Przetwarzanie tekstu 5-6

Automaty i przetworniki skończone, zastosowania

Automaty skończone

Automaty skończone

definicja

Automat skończony definiuje się jako piątkę

$$< Q, \Sigma, \delta, q_0, F >$$

gdzie

Q – skończony zbiór stanów,

Σ – alfabet automatu,

$\delta : Q \times \Sigma \cup \{\varepsilon\} \rightarrow 2^Q$ – funkcja przejść,

$q_0 \in Q$ – stan początkowy,

$F \subseteq Q$ – zbiór stanów końcowych.

Automaty skończone

przykład automatu

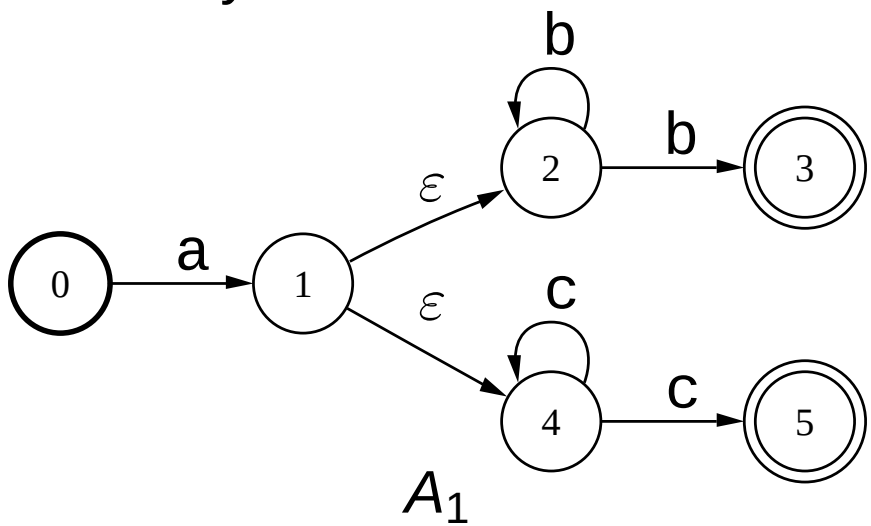
$$A_1 = < Q, \Sigma, \delta, 0, F >$$

$Q = \{0, 1, 2, 3, 4, 5\}$
$\Sigma = \{a, b, c\}$
$F = \{3, 5\}$

funkcja przejść:

δ	a	b	c	ε
0	$\{1\}$	$\emptyset$	$\emptyset$	$\emptyset$
1	$\emptyset$	$\emptyset$	$\emptyset$	$\{2, 4\}$
2	$\emptyset$	$\{2, 3\}$	$\emptyset$	$\emptyset$
3	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
4	$\emptyset$	$\emptyset$	$\{4, 5\}$	$\emptyset$
5	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

rysunek automatu



Automaty skończone

uwagi notacyjne

- stany będziemy oznaczać liczbami całkowitymi nieujemnymi (w literaturze często spotyka się oznaczenia $q_0, q_1, q_2, \ldots$)
- stan początkowy na rysunku będzie wyróżniony pogrubioną linią (będzie to zawsze stan 0).
- stany końcowe będą na rysunku wyróżnione podwójną linią

Automaty skończone

język automatu

- mówimy, że automat skończony akceptuje słowo $a_1 a_2 \ldots a_n$, jeśli istnieje w nim ścieżka etykietowana kolejnymi symbolami tego słowa, prowadząca od stanu początkowego do stanu końcowego (na ścieżce tej może być też dowolnie wiele przejść przez ε)
- zbiór wszystkich słów akceptowanych przez automat A nazywamy językiem automatu A (językiem akceptowanym/definiowanym przez automat A) i oznaczamy $L(A)$
- automat A_1 akceptuje język opisywany wyrażeniem regularnym `a(b+|c+)`

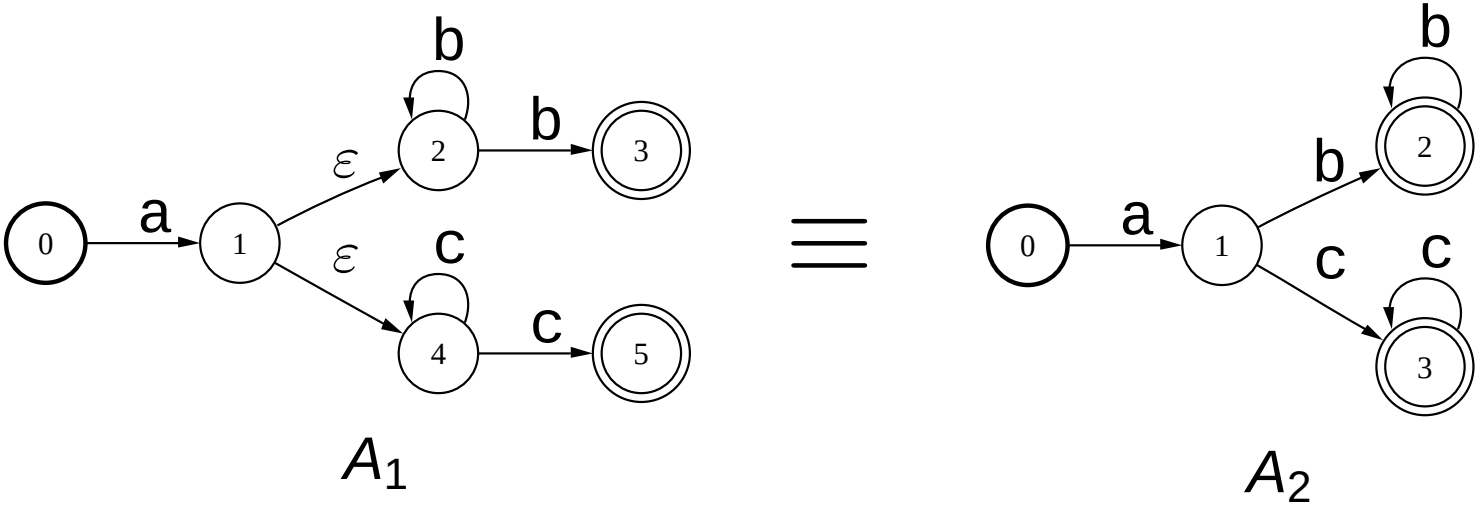
Automaty skończone

równoważność automatów

- o dwóch automatach mówimy, że są równoważne, jeśli języki akceptowane przez jeden i drugi są równe:

$$A_1 \equiv A_2 \quad \Leftrightarrow \quad L(A_1) = L(A_2)$$

- przykład:



A_1 A_2

Automaty skończone

najważniejsze podklasy

- automaty skończone bez ε-przejść
- automaty skończone deterministyczne

Automaty skończone

deterministyczne

- deterministycznym automatem skończonym nazywamy automat skończony spełniający warunki:
  1. nie posiada ε-przejść ($\forall_{q \in Q}\ \delta(q, \varepsilon) = \emptyset$),
  2. dla każdego stanu q i symbolu a, z q wychodzi co najwyżej jedno przejście przez a ($\forall_{q \in Q} \forall_{a \in \Sigma}\ \ |\delta(q, a)| \leq 1$)
- w automatach deterministycznych funkcję przejść δ definiuje się najczęściej jako funkcję (częściową) z $Q \times \Sigma$ w Q

Automaty skończone

niedeterministyczne

- automaty nie spełniające przynajmniej jednego z warunków 1. i 2. nazywamy niedeterministycznymi
- automat A_1 jest automatem niedeterministycznym, natomiast A_2 – automatem deterministycznym

Automaty skończone

podstawowe własności

- dla każdego niedeterministycznego automatu skończonego z ε-przejściami istnieje równoważny mu niedeterministyczny automat skończony bez ε-przejść (ε-przejścia można zawsze wyeliminować)
- dla każdego niedeterministycznego automatu skończonego istnieje równoważny mu deterministyczny automat skończony
- zatem : wszystkie omawiane tu podklasy automatów skończonych akceptują tę samą klasę języków

Automaty skończone

podstawowe własności c.d.

- klasa języków akceptowanych przez automaty skończone to klasa języków regularnych, czyli ta sama klasa języków, którą da się opisać za pomocą wyrażeń regularnych
- na podstawie każdego wyrażenia regularnego można utworzyć automat skończony akceptujacy język definiowany przez to wyrażenie

Automaty skończone

podstawowe własności c.d.

złożoność czasowa i pamięciowa problemu stwierdzenia, czy słowo w należy do języka definiowanego przez wyrażenie regularne r (tabela za ASU)

automat	**pamięć**	**czas**
NAS	$O(\|r\|)$	$O(\|r\| \times \|w\|)$
DAS	$O(2^{\|r\|})$	$\boxed{O(\|w\|)}$

$|r|$ – długość wyrażenia regularnego
$|w|$ – długość słowa

Automaty skończone

podstawowe własności c.d.

- dla każdego języka regularnego istnie akceptujący do deterministyczny automat skończony o minimalnej liczbie stanów i jest on unikalny

Automaty skończone

podstawowe operacje na automatach

przekształcenia

- usuwanie ε-przejść
- determinizacja
- minimalizacja

Automaty skończone

podstawowe operacje na automatach c.d.

konstrukcja automatu

- z listy słów
- z wyrażenia regularnego

Automaty skończone

podstawowe operacje na automatach c.d.

operacje na automatach

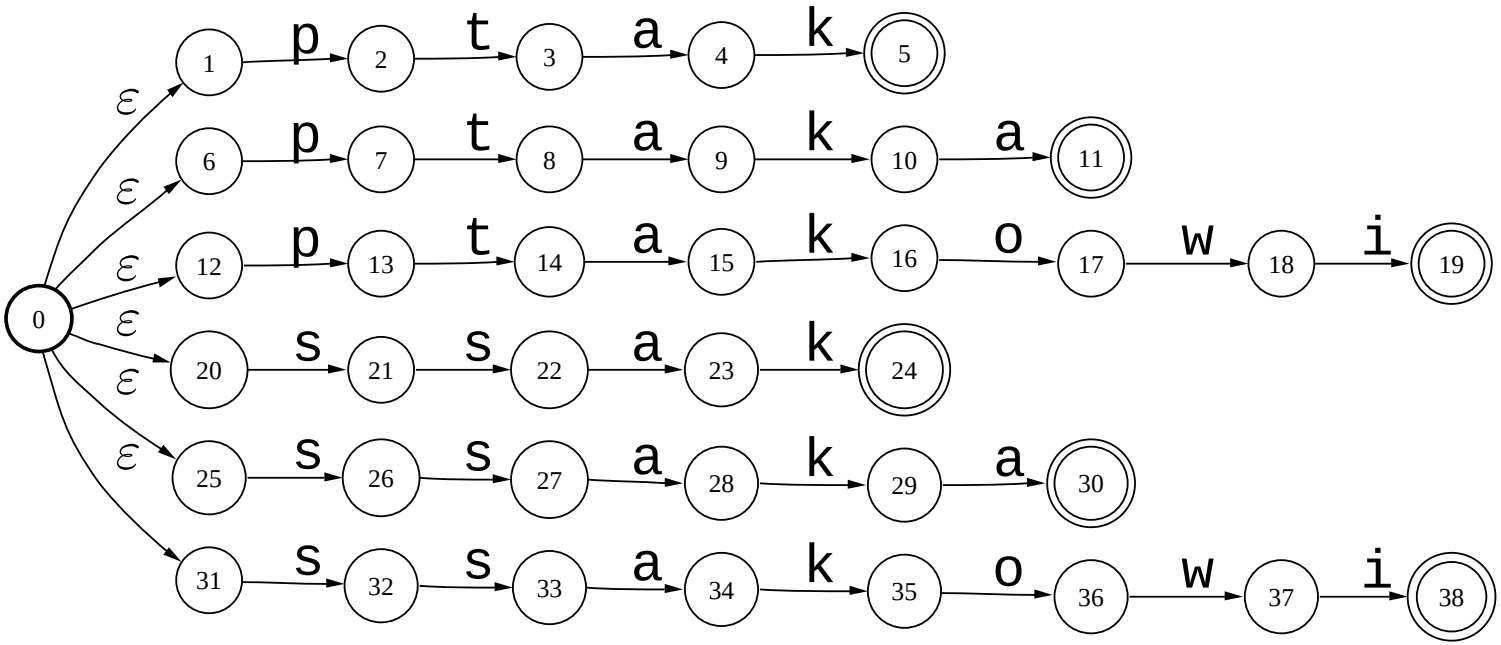
- suma
- przecięcie
- różnica
- konkatenacja
- domknięcie

  - wynikiem sumy (przecięcia, różnicy, konkatenacji) automatów A i B jest automat akceptujący język będący sumą (przecięciem, różnicą, konkatenacją) języków akceptowanych przez automaty A i B
  - wynikiem domknięcia automatu A jest automat akceptujący domknięcie języka akceptowanego przez A

Automaty skończone

konstrukcja z listy słów (1)

Tworzymy jednościeżkowe automaty dla wszystkich słów i sumujemy je.

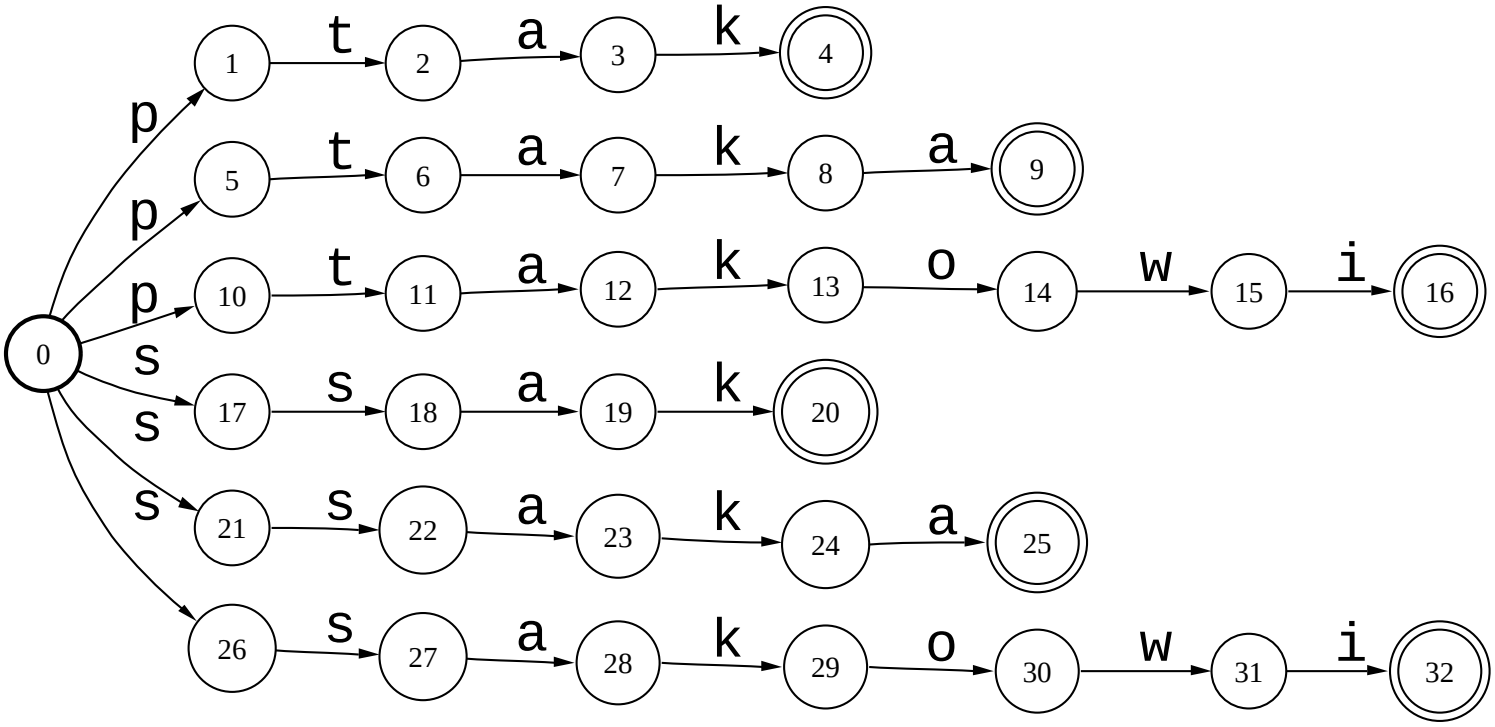


S_1

Automaty skończone

konstrukcja z listy słów (2)

$S_2 = S_1$ po usunięciu ε-przejść:

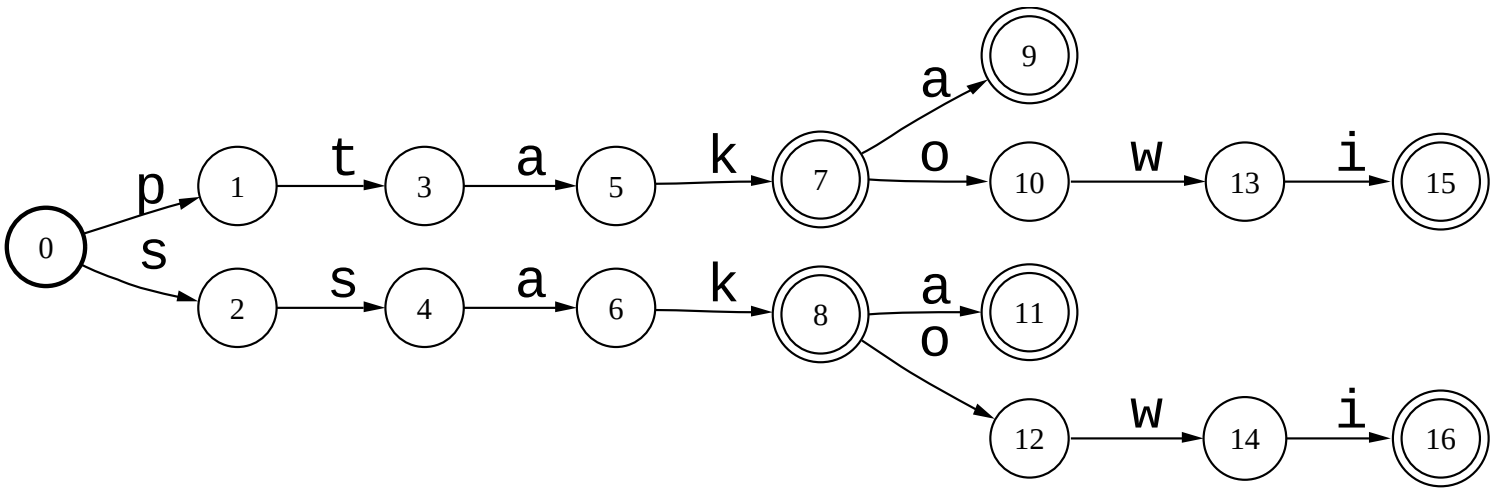


S_2

Automaty skończone

konstrukcja z listy słów (3)

$S_3 = S_2$ po determinizacji:

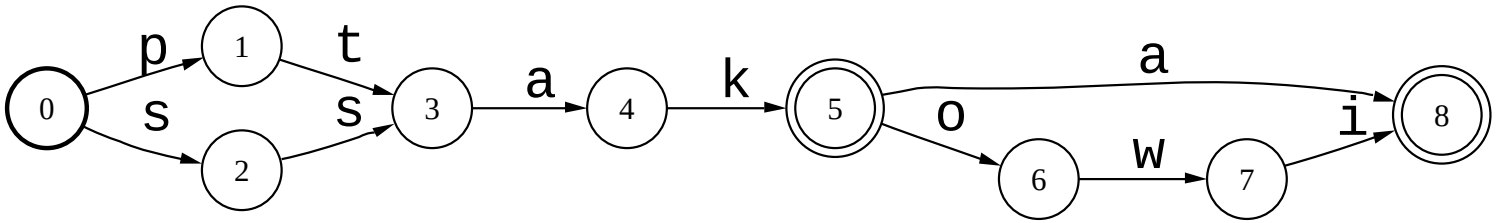


S_3

Automaty skończone

konstrukcja z listy słów (4)

$S_4 = S_3$ po minimalizacji:



S_4

Automaty skończone

konstrukcja z lisy słów c.d.

- efektywniejsze pamięciowo sposoby budowania automatu z listy słów:
  - buduje się od razu automat deterministyczny o strukturze drzewiastej, w której uwzględnione są wspólne prefiksy słów (taki jak S_3).
  - buduje się od razu automat minimalny (taki jak S_4) (autor: Jan Daciuk, Politechnika Gdańska)

Automaty skończone

automatowa reprezentacja listy słów: przykład ilościowy

liczba słów	1 072 565
rozmiar pliku tekstowego	12.7 MB
automat minimalny:	
liczba stanów	56 268
liczba przejść	164 240
rozmiar (format FSM Library)	3.3 MB
rozmiar (wektor przejść, 4B/przejście)	0.66 MB

Automaty skończone

konstrukcja z wyrażenia regularnego

- patrz J. E. Hopcroft, J. D. Ullman, *Wprowadzenie do teorii automatów, języków i obliczeń*, Wydawnictwo Naukowe PWN, 1994, rozdział 2.5

Automaty skończone

zastosowania

- analiza leksykalna
- przeszukiwanie tekstu
- reprezentacja słowników
- korekta ortografii, błędów literowych, wyników OCR

Automaty skończone

przeszukiwanie tekstu

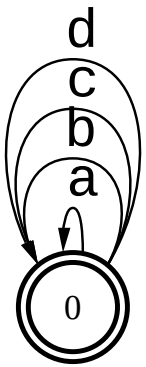
- chcemy znaleźć w tekście wyrażenie pasujące do wzorca zdefiniowanego wyrażeniem regularnym r.
  1. budujemy automat akceptujący język $\Sigma^* L(r)$
  2. podajemy na wejście kolejne znaki tekstu tak długo, aż automat osiągnie stan końcowy
- w ten sposób znajdujemy pierwsze najkrótsze dopasowanie
- jak znaleźć pierwsze najdłuższe dopasowanie?

Automaty skończone

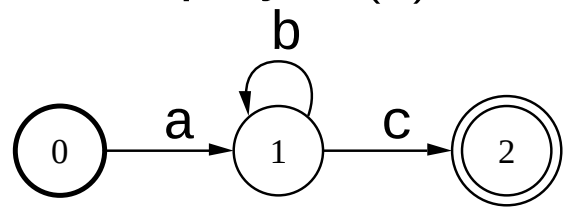
przeszukiwanie tekstu

$$r = \mathrm{ab}^*\mathrm{c}, \Sigma = \{\mathrm{a}, \mathrm{b}, \mathrm{c}, \mathrm{d}\}$$

A akceptuje Σ^*



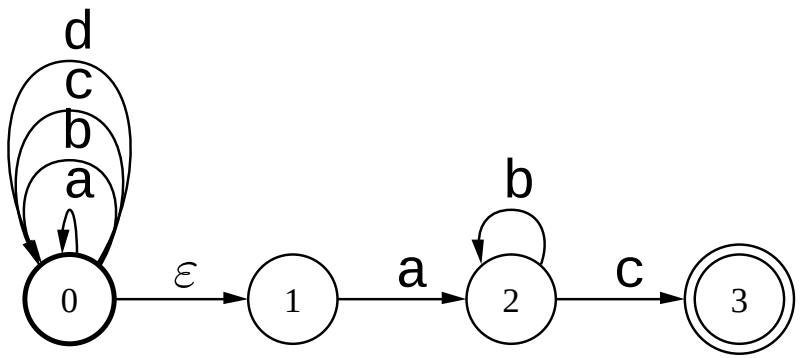
B akceptuje $L(r)$



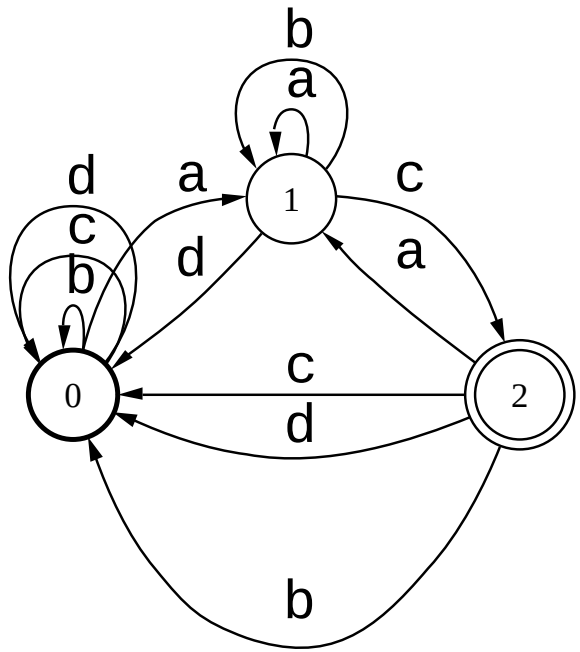
Automaty skończone

przeszukiwanie tekstu

$C = AB$ akceptuje $\Sigma^* L(r)$



C po minimalizacji



Automaty skończone

reprezentacja słowników

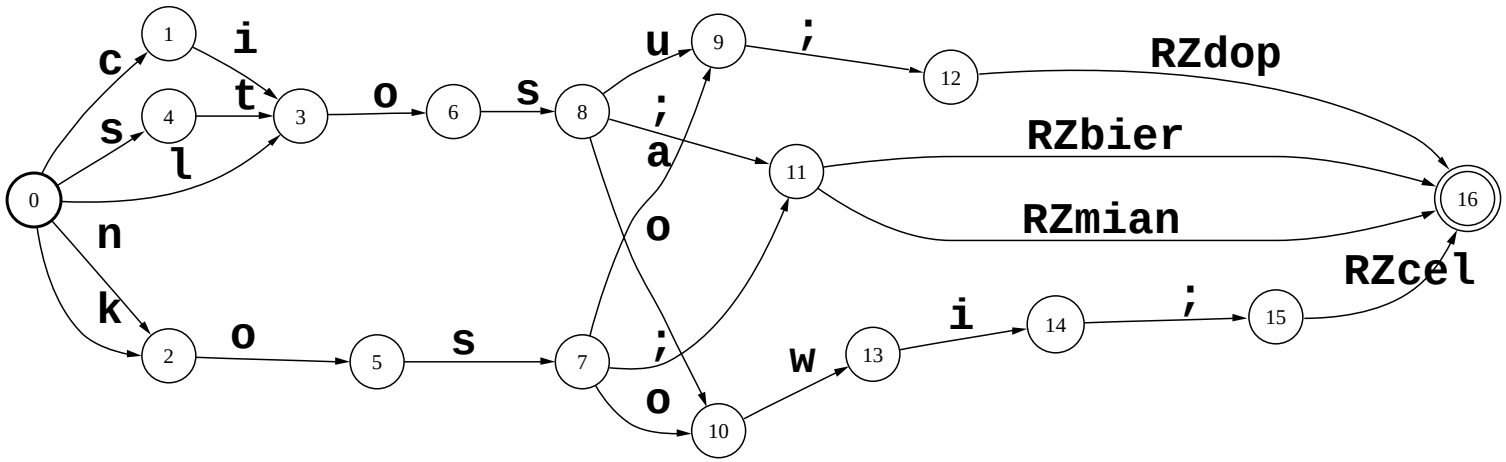
- ścieżki automatu etykietowane są napisami składającymi się z hasła, separatora i opisu, np.:

cios;RZmian
cios;RZbier
ciosu;RZdop
ciosowi;RZcel
kos;RZmian
kos;RZbier
kosa;RZdop
kosowi;RZcel
los;RZmian
los;RZbier
losu;RZdop
losowi;RZcel

Automaty skończone

reprezentacja słowników

- automat słownika:



- aby uzyskać informację o słowie *w*, przechodzimy ścieżkę etykietowaną przez *w*`;` (*w* plus separator). Następnie, wyszukujemy w automacie wszystkie ścieżki od osiągniętego stanu do stanu końcowego, uzyskując wszystkie opisy słowa.

Automaty skończone

korekta ortografii

- problem korekty błędnie zapisanego słowa w sprowadza się do wyszukania w automacie (reprezentującym listę słów) ścieżek, etykietowanych słowami, których odległość edycyjna od w jest najmniejsza (mniejsza niż zadany próg)
- algorytm: patrz K. Oflazer, Error-tolerant Finite State Recognition with Applications to Morphological Analysis and Spelling Correction, *Computational Linguistics* 22:1
- odległość edycyjna (ed) między słowami w_1 i w_2 to najmniejsza liczba elementarnych operacji edycyjnych potrzebnych do przekształcenia w_1 w w_2.
- elementarne operacje edycyjne to:
  - wymazanie litery
  - dodanie litery
  - zastąpienie litery inną
  - przestawienie miejscami dwóch sąsiadujących liter
- przykłady
  - $ed(\texttt{krzesło}, \texttt{krzesło}) = 0$
  - $ed(\texttt{kszesło}, \texttt{krzesło}) = 1$
  - $ed(\texttt{kzresł}, \texttt{krzesło}) = 2$

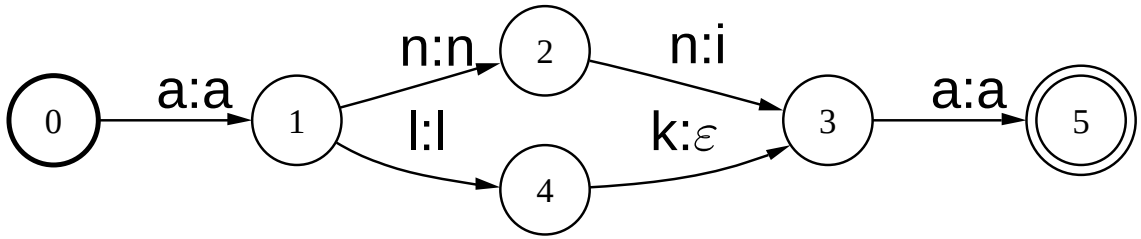
Przetworniki skończone

Przetworniki skończone

- przetworniki skończone /in. automaty z wyjściem, automaty Mealy'ego, transducery, transduktory; ang. finite-state transducers/ stanowią uogólnienie automatu skończonego
- z przejściem, oprócz symbolu wejściowego, jest związany też symbol wyjściowy
- symbole wyjściowe należą do alfabetu wyjściowego, który może być inny niż alfabet wejściowy
- rozważa się też przetworniki skończone, w których symbole wyjściowe są związane ze stanami, a nie przejściami (automaty Moore'a).

Przetworniki skończone – przykład

- przetworniki skończone definiują zbiór par napisów (relację miedzy napisami).



- przetwornik pokazany powyżej definiuje relację:

  anna:ania
  alka:ala

Przetworniki skończone

definicja

- Przetwornikiem skończonym nazywamy szóstkę

$$\langle Q, \Sigma_1, \Sigma_2, E, q_0, F \rangle$$

gdzie:

Q		– zbiór stanów
Σ_1		– alfabet wejsciowy
Σ_2		– alfabet wyjściowy
E	$\subseteq \quad Q \times (\Sigma_1 \cup \{\varepsilon\}) \times (\Sigma_2 \cup \{\varepsilon\}) \times Q$	– zbiór przejść
q_0	$\in \quad Q$	– stan początkowy
F	$\subseteq \quad Q$	– zbiór stanów końcowych

- Jeśli $(q, \varepsilon, \varepsilon, q') \notin E$ dla żadnej pary q, q'– przetwornik literowy (klasa równoważna).
- Jeśli $E \subseteq Q \times \Sigma_1 \times \Sigma_2 \times Q$ – przetwornik literowy ε-wolny (podklasa właściwa). (terminologia za E. Roche, Y. Schabes, *Finite-State Language Processing*, MIT Press, 1997)

Przetworniki skończone

transdukcja, równoważność

- relację między napisami definiowaną przez przetwornik T nazywamy transdukcją /ang. transduction/ i oznaczamy $|T|$.
- $(w, w') \in |T|$ jeśli istnieje w T ścieżka od stanu początkowego do końcowego etykietowana wejściem w i wyjściem w'.
- przetworniki T_1 i T_2 są równoważne, jeśli definiują tę samą transdukcję: $|T_1| = |T_2|$

Przetworniki skończone

projekcja

- pierwsza projekcja przetwornika T ($p_1(T)$): automat skończony powstający przez ograniczenie etykiet przejść do symboli wejściowych
- druga projekcja przetwornika T ($p_2(T)$): automat skończony powstający przez ograniczenie etykiet przejść do symboli wyjściowych
- automaty uzyskane w wyniku projekcji nazywamy, odpowiednio, automatem wejściowym i automatem wyjściowym
- automat wejściowy definiuje język wejściowy przetwornika, a automat wyjściowy – język wyjściowy przetwornika; oba języki muszą być zatem językami regularnymi.

Przetworniki skończone

ważne podklasy

- przetwornik literowy: $(q, \varepsilon, \varepsilon, q') \notin E$ dla żadnej pary q, q' – podklasa równoważna, definiuje tę samą klasę transdukcyj
- przetwornik literowy ε-wolny: $E \subseteq Q \times \Sigma_1 \times \Sigma_2 \times Q$ – podklasa słabsza (definiuję podklasę właściwą transdukcyj)
- przetwornik sekwencyjny: automat wejściowy jest deterministyczny – podklasa słabsza

Przetworniki skończone

operacje

konstrukcja:

- z rozszerzonych wyrażeń regularnych (ich operandami są pary symboli lub pary napisów)
- ze zbioru par napisów
- z reguł kontekstowej zamiany symboli

Przetworniki skończone

operacje

transformacje:

- usunięcie ε:ε-przejść
- deternimizacja (tak, by automat wejściowy $p_1(T)$ był deterministyczny) – w ogólnym przypadku niemożliwa
- minimalizacja – algorytmy znane dla niektórych podklas przetworników (np. dla przetworników sekwencyjnych)
- inne

Przetworniki skończone

operacje

suma	$\lvert T_1 \cup T_2 \rvert = \lvert T_1 \rvert \cup \lvert T_2 \rvert$	zdefiniowana zawsze
złożenie	$\lvert T_1 \circ T_2 \rvert = \lvert T_1 \rvert \circ \lvert T_2 \rvert$	zdefiniowane zawsze
inwersja	$\lvert T^{-1} \rvert = \lvert T \rvert^{-1}$	zdefiniowana zawsze
przecięcie	$\lvert T_1 \cap T_2 \rvert = \lvert T_1 \rvert \cap \lvert T_2 \rvert$	zdefiniowane dla przetworników literowych ε-wolnych

Przetworniki skończone

przykłady zastosowań

- reprezentacja słowników

  przekształcenie: słowo hasłowe $\longrightarrow$ opis

- analiza/synteza mowy

  przekształcenie: ciąg fonemów $\longrightarrow$ ciąg liter, ciąg liter $\longrightarrow$ ciąg fonemów

- analiza/synteza fleksyjna

  przekształcenie: forma fleksyjna $\longrightarrow$ temat+końcówka, temat+końcówka $\longrightarrow$ forma fleksyjna

  kot+e $\longrightarrow$ kocie

- korekta ortografii, błędów literowych, wyników OCR

  przekształcenie: słowo z błędem $\longrightarrow$ (’najbliższe’) słowo poprawne

  ktury $\longrightarrow$ który

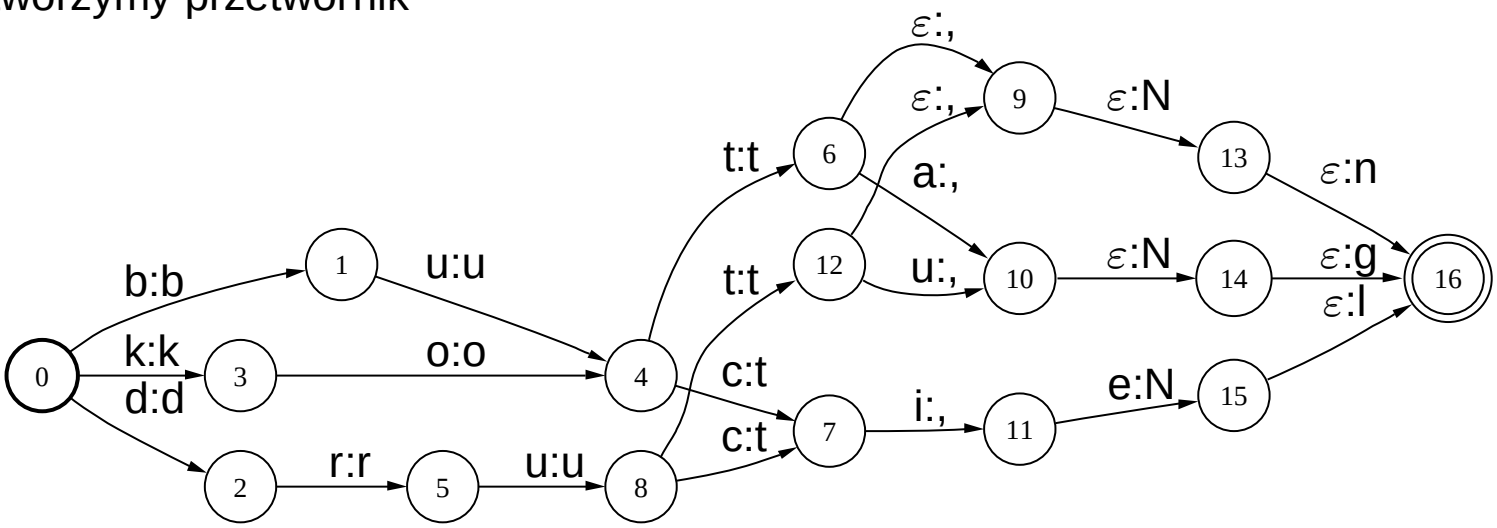
Przetworniki skończone

reprezentacja słownika

z listy par *hasło*:*opis*

kot:kot,Nn	but:but,Nn	drut:drut,Nn
kota:kot,Ng	buta:but,Ng	drutu:drut,Ng
kocie:kot,Nl	bucie:but,Nl	drucie:drut,Nl

tworzymy przetwornik



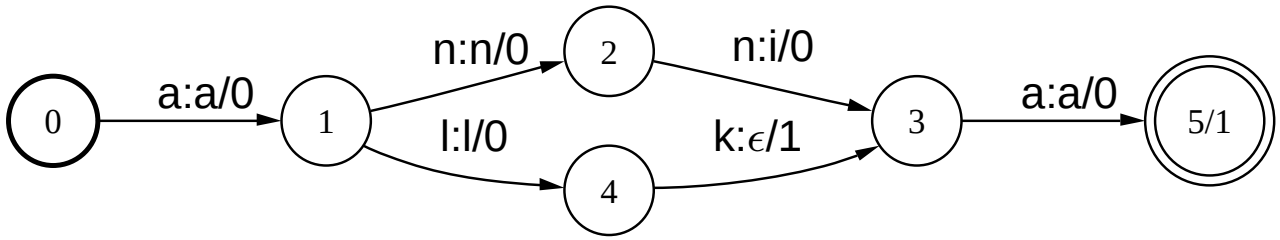
Reprezentacja przetwornikowa słownika jest zazwyczaj mniejsza od automatowej. Jeśli uda się stworzyć przetwornik sekwencyjny, informację uzyskujemy szybciej niż z automatu.

Automaty i przetworniki skończone z ważonymi przejściami

- z przejściami, a także ze stanami końcowymi, wiąże się wagę (koszt)
- koszt przejścia ścieżki to suma (iloczyn) kosztów związanych z przejściami należącymi do tej ścieżki plus (razy) koszt związany ze stanem końcowym

Przetworniki skończone z ważonymi przejściami

przykład



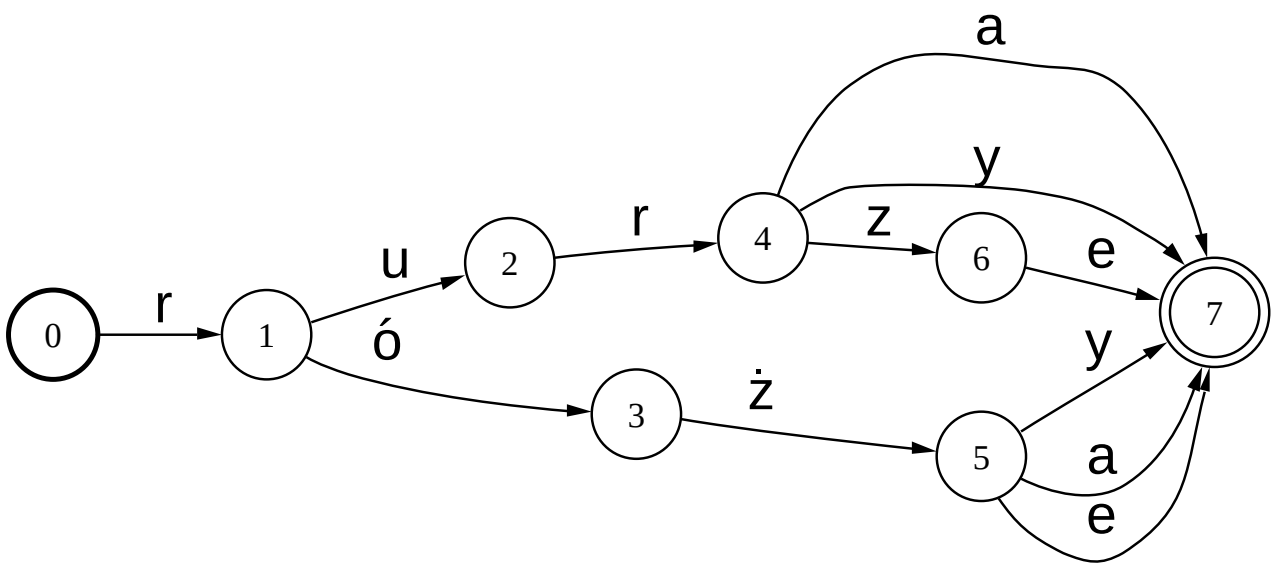
- przetworniki skończone definiują zbiór par napisów, przy czym dla każdej pary napisów określony jest pewien koszt, interpretowany np. jako koszt przekształcenia jednego napisu w drugi, odległość między napisami
- w przykładowym przetworniku:

  anna:ania <koszt 1>
  alka:ala <koszt 2>

Przetworniki skończone z ważonymi przejściami

korektor ortografii (1)

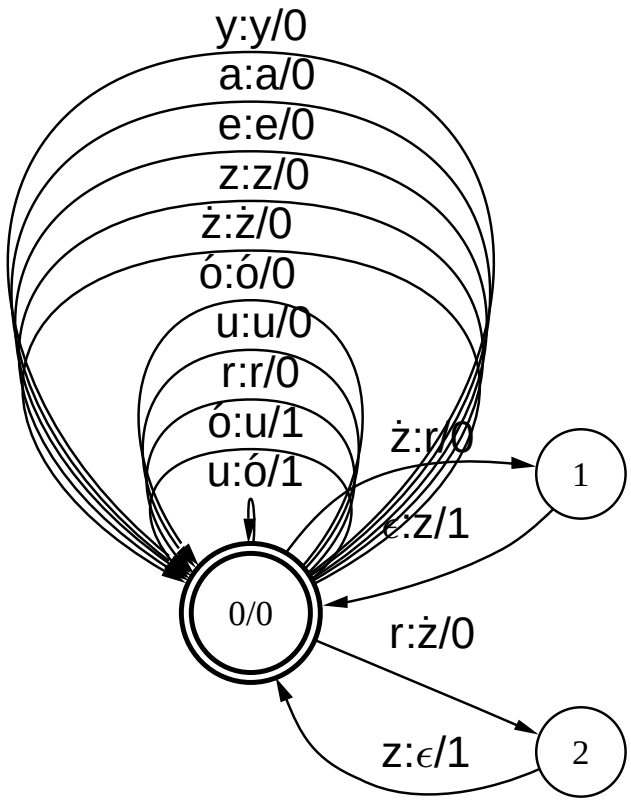
dany mamy słownik S (rura, rury, rurze, róża, róży, róże).



Przetworniki skończone z ważonymi przejściami

korektor ortografii (2)

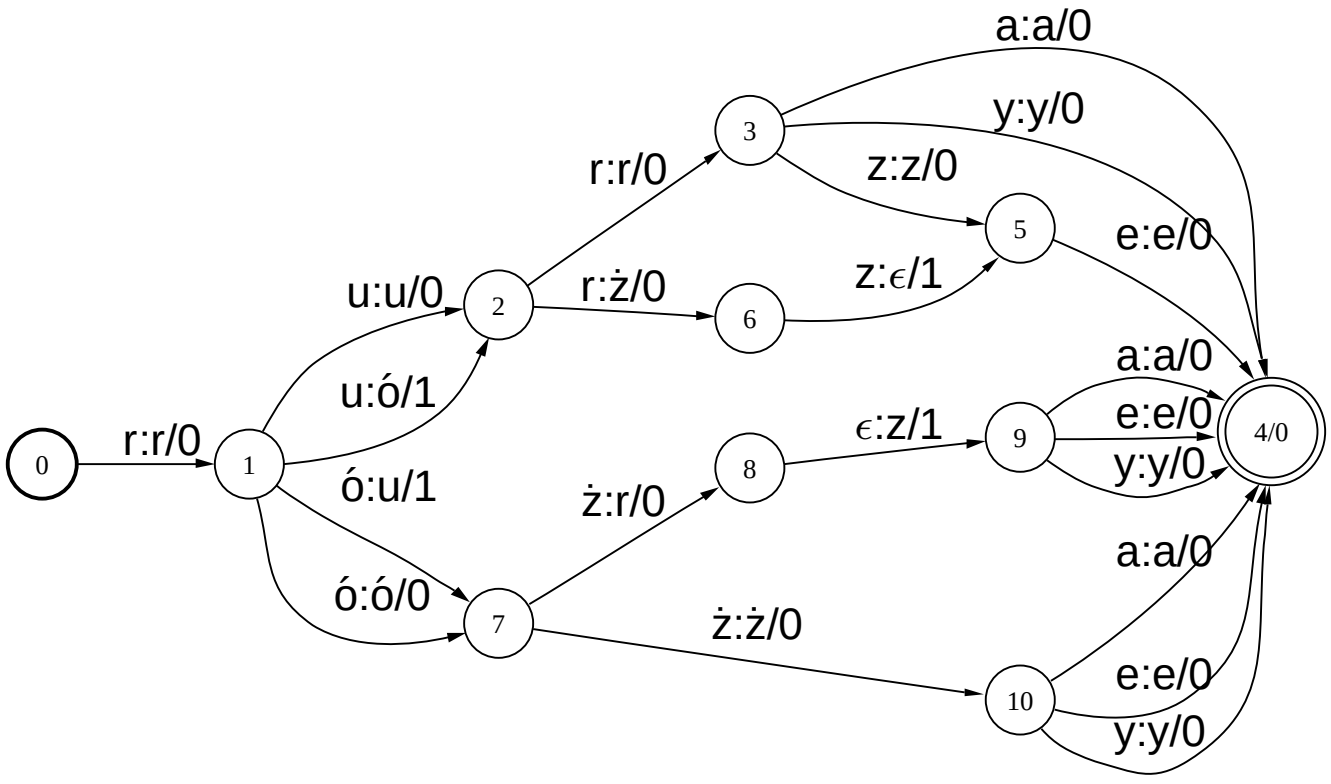
Tworzymy przetwornik *B*, który przetwarza tekst w taki sposób, że może wprowadzać weń błędy ortograficzne: każdy błąd kosztuje 1.



Przetworniki skończone z ważonymi przejściami

korektor ortografii (3)

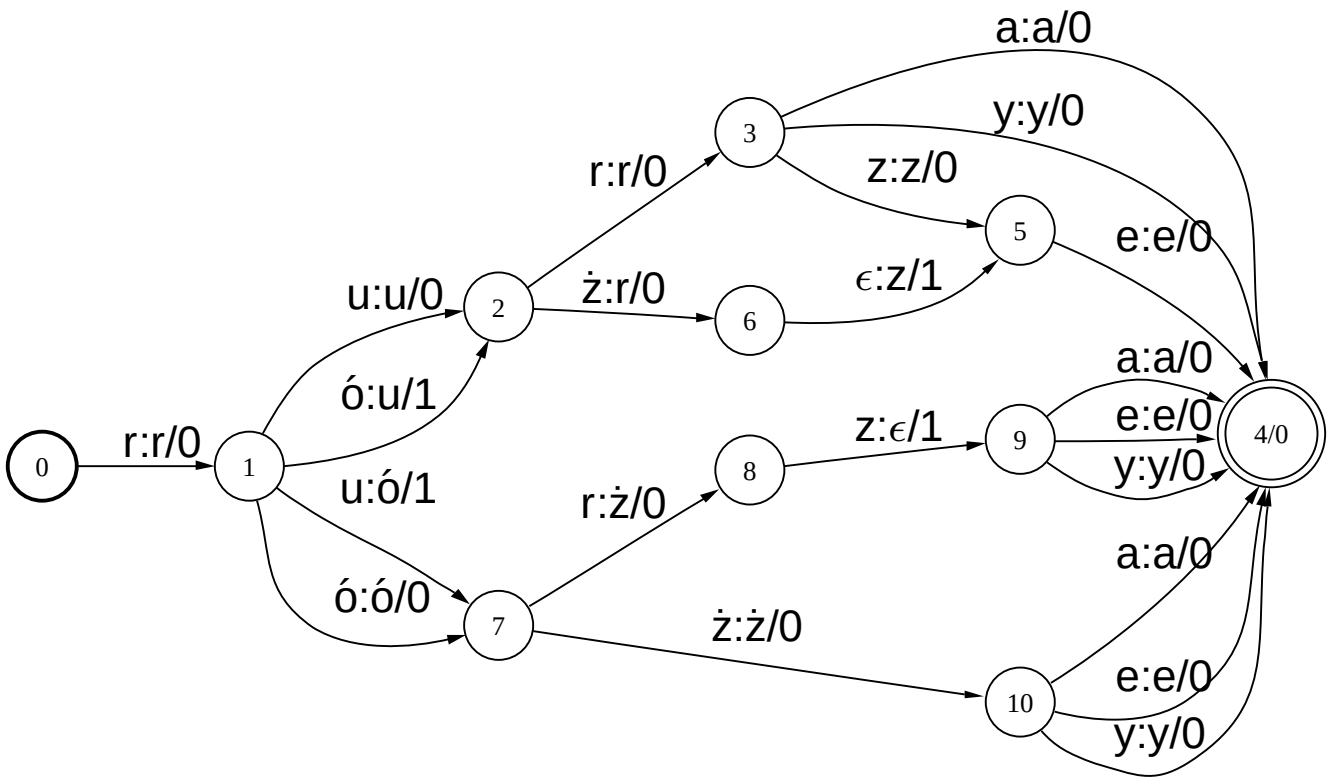
Złożenie S i B ($S \circ B$) daje przetwornik, który umie wprowadzać błędy ortograficzne do słów ze słownika S, za każdy błąd licząc koszt 1.



Przetworniki skończone z ważonymi przejściami

korektor ortografii (4)

Poprzez inwersję $(S \circ B)^{-1}$ otrzymujemy przetwornik, który umie poprawiać błędy ortograficzne w słowach ze słownika S, za jedną poprawkę licząc koszt 1.



Przetworniki skończone z ważonymi przejściami

korektor ortografii (5)

- ścieżki automatu $(S \circ B)^{-1}$:

```
rura:rura<0>          rurzy:róży<2>
rury:rury<0>          ruża:róża<1>
rurze:rurze<0>        ruże:róże<1>
ruże:rurze<1>         ruży:róży<1>
róra:rura<1>          rórza:róża<1>
róry:rury<1>          rórze:róże<1>
rórze:rurze<1>        rórzy:róży<1>
róże:rurze<2>         róża:róża<0>
rurza:róża<2>         róże:róże<0>
rurze:róże<2>         róży:róży<0>
```

- problem korekty błędu sprowadza się do wyszukania ścieżek, których koszt przejścia jest nie większy niż zadany próg (algorytm Viterbi)

Automaty i przetworniki skończone

literatura

- J. E. Hopcroft, J. D. Ullman, *Wprowadzenie do teorii automatów, języków i obliczeń*, Wydawnictwo Naukowe PWN, 1994, **(rozdział 2.)**