AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Analiza leksykalna – 2

Języki formalne i automaty

Dr inż. Janusz Majewski
Katedra Informatyki

Zadanie analizy leksykalnej

- wyodrębnianie symboli leksykalnych (tokenów)
- ignorowanie komentarzy
- ignorowanie białych znaków (spacji, tabulacji, znaków nowej linii…)
- korelowanie błędów zgłaszanych przez kompilator z numerami linii
- tworzenie kopii zbioru wejściowego (źródłowego) łącznie z komunikatami o błędach
- czasami realizacja funkcji preprocessingu, rozwijanie makrodefinicji

Specyfikacja analizatora leksykalnego (1)

wyrażenie_regularne_1	{akcja_1}
wyrażenie_regularne_2	{akcja_2}
…….	……..
wyrażenie_regularne_n	{akcja_n}

Zasady interpretacji specyfikacji:

- Zawsze rozpoznawany i uzgadniany jest token odpowiadający możliwie najdłuższemu leksemowi.
- W przypadku, gdy ten sam leksem odpowiada więcej niż jednemu tokenowi, uzgadniany jest token odpowiadający najwcześniejszej z „pasujących” pozycji specyfikacji.

Specyfikacja analizatora leksykalnego (2)

Przykład:

a	{akcja_1}
abb	{akcja_2}
a^*b^+	{akcja_3}

Analizowany łańcuch: aabba

Możliwe interpretacje: **a|abb|a**, **a|a|bb|a, a|a|b|b|a, aabb|a**

1 2 1 1 1 3 1 1 1 3 3 1 3 1

Poprawna interpretacja: **aabb|a**

3 1

gdyż:

- Zawsze rozpoznawany i uzgadniany jest token odpowiadający możliwie najdłuższemu leksemowi.

Specyfikacja analizatora leksykalnego (3)

Przykład:

a	{akcja_1}
abb	{akcja_2}
a^*b^+	{akcja_3}

Analizowany łańcuch: abba

Niektóre możliwe interpretacje: **abb|a**, **abb|a**

2 1 3 1

Poprawna interpretacja: **abb|a**

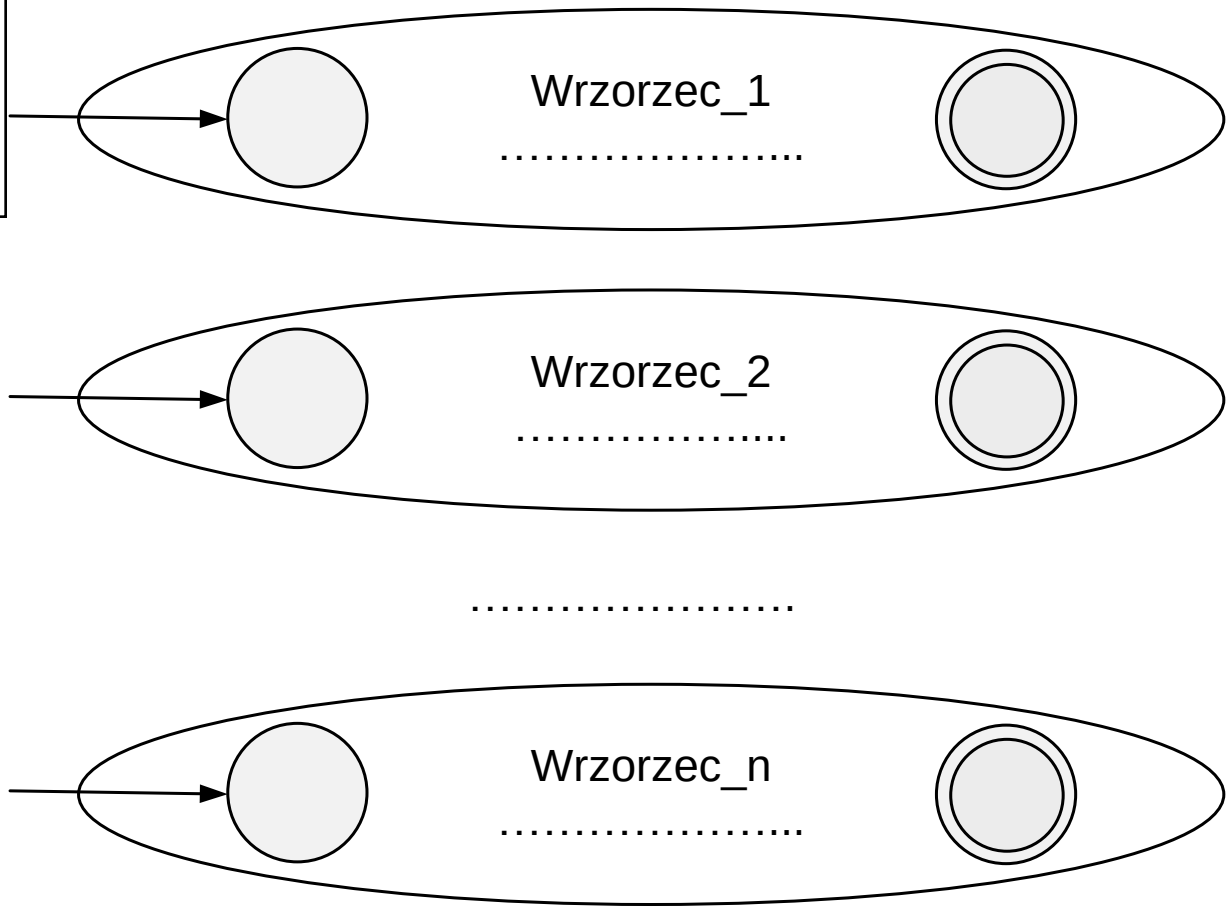
2 1

gdyż:

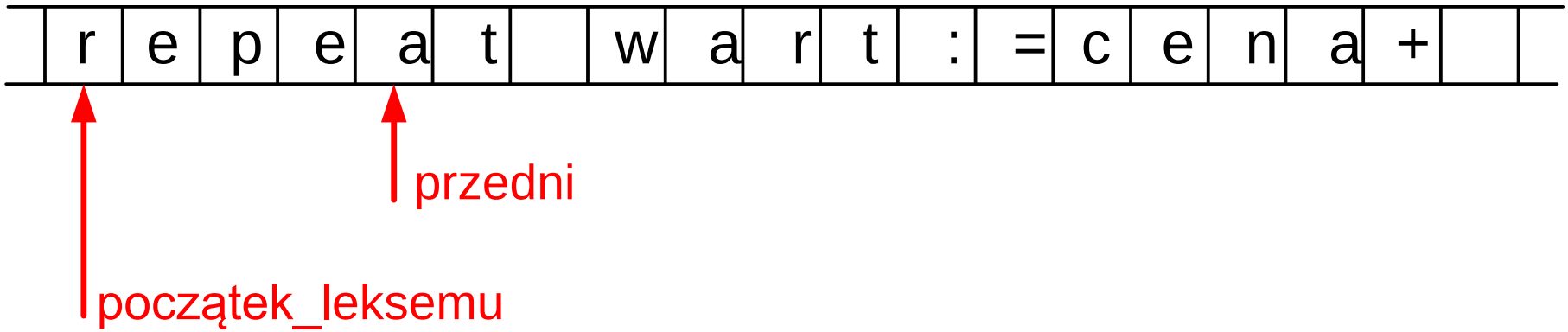
- W przypadku, gdy ten sam leksem odpowiada więcej niż jednemu tokenowi, uzgadniany jest token odpowiadający najwcześniejszej z „pasujących” pozycji specyfikacji.

Izolowane automaty deterministyczne

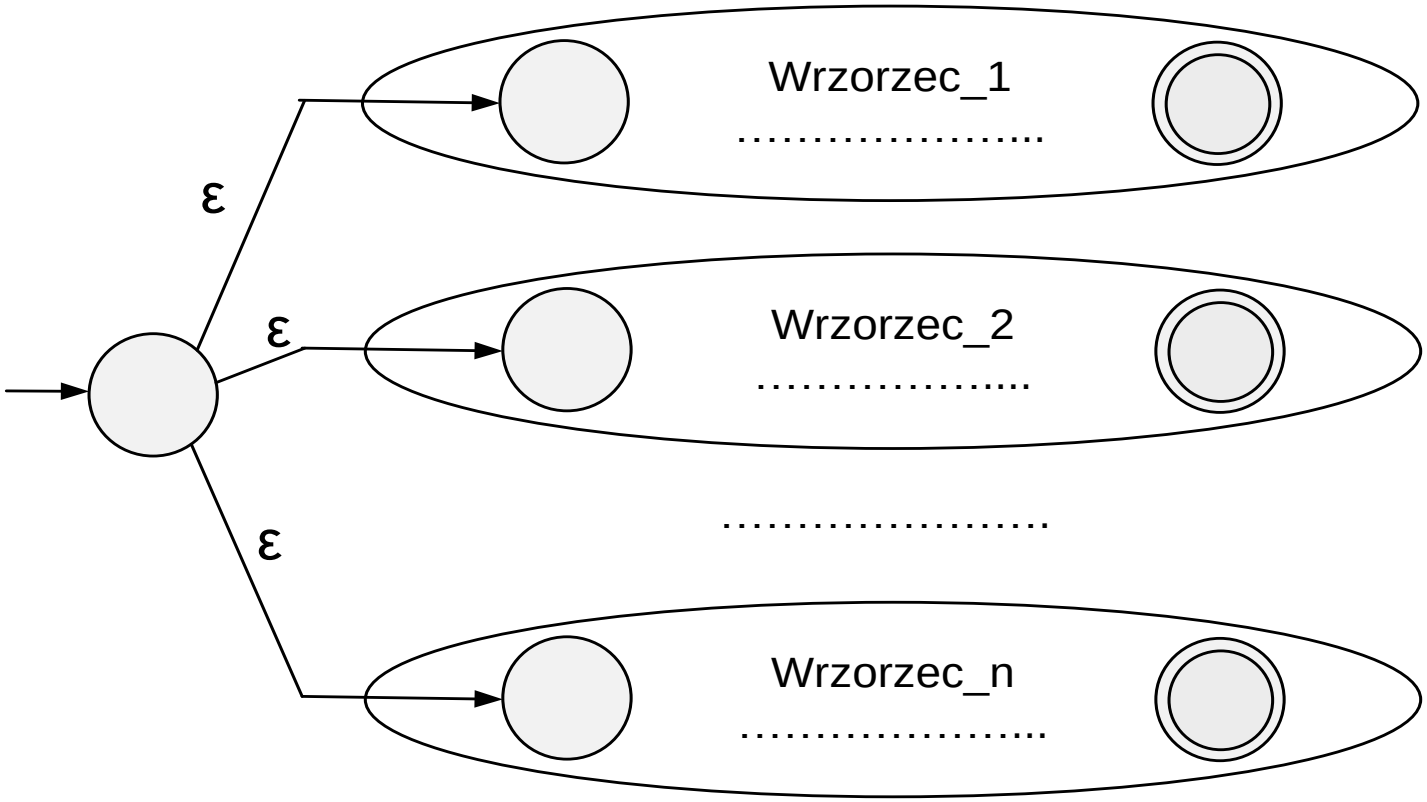
wzorzec_1	{akcja_1}
wzorzec_2	{akcja_2}
…….	……..
wzorzec_n	{akcja_n}



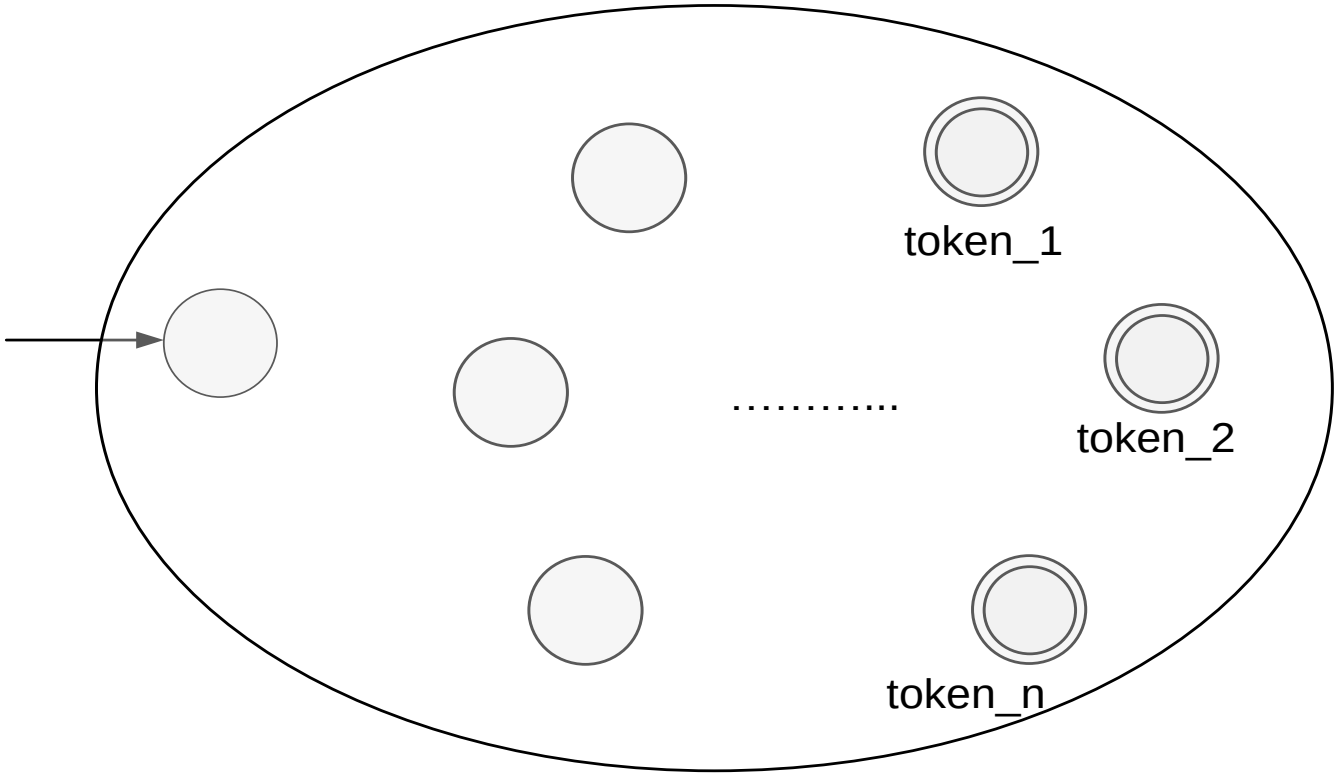
Najprostsza implementacja – algorytm z powrotami



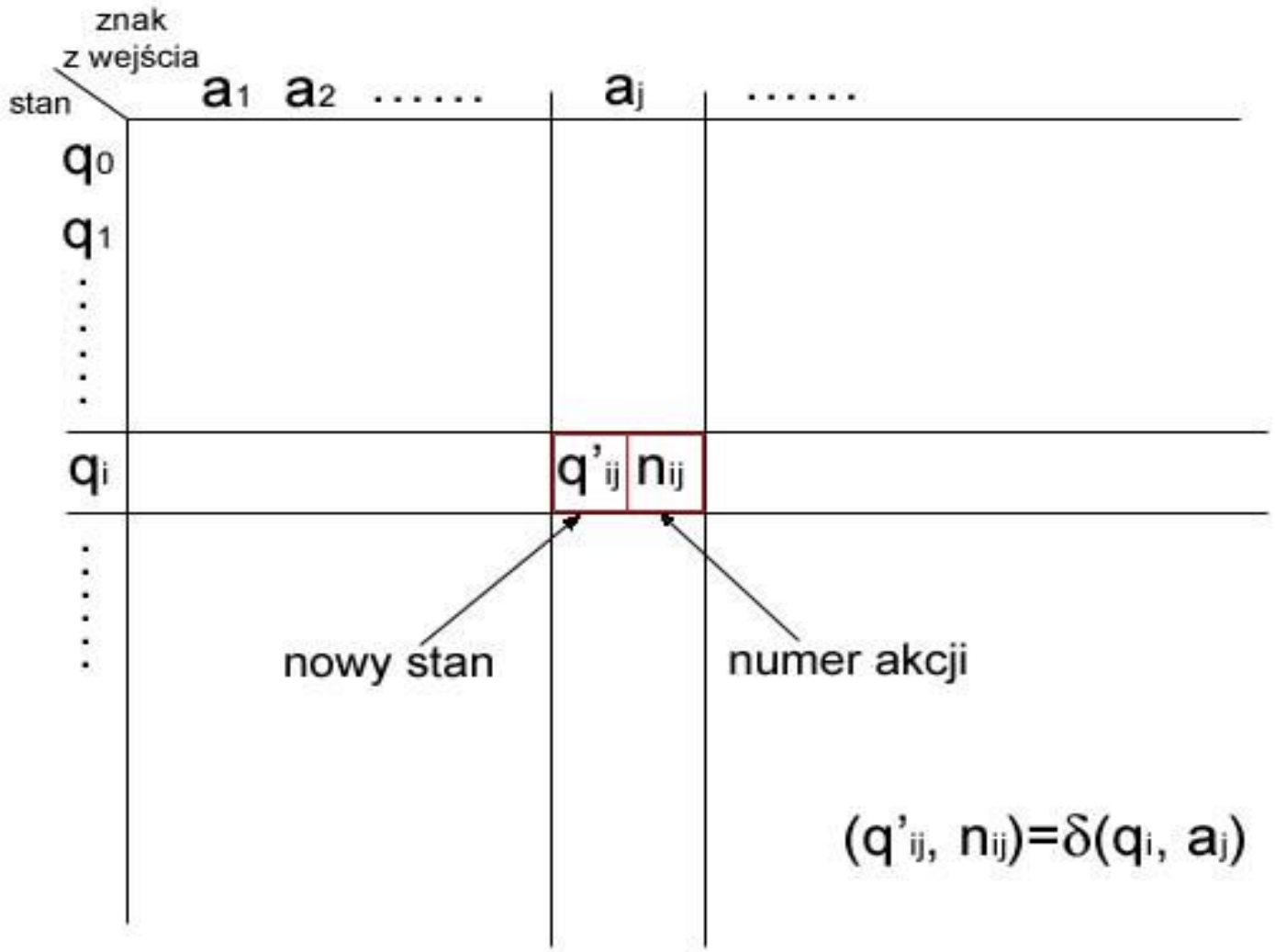
Kombinowany automat niedeterministyczny



Automat deterministyczny



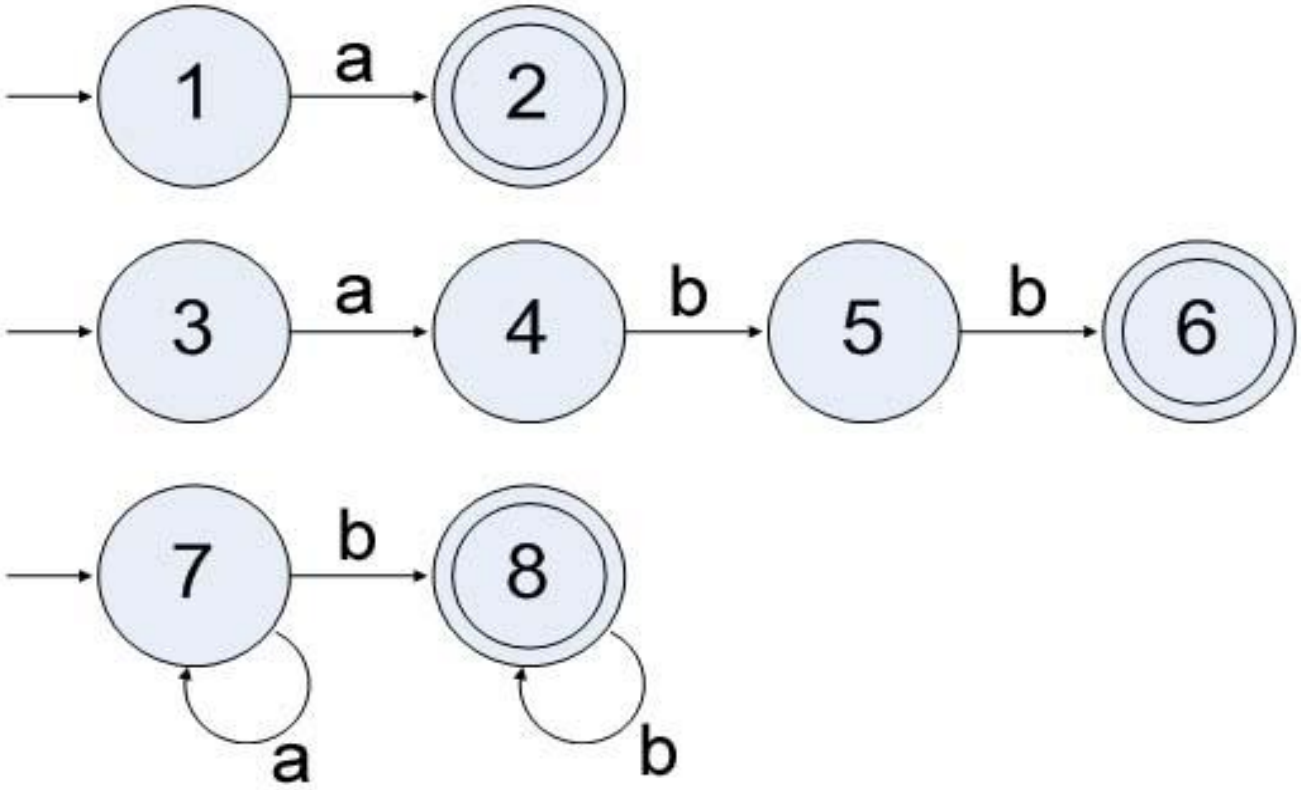
Implementacja



Budowa automatu skończonego (1)

a {akcja 1}
abb {akcja 2}
a^*b^+ {akcja 3}

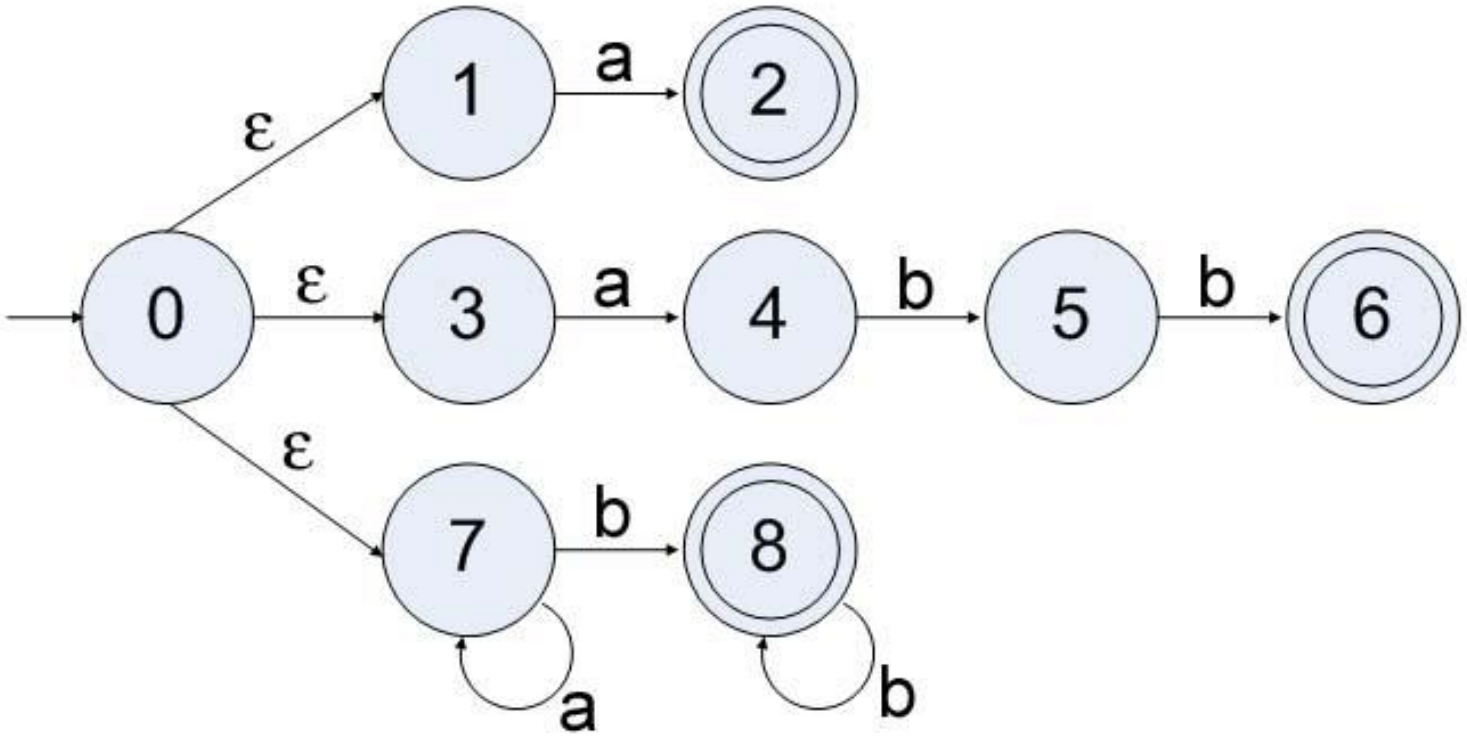
pojedyncze NFA



Budowa automatu skończonego (2)

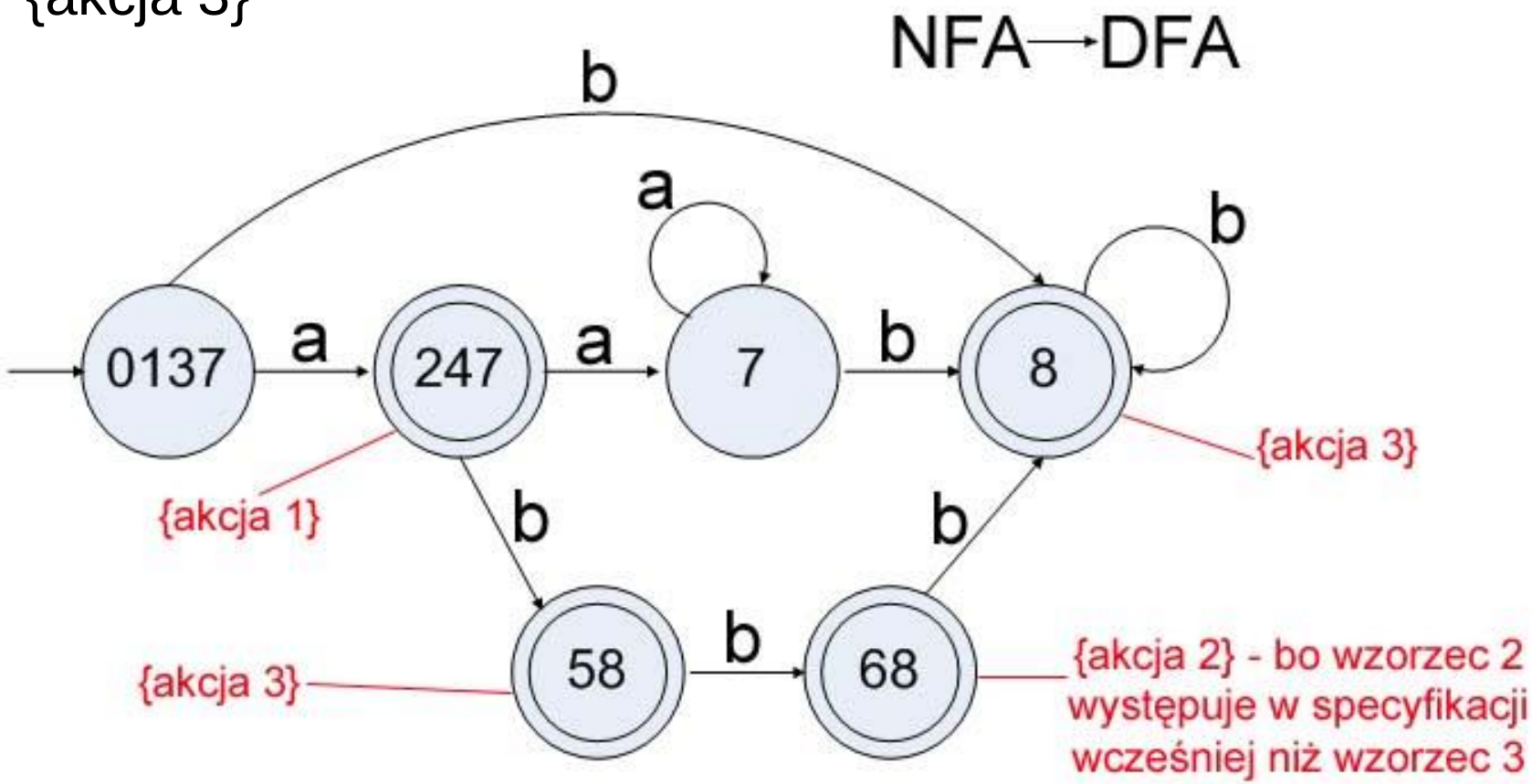
a {akcja 1}
abb {akcja 2}
a^*b^+ {akcja 3}

kombinowany NFA



Budowa automatu skończonego (3)

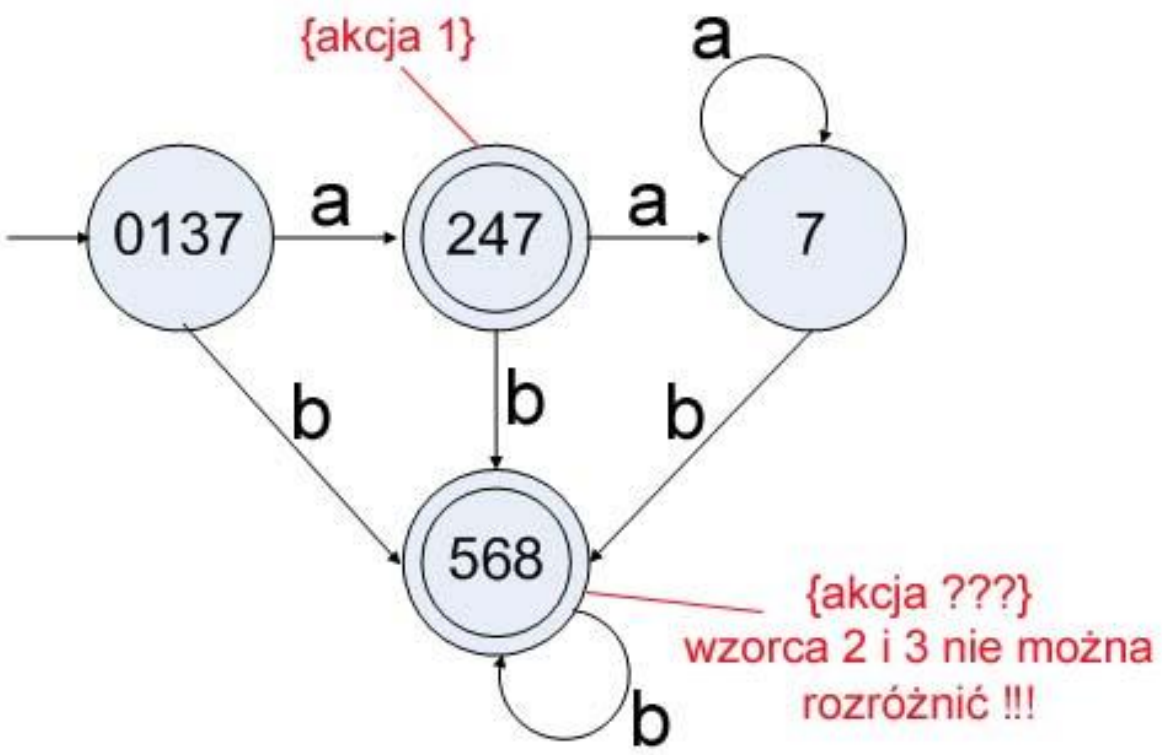
a {akcja 1}
abb {akcja 2}
a^*b^+ {akcja 3}



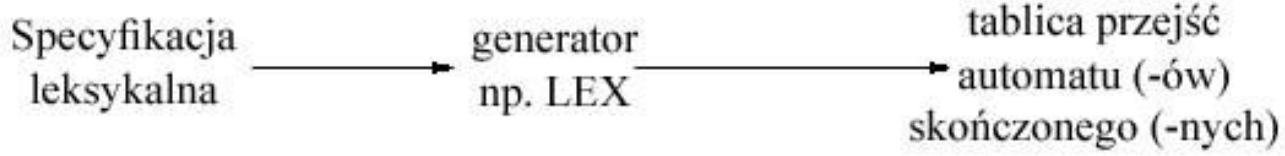
Budowa automatu skończonego (4)

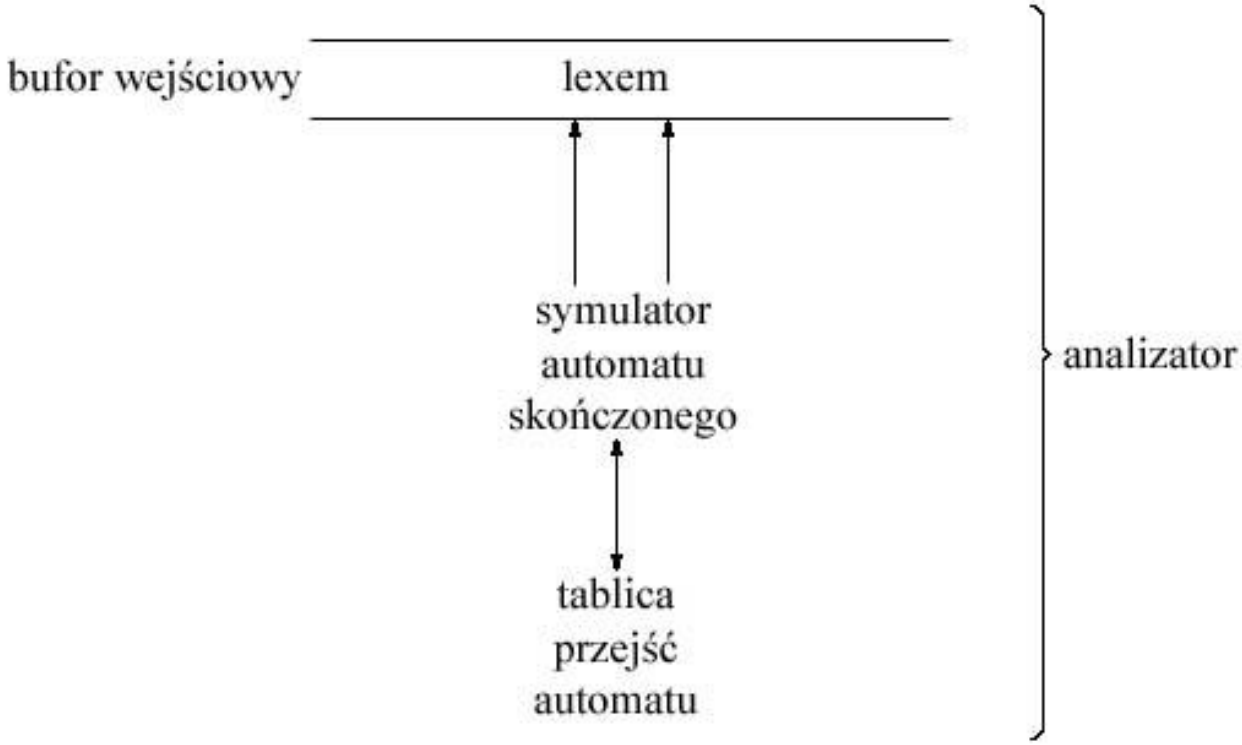
a	{akcja 1}
abb	{akcja 2}
a^*b^+	{akcja 3}

optymalizacja DFA

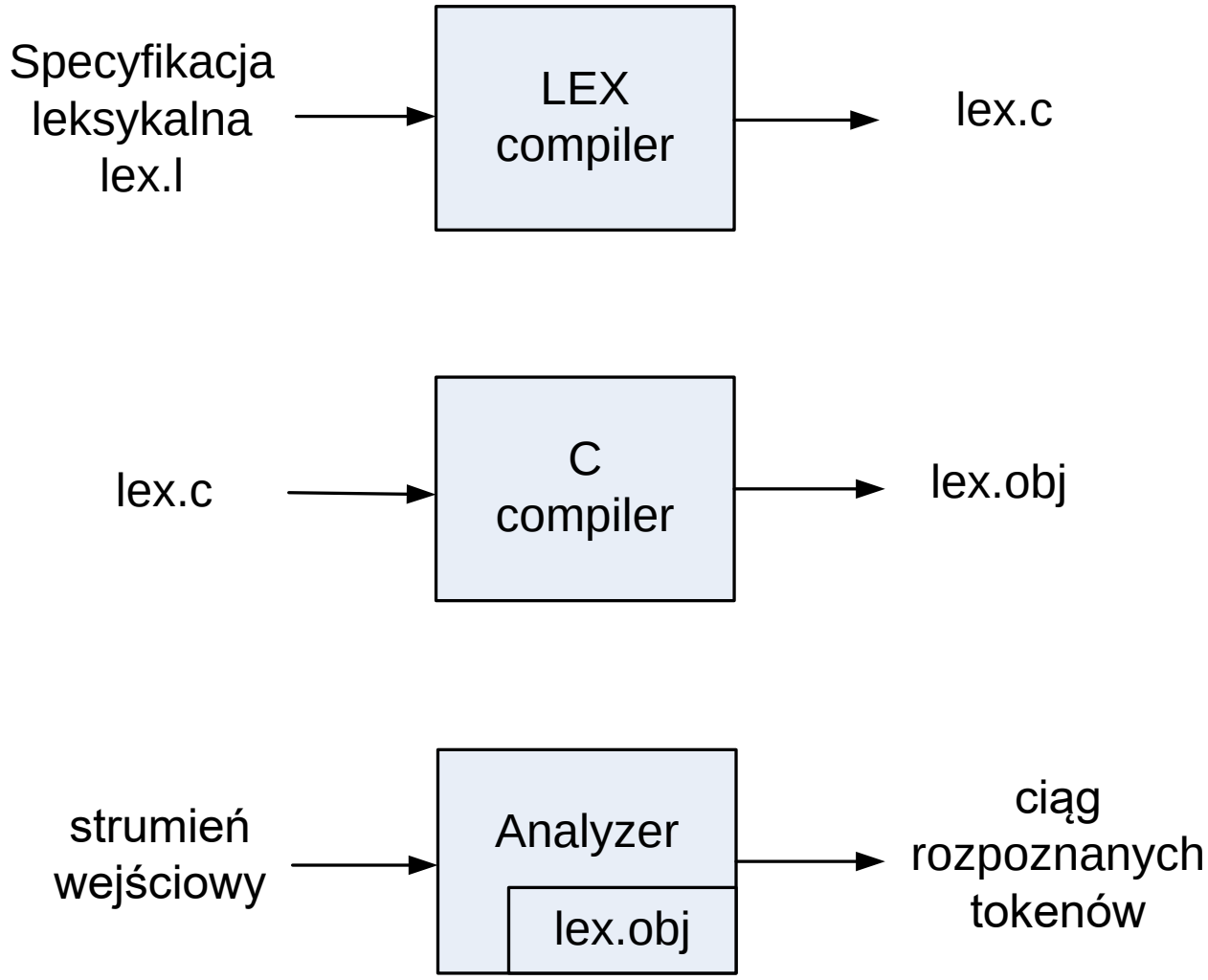


Generatory analizatorów leksykalnych

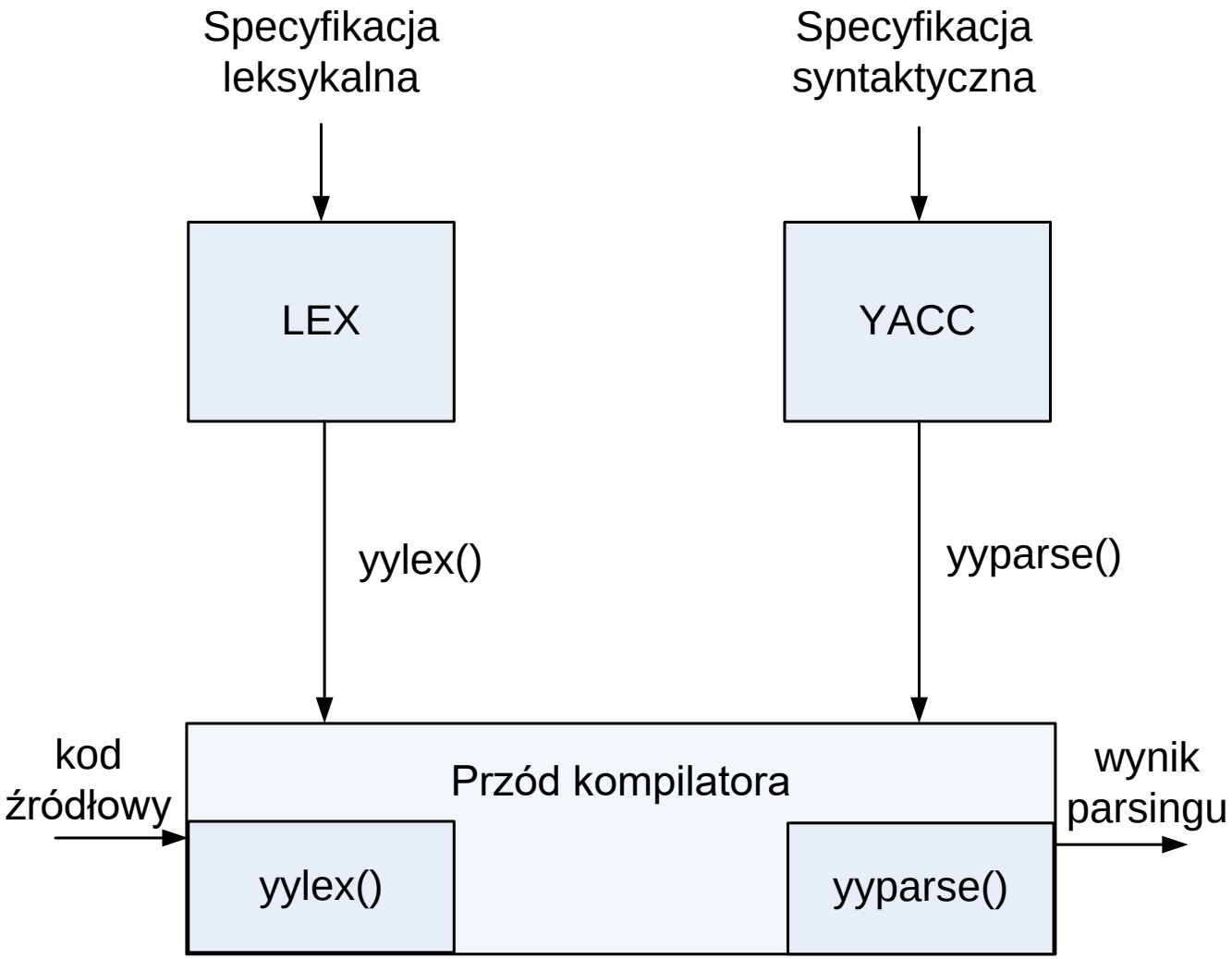




Generatory analizatorów leksykalnych



Generatory analizatorów leksykalnych i syntaktycznych



„Język” tworzenia specyfikacji leksykalnych

x	znak „x”
„x”	znak „x” nawet gdy x jest operatorem
\x	znak „x” nawet gdy x jest operatorem
[xy]	znak „x” lub „y”
[x-z]	znak „x” lub „y” lub „z”
[^x]	dowolny znak z wyjątkiem „x”
.	dowolny znak z wyjątkiem „\n”
^x	znak „x” na początku linii
<y>x	znak „x” gdy LEX jest w stanie początkowym „y”
x$	znak „x” na końcu linii
x?	0 lub 1 wystąpienie „x”
x*	0, 1, 2,… wystąpień „x”
x+	1, 2,… wystąpień „x”
x|y	„x” lub „y”
(x)	„x”
x/y	„x”, ale wtedy, gdy występuje przed „y”
{xx}	wyrażenie regularne opisane w sekcji deklaracji
x{m,n}	m, m+1,… ,n wystąpień „x”