konečný automat, který přijímá všechna taková slova  $w$  nad abecedou  $A$ , pro která platí, že alespoň jeden prefix slova  $w$  leží v množině  $M_1$  a alespoň jeden suffix  $w$  leží v množině  $M_2$ . Připomeňme, že celé slovo se považuje za svůj vlastní prefix i suffix. Sestavte příklad pro  $|M_1| = |M_2| = 2$ .

## Př. 8/12: předpona a přípona

Nad abecedou  $A$  jsou dány dvě konečné množiny řetězců  $M_1$  a  $M_2$ . Popište, jak sestavíte konečný automat, který přijímá všechna taková slova  $w$  nad abecedou  $A$ , pro která platí, že alespoň jeden prefix slova  $w$  leží v množině  $M_1$  a alespoň jeden suffix  $w$  leží v množině  $M_2$ . Připomeňme, že celé slovo se považuje za svůj vlastní prefix i suffix. Sestavte příklad pro  $|M_1| = |M_2| = 2$ .



$$L_1 = \{abc, bcd\}$$

abcd

$$L_2 = \{bcd, efd\}$$

$$L_1 \cap L_2 \dots$$

$$L_{12} = L_1 \cap L_2$$

$$A_{12} = A_1 \cap A_2$$

Yes

7

100%

No 0

110% 48



## Př. 8/3: hledání Hammingovsky pozměněného slova

---

V textu nad abecedou  $\{a, b, c, d\}$  máme určit všechny výskyty takových podřetězců, které začínají i končí znakem  $b$  a zároveň mají od daného vzorku  $abbbcdabcdab$  Hammingovu vzdálenost ~~větší~~ než 2. Navrhněte konečný ~~nedeterministický~~ automat pro řešení této úlohy.

## Př. 8/3: hledání Hammingovsky pozměněného slova

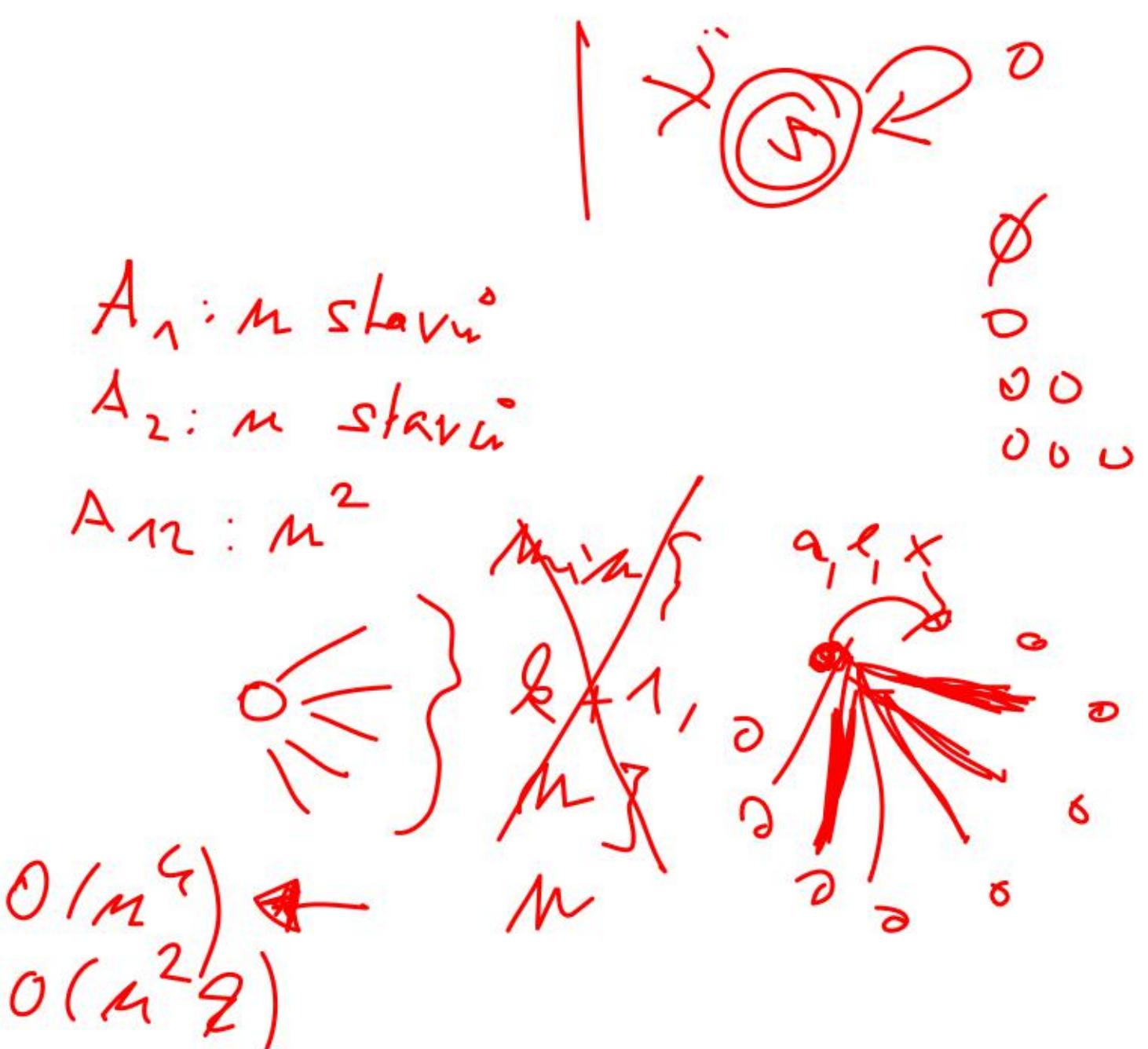
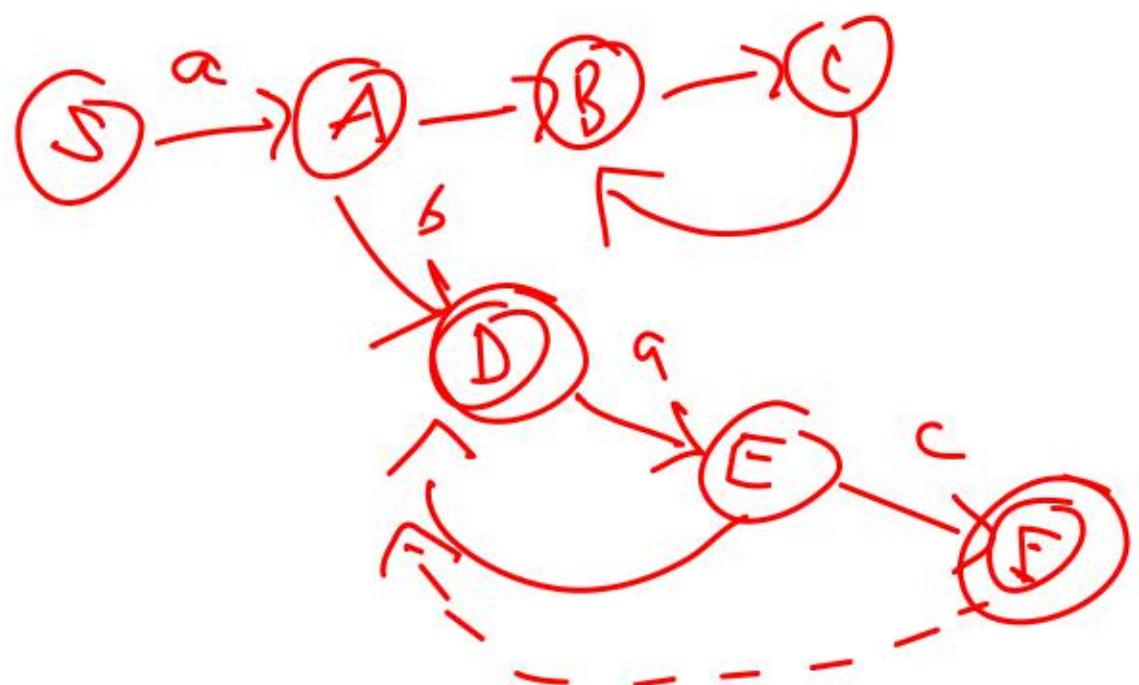
---

V textu nad abecedou  $\{a, b, c, d\}$  máme určit všechny výskyty takových podřetězců, které začínají i končí znakem  $b$  a zároveň mají od daného vzorku  $abbbcdabcdab$  Hammingovu vzdálenost ~~větší~~ než 2. Navrhněte konečný ~~nedeterministický~~ automat pro řešení této úlohy.



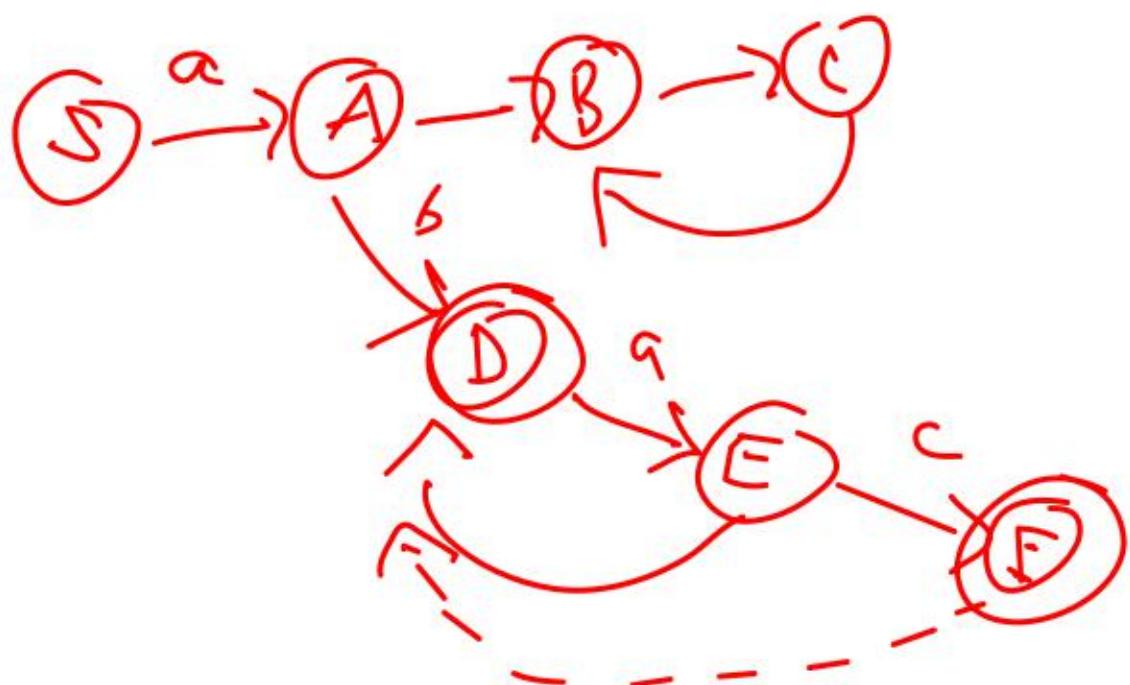
## Př. 8/2: konečný průnik

Automat  $A_1$  rozpoznává jazyk  $L_1$ , automat  $A_2$  rozpoznává jazyk  $L_2$ . Oba automaty mají  $n$  stavů. Abeceda pro oba jazyky je shodná a má  $k$  znaků. Jaká je asymptotická složitost algoritmu, který efektivně určí, zda jazyk  $L_1 \cap L_2$  je konečný?



## Př. 8/2: konečný průnik

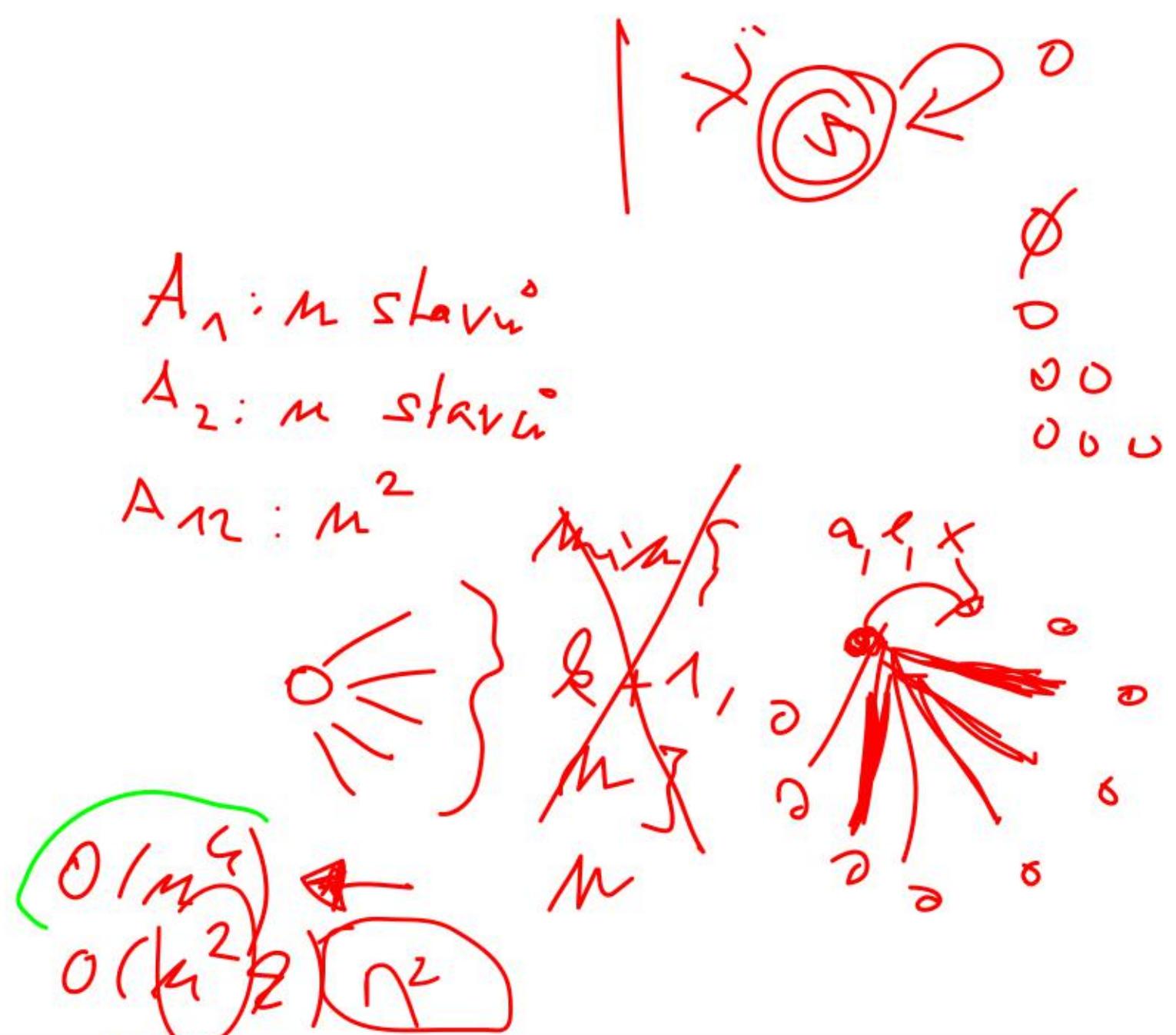
Automat  $A_1$  rozpoznává jazyk  $L_1$ , automat  $A_2$  rozpoznává jazyk  $L_2$ . Oba automaty mají  $n$  stavů. Abeceda pro oba jazyky je shodná a má  $k$  znaků. Jaká je asymptotická složitost algoritmu, který efektivně určí, zda jazyk  $L_1 \cap L_2$  je konečný?



$$|V| = n^2$$

$$|E| = \binom{n^2}{2} = O(n^4)$$

$$O(|V| + |E|) = O(n^2 + n^4) = O(n^4)$$





## Př. 8/3: hledání Hammingovsky pozměněného slova

---

V textu nad abecedou  $\{a, b, c, d\}$  máme určit všechny výskyty takových podřetězců, které začínají i končí znakem  $b$  a zároveň mají od daného vzorku  $abbbcdabcdab$  Hammingovu vzdálenost ~~větší~~ než 2. Navrhněte konečný ~~nedeterministický~~ automat pro řešení této úlohy.

## Př. 8/3: hledání Hammingovsky pozměněného slova

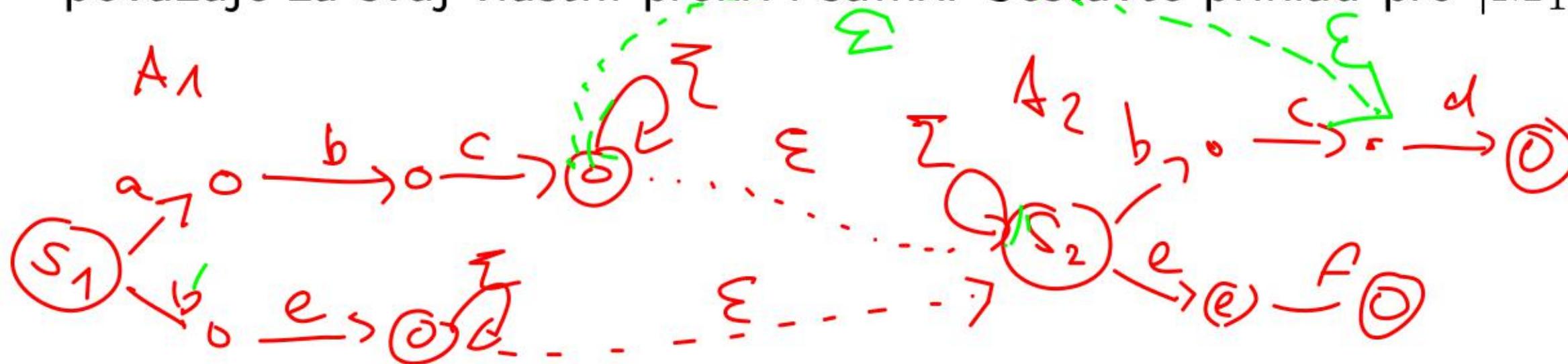
---

V textu nad abecedou  $\{a, b, c, d\}$  máme určit všechny výskyty takových podřetězců, které začínají i končí znakem  $b$  a zároveň mají od daného vzorku  $abbbcdabcdab$  Hammingovu vzdálenost ~~větší~~ než 2. Navrhněte konečný ~~nedeterministický~~ automat pro řešení této úlohy.



## Př. 8/12: předpona a přípona

Nad abecedou  $A$  jsou dány dvě konečné množiny řetězců  $M_1$  a  $M_2$ . Popište, jak sestavíte konečný automat, který přijímá všechna taková slova  $w$  nad abecedou  $A$ , pro která platí, že alespoň jeden prefix slova  $w$  leží v množině  $M_1$  a alespoň jeden suffix  $w$  leží v množině  $M_2$ . Připomeňme, že celé slovo se považuje za svůj vlastní prefix i suffix. Sestavte příklad pro  $|M_1| = |M_2| = 2$ .



$$L_1 = \{abc, bcd\}$$

abcd

$$L_2 = \{bcd, e\}$$

$$\begin{matrix} L_1 & L_2 & \dots \\ L_2 & \dots & L_1 \end{matrix}$$

$$L_{12} = L_1 \cap L_2$$

$$A_{12} = A_1 \cap A_2$$

Yes

7

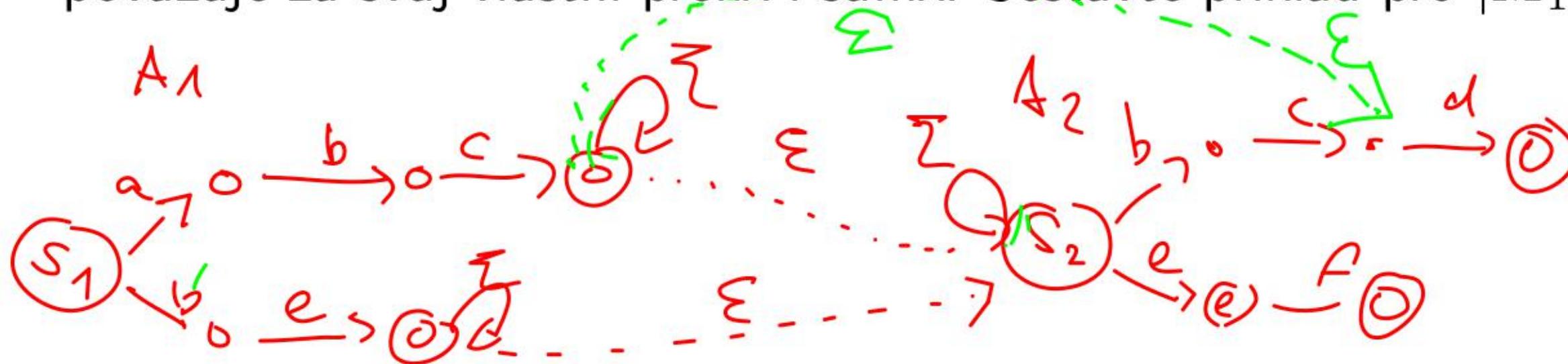
100%

No 0

110%  
48

## Př. 8/12: předpona a přípona

Nad abecedou  $A$  jsou dány dvě konečné množiny řetězců  $M_1$  a  $M_2$ . Popište, jak sestavíte konečný automat, který přijímá všechna taková slova  $w$  nad abecedou  $A$ , pro která platí, že alespoň jeden prefix slova  $w$  leží v množině  $M_1$  a alespoň jeden suffix  $w$  leží v množině  $M_2$ . Připomeňme, že celé slovo se považuje za svůj vlastní prefix i suffix. Sestavte příklad pro  $|M_1| = |M_2| = 2$ .



$$M_1 = \{abc, b\epsilon\}$$

abcd

$$M_2 = \{bcd, \epsilon f\}$$

$$\begin{matrix} L_1 & M_1 & \dots \\ L_2 & \dots & M_2 \end{matrix}$$

$$L_{12} = L_1 \cap L_2$$

$$A_{12} = A_1 \cap A_2$$

Yes

9 100%

No 0

11 0% 48



---

# Operace nad jazyky. Přibližné vyhledávání v textu pomocí konečných automatů

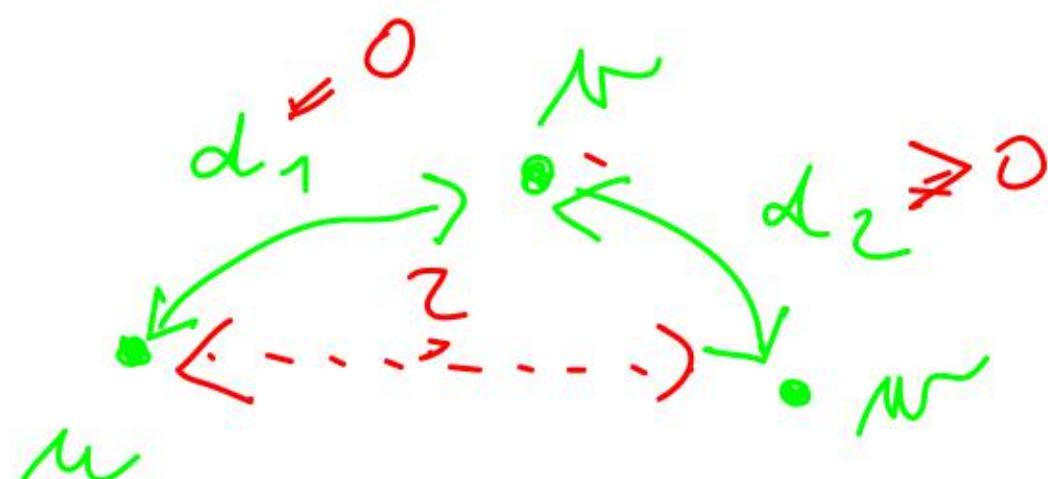
## Př. 8/6: vlastnosti Levenshteinovy vzdálenosti

---

Označme symbolem  $d(x, y)$  Levenshteinovu vzdálenost slov  $x$  a  $y$ . Víme že, pro tři slova  $u, v, w$  platí  $d(u, v) = d_1$ ,  $d(v, w) = d_2$ . Jakých hodnot může nabývat  $d(u, w)$  v závislosti na  $d_1, d_2$ ? Abeceda je pro všechna slova společná.

## Př. 8/6: vlastnosti Levenshteinovy vzdálenosti

Označme symbolem  $d(x, y)$  Levenshteinovu vzdálenost slov  $x$  a  $y$ . Víme že, pro tři slova  $u, v, w$  platí  $d(u, v) = d_1, d(v, w) = d_2$ . Jakých hodnot může nabývat  $d(u, w)$  v závislosti na  $d_1, d_2$ ? Abeceda je pro všechna slova společná.



$$0 \leq d_{12} \leq d_1 + d_2$$

$\Leftarrow$

$$\begin{aligned} u &= ab \\ v &= cd \\ w &= ab \end{aligned} \quad \begin{array}{l} \nearrow \\ \searrow \end{array}$$



## Př. 8/8: Hamming vs. Levenshtein

---

Označme symbolem  $HD(v, w)$  Hammingovu vzdálenost slov  $v$  a  $w$  nad abecedou  $A$ , symbolem  $LD(v, w)$  Levenshteinovu vzdálenost těchž slov. Rozhodněte, který z následujících případů může nastat a pro možné případy uveďte příklad slov  $v$  a  $w$  délky alespoň 5.

- a)  $HD(v, w) < LD(v, w)$ ,
- b)  $HD(v, w) = LD(v, w)$ ,
- c)  $HD(v, w) > LD(v, w)$ .

## Př. 8/8: Hamming vs. Levenshtein

Označme symbolem  $HD(v, w)$  Hammingovu vzdálenost slov  $v$  a  $w$  nad abecedou  $A$ , symbolem  $LD(v, w)$  Levenshteinovu vzdálenost těchž slov. Rozhodněte, který z následujících případů může nastat a pro možné případy uveďte příklad slov  $v$  a  $w$  délky alespoň 5.

- a)  $HD(v, w) < LD(v, w)$ ,
- b)  $HD(v, w) = LD(v, w)$ ,
- c)  $HD(v, w) > LD(v, w)$ .

$$\xrightarrow{\quad} =$$

$$ab \subset A^6 = \emptyset$$
$$ab \dashv A^6 = \emptyset$$

$$abacd \quad bf = 1$$

$$ababc$$
$$ab\cancel{c}ab$$
$$abca$$
$$abca$$

$$HD = 3$$
$$HD = LD = 3$$
$$LD \leq 2$$



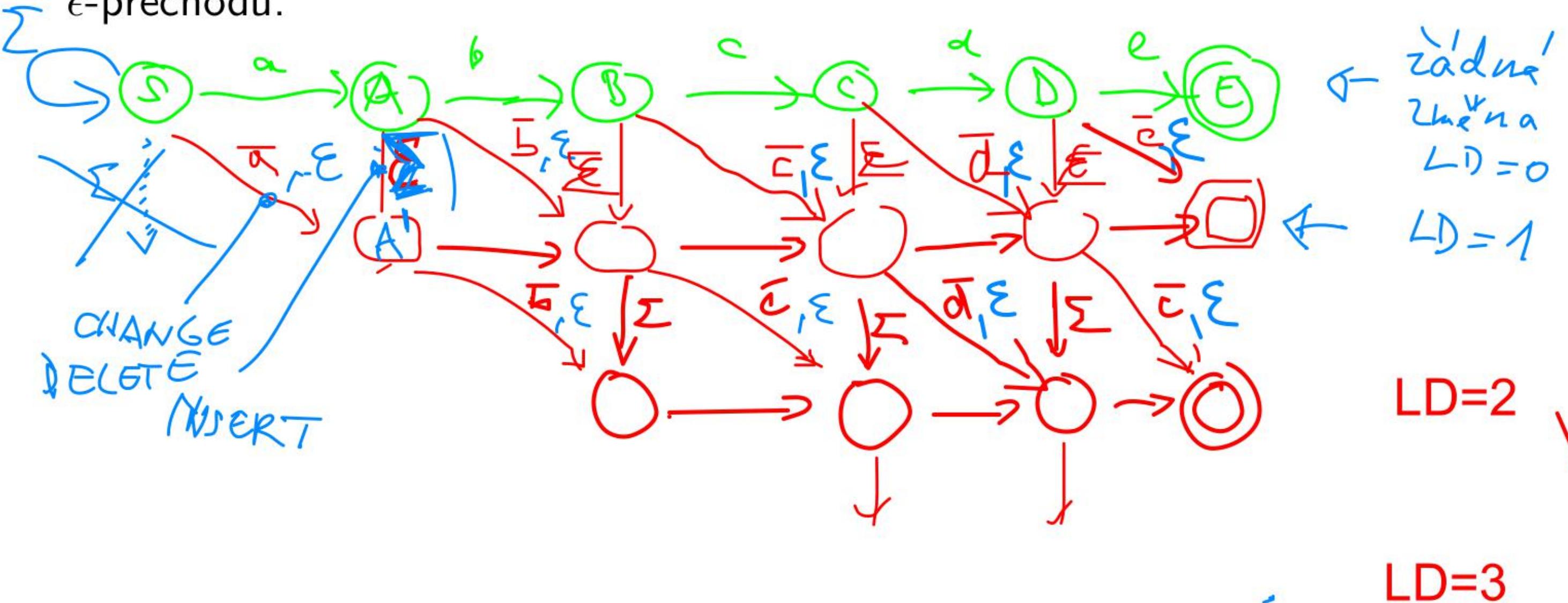
## Př. 8/4: hledání Levenshteinovsky změněného slova

---

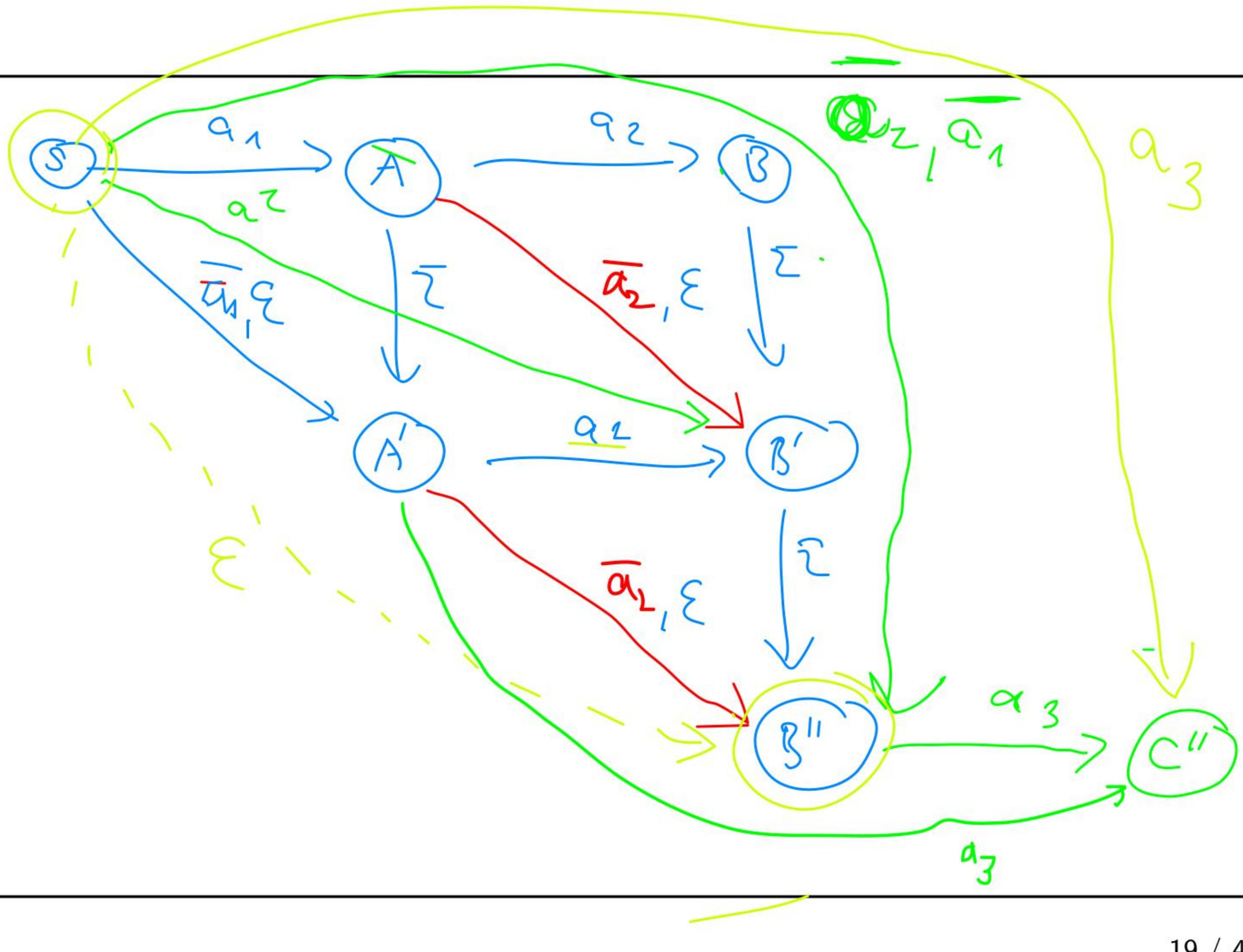
Konečný automat pro hledání v textu, který hledá všechny podřetězce mající od daného vzorku Levenshteinovu vzdálenost menší než dané  $k$ , obsahuje  $\epsilon$ -přechody. Nakreslete příklad tohoto automatu pro délku vzorku 6 a hodnotu  $k = 3$ . Dále nakreslete, jak bude tento automat vypadat po odstranění všech  $\epsilon$ -přechodů.

## Př. 8/4: hledání Levenshteinovský změněného slova

Konečný automat pro hledání v textu, který hledá všechny podřetězce mající od daného vzorku Levenshteinovu vzdálenost menší než dané  $k$ , obsahuje  $\epsilon$ -přechody. Nakreslete příklad tohoto automatu pro délku vzorku ~~6~~<sup>5</sup> a hodnotu  $k = 3$ . Dále nakreslete, jak bude tento automat vypadat po odstranění všech  $\epsilon$ -přechodů.







## Př. 8/5: hledání Levenshteinovsky změněného slova 2

---

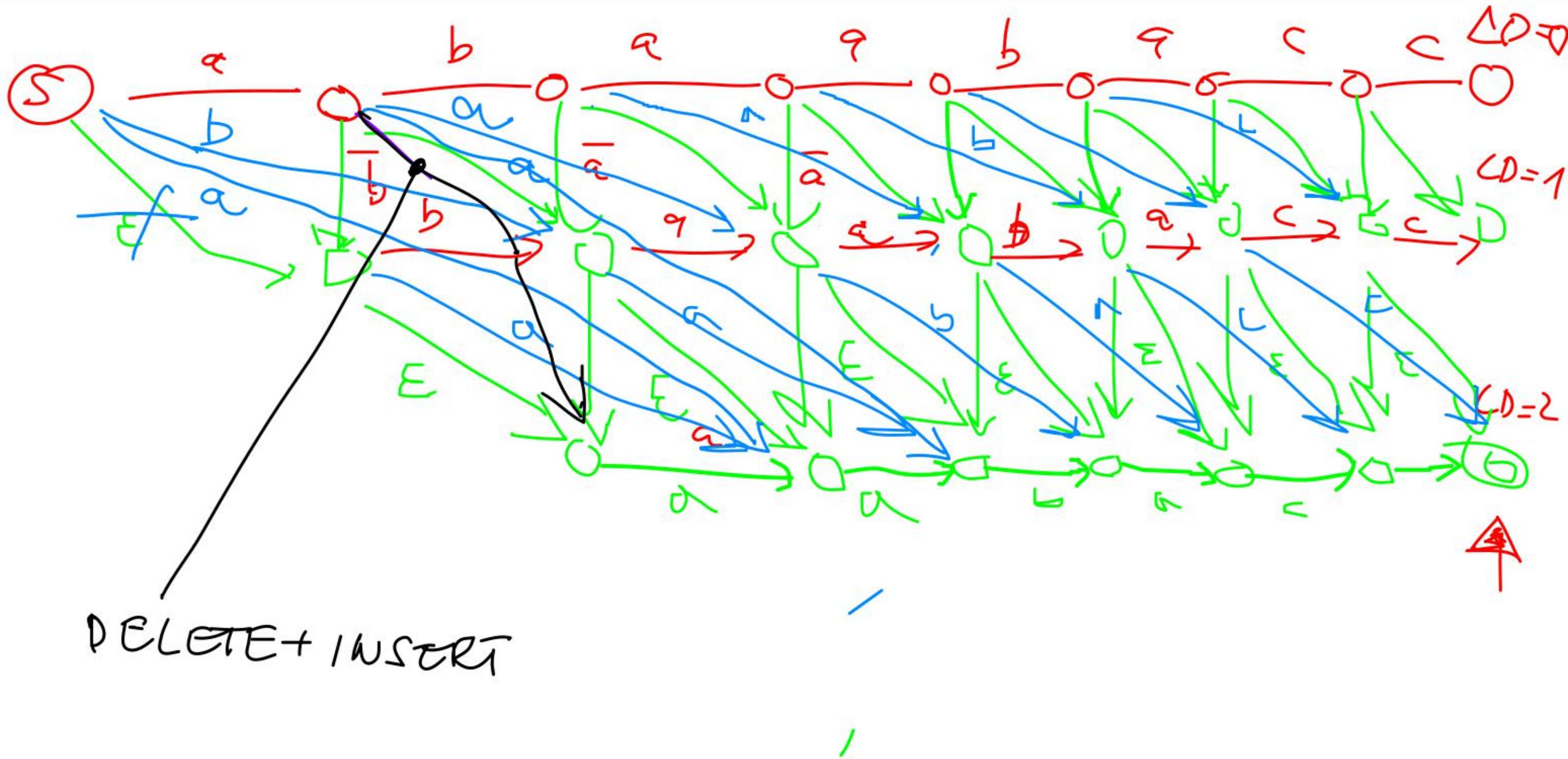
Dvě slova  $V$ ,  $W$  nad abecedou  $A$  mají redukovanou Levenshteinovu vzdálenost rovnu  $k$ , pokud  $k$  je nejmenší možný počet editačních operací, po jejichž provedení ze slova  $V$  vznikne slovo  $W$ . Za editační operace považujeme v tomto případě pouze operace *Insert* a *Delete*. Sestavte nedeterministický automat bez  $\epsilon$ -přechodů, který v textu určí všechny výskytu řetězců, které mají od daného vzorku *abaabacc* redukovanou Levenshteinovu vzdálenost rovnou právě 2.

## Př. 8/5: hledání Levenshteinovsky změněného slova 2

---

Dvě slova  $V$ ,  $W$  nad abecedou  $A$  mají redukovanou Levenshteinovu vzdálenost rovnu  $k$ , pokud  $k$  je nejmenší možný počet editačních operací, po jejichž provedení ze slova  $V$  vznikne slovo  $W$ . Za editační operace považujeme v tomto případě pouze operace *Insert* a *Delete*. Sestavte nedeterministický automat bez  $\epsilon$ -přechodů, který v textu určí všechny výskytu řetězců, které mají od daného vzorku *abaabacc* redukovanou Levenshteinovu vzdálenost rovnou právě 2.





## Př. 8/5: hledání Levenshteinovsky změněného slova 2

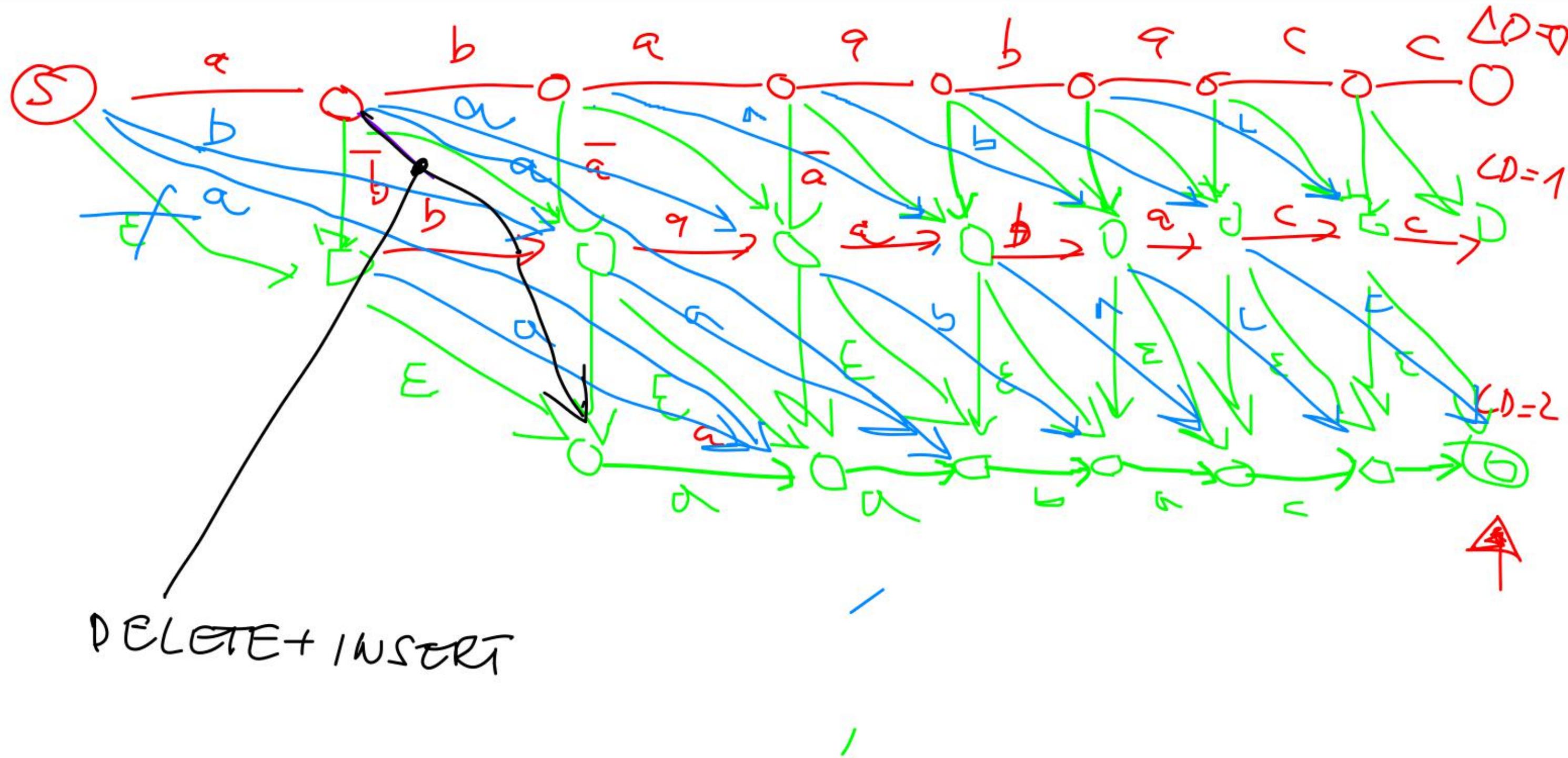
---

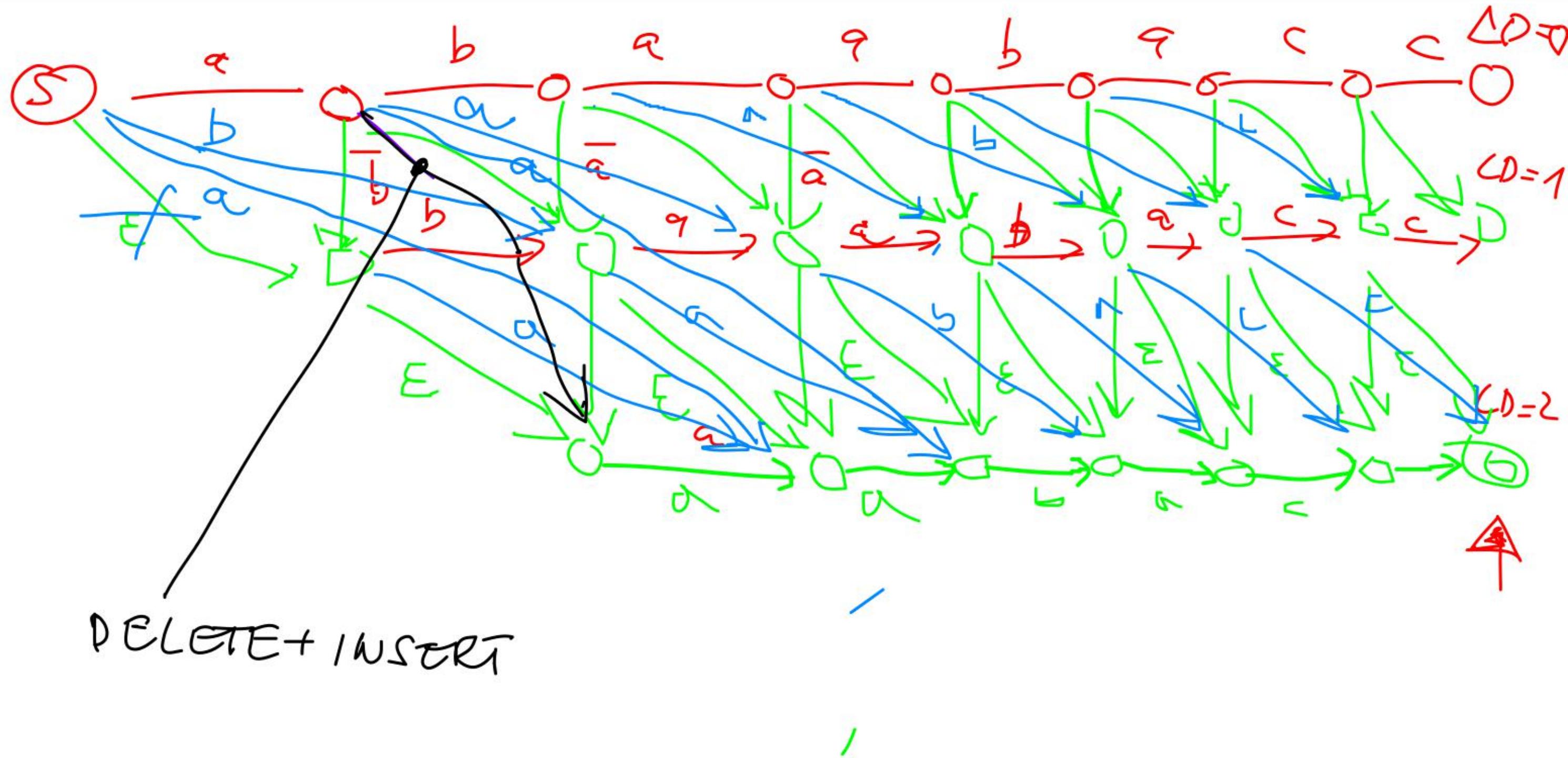
Dvě slova  $V$ ,  $W$  nad abecedou  $A$  mají redukovanou Levenshteinovu vzdálenost rovnu  $k$ , pokud  $k$  je nejmenší možný počet editačních operací, po jejichž provedení ze slova  $V$  vznikne slovo  $W$ . Za editační operace považujeme v tomto případě pouze operace *Insert* a *Delete*. Sestavte nedeterministický automat bez  $\epsilon$ -přechodů, který v textu určí všechny výskytu řetězců, které mají od daného vzorku *abaabacc* redukovanou Levenshteinovu vzdálenost rovnou právě 2.

## Př. 8/5: hledání Levenshteinovsky změněného slova 2

---

Dvě slova  $V$ ,  $W$  nad abecedou  $A$  mají redukovanou Levenshteinovu vzdálenost rovnu  $k$ , pokud  $k$  je nejmenší možný počet editačních operací, po jejichž provedení ze slova  $V$  vznikne slovo  $W$ . Za editační operace považujeme v tomto případě pouze operace *Insert* a *Delete*. Sestavte nedeterministický automat bez  $\epsilon$ -přechodů, který v textu určí všechny výskytu řetězců, které mají od daného vzorku *abaabacc* redukovanou Levenshteinovu vzdálenost rovnou právě 2.





## Př. 8/9: Levenshtein

---

Napište všechna slova, která mají od vzorku  $aba$  nad abecedou  $\{a, b, c\}$  Levenshteinovu vzdálenost rovnu 1.

# Př. 8/9: Levenshtein

Napište všechna slova, která mají od vzorku  $aba$  nad abecedou  $\{a, b, c\}$  Levenshteinovu vzdálenost rovnu 1.

DELE E

四

b5

10

insert 0

aaba  
baba  
caba

2 abax  
abba  
abcc

1 aaba  
aaba  
aaba

3 about  
about  
about

## INSERT

bab a  
bab a  
bab a  
bab a  
bab a

## CHANGE

$$1. \quad \begin{matrix} 6 \\ 5 \end{matrix} \quad \begin{matrix} 7 \\ 5 \end{matrix}$$

б

$$a \leq b$$



## Př. 8/10: SWAP & REWRITE

---

V textu hledáme podřetězec  $Q$ , který se od daného vzorku  $P$  může lišit právě jedním z následujících způsobů:

- $Q$  vznikl z  $P$  právě jednou operací *SWAP* (vzájemné prohození dvou sousedních znaků),
- $Q$  vznikl z  $P$  právě jednou operaci *REWRITE* (náhrada jednoho znaku jiným znakem abecedy)

Sestavte NKA pro hledání  $Q$ , když víme, že  $P = abbaac$ , abeceda je  $\{a, b, c\}$ .



## Př. 8/13: generování podobných textů: Hamming

---

Navrhněte algoritmus pro vypsání všech slov nad abecedou  $A$ , která mají od daného vzorku  $p$  Hammingovu vzdálenost právě  $k > 0$ . Hodnota  $k$  je pevně dána. Jaká bude asymptotická složitost tohoto algoritmu?



## Př. 8/14: generování podobných textů: Levenshtein

---

Navrhněte algoritmus pro vypsání všech slov nad abecedou  $A$ , která mají od daného vzorku  $p$  Levenshteinovu vzdálenost nejvýše  $k > 0$ . Hodnota  $k$  je pevně dána. Jaká bude asymptotická složitost tohoto algoritmu?



---

# Slovníkové automaty. Implementace automatů.

## Př. 9/5b: bitový paralelizmus

---

Sestavte tabulky pro simulaci činnosti vyhledávacího automatu metodou bitového paralelizmu pro daný text  $T$ , vzorek  $P$  a Hammingovu vzdálenost  $k$ .

- a)  $T = abcbcaaccbbbaa$ ,  $P = bbac$ ,  $k = 2$ ,
- b)  $T = accbbaaabcbba$ ,  $P = acbb$ ,  $k = 2$ .

---

# Slovníkové automaty. Implementace automatů.



## Př. 8/14: generování podobných textů: Levenshtein

---

Navrhněte algoritmus pro vypsání všech slov nad abecedou  $A$ , která mají od daného vzorku  $p$  Levenshteinovu vzdálenost nejvýše  $k > 0$ . Hodnota  $k$  je pevně dána. Jaká bude asymptotická složitost tohoto algoritmu?



## Př. 8/13: generování podobných textů: Hamming

---

Navrhněte algoritmus pro vypsání všech slov nad abecedou  $A$ , která mají od daného vzorku  $p$  Hammingovu vzdálenost právě  $k > 0$ . Hodnota  $k$  je pevně dána. Jaká bude asymptotická složitost tohoto algoritmu?



## Př. 8/10: SWAP & REWRITE

---

V textu hledáme podřetězec  $Q$ , který se od daného vzorku  $P$  může lišit právě jedním z následujících způsobů:

- $Q$  vznikl z  $P$  právě jednou operací *SWAP* (vzájemné prohození dvou sousedních znaků),
- $Q$  vznikl z  $P$  právě jednou operaci *REWRITE* (náhrada jednoho znaku jiným znakem abecedy)

Sestavte NKA pro hledání  $Q$ , když víme, že  $P = abbaac$ , abeceda je  $\{a, b, c\}$ .



## Př. 8/13: generování podobných textů: Hamming

---

Navrhněte algoritmus pro vypsání všech slov nad abecedou  $A$ , která mají od daného vzorku  $p$  Hammingovu vzdálenost právě  $k > 0$ . Hodnota  $k$  je pevně dána. Jaká bude asymptotická složitost tohoto algoritmu?



## Př. 8/14: generování podobných textů: Levenshtein

---

Navrhněte algoritmus pro vypsání všech slov nad abecedou  $A$ , která mají od daného vzorku  $p$  Levenshteinovu vzdálenost nejvýše  $k > 0$ . Hodnota  $k$  je pevně dána. Jaká bude asymptotická složitost tohoto algoritmu?



---

# Slovníkové automaty. Implementace automatů.

## Př. 9/5b: bitový paralelizmus

---

Sestavte tabulky pro simulaci činnosti vyhledávacího automatu metodou bitového paralelizmu pro daný text  $T$ , vzorek  $P$  a Hammingovu vzdálenost  $k$ .

- a)  $T = abcbcaaccbbbaa$ ,  $P = bbac$ ,  $k = 2$ ,
- b)  $T = accbbaaabcbba$ ,  $P = acbb$ ,  $k = 2$ .



## Př. 9/5b: bitový paralelizmus

---

Sestavte tabulky pro simulaci činnosti vyhledávacího automatu metodou bitového paralelizmu pro daný text  $T$ , vzorek  $P$  a Hammingovu vzdálenost  $k$ .

- a)  $T = abcbcaaccbbbaa$ ,  $P = bbac$ ,  $k = 2$ ,
- b)  $T = accbbaaabcbba$ ,  $P = acbb$ ,  $k = 2$ .



## Př. 9/1a: Hammingovsky blízká slova - dynamicky

---

Najděte v textu  $T$  všechny výskyty řetězců, které mají od vzorku  $P$  Hammingovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

- a)  $T = ccacbaabccaccbcabccc, P = abcba, k = 2,$
- b)  $T = 000111011000101010111110, P = 110010, k = 3.$



## Př. 9/1b: Hammingovsky blízká slova - dynamicky

---

Najděte v textu  $T$  všechny výskytu řetězců, které mají od vzorku  $P$  Hammingovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

- a)  $T = ccacbaabccaccbcabccc, P = abcba, k = 2,$
- b)  $T = 000111011000101010111110, P = 110010, k = 3.$



## Př. 9/2a: Levenshteinovsky blízká slova - dynamicky

---

Najděte v textu  $T$  všechny výskyty řetězců, které mají od vzorku  $P$  Levenshteinovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

- a)  $T = aacacacbaabbbcbcacc, P = cbbba, k = 3,$
- b)  $T = 010011101000010101011100, P = 11100, k = 1.$



## Př. 9/1b: Hammingovsky blízká slova - dynamicky

---

Najděte v textu  $T$  všechny výskytu řetězců, které mají od vzorku  $P$  Hammingovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

- a)  $T = ccacbaabccaccbcabccc, P = abcba, k = 2,$
- b)  $T = 000111011000101010111110, P = 110010, k = 3.$



## Př. 9/1a: Hammingovsky blízká slova - dynamicky

---

Najděte v textu  $T$  všechny výskyty řetězců, které mají od vzorku  $P$  Hammingovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

- a)  $T = ccacbaabccaccbcabccc, P = abcba, k = 2,$
- b)  $T = 000111011000101010111110, P = 110010, k = 3.$

## Př. 9/1a: Hammingovsky blízká slova - dynamicky

Najděte v textu  $T$  všechny výskytu řetězců, které mají od vzorku  $P$  Hammingovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

a)  $T = ccacbaabccaccbcabccc, P = abcba, k = 2,$

~~b)  $T = 000111011000101010111110, P = 110010, k = 3.$~~

The handwritten notes show a grid for the KMP-like algorithm. The first row contains the string  $ccacbaabccaccbcabccc$  with red annotations: 'c' has a circled '1' above it, 'a' has a circled '1' above it, 'b' has a circled '1' above it, and 'c' has a circled '1' above it. The second row contains the pattern  $abcba$  with red annotations: 'a' has a circled '1' above it, 'b' has a circled '1' above it, and 'b' has a circled '1' above it. The third row contains the numbers '1 0 1 1'. The fourth row contains the number '2'. The fifth row contains the letter 'c'.



## Př. 9/1b: Hammingovsky blízká slova - dynamicky

---

Najděte v textu  $T$  všechny výskytu řetězců, které mají od vzorku  $P$  Hammingovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

- a)  $T = ccacbaabccaccbcabccc, P = abcba, k = 2,$
- b)  $T = 000111011000101010111110, P = 110010, k = 3.$



## Př. 9/2a: Levenshteinovsky blízká slova - dynamicky

---

Najděte v textu  $T$  všechny výskyty řetězců, které mají od vzorku  $P$  Levenshteinovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

- a)  $T = aacacacbaabbbcbcacc, P = cbbba, k = 3,$
- b)  $T = 010011101000010101011100, P = 11100, k = 1.$



## Př. 9/1b: Hammingovsky blízká slova - dynamicky

---

Najděte v textu  $T$  všechny výskytu řetězců, které mají od vzorku  $P$  Hammingovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

- a)  $T = ccacbaabccaccbcabccc, P = abcba, k = 2,$
- b)  $T = 000111011000101010111110, P = 110010, k = 3.$



## Př. 9/1a: Hammingovsky blízká slova - dynamicky

Najděte v textu  $T$  všechny výskytu řetězců, které mají od vzorku  $P$  Hammingovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

a)  $T = ccacbaabccaccbcabccc, P = abcba, k = 2,$

~~b)  $T = 000111011000101010111110, P = 110010, k = 3.$~~

The handwritten notes show a dynamic programming table for finding approximate matches between two strings. The columns represent positions in string T and the rows represent positions in string P. Red numbers indicate the Hamming distance at each position. A red box highlights the value 1 in the second column of the first row, and another red box highlights the value 2 in the third column of the second row. The table is as follows:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	
c	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
c	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
a	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
b	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100
c	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100

## Př. 9/1a: Hammingovsky blízká slova - dynamicky

Najděte v textu  $T$  všechny výskyty řetězců, které mají od vzorku  $P$  Hammingovu vzdálenost rovnou nejvýše  $k$ . Použijte metodu dynamického programování.

a)  $T = ccacbaabccaccbcabccc, P = abcba, k = 2,$

b)  ~~$T = 000111011000101010111110, P = 110010, k = 3.$~~