Operačná stavová zložitosť operácii zjednotenie, prienik a zreťazenie na unárnych automatoch s polovicou stavov koncových:

(prezentácia vrámci predmetu SDIa)

Vypracovala: Petra Plšková

Vedúci práce: RNDr. Juraj Šebej, PhD.

Obsah

Motivácia

Prečo študujeme práve (unárne) automaty s polovicou stavov koncových?

Hra s dvomi hráčmi

Alica a Bob hrajú proti sebe nejakú hru. Ako zistíme, či existuje v danom momente výherná stratégia pre Alicu?

Hra s dvomi hráčmi

Alica a Bob hrajú proti sebe nejakú hru. Ako zistíme, či existuje v danom momente výherná stratégia pre Alicu?

- Pokiaľ je Alica na ťahu musí platiť:
  Existuje ťah pre Alicu taký, že pre každý Bobov ťah, existuje Alicim ťah taký, že pre každý Bobov ťah...

Hra s dvomi hráčmi

Alica a Bob hrajú proti sebe nejakú hru. Ako zistíme, či existuje v danom momente výherná stratégia pre Alicu?

- Pokiaľ je Alica na ťahu musí platiť:
  Existuje ťah pre Alicu taký, že pre každý Bobov ťah, existuje Alicim ťah taký, že pre každý Bobov ťah...
- Striedanie kvantifikátorov $\exists \forall \exists \forall \exists \forall \exists \forall \exists \ldots$.

Hra s dvomi hráčmi

Alica a Bob hrajú proti sebe nejakú hru. Ako zistíme, či existuje v danom momente výherná stratégia pre Alicu?

- Pokiaľ je Alica na ťahu musí platiť:
  Existuje ťah pre Alicu taký, že pre každý Bobov ťah, existuje Alicim ťah taký, že pre každý Bobov ťah...
- Striedanie kvantifikátorov $\exists \forall \exists \forall \exists \forall \exists \forall \exists \ldots$.
- modelujeme pomocou tzv. **Alternujúceho automatu**

Ekvivalencia AFA a DFA s polovicou stavov koncových

Veta (A.FELLAH; JÜRGENSEN; YU, 1990)

Nech L je jazyk rozpoznateľný n stavovým alternujúcim konečným automatom (AFA), potom jazyk L^R je rozpoznateľný 2^n stavovým deterministickým konečným automatom (DFA) s polovicou stavov koncových.

Ekvivalencia AFA a DFA s polovicou stavov koncových

Veta (A.FELLAH; JÜRGENSEN; YU, 1990)

Nech L je jazyk rozpoznateľný n stavovým alternujúcim konečným automatom (AFA), potom jazyk L^R je rozpoznateľný 2^n stavovým deterministickým konečným automatom (DFA) s polovicou stavov koncových.

Pre jednoduchosť uvažujeme iba jednoprvkovú, t.j. unárnu, abecedu a súčasne naše skúmanie rozšírime na množinu automatov s párnym počtom stavov.

Stavová zložitosť unárnych operácii pre unárny DFA

Náväznosť na diplomovú prácu od (JAVORSKÝ, 2021), ktorý skúmal stavovú zložitosť pre operáciu hviezdička. Vo svojej práci preukázal nasledujúci dolný odhad.

Stavová zložitosť unárnych operácii pre unárny DFA

Náväznosť na diplomovú prácu od (JAVORSKÝ, 2021), ktorý skúmal stavovú zložitosť pre operáciu hviezdička. Vo svojej práci preukázal nasledujúci dolný odhad.

Veta

Pre každé $n = 6k$, $k \in \mathbb{N}$, existuje unárny jazyk L, taký že $sc(L) = n$ rozpoznaný minimálnym n-stavovým DFA s polovicou stavov akceptujúcimi taký, že $sc(L^+) = n + 3$.

Veta

Pre každé $n = 6k + 2$ alebo $n = 6k + 4$, $k \in \mathbb{N}$, existuje unárny jazyk L, taký že $sc(L) = n$ rozpoznaný minimálnym n-stavovým DFA s polovicou stavov akceptujúcimi taký, že $sc(L^+) = n + 5$.

Deterministický konečný automat

Definícia

Deterministickým konečný automatom (DFA) nazveme $A = (Q, \Sigma, \delta, q_0, F)$, kde:

- Q je konečná množina stavov
- Σ je konečná neprázdna množina vstupných symbolov (abeceda)
- $\delta : Q \times \Sigma \to Q$ prechodová funkcia
- $q_0 \in Q$ počiatočný stav
- $F \subseteq Q$ množina koncových (prijímajúcich) stavov (final states)

Alternujúci konečný automat

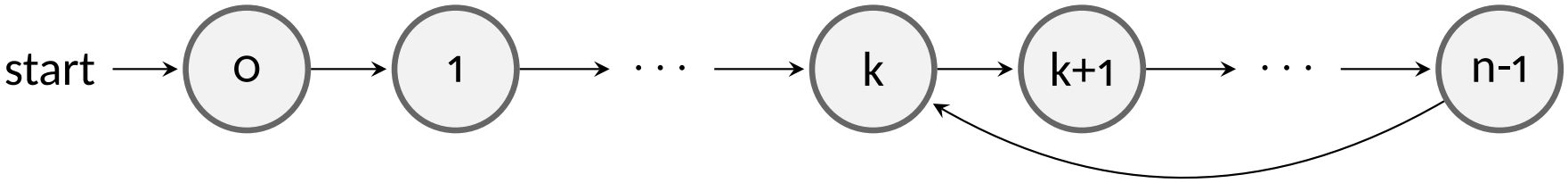
Definícia

Alternujúci konečný automatom (AFA) nazveme $A = (Q_\exists, Q_\forall, \Sigma, \delta, q_0, F)$, označme $Q := Q_\exists \cup Q_\forall$, potom:

- $Q_\exists$ je konečná množina existenciálnych stavov, tiež sa značí $S(\vee)$
- $Q_\forall$ je konečná množina všeobecných stavov, tiež sa značí $S(\wedge)$
- Σ je konečná neprázdna množina vstupných symbolov (abeceda)
- $\delta : Q \times \Sigma \to Q$ prechodová funkcia
- $q_0 \in Q$ počiatočný stav
- $F \subseteq Q$ množina koncových (prijímajúcich) stavov (final states)

Unárny DFA

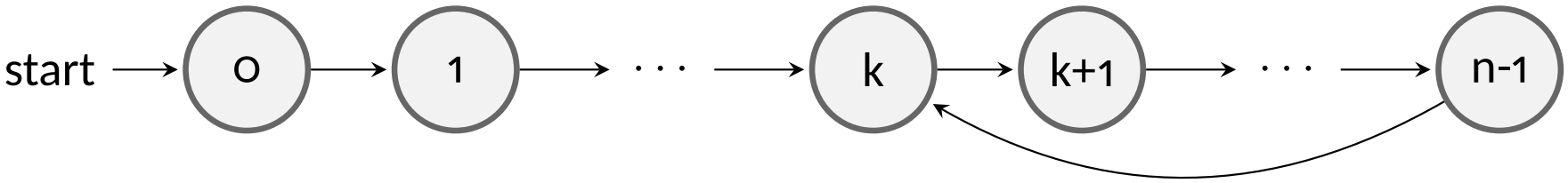
Každý unárny automat bude vyzerať nasledovne (až na množinu koncových stavov).



Možno ho rozdeliť na dva disjunktné grafy - cestu dĺžky k a kružnicu dĺžky $n - k$.

Unárny DFA

Každý unárny automat bude vyzerať nasledovne (až na množinu koncových stavov).



Možno ho rozdeliť na dva disjunktné grafy - cestu dĺžky k a kružnicu dĺžky $n - k$.
Unárny automat budeme reprezentovať ako trojicu **(n,k,F)**:

- $n \in \mathbb{N}$ je počet stavov
- $k \in \mathbb{N}, k \leq n$ označenie stavu, do ktorého vedú dva prechody
- F je postupnosť núl a jedničiek dĺžky n, kde jednička na i-tej pozícii znamená, že i-tý stav je koncový, nula naopak, že nie je

Stavová zložitosť binárnej operácie nad DFA

Definícia (Stavová zložitosť binárnej operácie)

Nech $\square$ je binárna operácia nad deterministickými konečnými automatmi. Stavovou zložitosťou operácie $\square$ je funkcia $sc_{\square}(m, n)$ taká, že:

- pre každý m-stavový DFA A a n-stavový DFA B existuje $sc_{\square}(m, n)$-stavový automat pre $L(A) \circ L(B)$,
- pre každé $m \in \mathbb{N}$, $n \in \mathbb{N}$ existuje there is an mm-stavový DFA A a n-stavový DFA B taký, že každý DFA pre $L(A) \circ L(B)$ má minimálne $sc_{\square}(m, n)$ stavov.

Princíp hľadania stavovej zložitosti

Stavová zložitosť prieniku a zjednotenia

Uvedené tiež v (PIGHIZZINI; SHALLIT, 2002)

Stavová zložitosť prieniku a zjednotenia pre unárne automaty

Nech graf unárneho DFA A pozostáva z cesty dĺžky μ_A a kružnice dĺžky λ_A a nech DFA B pozostáva z cesty dĺžky μ_B a kružnice dĺžky λ_B. Potom prienik a tiež zjednotenie sú pre jazyky $L(A)$ a $L(B)$ sú akceptované DFA dĺžky s kružnicou dĺžky $nsn(\lambda_A, \lambda_B)$ a cestou dĺžky $\max\{\mu_A, \mu_B\}$.

Dôkaz je dôsledkom Čínskej zvyškovej vety.

Stavová zložitosť prieniku a zjednotenia

$$sc_{\cap}(m,n) = \max_{\lambda_A \in \{1,\ldots,n\}, \lambda_B \in \{1,\ldots,m\}} (\max\{n-\lambda_A, m-\lambda_B\} + nsn(\lambda_A, \lambda_B))$$

Stavová zložitosť prieniku a zjednotenia

$$sc_{\cap}(m,n) = \max_{\lambda_A \in \{1,\ldots,n\}, \lambda_B \in \{1,\ldots,m\}} (\max\{n-\lambda_A, m-\lambda_B\} + nsn(\lambda_A, \lambda_B))$$

Tvrdenie ((PIGHIZZINI; SHALLIT, 2002))

Funkcia $f(\lambda_A, \lambda_B) = \max\{n-\lambda_A, m-\lambda_B\} + nsn(\lambda_A, \lambda_B)$ nadobúda pre pevne zvolené $m, n \in \mathbb{N}$ svoje maximum na $\{1, \ldots, n\} \times \{1, \ldots, m\}$ práve vtedy, keď $NSD(\lambda_A, \lambda_B) = 1$

Stavová zložitosť prieniku a zjednotenia

*P*latí

$$sc_{\cap}(m,n) = sc_{\cup}(m,n)$$

pretože

$$L(A) \cup L(B) = (L(A)^C \cap L(B)^C)^C$$

Stavová zložitosť konkatenácie

Uvedené tiež v (PIGHIZZINI; SHALLIT, 2002)

Stavová zložitosť konkatenácie pre unárne automaty

Nech graf unárneho DFA A pozostáva z cesty dĺžky μ_A a kružnice dĺžky λ_A a nech DFA B pozostáva z cesty dĺžky μ_B a kružnice dĺžky λ_B. Potom jazyk $L(A)L(B)$ je prijímaný automatov s kružnicou dĺžky $nsn(\lambda_A, \lambda_B)$ a cestou dĺžky $\mu_A + \mu_B + nsn(\lambda_A, \lambda_B) - 1$

Minimálny DFA

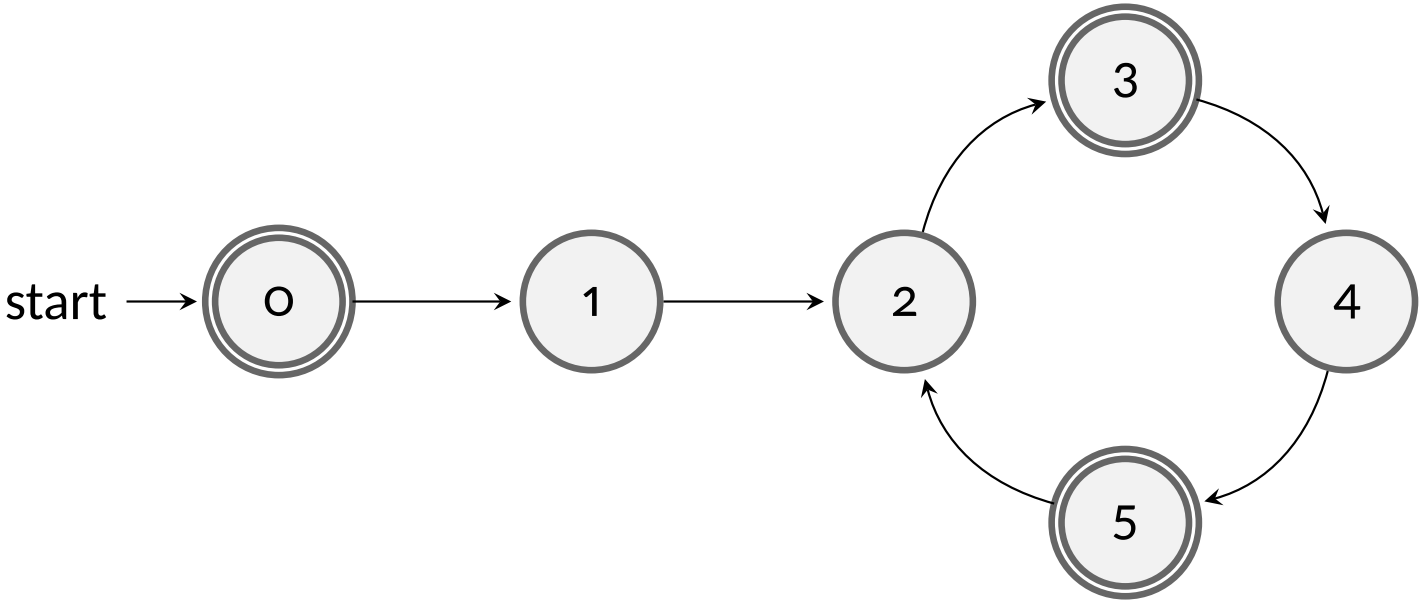
Uvedené tiež v (PIGHIZZINI, 2001)

Nutná a postačujúca podmienka pre minimalitu DFA

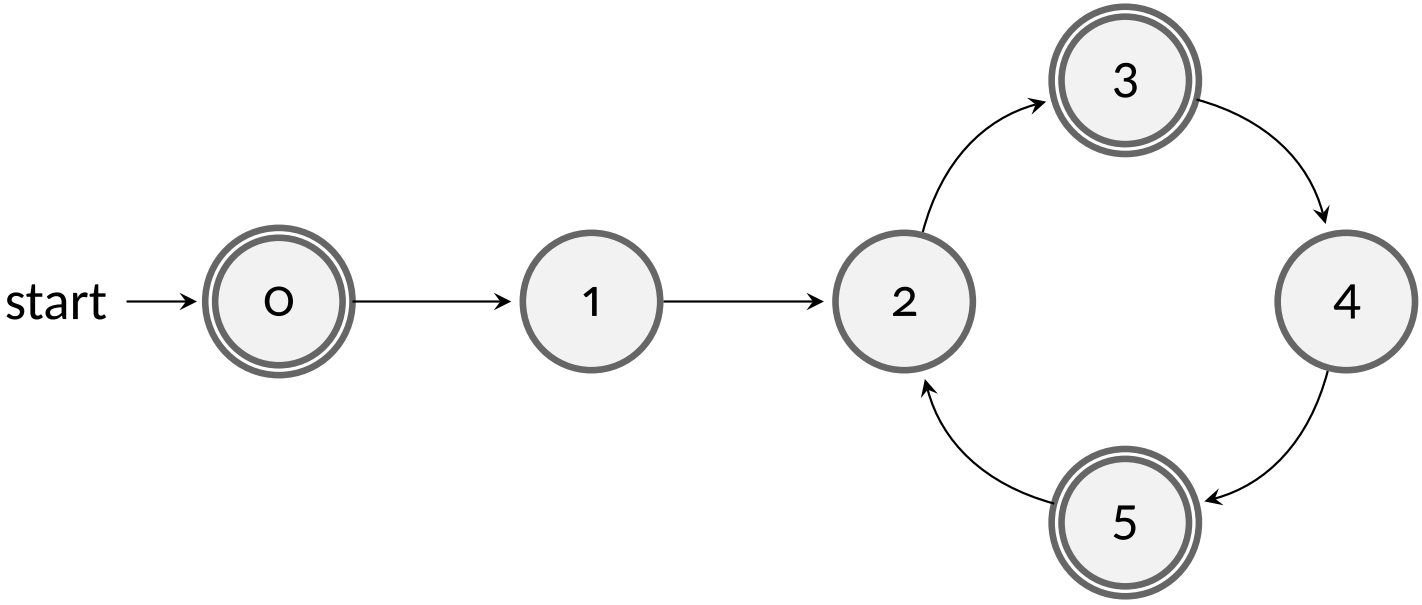
Nech graf unárneho DFA A pozostáva z cesty dĺžky λ a kružnice dĺžky μ. Nech $p_0, \ldots, p_{\lambda-1}$ sú po poradí stavy na kružnici a $q_0, \ldots, q_{\mu-1}$ sú po poradí stavy na ceste. Potom A je minimálny práve vtedy, keď:

- pre ľubovoľný maximálny vlastný deliteľ d čísla μ existuje $h \in \mathbb{N}$, $0 < h < X$, taký, že $p_h \in F$ práve vtedy, keď $p_{(h+d) \bmod \lambda} \notin F$
- $p_{\lambda-1} \in F$ práve vtedy, keď $q_{\mu-1} \notin F$

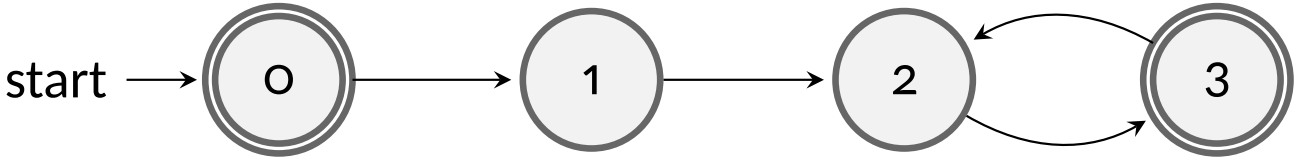
Minimálny DFA



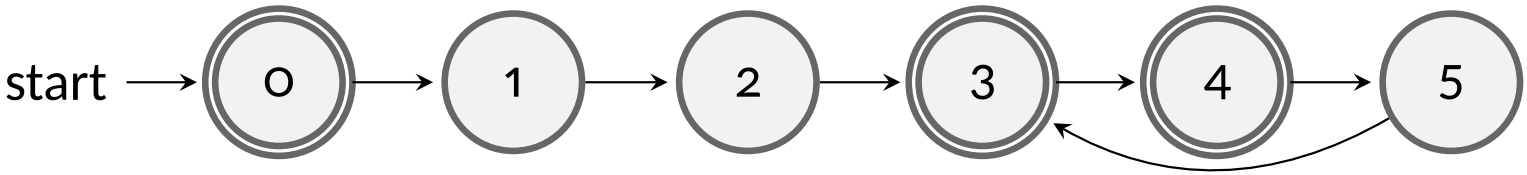
Minimálny DFA



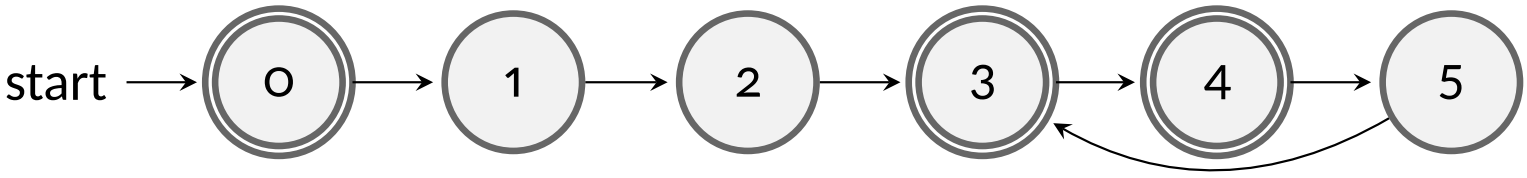
Možno zredukovať na:



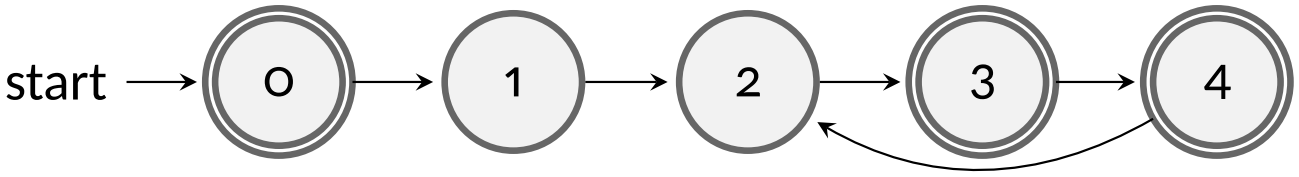
Minimálny DFA



Minimálny DFA



Možno zredukovať na:



Redukcia (minimalizácia) DFA

Z predchádzajúceho tvrdenia vyplýva, že na nájdenie reduktu daného DFA nie je nutné využiť štandartný Hopcroftov minimalizačný algoritmus, ktorého časová zložisť je $O(n \log n)$, ale postačí na to algoritmus pracujúci v čase $O(n)$.

Redukcia a (minimalizácia) DFA

INPUT: DFA A = (n,k,F)

OUTPUT: redukt DFA A' = (n',k',F')

cesta $\leftarrow$ zoznam prvých $k-1$ znakov z F

kruznica $\leftarrow$ zvyšné znaky z F

for $d \in$ *VlastneDelitele*(*length*(*kruznica*)) **do**

    rozdeľ kružnicu na disjunktné dieliky dĺžky d

    if diely sú rovnaké **then**

        kruznica $\leftarrow$ jeden dielik

    end if

end for

while cesta[-1]=kruznica[-1] **do**

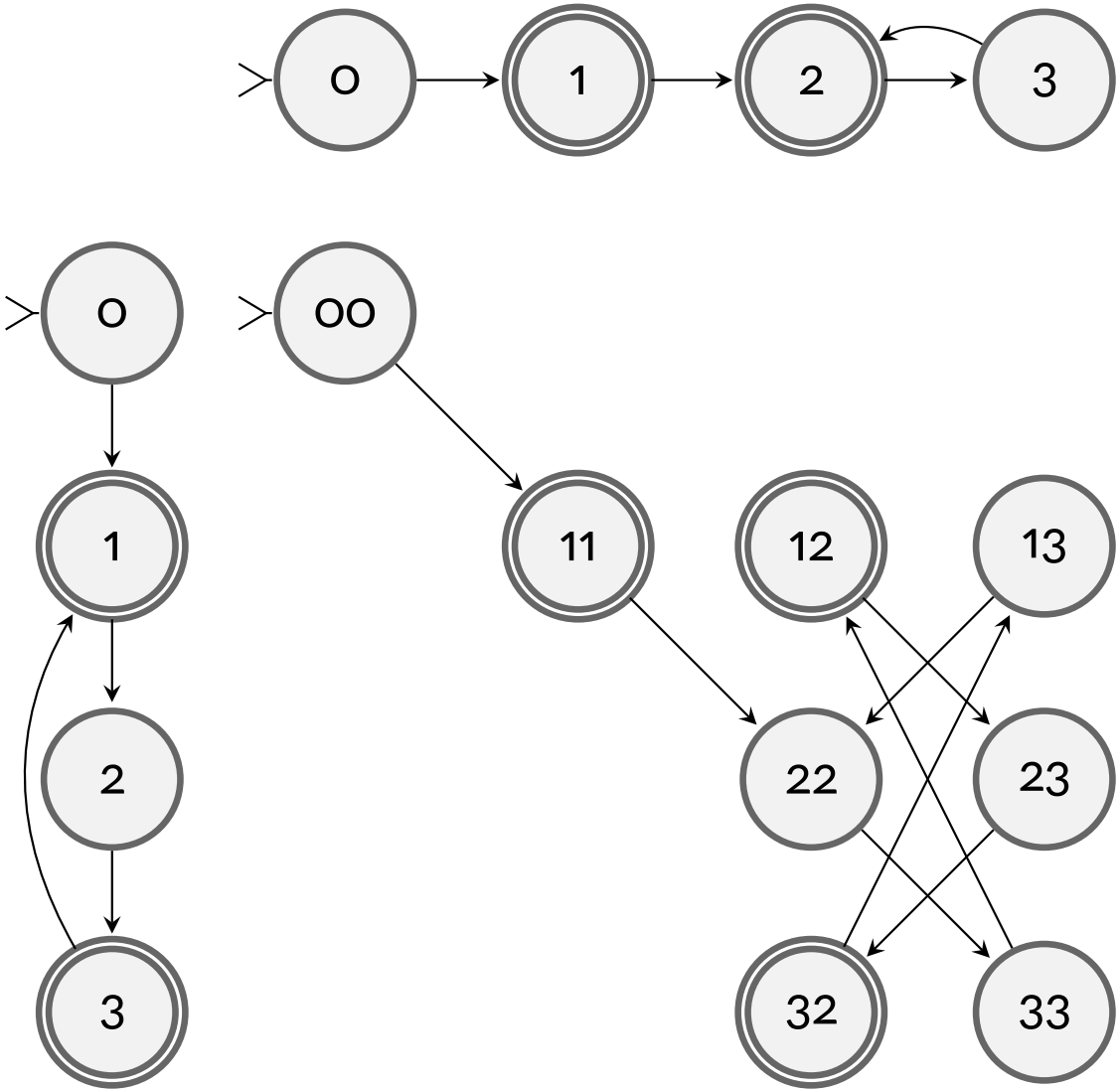
    kruznica $\leftarrow$ cesta[-1] + kruznica[:-1]

    cesta $\leftarrow$ cesta[:-1]

end while

return (length(cesta)+length(kruznica), length(cesta), cesta+kruznica)

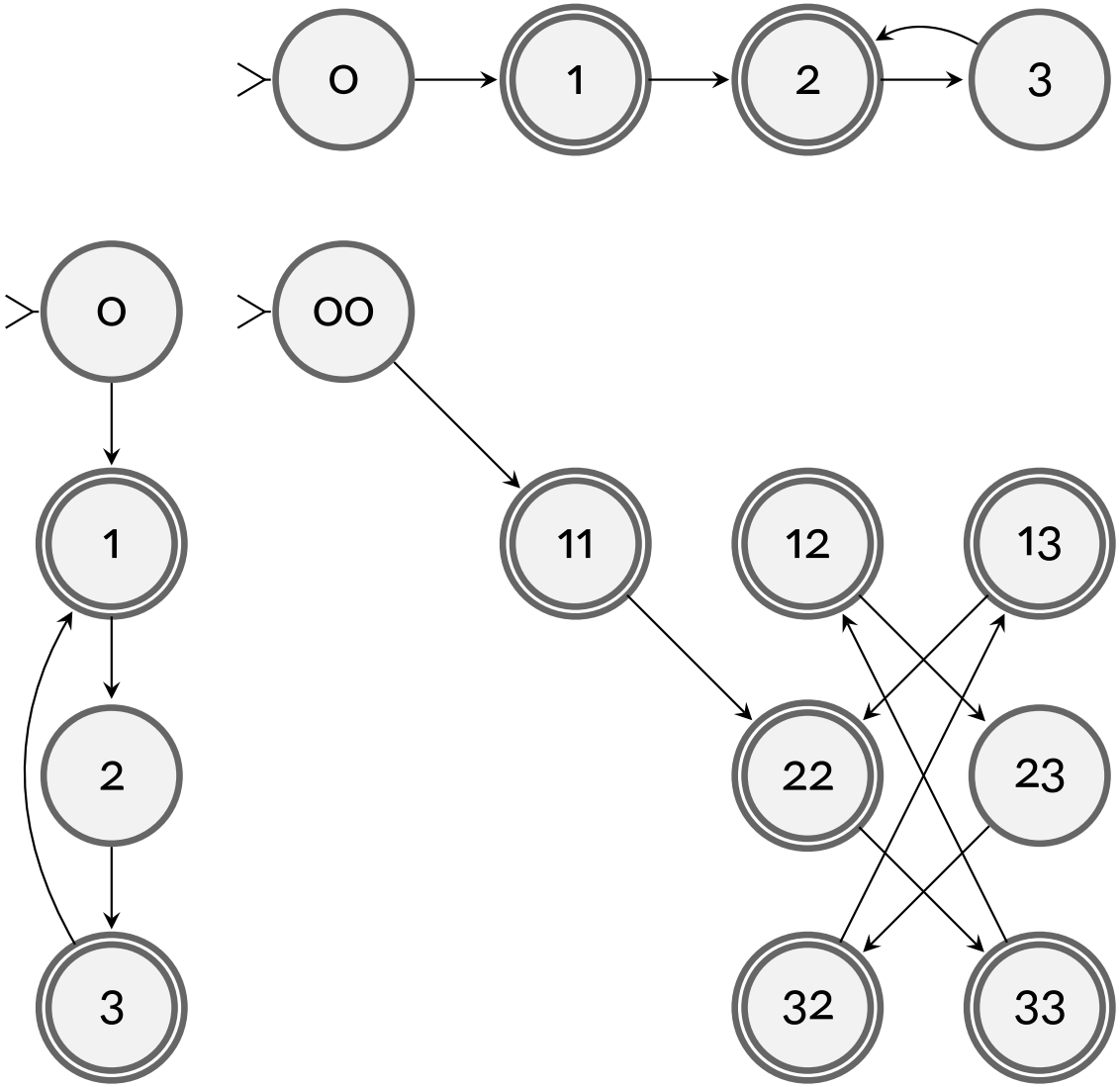
Prienik(DFA)



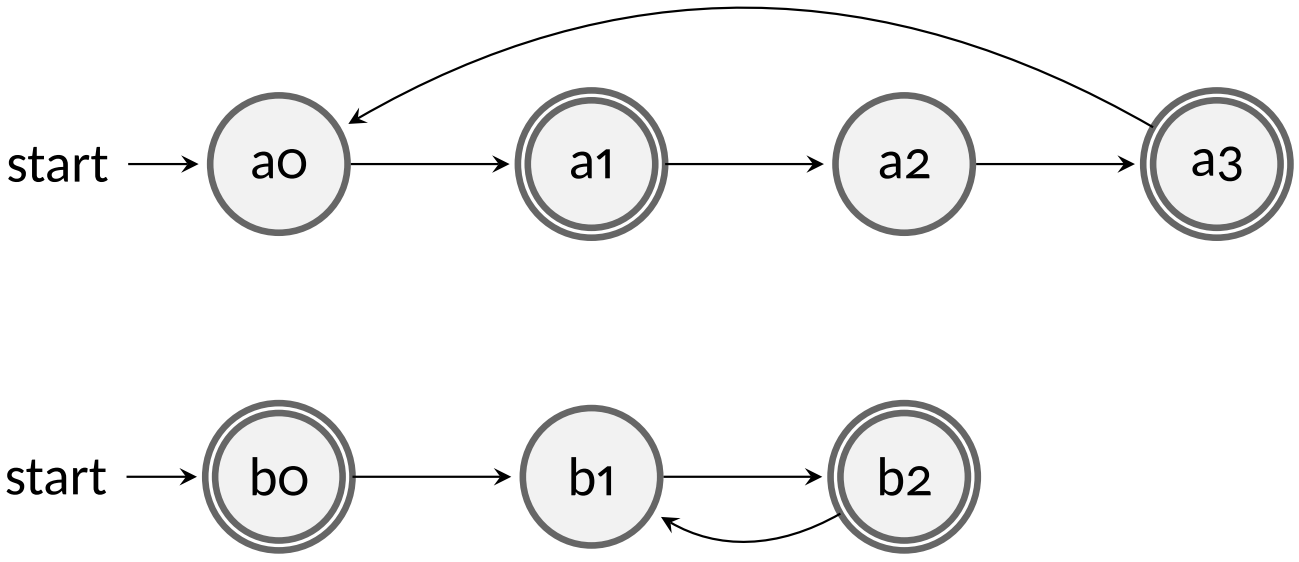
Prienik

```python
def intersection(A, B):
    # INPUT: unary DFA A,B both represented as a tuple (number_of_states, loop_state, bool_list_of_finals)
    # OUTPUT: unary DFA A∩B represented as a tuple (number_of_states, loop_state, bool_list_of_finals)
    state_label = 0
    cartesian_product_matrix = np.zeros((A[0], B[0]))
    intersection_finals = []
    diag_range = min(A[0], B[0]) - 1
    for i in range(diag_range + 1):
        state_label += 1
        cartesian_product_matrix[i, i] = state_label
        intersection_finals.append(A[2][i] and B[2][i])
    i, j = diag_range, diag_range
    while True:
        if i < (A[0] - 1):
            i += 1
        else:
            i = A[1]
        if j < (B[0] - 1):
            j += 1
        else:
            j = B[1]
        if cartesian_product_matrix[i, j] > 0:
            break
        else:
            state_label += 1
            cartesian_product_matrix[i, j] = state_label
            intersection_finals.append(A[2][i] and B[2][j])
    return (int(state_label), int(cartesian_product_matrix[i, j] - 1), intersection_finals)
```

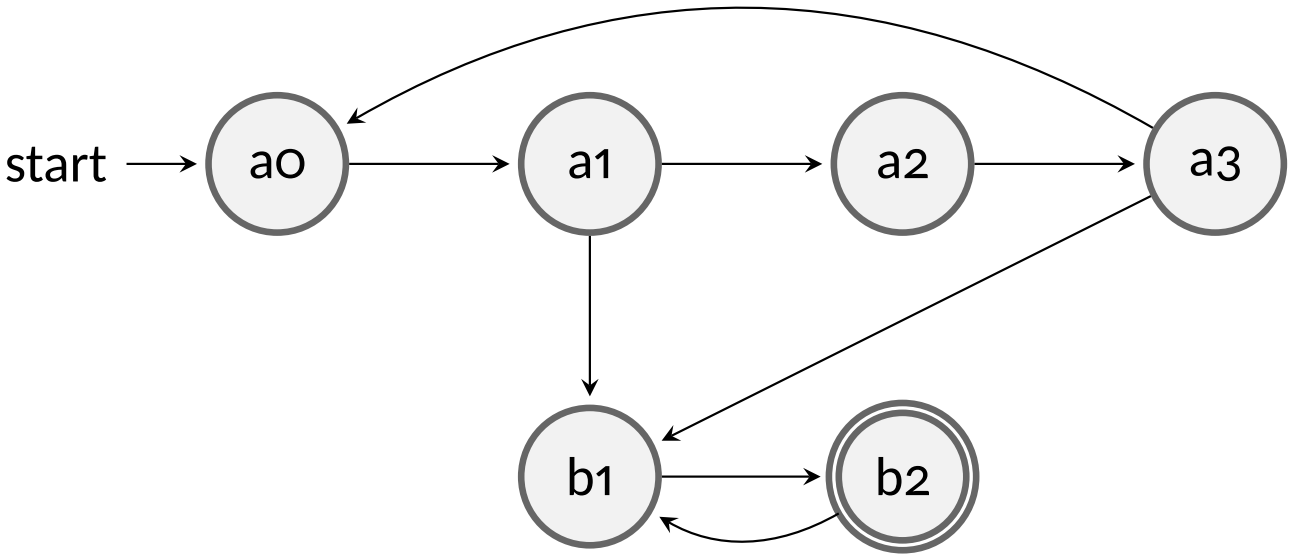
Zjednotenie (DFA)



Konkatenácia



Konkatenácia (NFA)



Konkatenácia

```
def concatenation(A,B):
# INPUT: unary DFA A,B both represented as a tuple (number_of_states, loop_state, bool_string_of_finals)
# OUTPUT: unary DFA AB represented as a tuple (number_of_states, loop_state, bool_string_of_finals)
    lst_of_states = []
    concatenation_finals = []

    if A[2][0]: #the first state in automata A is final
        new_state = set()
        new_state.add(0)
        lst_of_states.append(new_state)
        concatenation_finals.append(True)
    else:
        i = 0
        while not A[2][i]:
            new_state = set()
            new_state.add(i)
            lst_of_states.append(new_state)
            concatenation_finals.append(False)
            i+=1

    while True:
        new_state = set()
        is_final = False
        for x in lst_of_states[len(lst_of_states)-1]:
            if x//A[0] == 0:
                if x<(A[0]-1):
                    new_state.add(x+1)
                    if A[2][x+1]:
                        is_final = True
                else:
                    new_state.add(A[1])
                    if A[2][A[1]]:
                        is_final = True
                if A[2][x]:
                    new_state.add(A[0]+1)
                    if B[2][1]:
                        is_final = True

            else:
                if x<(A[0]+B[0]-1):
                    new_state.add(x+1)
                    if B[2][x+1-A[0]]:
                        is_final = True
                else:
                    new_state.add(A[0]+B[1])
                    if B[2][B[1]]:
                        is_final = True
        if new_state in lst_of_states:
            break
        else:
            lst_of_states.append(new_state)
            concatenation_finals.append(is_final)
    return (len(lst_of_states),lst_of_states.index(new_state),concatenation_finals)
```

Hypotéza

Hypotéza pre stavovú zložitosť prieniku unárnych DFA s polovicou stavov koncových (svedok)

Nech je daný unárny DFA $A = (n, \mathrm{o}, 1^{\frac{n}{2}}\mathrm{o}^{\frac{n}{2}})$ a DFA $B = (m, 1, 1^{\frac{m}{2}}\mathrm{o}^{\frac{m}{2}})$, $m, n \in \mathbb{N}$ sú mocniny dvojky, $n \leq m$. Potom minimálny automat akceptujúci jazyk $L(A) \cap L(B)$ bude mať $mn - n + 1$ stavov.

Idea dôkazu

- graf automatu A je kružnicou dĺžky n
- graf automatu B možno disjunktne rozdeliť na cestu dĺžky 1 a kružnicu dĺžky m-1
- podľa (ref) potom bez ohľadu na charakter koncových stavov je stavová zložitosť prieniku

$$sc_{\cap}(m,n) = nsn(m-1,n) + max\{0,1\} = n(m-1)+1 = mn-n+1$$

keďže $m-1$ a n sú nesúdelné

Idea dôkazu

Ostáva ukázať, že vzniknutý DFA z prieniku A a B sa nezredukuje. Robíme bitový AND pre následujúce regulárne výrazy dĺžky $n(m-1)+1$

$$(1^{\frac{n}{2}}0^{\frac{n}{2}})^{m-1}1$$

a

$$1(1^{\frac{m}{2}-1}0^{\frac{m}{2}})^{n}$$

Keďže cesta vo výslednom grafe bude dĺžky $max\{0,1\} = 1$, tak z výsledku odobereme prvý bit. Ostáva ukázať, že výsledný výraz nebude periodický

Idea dôkazu

Platí: $(1^{\frac{n}{2}} \circ 0^{\frac{n}{2}})^{m-1} 1 = 1(1^{\frac{n}{2}-1} \circ 0^{\frac{n}{2}} 1)^{m-1}$

Teda stačí overiť neperiodickosť bitového ANDu pre výrazy:

$$(1^{\frac{n}{2}-1} \circ 0^{\frac{n}{2}} 1)^{m-1}$$

a

$$(1^{\frac{m}{2}-1} \circ 0^{\frac{m}{2}})^{n}$$

Špeciálny prípad

Ukážeme neperiodickosť bitového ANDu pre špeciálny prípad, kde $n = 2$, teda:

$$(01)^{m-1} \quad \text{a} \quad 1^{\frac{m}{2}-1}0^{\frac{m}{2}}1^{\frac{m}{2}-1}0^{\frac{m}{2}}$$

Platí:

$1^{\frac{m}{2}-1}0^{\frac{m}{2}}1^{\frac{m}{2}-1}0^{\frac{m}{2}}$	=	$1^{\frac{m}{2}-2}$	1	$0^{\frac{m}{2}}$	$1^{\frac{m}{2}-2}$	1	$0^{\frac{m}{2}}$
$(01)^{m-1}$	=	$(01)^{\frac{m}{4}-1}$	0	$(10)^{\frac{m}{4}}$	$(10)^{\frac{m}{4}-1}$	1	$(01)^{\frac{m}{2}}$
VÝSLEDOK:		$(01)^{\frac{m}{4}-1}$	0	$0^{\frac{m}{2}}$	$(10)^{\frac{m}{4}-1}$	1	$0^{\frac{m}{2}}$

Dostaváme výraz $(01)^{\frac{m}{4}-1}0^{\frac{m}{2}+1}(10)^{\frac{m}{4}-1}10^{\frac{m}{2}}$, ktorý je zrejme neperiodický.

Výsledky pre prienik (max DFA z hypotézy)

m	n	$sc_{\cap}(m, n)$	$F_{\cap}$
2	2	3	001
2	4	7	1000100
2	6	11	10100010000
2	8	15	101000001010000
2	10	19	1010100000101000000
4	4	13	1100100000000
4	6	21	110000000000100011000
4	8	29	11000000110000001100000010000
4	10	37	1100100000001100000011000000110000000
6	6	31	1110001100001000000000000010000
6	8	43	1110000010000000000000001000001100001110000
6	10	46	1110000000011000000001100000000100000100000000
8	8	57	1111000011100000110000001000000000000000000010000001100000
8	10	73	1111000000110000000100000000000000000000010000000110000001110000011110000
10	10	91	1111100000111100000011100000000110000000010000000000000000000000010000000011000000011100000

Výsledky pre konkatenáciu

		A			B			AB			reduct(AB)			how much it reduces
		n	k	F	n	k	F	n	k	F	n	k	F	
m	n													
4	2	4	1	1010	2	0	10	9	6	101010111	7	6	1010101	2
	4	4	0	1001	4	1	1001	13	9	1001001101111	10	9	1001001101	3
	6	4	0	1001	6	3	110001	15	11	110011001101111	12	11	110011001101	3
	8	4	1	1010	8	4	11100001	18	15	111110110111110111	16	15	1111101101111101	2
	10	4	1	1010	10	6	1101100001	20	17	11011110110111110111	18	17	110111101101111101	2
6	2	6	1	101010	2	0	10	13	8	1010101011111	9	8	101010101	4
	2	6	3	101010	2	0	10	11	8	10101010111	9	8	101010101	2
	4	6	1	110010	4	1	1001	17	12	10011011011011111	13	12	1001101101101	4
	4	6	2	110100	4	1	1001	16	12	1001101101101111	13	12	1001101101101	3
	6	6	1	110010	6	3	110001	19	14	1110011111111011111	15	14	111001111111101	4
	6	6	2	110100	6	3	110001	18	14	1110111011011011111	15	14	111011101101101	3
	8	6	1	100011	8	4	11100001	24	19	111001110011100111011111	20	19	11100111001110011101	4
	10	6	0	110001	10	5	1111000001	30	24	111110111110111110111110111111	25	24	111110111110111101111101	5
	10	6	2	110100	10	5	1111000001	28	24	111111101110111111111101111	25	24	111111101110111111111101	3
8	2	8	1	10101010	2	0	10	17	10	10101010111111111	11	10	10101010101	6
	2	8	3	10101010	2	0	10	15	10	1010101011111	11	10	10101010101	4
	2	8	5	10101010	2	0	10	13	10	1010101010111	11	10	10101010101	2
	4	8	0	01001011	4	1	1001	23	15	000010010110110111111111	16	15	0000100101101101	7
	4	8	1	10010011	4	1	1001	22	15	100100100110110111111	16	15	1001001001101101	6
	4	8	3	11001001	4	1	1001	20	15	10011011011011011111	16	15	1001101101101101	4
	4	8	4	11010010	4	1	1001	19	15	1001101101101101111	16	15	1001101101101101	3
	6	8	2	11000101	6	2	110010	21	9	111011101101111111111	20	8	11101110010111111111	1
	8	8	3	11101000	8	4	11100001	28	23	11111111111111011101110111111	24	23	111111111111110111011101	4
	10	8	2	11000101	10	2	1110001010	37	13	1111001111111111111111111111011111111	30	6	1111000111111111111111111111101	7
	10	8	2	11000110	10	5	1111000001	35	29	11111011111101111001111001110111111	30	29	11111011111101111001111001110111101	5

Výsledné stavové zložitosti

m \ n	2	4	6	8	10
2	3	5	11	15	19
4	5	13	21	29	37
6	11	21	31	43	46
8	15	29	43	57	73
10	19	37	46	73	91

Table: Stavová zložitosť prieniku

m \ n	2	4	6	8	10
2	4	7	9	11	13
4	7	10	12	16	18
6	9	13	15	20	25
8	11	16	20	24	30
10	13	19	21	30	35

Table: Stavová zložitosť konkatenácie

Ďakujem za pozornosť!

Zoznam literatúry

 A.FELLAH; JÜRGENSEN, H.; YU, S. Constructions for alternating finite automata. **International Journal of Computer Mathematics**, Taylor Francis, v. 35, n. 1-4, p. 117–132, 1990. 8, 9

 JAVORSKÝ Šimon. structural properties and state complexity of unary operations on deterministic state finite state automata. **Master thesis, Pavol Jozef Šafárik University**, 2021. 10, 11

 PIGHIZZINI, G. Unary language concatenation and its state complexity. In: YU, S.; PĂUN, A. (Ed.). **Implementation and Application of Automata**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001. p. 252–262. ISBN 978-3-540-44674-3. 23

 PIGHIZZINI, G.; SHALLIT, J. Unary language operations, state complexity and jacobsthal's function. **International Journal of Foundations of Computer Science**, v. 13, n. 01, p. 145–159, 2002. 18, 19, 20, 22