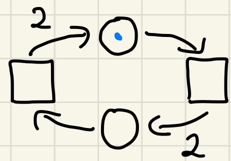
# Ćwiczenia 3 24.10

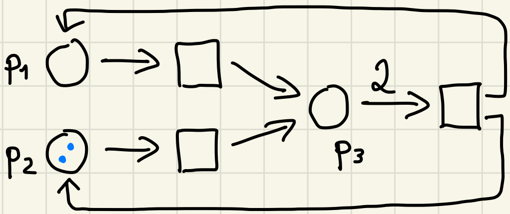
1. Ogólna sieć Petriego N z konfiguracją M jest żywa. Czy wynika stąd, że z dowolnej osiągalnej konfiguracji M' można wrócić do M?

* wystarczy wziąć sieć, która przemnaża żetony




1'. Co jeśli $(N, M)$ jest 2-ograniczone?

* znowu można pokazać kontrprzykład




* dla dowolnej osiągalnej konfiguracji M' powyższej sieci mamy $M'(p_2) \leqslant 1$

1''. A co dla 1-ograniczonych?

zadanie dla chętnych
(odpowiedź na ostatniej stronie)

2. Zaproponuj transformację $n$-wymiarowego VASSu do równoważnego $(n+s)$-wymiarowego VASu. Jak małe może być $s$?

VASS: graf $(Q, T)$, konfiguracja $(p, u)$

**Podejście 1** – 1 dodatkowy wymiar

* każdemu stanowi z $Q$ przypisujemy inną etykietę ze zbioru $K = \{1, 2, \dots, |Q|\}$ (funkcja $\ell : Q \to K$)
* konfigurację $(p, u)$ VASSu reprezentujemy w tworzonym VASie jako $(\ell(p), u) =: c_{p,u}$
* narzuca to tłumaczenie tranzycji w następujący sposób: $(p, v, q) \rightsquigarrow (\ell(q) - \ell(p), v) =: m_{p,v,q}$
* **problem:** $m_{p,v,q}$ można dodać do każdego $c_{r,u}$ dla $r$ spełniającego $\ell(r) \geq \ell(p)$, czyli są przejścia niewystępujące w VASSie ↯

* trzeba poszukać takiego kodowania stanów, które pozwoli poprawnie zdefiniować tranzycje

**Podejście 2** – $|Q| := k$ dodatkowych wymiarów

* nowy wymiar dla każdego stanu z $Q$
* wymiar reprezentujący $q \in Q$ zawiera 1 wtw gdy opisywana konfiguracja VASSu jest w stanie $q$ i wpp zawiera 0
* formalnie dostajemy transformację

| | VASS | | VAS |
|---|---|---|---|
| konfiguracja | $(p, u)$ | $\rightsquigarrow$ | $(0, 0, \ldots, 0, \underset{\uparrow_p}{1}, 0, \ldots, 0, u)$ |
| tranzycja | $(p, v, q)$ | $\rightsquigarrow$ | $(0, \ldots, 0, \underset{\uparrow_q}{1}, 0, \ldots, 0, \underset{\uparrow_p}{-1}, 0, \ldots, 0, v)$ |

* wektor reprezentujący $(p, v, q)$ można dodać tylko do wektora odpowiadającego konfiguracji $(p, -)$, więc kodujemy tylko i wyłącznie biegi w VASSie
* problem: rozmiar $Q$ może znacznie przekraczać wymiar wektora $u$ – chcemy znaleźć bardziej oszczędną transformację

## Podejście 3 – 4 dodatkowe wymiary

pomysł: na dwóch z nowych miejsc kodujemy obecny stan, a pozostałe dwa używamy przy kodowaniu tranzycji (aby umożliwić wpisanie stanu, do którego przechodzimy)

* jak zakodować stan na dwóch miejscach, aby uzyskać łatwe kodowanie tranzycji?

* szkic modelu:

$$(p, u) \rightsquigarrow (\underbrace{a(p), b(p)}_{\text{kodowanie } p \text{ na dwóch miejscach przy użyciu pewnych funkcji } a \text{ i } b}, 0, 0, u) =: c_{p,u}$$

$$(p, v, q) \rightsquigarrow (-a(p), -b(p), \underbrace{a(q), b(q)}_{\text{wpisanie kodowania stanu } q}, v) =: m_{p,v,q}$$

* chcemy, żeby fragment $-a(p), -b(p)$ uniemożliwiał dodanie $m_{p,v,q}$ do $c_{r,u}$ dla każdego $r \neq p$ (innymi słowy dla stanu $r \neq p$ ma zachodzić $a(r) - a(p) < 0$ lub $b(r) - b(p) < 0$)

* rozwiązanie: niech $a(p) = \ell(p)$, $b(p) = k + 1 - \ell(p)$ $\quad k := |Q|$

* dla $r \neq p$ mamy $a(r) < a(p)$, jeśli $\ell(r) < \ell(p)$ oraz $b(r) < b(p)$, gdy $\ell(r) > \ell(p)$

przykład:

| stan $p$ | $\ell(p)$ | $a(p)$ | $b(p)$ |
|---|---|---|---|
| $p_1$ | 1 | 1 | 5 |
| $p_2$ | 2 | $\boxed{2}$ | $\boxed{4}$ |
| $p_3$ | 3 | 3 | 3 |
| $p_4$ | 4 | 4 | 2 |
| $p_5$ | 5 | 5 | 1 |

$m_{p_2,v,p_3}$ zaczyna się od $\boxed{-2, -4}$, które można dodać tylko do $m_{p_2,-}$ bez „spadania poniżej zera"

* przy obecnym zapisie po dodaniu $m_{p,v,q}$ do wektora $c_{p,u}$ kodowanie stanu $q$ znajdzie się na 3. i 4. pozycji, więc formalnie trzeba zaznaczyć, że każdy stan ma też reprezentację

$$c'_{p,u} = (0, 0, a(p), b(p), u),$$

a każda tranzycja musi jeszcze mieć drugie możliwe kodowanie (aby umożliwić aplikację do $c'_{\dots}$)

$$m'_{p,v,q} = (a(q), b(q), -a(p), -b(p), v)$$

* podana transformacja generuje $2|T|$ wektorów przejścia ($m_{p,v,q}$ i $m'_{p,v,q}$) – czy da się lepiej?
* tak, można zachowywać kodowanie stanu na pierwszych dwóch miejscach i dla każdej tranzycji z $T$ generować tylko jeden wektor przejścia
* koszt: $|Q| + |T|$, kodowanie:

$$(p, u) \rightsquigarrow (a(p), b(p), 0, 0, u)$$

$$\Downarrow$$

pomocnicze przejście $\quad (-a(p), -b(p), a(p), b(p), 0)$

$$\Downarrow$$

pomocnicza konfiguracja $\quad (0, 0, a(p), b(p), u)$

$$\Downarrow$$

$$(p, v, q) \rightsquigarrow (a(q), b(q), -a(p), -b(p), v)$$

$$\Downarrow$$

$$(q, u+v) \rightsquigarrow (a(q), b(q), 0, 0, u+v)$$

* czy da się zejść do trzech dodatkowych wymiarów? jeśli tak, w jaki sposób?

## Podejście 4 – 3 dodatkowe wymiary

* podobnie jak w poprzednim podejściu zaczniemy od sposobu, który nie jest optymalny, ale w tym przypadku jest zdecydowanie prostszy do analizy
* każdej konfiguracji $(p,u)$ VASSu będą odpowiadały trzy możliwe konfiguracje tworzonego VASu:

$$c_{p,u} = (a(p), b(p), 0, u),$$
$$c'_{p,u} = (0, a(p), b(p), u),$$
$$c''_{p,u} = (b(p), 0, a(p), u)$$

dla pewnych nowych funkcji $a$ i $b$

* każdej tranzycji będą z kolei odpowiadały trzy wektory przejścia, np. na zaaplikowanie $(p,v,q)$ do $c_{p,u}$ będzie pozwalał wektor

$$m_{p,v,q} = (-a(p), -b(p)+a(q), b(q), v),$$

co spowoduje przejście z reprezentacji $c_{-,-}$ do $c'_{-,-}$

* podobnie z $c'_{-,-}$ będziemy przechodzili do $c''_{-,-}$, a następnie do $c_{-,-}$ (tak definiujemy $m'_{p,v,q}$ i $m''_{p,v,q}$)

* przyjmijmy $a(p) = \ell(p)$, szukamy dobrego kandydata na $b(p)$
* musimy zapewnić, że wektor

$$m_{p,v,q} = (-a(p), -b(p)+a(q), b(q), v)$$

nie będzie dodany do konfiguracji

$$c_{r,u} = (a(r), b(r), 0, u)$$

dla $r \neq p$

* w wyniku dostalibyśmy

$$(a(r)-a(p), b(r)-b(p)+a(q), b(q), u+v)$$

* dla $\ell(r) < \ell(p)$ pierwsza współrzędna będzie $< 0$
* dla $\ell(r) > \ell(p)$ chcemy, żeby na drugim miejscu pojawiła się wartość ujemna, czyli

$$b(r)-b(p) < -a(q),$$

$$b(p)-b(r) > a(q)$$

* wystarczy $b(p)-b(r) > k$, zatem możemy przyjąć $b(p) = (k+1)(k+1-\ell(p))$
* jako formalność pozostaje sprawdzić resztę przejść

* podana transformacja generuje $3|T|$ wektorów przejścia — podobnie jak poprzednio, można poprawić złożoność do $|T| + 2|Q|$

* tym razem generujemy przejścia następująco:

$$(p, u) \rightsquigarrow (a(p), b(p), 0, u)$$

pomocnicze przejścia

będzie omówione później

$$\downarrow$$

pomocnicza konfiguracja

$$(b(p), 0, a(p), u) =: c_{p,u}$$

$$\downarrow$$

$$(p, v, q) \rightsquigarrow (a(q) - b(p), b(q), -a(p), v) =: m_{p,v,q}$$

$$\downarrow$$

$$(q, u+v) \rightsquigarrow (a(q), b(q), 0, u+v)$$

* musimy zagwarantować, że $m_{p,v,q}$ nie będzie dodane do konfiguracji innej niż $c_{p,u}$

* proste rachunki pokazują, że wystarczy przyjąć $a(p) = \ell(p)$ (co będzie blokowało dodawanie dla $\ell(r) < \ell(p)$) oraz $b(p) = (k+1)(k+1-\ell(p))$ (blokujące $\ell(r) > \ell(p)$)

Pełne rozwiązanie:

**Lemma 2.1.** *An n-dim VASS can be simulated by an* $(n+3)$*-dim VAS.*

**Proof.** We give the construction of the VAS. The last three coordinates encode the state while the first $n$ coordinates are as in the VASS. Assume that the VASS has $k$ states $q_1, \ldots, q_k$. Let $a_i = i$ and $b_i = (k+1)(k+1-i)$ for $i = 1$ to $k$. If the VASS is at $v$ in state $q_i$ then the VAS will be at $(v, a_i, b_i, 0)$. For each $i$ the VAS has two dummy transitions $t_i$ and $t_i'$ defined so that $t_i$ goes from $(v, a_i, b_i, 0)$ to $(v, 0, a_{k-i+1}, b_{k-i+1})$ and $t_i'$ goes from $(v, 0, a_{k-i+1}, b_{k-i+1})$ to $(v, b_i, 0, a_i)$. Note that $t_i$ and $t_i'$ modify only the last three components. In addition there is a transition $t_i''$ for each transition $i \rightarrow (j, w)$ of the VASS, defined by

$$t_i'' = (w, a_j - b_i, b_j, -a_i).$$

Clearly any path of the VASS can be mimicked by the VAS. It remains to be shown that the VAS cannot do something unintended. We will only show that $t_i''$ can only be applied if the last three components are $b_i$, 0 and $a_i$ respectively. The other cases are similar. Observe that for each $i$ and $j$, $a_i < a_{i+1}$, $b_i > b_{i+1}$, $a_i < b_j$ and $b_i - b_{i+1} = k + 1 > a_j$. Let $v_i''$ be the vector $(w, a_j - b_i, b_j, -a_i)$ which accomplishes the transition $t_i''$. Note that the $n+1$st and last components are negative. Hence $t_i''$ cannot be applied when the last three coordinates are $(a_i, b_i, 0)$ or $(0, a_{k-i+1}, b_{k-i+1})$ since either the first or third components are 0. Let the last three coordinates be $(b_m, 0, a_m)$. Then if $m < i$, $t_i''$ cannot be applied since $a_m - a_i < 0$. If $m > i$, then $t_i''$ cannot be applied since $b_m + a_j - b_i \leq a_j - (k+1) < 0$. □

Źródło: John Hopcroft, Jean-Jacques Pansiot. On the reachability problem for 5-dimensional vector addition systems

* można pomyśleć nad tym, jakie problemy pojawią się przy użyciu tylko 2 dodatkowych wymiarów

## Wprowadzenie do zadań 3. i 4. (teoria śladów)

**Alfabet z zależnością** $(\Sigma, D)$, gdzie

* $D \subseteq \Sigma^2$ relacja zależności zwrotna, symetryczna
* $I = \Sigma^2 \setminus D$ relacja niezależności

**Równoważność śladowa** $\equiv_D \subseteq \Sigma^* \times \Sigma^*$ to najmniejsza równoważność, taka że $xaby \equiv_D xbay$ dla każdych $x, y \in \Sigma^*$ i $(a,b) \in I$.

Inne spojrzenie na $\equiv_D$:

* niech $\sim$ będzie binarną relacją na $\Sigma^*$, taką że $u \sim v$ wtw, gdy istnieją $x, y \in \Sigma^*$ i $(a,b) \in I$ spełniające $u = xaby$, $v = xbay$
* wtedy $\equiv_D$ jest symetrycznym, zwrotnym i przechodnim domknięciem $\sim$, innymi słowy $u \equiv_D v$ wtw, gdy istnieje sekwencja $(w_0, w_1, \ldots, w_n)$, taka że $w_0 = u$, $w_n = v$ i $\forall\, 0 < i \leq n \quad w_{i-1} \sim w_i$

**Śladem** nazywamy klasę abstrakcji $\equiv_D$, przyjmujemy oznaczenie $[w]_D = \{v : v \equiv_D w\}$. Gdy $D$ wynika z kontekstu, pomijamy je, pisząc $[w]$. **Językiem śladów** nazywamy dowolny ich podzbiór.

Język śladów $T$ nad $(\Sigma, D)$ jest **regularny**, jeśli istnieje istnieje język regularny $L \subseteq \Sigma^*$, który jest zamknięty na równoważność śladową i spełnia warunek:

$$T = [L] = \{[w] : w \in L\}$$

↙ dla języków śladów analogicznie

Dla języka $L \subseteq \Sigma^*$ i słowa $u$ **ilorazem lewostronnym** $L$ względem $u$ nazywamy $u^{-1}L = \{v : uv \in L\}$

Przykłady:

te pary liter można zamieniać w słowie

* $\Sigma = \{a, b, c\}$, $\quad D = a\!<\!\begin{smallmatrix} b \\ c \end{smallmatrix} \quad \Rightarrow \quad I = \{(b,c), (c,b)\}$

  $[abbca]_D = \{abbca, abcba, acbba\}$

* $\Sigma = \{a, b, c\}$, $\quad D = a - b \quad c$

  $L = \{ab, abc, acb, cab\}$ zamknięty na $\equiv_D$

* $\Sigma = \{a, b, c\}$, $L = \{ab, acb, abca, acaa, acbca\}$

  $(ab)^{-1}L = \{\varepsilon, ca\} = (acb)^{-1}L$

* $\Sigma = \{a, b\}$ $L = (ab)^*$

  istnieją tylko dwa ilorazy lewostronne języka $L$:

  $L = \varepsilon^{-1}L = (ab)^{-1}L = (abab)^{-1}L = \ldots$ ,

  $bL = a^{-1}L = (aba)^{-1}L = (ababa)^{-1}L = \ldots$

Inne spojrzenie na ilorazy:

* słowa $x, y \in \Sigma^*$ mają taką samą przyszłość w języku $L$, oznaczenie $x \sim_L y$, jeśli nie istnieje $z \in \Sigma^*$, takie że $xz \in L \wedge yz \notin L$ lub $xz \notin L \wedge yz \in L$
* oczywiście $x \sim_L y \iff x^{-1}L = y^{-1}L$

Twierdzenie (Myhill, Nerode)

Język $L$ jest regularny wtw $\sim_L$ ma skończoną liczbę klas abstrakcji. Co więcej, jest to liczba stanów w minimalnym automacie deterministycznym rozpoznającym język $L$.

3. Pokaż, że język śladów $T$ jest regularny wtedy i tylko wtedy, gdy ma skończenie wiele ilorazów lewostronnych.

Mamy do udowodnienia dwie implikacje

1) $T$ regularny $\Rightarrow$ ma skończenie wiele ilorazów

* oznaczmy alfabet z zależnością przez $(\Sigma, D)$
* niech $L \subseteq \Sigma^*$ będzie językiem regularnym zamkniętym na równoważność śladową, który odpowiada $T$
* $L$ regularny, czyli z twierdzenia M-N ma skończenie wiele ilorazów lewostronnych
* oznaczmy je przez $L_1, L_2, \dots, L_k$
* załóżmy, że $T$ ma $> k$ ilorazów lewostronnych i weźmy dowolne $k+1$ z nich oraz dla każdego wybierzmy dowolny ślad go generujący:

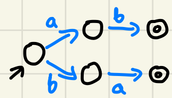
$$[s_1]^{-1} T_1, [s_2]^{-1} T_2, \dots, [s_{k+1}]^{-1} T_{k+1}$$

* z zasady szufladkowej co najmniej dwa ze słów $s\_$ generują ten sam iloraz lewostronny dla języka $L$, niech będą to $s_i$ oraz $s_j$
* z drugiej strony $T_i \neq T_j$, więc BSO możemy przyjąć $[t] \in T_i$ i $[t] \notin T_j$ dla pewnego $t \in \Sigma^*$
* prowadzi to do sprzeczności, bo $s_i t \in L$ i $s_j t \in L$, a co za tym idzie (z zamkniętości $L$ na $\equiv_D$) $\forall s' \in [s_i] \, \forall t' \in [t] \; s't' \in L$ i $\forall s' \in [s_j] \, \forall t' \in [t] \; s't' \in L$, czyli $[t]$ nie może rozróżniać przyszłości $[s_i]$ i $[s_j]$
* tym samym udowodniliśmy, że $T$ ma dokładnie $k$ ilorazów lewostronnych: $[L_1], [L_2], \ldots, [L_k]$

2) $T$ ma skończenie wiele ilorazów $\Rightarrow$ jest regularny

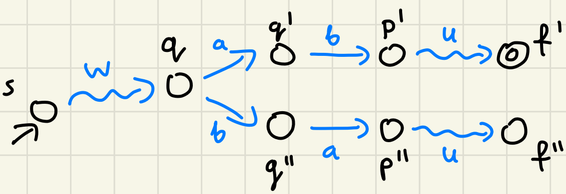
* pokazujemy, że język $L = \bigcup \{ s : s \in T \}$ jest regularny i zamknięty na $\equiv_D$
* kolejne kroki przebiegają analogicznie do tych z pierwszej części dowodu

4. Automat deterministyczny jest „diamentowy", jeśli dla każdego stanu $q$ i każdych dwóch niezależnych liter $a, b$, takich że $q \xrightarrow{a} q'$ i $q \xrightarrow{b} q''$, istnieje stan $p$, spełniający $q' \xrightarrow{b} p$ oraz $q'' \xrightarrow{a} p$. Czy każdy język regularny zamknięty na równoważność śladową jest rozpoznawany przez automat „diamentowy"?

* weźmy dowolny deterministyczny automat $A$ rozpoznający dany język $L$
* potencjalny problem: 

może mieć za dużo stanów
* przekształcamy $A$, żeby otrzymać automat minimalny $A'$ rozpoznający $L$ (np. algorytmem Hopcrofta, $n \log n$)
* zgodnie z twierdzeniem M-N każdy stan w $A'$ reprezentuje inną klasę abstrakcji relacji $\sim_L$ (w szczególności istnieje stan „śmietnik")
* pokażemy, że $A'$ jest automatem diamentowym

* załóżmy, że $q \xrightarrow{a} q'$ i $q \xrightarrow{b} q''$ dla pewnego stanu $q$ oraz niezależnych liter $a, b$
* skoro $A'$ jest deterministyczny, mamy $q' \xrightarrow{b} p'$ i $q'' \xrightarrow{a} p''$ dla pewnych stanów $p', p''$
* jeśli $p' = p''$, własność automatu diamentowego jest spełniona, dlatego załóżmy $p' \neq p''$
* $p'$ i $p''$ muszą mieć inne „przyszłości” (na mocy twierdzenia M-N)
* stąd BSO istnieje słowo $u$, takie że z $p'$ bieg po $u$ jest akceptujący, a bieg z $p''$ po $u$ – nie
* wystarczy wziąć dowolne słowo $w$, po którym istnieje bieg ze stanu początkowego do $q$, oraz zauważyć, że $wabu \in L$ i $wbau \notin L$ – sprzeczność z zamkniętością $L$ na równoważność śladową



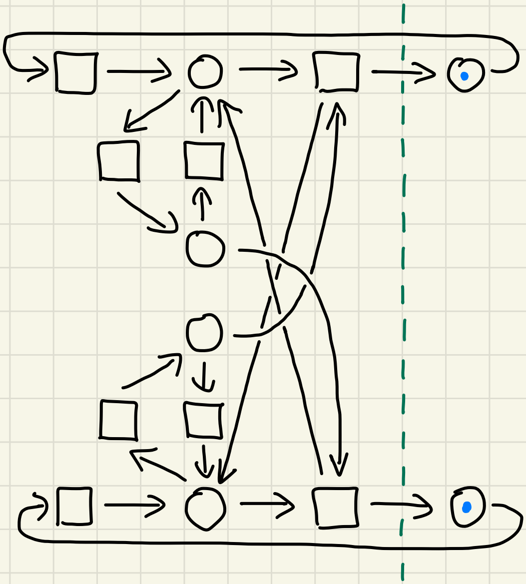
5. Dla danego alfabetu z zależnością $(\Sigma, D)$ i języka $L$ rozważmy pytanie, czy $L$ jest zamknięty na równoważność śladową. Pokaż, że pytanie to jest rozstrzygalne dla $L$ regularnego. Udowodnij, że problem staje się nierozstrzygalny, gdy założymy, że $L$ jest bezkontekstowy.

zadanie do pomyślenia w domu (nieobowiązkowe)

Uwaga: na następnej stronie jest odpowiedź do zadania 1"

Rozwiązanie 1'':

* dla sieci 1-ograniczonych odpowiedź też brzmi nie



* w powyższej sieci jedynie początkowa konfiguracja ma dwa żetony po prawej stronie zielonej linii
* cały czas zachowany jest niezmiennik: jeden żeton w górnej części sieci i jeden w dolnej
* każde przeniesienie żetona na prawą stronę zielonej linii wymaga jednak, aby drugi żeton pozostawał po lewej stronie