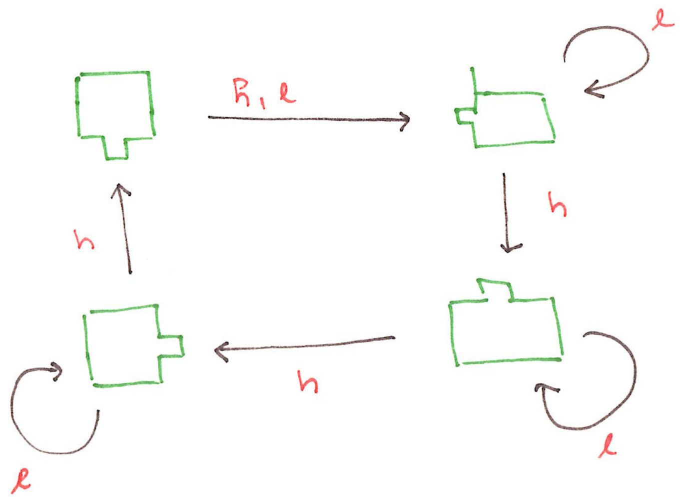
Synchronizing words

Motivace: B.K. Natarjan: An algorithmic approach to the automated design of part orienters.

- máme součástky, které musí být na páse správně orientované
- máme základní „zarážky", které mění orientaci v závislosti na orientaci ~~příslušný~~ součástky.

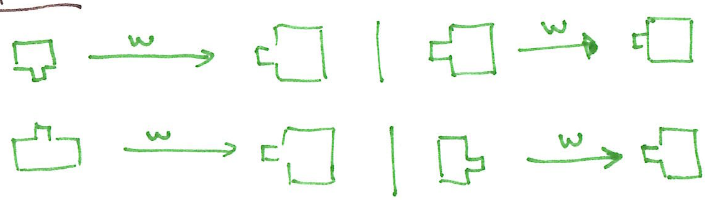
Př:



zarážky h, l

Otázka: ~~Otázka~~ Existuje posloupnost zarážek tak, že ~~p~~ z libovolné orientace převede součástky do jedné orientace?

Odpověď: ANO $w = l\,h\,h\,h\,l\,h\,h\,h\,l$



Formalizace:

stavový stroj (deterministický, úplný)

$\mathcal{A} = (Q, \Sigma, \delta)$

Q … konečná množina stavů

Σ … abeceda akcí

δ … přechodová fce

$$Q \times \Sigma \longrightarrow Q$$

(Tj. DKA, ale nezajímá nás akceptace slov, proto nemáme počáteční a akceptující stavy) budeme také říkat jenom automat)

Rozšiřme δ na řetězce

$$\hat{\delta}(q, \varepsilon) = q$$

$$\hat{\delta}(q, xa) = \hat{\delta}(\delta(q,x), a)$$

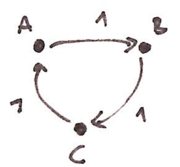
a na množiny stavů $\hat{\hat{\delta}}$

$$\hat{\hat{\delta}}(S, w) = \bigcup_{q \in S} \{\hat{\delta}(q, w)\} = \{ q \in Q \mid \text{ex } p \in S\ \delta(p, w) = q \}$$

Abychom nemuseli psát stříšky na δ, budeme psát jenom δ, o kterou fci se jedná poznáme z kontextu.

def: Slovo $w \in \Sigma^*$ synchronizuje pokud $|\delta(Q,w)| = 1$.

Alg. problém: Pro daný automat nalezněte synchronizující slovo, nebo zjistěte, že takové neexistuje.

př:  nemá synchronizující slovo

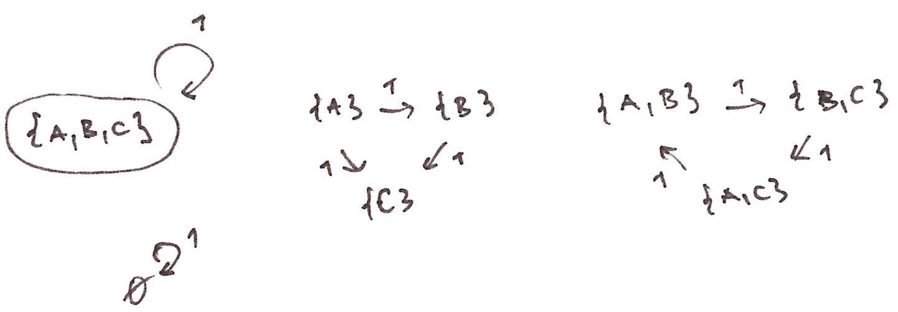
Algoritmický nápad I

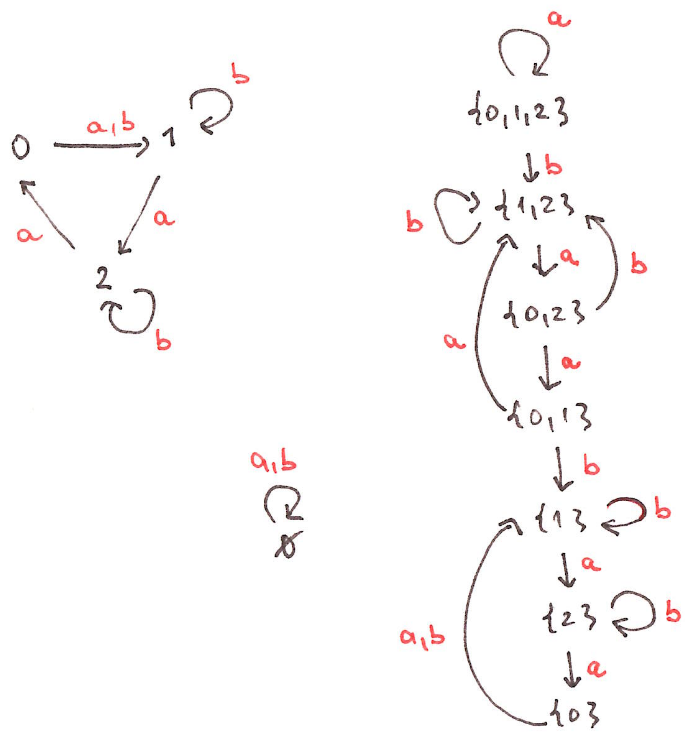
Pro automat $\mathcal{A}$ vytvoříme podmnožinový automat $2^{\mathcal{A}}$:

$$2^{\mathcal{A}} = (2^Q, \Sigma, \hat{\delta})$$ δ rozšířená na množiny stavů

V tomto automatu pak najdeme slovo w vedoucí ze stavu Q do nějakého stavu tvořeného jednoprvkovou množinou (nebo zjistíme, že neexistuje)

Př:

1) 

2) 

Synchronizační slovo: baab

Složitost algoritmu je $O(2^n)$, kde $n = |Q|$ ∎

Vylepšení:

def: $w \in \Sigma^*$ je slévací slovo pro $s,t \in Q$, pokud $\delta(s,w) = \delta(t,w)$

Pozorování: Pokud $w \in \Sigma^*$ synchronizuje, ~~pak~~ je pro $(\forall s,t \in Q)$ slévacím slovem.

3

Co opačný směr?

Tj. pokud existuje pro každou dvojici stavů slévací slovo, existuje také synchronizační slovo?

ANO:

Algoritmus 2 (vstup: automat $\mathcal{A}=(Q,\Sigma,\delta)$)

$x \in \Sigma$

while $|\delta(Q,x)|>1$

a) vybereme dva různé stavy $s,t \in \delta(Q,x)$

b) najdeme ~~s~~ slévací ~~se~~ slovo w pro s,t

(pokud neexistuje, vrátíme false)

c) nastavíme $x \leftarrow xw$

vrátíme x

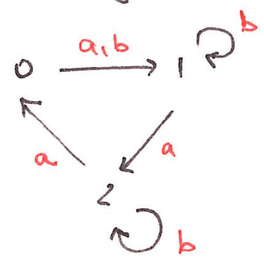
Korektnost:

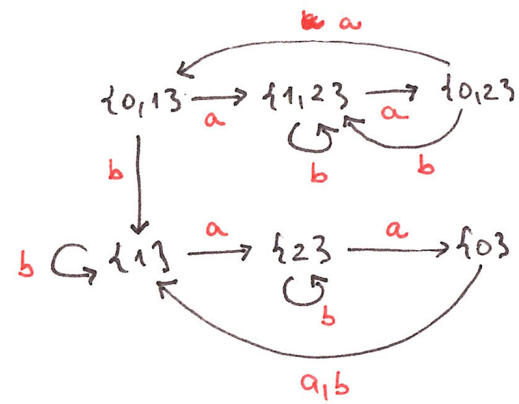
1. Pokud ~~vždy~~ v b) vždy najdeme slévací slovo, máme v každé iteraci $|\delta(Q,x)| = |\delta(Q,xw)|+1$, nakonec dojdeme do situace, kdy $|\delta(Q,x)|=1$ a tedy x synchronizuje.

2. Z předchozího pozorování vidíme, že pokud ~~synch. slovo neexistuje, pak~~ existuje dvojice stavů s,t taková, že pro ně neexistuje slévací slovo, pak synch. slovo neexistuje.

Jak hledat slévací slova?

Podobně jako v exponenciálním algoritmu, ale omezíme se na maximálně dvouprvkové množiny.





slévací sekvence:

$\{0,1\} \xrightarrow{b} \{1\}$

$\{0,2\} \xrightarrow[aab]{\cancel{ab}\ ab} \{1\}$

$\{1,2\} \xrightarrow{\cancel{abb}} \{1\}$

Alg 2 tedy může vybrat

~~1. $\{0,1\}$, potom $\{1,2\}$ → baab~~

~~2. $\{1,2\}$, potom $\{0,1\}$ → baab~~

$\{0,1,2\} \xrightarrow{b} \{1,2\} \xrightarrow{aab} \{1\}$ 1. slévá me $\{0,1\}$

$\{0,1,2\} \xrightarrow{ab} \{1,2\} \xrightarrow{aab} \{1\}$ 1. slévá me $\{0,2\}$

$\{0,1,2\} \xrightarrow{aab} \{1,2\} \xrightarrow{aab} \{1\}$ 1. slévá me $\{1,2\}$

Složitost:

→ délka slévaného slova je $O(n^2)$, protože použitý automat (s max dvouprvkovými ~~hodnotami~~ stavy) má $O(n^2)$ stavů.

→ jedno slévací slovo najdeme v čase $O(n^2)$ např. průchodem do šířky

→ v jedné iteraci Algoritmu 2 počítáme $\delta(Q, xw)$ z $\delta(Q, x)$ a protože $|\delta(Q,x)| \in O(n)$, a $|w| \in O(n^2)$, máme v jedné iteraci $O(n^3)$ operací.

→ Algoritmus 2 má $O(n)$ iterací, složitost je tedy $O(n^4)$

→ Lze jej implementovat tak, aby běžel v čase $O(n^3)$: D. Eppstein: Reset sequences for monotonic automata.

Délka nejkratšího synchronizujícího slova

→ $\binom{n}{2} = n \cdot (n-1)/2$ a tedy délka slévacího slova je omezena $n \cdot (n-1)/2$

→ synchronizující slovo nalezené algoritmem má tedy délku omezenou $n \cdot (n-1)^2/2$ protože slovo skládáme z max $n-1$ slévacích slov.

→ Existují lepší odhady, např. $(n^3-n)/6$

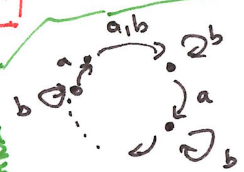
→ Existuje ~~množina~~ rodina automatů, ve které mají nejkratší synchronizující slova délky alespoň $(n-1)^2$

(ve skutečnosti je to právě $(n-1)^2$, to vede k Černého domněnce:

ČERNÉHO DOMNĚNKA

Délka nejkratšího synchronizujícího slova je seshora omezena $(n-1)^2$.

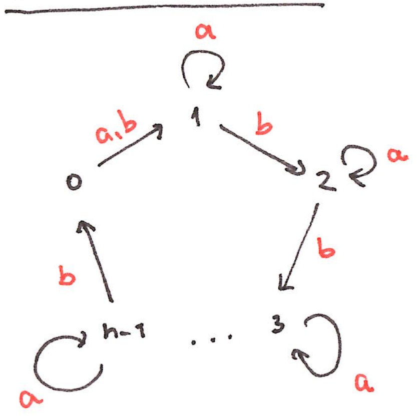
ČERNÉHO RODINA



→ nejkratší synch. slova mají délku $(n-1)^2$

$b(a^{n-1}b)^{n-2}$ délky $1 + (n-1)\cdot(n-2) + (n-2) = (n-1)^2$

Černého automat



označíme C_n

$(ab^{n-1})^{n-2}a$ je synchronizační slovo

délka slova je $n\cdot(n-2)+1 = n^2-2n+1 = (n-1)^2$

Synchronizuje, pro $k \geq 1$

→ $\{0,\dots,k\} \xrightarrow{a} \{1,\dots,k\}$

→ $\{1,\dots,k\} \xrightarrow{b^{n-1}} \{0,\dots,k-1\}$

Pokud předchozí zopakujeme

$\{0,\dots,n-1\} \xrightarrow{ab^{n-1}} \{0,\dots,n-2\}$ } ubude $n-2$ prvků

⋮

$\longrightarrow \{0,1\}$

a $\{0,1\} \xrightarrow{a} \{1\}$

Věta: [Černý]: každé synchronizující slovo v C_n má délku nejméně $(n-1)^2$.

Důkaz. Nechť w je nejkratší synchr. slovo. Potom

→ w nekončí symbolem b, protože tem by šel odebrat, máme tedy $w = w'a$ a $Q \xrightarrow{w'} \{0,1\}$

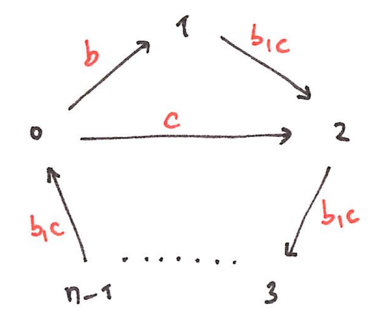
→ uvnitř w' je každé a následováno b (ze slova aa můžeme odebrat druhé a: $A \xrightarrow{a} A$, pokud $0 \notin A$; To ovšem první a zajistí)

zavedeme nový symbol c a řekneme, že

$c = ab$ (1)

Ve slově w' nahradíme ab pomocí c a dostaneme slovo v nad abecedou $\{b,c\}$.

Teď můžeme odstranit s pomocí (1) znak a z C_n. Dostaneme:



označíme W_n

slovo v v automatu W_n:

$Q \xrightarrow{v} \{0,1\}$ a tedy slovo

vc synchronizuje W_n do stavu 2.

každé uvc (pro $u \in \{b,c\}^*$) je také synch. slovo, které synchronizuje do 2.

Pro $\ell \geq |vc|$ tedy pro každý stav i existuje cesta délky ℓ z i do 2. Pokud nastavíme $i=2$, máme kružnici.

každá kružnice ve W_n se skládá z jednoduchých kružnic délek n a $n-1$,

tj. $\ell = k_1\cdot n + k_2\cdot(n-1)$ pro $k_1,k_2 \in \mathbb{N}\cup\{0\}$

4c

Lemma: Pro nesoudělná $p, q \in \mathbb{N}$ je největší číslo r takové, že neexistují $k_1, k_2 \in \mathbb{N} \cup \{0\}$ tak, že $r = k_1 \cdot p + k_2 \cdot q$, rovno $p \cdot q - p - q$. ∎

Protože $n, n-1$ jsou nesoudělná, předchozí Lemma vyplývá

$$|vc| > n \cdot (n-1) - n - (n-1) = n^2 - 3n + 1$$

Dále si všimneme, že $|vc| \neq n^2 - 3n + 2$, protože

$$1 \xrightarrow{b|c} 2 \xrightarrow{n^2-3n+1} 2$$

provedení $|vc| \geq 1$ by vyžadovalo existenci kružnice délky n^2-3n+1, ale ta podle Lemmatu neexistuje.

Dohromady $|vc| \geq n^2 - 3n + 3$.

Protože vc synchronizuje, musí obsahovat aspoň $n-1$ c, symbolů.
tedy v obsahuje aspoň $n-2$ symbolů c.

Připomeneme-li si, že $c = ab$, pak máme

$$|w'| \geq n^2 - 3n + 2 + n - 2 = n^2 - 2n$$

a tedy

$$|w'a| \geq n^2 - 2n + 1 = (n-1)^2$$

∎

Najít nejkratší synchronizující slovo je NP-težké

Rozhodovací verze problému: (SYN-DEC)

Vstup: automat $\mathcal{A}$, přirozené číslo k

Otázka: má $\mathcal{A}$ synchronizující slovo délky $\leq k$?

Zredukujeme 3SAT na SYN-DEC

φ … Fle v 3-CNF, proměnné $x_1 \dots x_n$

klausule $C_1 \dots C_m$

Vytvoříme automat $\mathcal{A}$ s množinou stavů

$$\{s\} \cup \bigcup_{i=1}^{m} \{s_1^i, s_2^i, s_3^i, \dots, s_n^i, s_{n+1}^i\}$$

| horní index ~ klausule
| dolní index ~ proměnná (mimo $n+1$)

Přidáme přechody (pro abecedu $\{T, F\}$)

| T … pravda, F … nepravda

Pro klausuli i $(1 \leq i \leq m)$: $C_i = (\ell_1 \vee \ell_2 \vee \ell_3)$

- $s_{n+1}^i \xrightarrow{T,F} s$
- pokud $x_j \in \{\ell_1, \ell_2, \ell_3\}$ potom

  $s_j^i \xrightarrow{T} s$ [zkratka]

  $s_j^i \xrightarrow{F} s_{j+1}^i$
- pokud $\overline{x_j} \in \{\ell_1, \ell_2, \ell_3\}$

  $s_j^i \xrightarrow{F} s$ [zkratka]

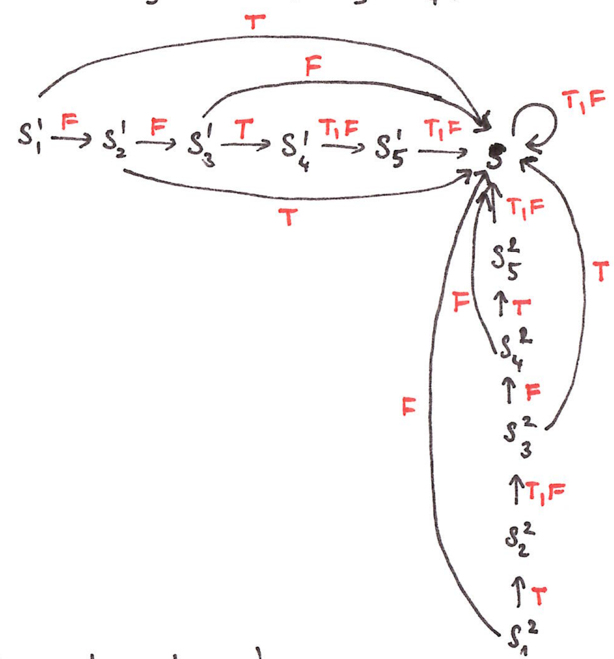
  $s_j^i \xrightarrow{T} s_{j+1}^i$
- jinak

  $s_j^i \xrightarrow{T,F} s_{j+1}^i$

Dále přidáme $s \xrightarrow{T,F} s$. Hodnotu k nastavíme na n.

Příklad:

$$\varphi = \underbrace{(x_1 \vee x_2 \vee \overline{x_3})}_{C_1} \wedge \underbrace{(\overline{x_1} \vee x_3 \vee \overline{x_4})}_{C_2}$$



→ Vidíme, že automat má $m \cdot (n+1) + 1$ stavů a jde sestavit v pol. čase

→ pokud je φ pravdivá v ohodnocení $e: \{x_1 \dots x_n\} \to \{T, F\}$ pak $e(x_1)\,e(x_2)\,e(x_3)\dots$ je synchronizující slovo délky ~~nejvýše~~ n

[Pokud je v C_m literál proměnné x_j splněn, jdeme „zkratkou"]

→ Pokud je v $\mathcal{A}$ synchronizující slovo w délky $\leq n$, pak pro každé $1 \leq i \leq m$ existuje j tak, že z S_j^i jsme šli ztrátkou do S. To ovšem znamená, že literál proměnné x_j obsažený v C_i ohodnotilo w pravdivě a tedy C_i je pravdivá (*)

Protože $|w| \leq n$, můžeme brát $|w| = n$, protože cílový stav může být jenom $\mathcal{S}$. Potom můžeme brát w jako ohodnocení proměnných e:

$$e(x_k) = \begin{cases} F & \text{k-tý znak v } w \text{ je } F \\ T & \text{k-tý znak v } w \text{ je } T \end{cases}$$

→ Tedy, celá φ je pro ohodnocení dané w pravdivá.

□

Pro nalezení nejkratšího synchronizujícího slova tak můžeme zkonstruovat $2^{\mathcal{A}}$ a nalézt nejkratší cestu z Q do jednoprvkové množiny, např. průchodem do šířky.

Poznámka: SYN-DEC $\in$ NP, členství lze certifikovat synchronizačním slovem (o něm už víme, že je polynomiálně dlouhé).

Poznámka 2: z předchozího plyne, že pokud P$\neq$NP, pak nejkratší synchronizující slovo nelze nalézt v polynomiálním čase.

Zobecnění

def: Pro $R \subseteq Q$, slovo $w \in \Sigma^*$ R-synchronizuje, pokud $|\delta(R,w)| = 1$.

Alg. problém: Je dán $\mathcal{A}$ a $R \subseteq Q$. Naleznete R-synchronizující slovo, nebo zjistěte, že neexistuje.

Problém lze v čase $O(2^n)$ řešit pomocí automatu $2^{\mathcal{A}}$.

Jeho rozhodovací verze (zjišťujeme, jestli R-synchronizující slovo existuje) je PSPACE-úplná. Pojmenujeme ji R-SYN-DEC

1) Členství v PSPACE

Následující nedeterministický algoritmus pracuje v polynomické paměti

```
S ← R
while (|S| > 1)
    1. nedeterministicky vybereme a ∈ Σ
    2. S ← δ(S,a)
```

Zřejmě, pokud $\mathcal{A}$ má R-synchronizující slovo, algoritmus doběhne, ~~a na~~ (ve skutečnosti je to „právě když")

Protože NPSPACE = PSPACE, náš problém je v PSPACE.*

(* paměťově omezený algoritmus si může počítat kroky a zjišťovat, jestli se zacyklí).

2) PSPACE-obtížnost.

Najdeme PSPACE-obtížný problém a v pol. čase jej zredukujeme na R-SYN-DEC.

Budeme redukovat z problému zjištění prázdnosti průniku jazyků DKA. (FAI)

Problém FAI:

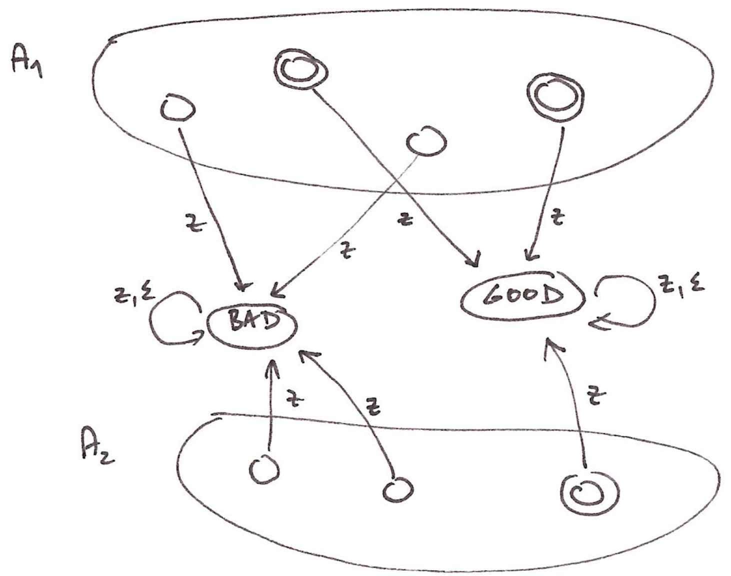
Vstup: DKA $A_1, A_2 \dots A_k$

Otázka: $L(A_1) \cap L(A_2) \cap \dots L(A_k) = \emptyset$?

Zkonstruujeme automat $\mathcal{A}$.

- vytvoříme kopie $A_1, A_2, A_3 \dots A_k$
- přidáme stavy GOOD, BAD a nový symbol z.
- ze všech přijímacích stavů v automatech $A_1, \dots A_k$ uděláme z-přechod do ~~BAD~~ GOOD
- z ostatních stavů přidáme z-přechod do BAD
- pro stavy GOOD a BAD přidáme smyčky pro všechny symboly

Obrázek (pro 2 automaty)



Vidíme, že konstrukci provede v poly. čase vzhledem k velikostem $A_1, A_2 \dots A_k$.

Množinu R vybereme ~~jako~~ jako množinu počátečních stavů $A_1, \dots A_k$ spolu se stavem GOOD.

→ Pokud $w \in \bigcap_{i=1}^{k} L(A_i)$, pak wz je R-synchronizující slovo.

→ Pokud v a $\mathcal{A}$ existuje R-synchronizující slovo w ~~(pokud)~~

potom: 1) $\delta(R, w) = \{GOOD\}$ (GOOD ∈ R a $\delta(GOOD, \cdot) = GOOD$)

2) ~~poslední symbol musí být~~ w obsahuje suffix tvořený pouze symboly z

Označíme-li $w = xz^m$, kde m je maximální možné a x ~~tak~~ neobsahuje z (protože $z \notin \Sigma$), potom ~~x~~ $x \in \bigcap_{i=1}^{k} L(A_i)$, protože $BAD \notin \delta(R, w)$ a tedy $\delta(R, x)$ neobsahuje neakceptující stav. Průnik $\bigcap_{i=1}^{k} L(A_i)$ je tak neprázdný.

∎

Důsledek: Nejkratší R-synchronizující slova mohou být superpolynomiálně dlouhá za předpokladu PSPACE ≠ NP.

(polynomiálně dlouhé slovo by znamenalo, že jím lze certifikovat členství a R-SYN-DEC ∈ NP, v důsledku čehož NP = PSPACE)

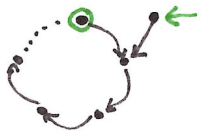
Předpoklad PSPACE ≠ NP není potřeba, platí totiž věta:

Věta: Nejkratší slovo $w \in \bigcap_{i=1}^{k} L(A_i)$ ~~může~~ je ~~být~~ exponenciálně dlouhé (v nejhorším případě), vzhledem k velikosti $A_1, A_2, \dots A_k$.

Důkaz:

$p_i \ldots$ i-té prvočíslo, $\Sigma = \{1\}$

$A_i \ldots$ $p_i + 1$ stavů, přijímá $1^{c \cdot p_i}$ pro $c \geq 1$



Pro $1^m \in \bigcap_{i=1}^{k} L(A_i)$ máme, že m je dělitelné $p_1 \cdot \ldots \cdot p_k \geq 2^k$.

Potom automat P_i má $\sim$ $i \log i$ stavů. (Prime theorem)

∎

Literatura:

[1] Sven Sandberg.
Homing and synchronizing sequences
Model-Based Testing of Reactive Systems, LNCS 3472, 2005

[2] B.K. Natarajan.
Algorithmic approach to the automated design of part orienters.
SFCS 1986

[3] David Eppstein.
Reset sequences for monotonic automata.
SIAM Journal on Computing, 19(3), 1990

[4] Černý Ján.
Poznámka k homogénnym experimentom s konečnými automatmi
Matematicko-fyzikálny časopis Slovenskej Akadémie Vied, 14, (1964)

[5] M.V. Volkov
Synchronizing automata and the Černý conjecture.
LNCS 5196, 2008.