

Teoretyczne podstawy informatyki

Wykład 11: Opis wzorców: wyrażenia regularne

Wyrażenia regularne (ang. regular expressions) stanowią algebraiczny sposób definiowania wzorców.

Wyrażenia regularne stanowią analogię do algebry wyrażeń arytmetycznych oraz do algebry relacyjnej.

Zbiór wzorców które można wyrazić w ramach algebry wyrażeń regularnych odpowiada dokładnie zbiorowi wzorców, które można opisać za pomocą automatów.

Operandy wyrażeń regularnych

Wyrażenia regularne posiadają pewne rodzaje operandów niepodzielnych (ang. atomic operands). Poniżej lista:

- (1) Znak
- (2) Symbol ε
- (3) Symbol \emptyset
- (4) Zmienna która może być dowolnym wzorcem zdefiniowanym za pomocą wyrażenia regularnego.

Wartość wyrażenia regularnego jest wzorcem składającym się ze zbioru ciągów znaków, który często określa się mianem języka (ang. language).

Język określony przez wyrażenia regularne E oznaczony będzie jako $L(E)$ lub określany jako język wyrażenia E .

Języki operandów niepodzielnych

Języki operandów niepodzielnych definiuje się w następujący sposób.

- (1) Jeżeli x jest dowolnym znakiem, to wyrażenie regularne x oznacza język $\{x\}$, to znaczy $L(x) = \{x\}$.
Należy zauważyć, że taki język jest zbiorem zawierającym jeden ciąg znakowy.
Ciąg ten ma długość 1 i jedyna pozycja tego ciągu określa znak x .
- (2) $L(\varepsilon) = \{\varepsilon\}$. Specjalny symbol ε jako wyrażenie regularne oznacza zbiór, którego jedynym ciągiem znakowym jest ciąg pusty, czyli ciąg o długości 0.
- (3) $L(\emptyset) = \emptyset$. Specjalny symbol \emptyset jako wyrażenie regularne oznacza zbiór pustych ciągów znakowych.

Istnieją trzy operatory w odniesieniu do wyrażeń regularnych.

Można je grupować przy użyciu nawiasów, podobnie jak ma to miejsce w przypadku innych znanych algebr. Definiuje się prawa kolejności działań oraz prawa łączności, które pozwalają na pomijanie niektórych par nawiasów – tak jak w przypadku wyrażeń arytmetycznych.

Operatory wyrażeń regularnych

Suma:

Symbol sumy (ang. union) oznacza się za pomocą symbolu $|$. Jeżeli R i S są dwoma wyrażeniami regularnymi, to $R | S$ oznacza sumę języków określanych przez R i S . To znaczy $L(R | S) = L(R) \cup L(S)$.

$L(R)$ i $L(S)$ są zbiorami ciągów znakowych, notacja sumowania jest uzasadniona

Złożenie:

Operator złożenia (ang. concatenation) nie jest reprezentowany przez żaden odrębny symbol. Jeżeli R i S są wyrażeniami regularnymi to RS oznacza ich złożenie. $L(RS)$, czyli język określony przez RS , jest tworzony z języków $L(R)$ i $L(S)$ w sposób następujący: Dla każdego ciągu znakowego r należącego do $L(R)$ oraz każdego ciągu znakowego s należącego do $L(S)$, ciąg rs , czyli złożenie ciągów r i s , należy do $L(RS)$.

Złożenie dwóch list takich jak ciągi znaków, jest wykonywane przez pobranie po kolei elementów pierwszej z nich i uzupełnienie ich po kolei elementami drugiej listy.

Domknięcie:

Operator domknięcia (ang. closure), jest to operator jednoargumentowy przyrostkowy. Domknięcie oznacza się za pomocą symbolu $*$, tzn. R^* oznacza domknięcie wyrażenia regularnego R .

Operator domknięcia ma najwyższy priorytet.

Efekt działania operatora domknięcia można zdefiniować jako „określenie występowania zera lub większej liczby wystąpień ciągów znaków w R ”.

Oznacza to że $L(R^*)$ składa się z:

(1) Ciągu pustego ε , który można interpretować jako brak wystąpień ciągów znaków w $L(R)$.

(2) Wszystkich ciągów znaków języka $L(R)$. Reprezentują one jedno wystąpienie ciągów znaków w $L(R)$.

(3) Wszystkich ciągów znaków języka $L(RR)$, czyli złożenia języka $L(R)$ z samym sobą. Reprezentują one dwa wystąpienia ciągów znaków w $L(R)$.

(4) Wszystkich ciągów znaków języka $L(RRR)$, $L(RRRR)$ i tak dalej, które reprezentują trzy, cztery i więcej wystąpień ciągów znaków w $L(R)$.

Nieformalnie można napisać: $R^* = \varepsilon \mid R \mid RR \mid RRR \mid \dots$

Wyrażenie po prawej stronie to nie jest wyrażenie regularne ponieważ zawiera nieskończoną liczbę wystąpień operatora sumy. Wszystkie wyrażenia regularne są tworzone ze skończonej liczby wystąpień operatorów.

Kolejność operatorów wyrażeń regularnych

Istnieje określona kolejność wykonywania trzech działań wyrażeń regularnych: sumy, złożenia oraz domknięcia.

Kolejność ta jest następująca:

- (1) Domknięcie (najwyższy priorytet)
- (2) Złożenie
- (3) Suma (najniższy priorytet)

Przykład: $a | bc^*d = (a | (b (c^*)) d)$

Prawa algebraiczne wyrażeń regularnych

Możliwe jest aby dwa wyrażenia regularne określały ten sam język.

Dwa wyrażenia regularne $R \mid S$ oraz $S \mid R$ określają ten sam język bez względu na postać wyrażeń regularnych jakie się podstawią za R i S . Wynika to z faktu że sumowanie jest przemienne.

Dwa wyrażenia regularne są równoważne (ang. equivalent) jeżeli $L(R) = L(S)$.

Tożsamość sumowania: $(\emptyset \mid R) \equiv (R \mid \emptyset) \equiv R$

Tożsamość złożenia: $\varepsilon R \equiv R \varepsilon \equiv R$

Anihilator złożenia: $\emptyset R \equiv R \emptyset \equiv \emptyset$

Przemienność sumowania: $(R \mid S) \equiv (S \mid R)$

Łączność sumowania: $((R \mid S) \mid T) \equiv (R \mid (S \mid T))$

Łączność złożenia: $((RS)T) \equiv (R(ST))$

Prawa algebraiczne wyrażeń regularnych

Lewostronna rozdzielność

złożenia względem sumowania: $(R (S \mid T)) \equiv (RS \mid RT)$

Prawostronna rozdzielność

złożenia względem sumowania: $((S \mid T) R) \equiv (SR \mid TR)$

Idempotencja sumowania:

$(R \mid R) \equiv R$

Równoważności operatora domknięcia:

$\emptyset^* \equiv \varepsilon$

$RR^* \equiv R^*R$

$(RR^* \mid \varepsilon) \equiv R^*$

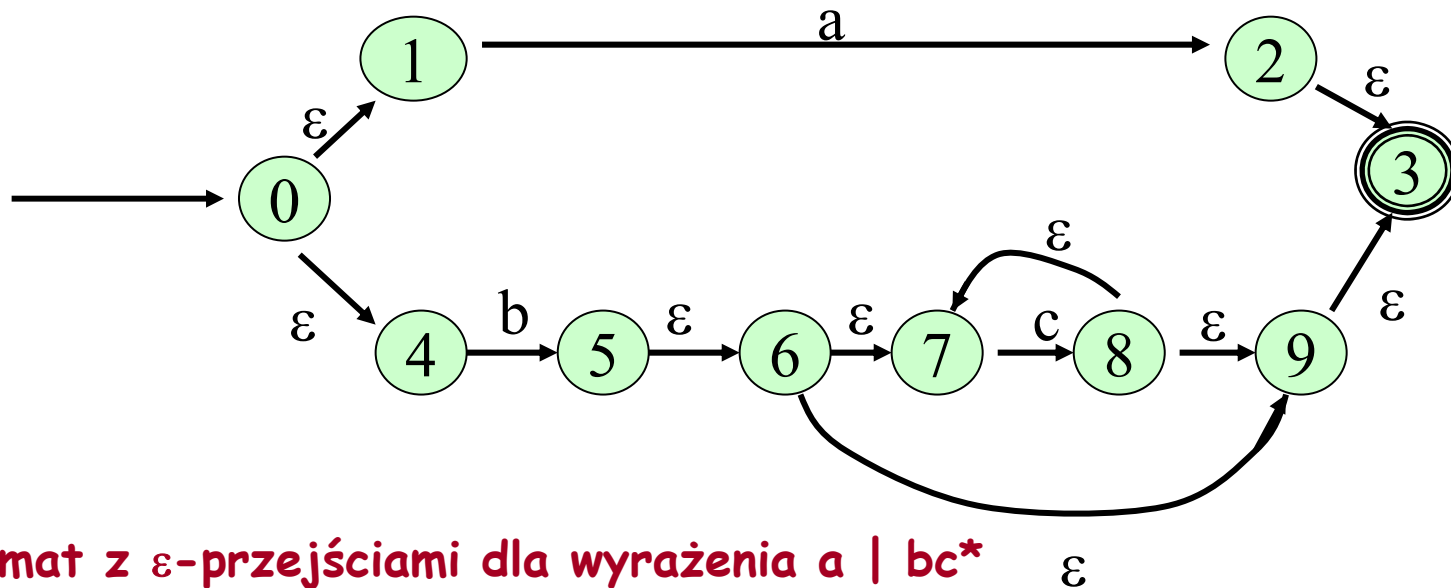
Od wyrażeń regularnych do automatów

Istnieje sposób na zamianę dowolnego wyrażenia regularnego na automat niedeterministyczny, a następnie przez użycie konstrukcji podzbiorów - zamiany takiego automatu na automat deterministyczny.

Istnieje także możliwość zamiany dowolnego automatu na wyrażenie regularne, którego język dokładnie odpowiada zbiorowi ciągów znaków akceptowanych przez automat. Stąd automaty i wyrażenia regularne dają te same możliwości opisywania języków.

Automaty z epsilon przejściami:

Należy rozszerzyć notację używaną w przypadku automatów w celu umożliwienia opisu krawędzi posiadających etykietę ϵ . Takie automaty wciąż akceptują ciąg znaków s wtedy i tylko wtedy, gdy ścieżka zaetykietowana ciągiem s wiedzie od stanu początkowego do stanu akceptującego. Symbol ϵ , ciąg pusty, jest „niewidoczny” w ciągach znaków, stąd w czasie konstruowania etykiety danej ścieżki w efekcie usuwa się wszystkie symbole ϵ i używa tylko rzeczywistych znaków.



Od wyrażeń regularnych do automatów z epsilon przejściami

Wyrażenie regularne zamienia się na automat przy użyciu algorytmu opracowanego na podstawie indukcji zupełnej względem liczby wystąpień operatorów w wyrażeniu regularnym.

Twierdzenie $S(n)$:

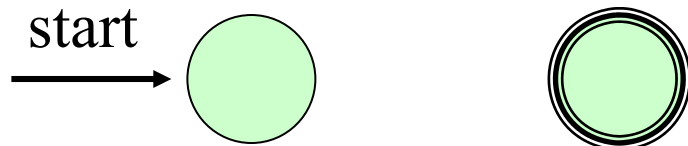
Jeżeli R jest wyrażeniem regularnym o n wystąpieniach operatorów i braku zmiennych jako operatorów niepodzielnych, to istnieje automat A z ε -przejściami, który akceptuje ciągi znaków należące do języka $L(R)$ i żadne inne.

Ponadto automat A :

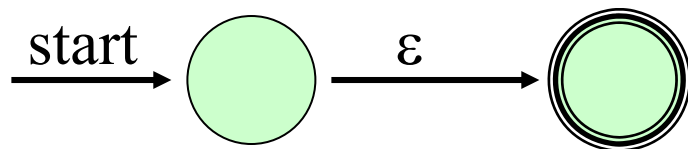
- (a) posiada tylko jeden stan akceptujący
- (b) nie posiada krawędzi wiodących do jego stanu początkowego
- (c) nie posiada krawędzi wychodzących z jego stanu akceptującego.

Podstawa:

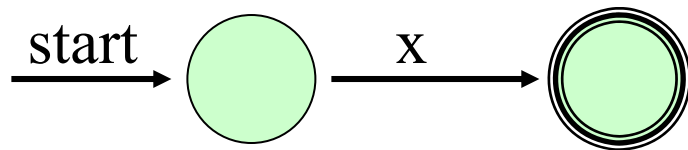
Jeżeli $n=0$, to R musi być operandem niepodzielnym, którym jest \emptyset , ε lub x dla pewnego symbolu x . Dla owych trzech przypadków można zaprojektować 2-stanowy automat, spełniający wymagania twierdzenia $S(0)$.



Automat dla \emptyset



Automat dla ε



Automat dla x

Automaty dla przypadków bazowych.

Każdy spełnia warunki (a) (b) (c) (patrz poprzednia strona).

Indukcja:

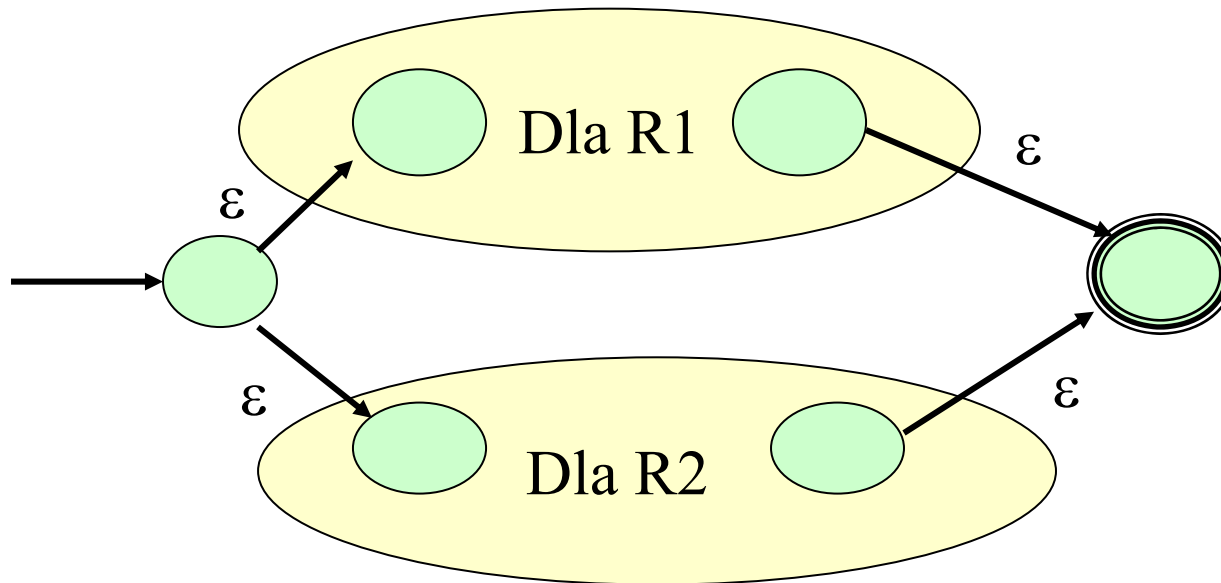
Zakładamy teraz, że $S(i)$ jest prawdziwe dla wszystkich $i \leq n$, to znaczy, że dla każdego wyrażenia regularnego R o maksymalnie n wystąpieniach istnieje automat spełniający warunek hipotezy indukcyjnej i akceptujący wszystkie ciągi znaków języka $L(R)$ i żadnych innych. Zajmiemy się tylko najbardziej zewnętrznym operatorem w R , co oznacza, że wyrażenie R może mieć tylko formę

$$R_1 \mid R_2, R_1 R_2, R_1^*$$

w zależności od tego czy ostatni użyty operator był operatorem sumy, złożenia lub domknięcia. Wyrażenie R_1, R_2 nie mogą posiadać więcej niż n operatorów.

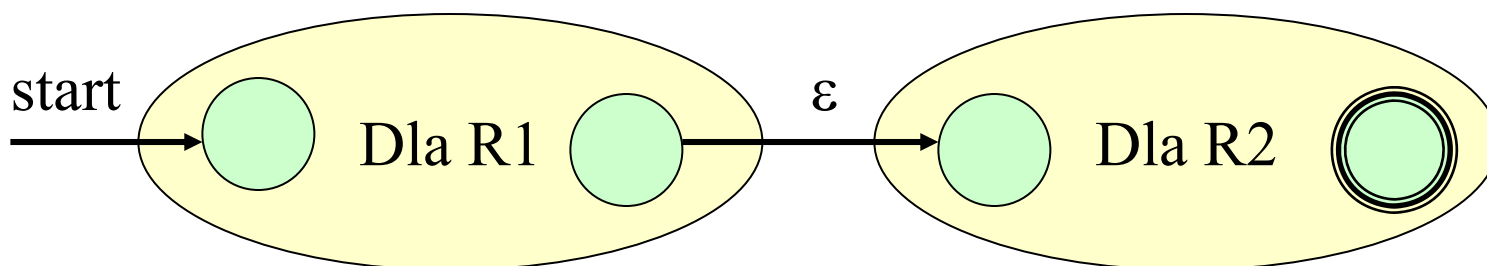
Przypadek 1: $R = R1 \mid R2$

Przechodzimy krawędzią zaetykietowaną symbolem ε do stanu początkowego automatu dla $R1$ lub automatu dla $R2$. Następnie przechodzimy do stanu akceptującego tego automatu, a później przejściem ε do stanu akceptującego automatu R .



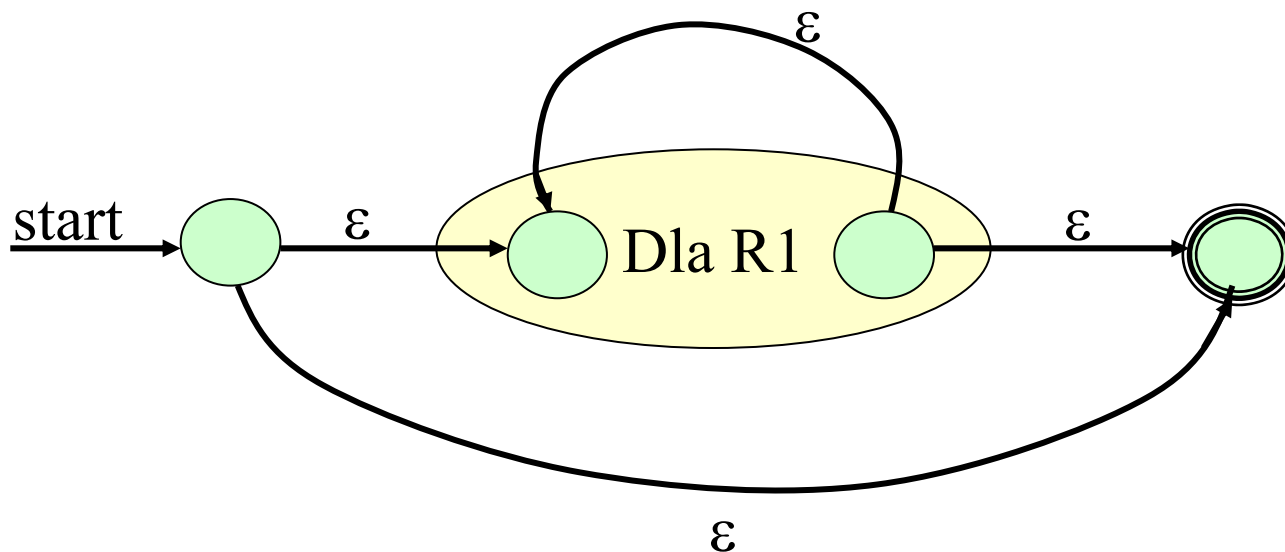
Przypadek 2: $R = R1 R2$

Automat posiada jako swój stan początkowy stan początkowy stan początkowy automatu dla wyrażenia R1, a jako swój stan akceptujący - stan akceptujący dla wyrażenia R2. Dodajemy także ε - przejście ze stanu akceptującego automatu dla wyrażenia R1 do stanu początkowego automatu dla wyrażenia R2. Stan akceptujący pierwszego automatu przestaje być stanem akceptującym, a stan początkowy drugiego automatu przestaje być stanem początkowym w skonstruowanym automacie.



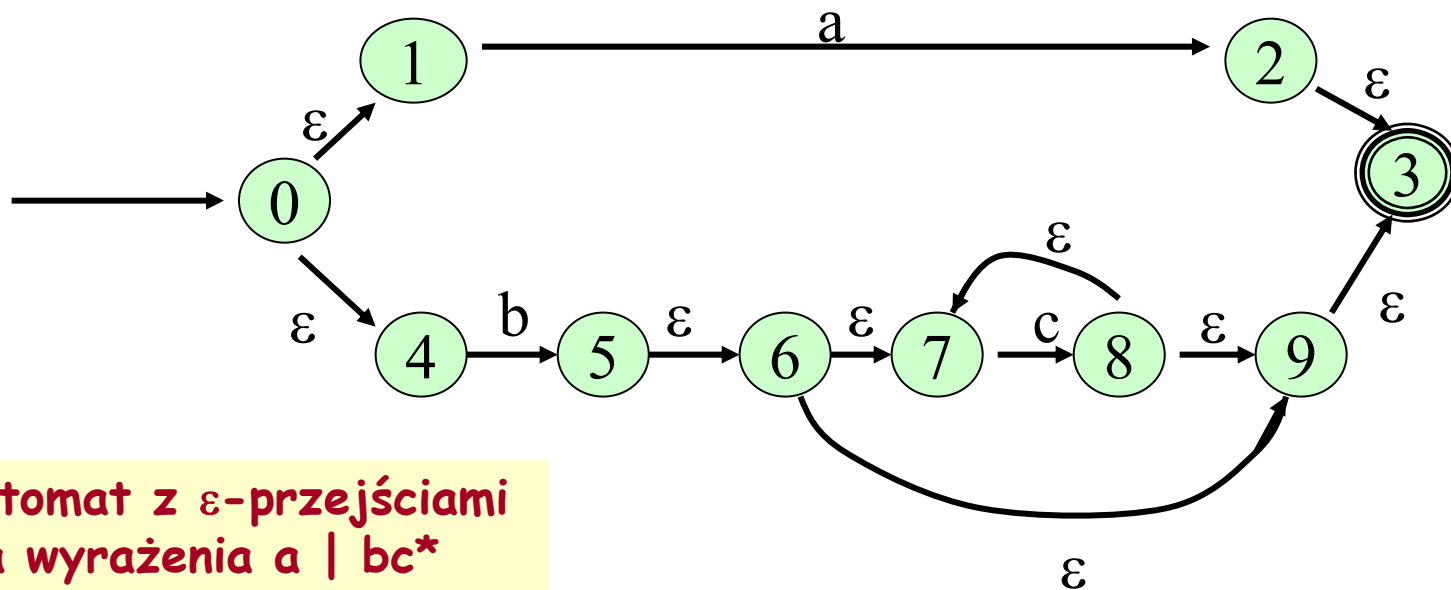
Przypadek 3: $R = R1^*$

Do automatu dla wyrażenia $R1$ dodajemy nowy stan początkowy i akceptujący. Stan początkowy posiada ϵ przejście do stanu akceptującego (a więc akceptowany jest ciąg ϵ) oraz do stanu początkowego automatu dla wyrażenia $R1$. Stan akceptujący automatu dla wyrażenia $R1$ otrzymuje ϵ -przejście z powrotem do swojego stanu początkowego oraz do stanu akceptującego automatu dla wyrażenia R . Stan początkowy i akceptujący automatu dla wyrażenia $R1$ nie są stanami początkowym i akceptującym konstruowanego automatu. Etykiety ścieżek odpowiadają ciągom należącym do języka $L(R1^*)$ czyli $L(R)$.



Eliminacja epsilon-przejęć

Jeżeli stanem bieżącym jest dowolny stan s automatu z ε -przejęciami, oznacza to że jednocześnie stanem bieżącym jest dowolny stan, do którego można się dostać z s w wyniku przejścia ścieżki zawierającej krawędzie zaetykietowane symbolem ε . Wynika to z faktu, że bez względu na to, jaki ciąg etykietuje wybraną ścieżkę prowadzącą do s , ten sam ciąg będzie także stanowił etykietę ścieżki rozszerzonej o ε -przejęcia.



**Automat z ε -przejęciami
dla wyrażenia $a \mid bc^*$**

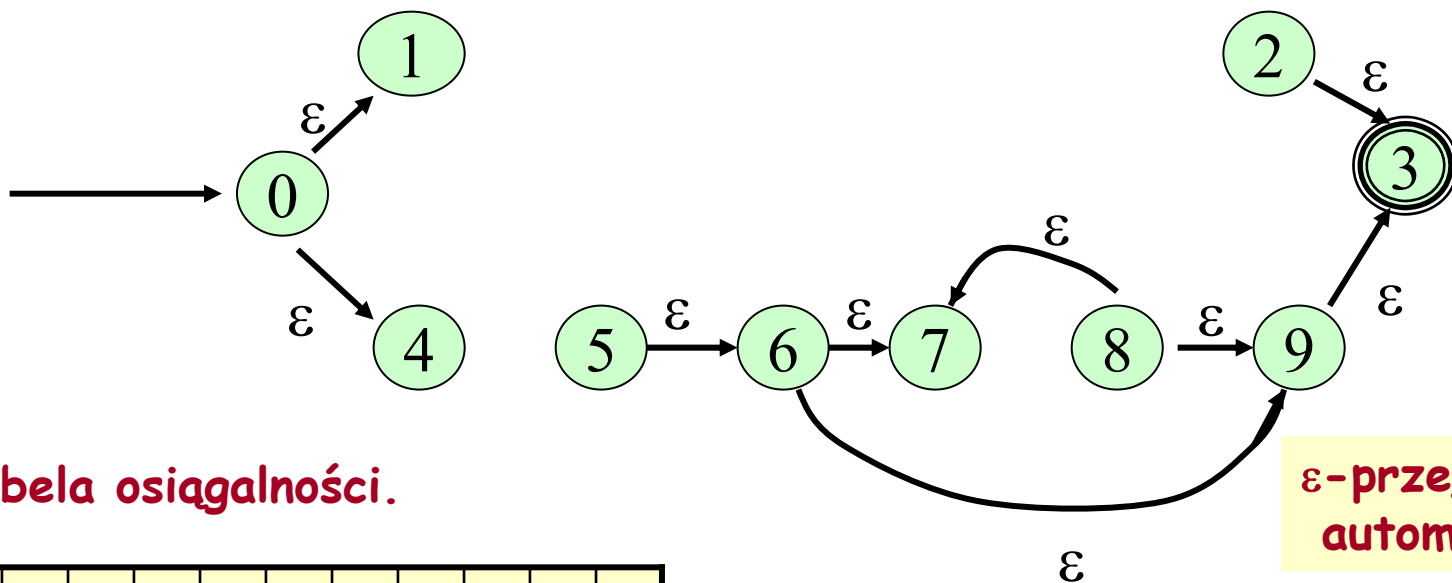


Tabela osiągalności.

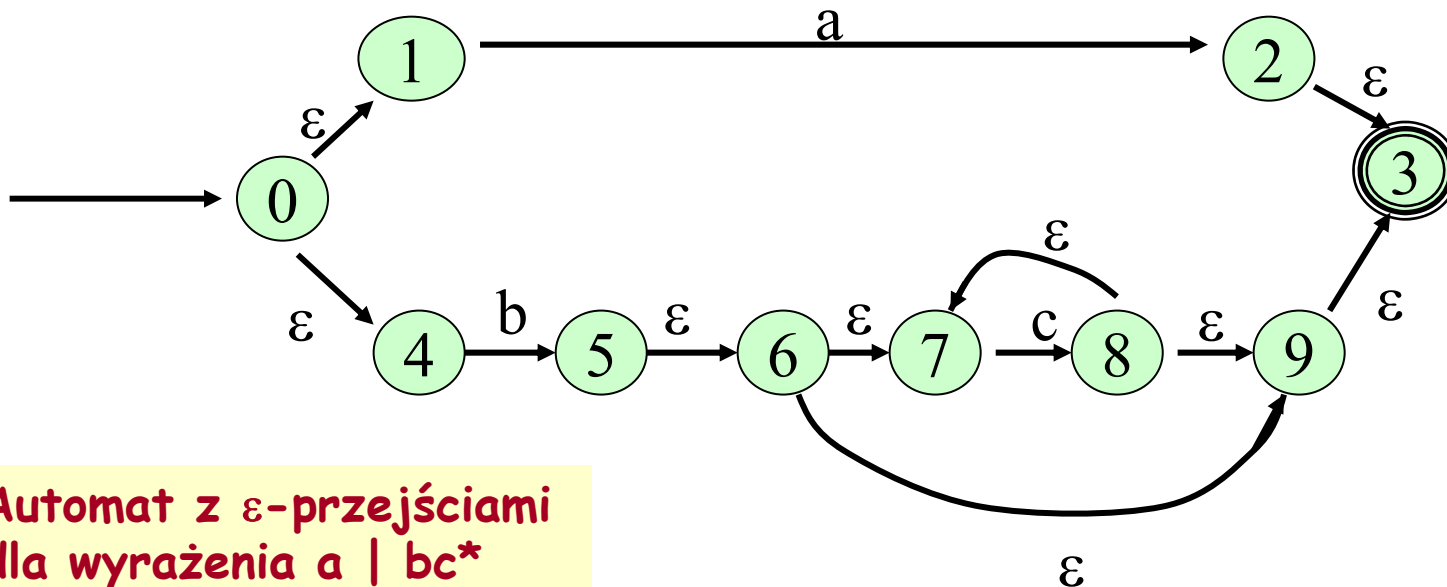
	0	1	2	3	4	5	6	7	8	9
0	1	1			1					
1		1								
2			1	1						
3				1						
4					1					
5				1		1	1	1		1
6				1			1	1		1
7				1				1		
8				1				1	1	1
9				1						1

ε-przejścia dla automatu z str.17.

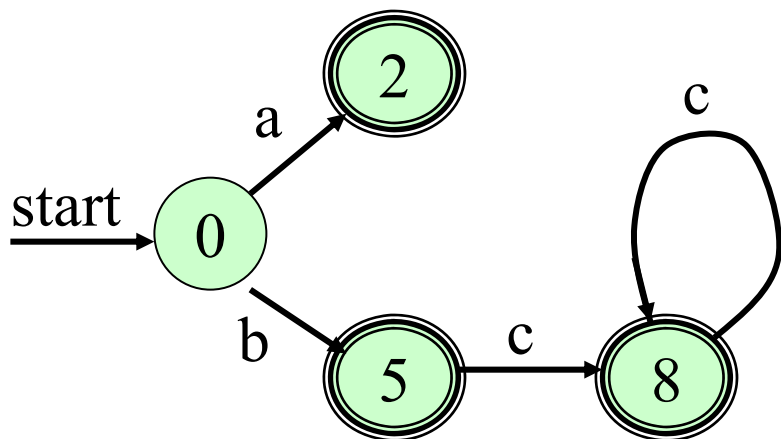
Dla grafu automatu usuwamy wszystkie ścieżki oznaczone rzeczywistymi etykietami. Przeprowadzamy przeszukiwanie w głąb pozostałego grafu.

Tabela osiągalności

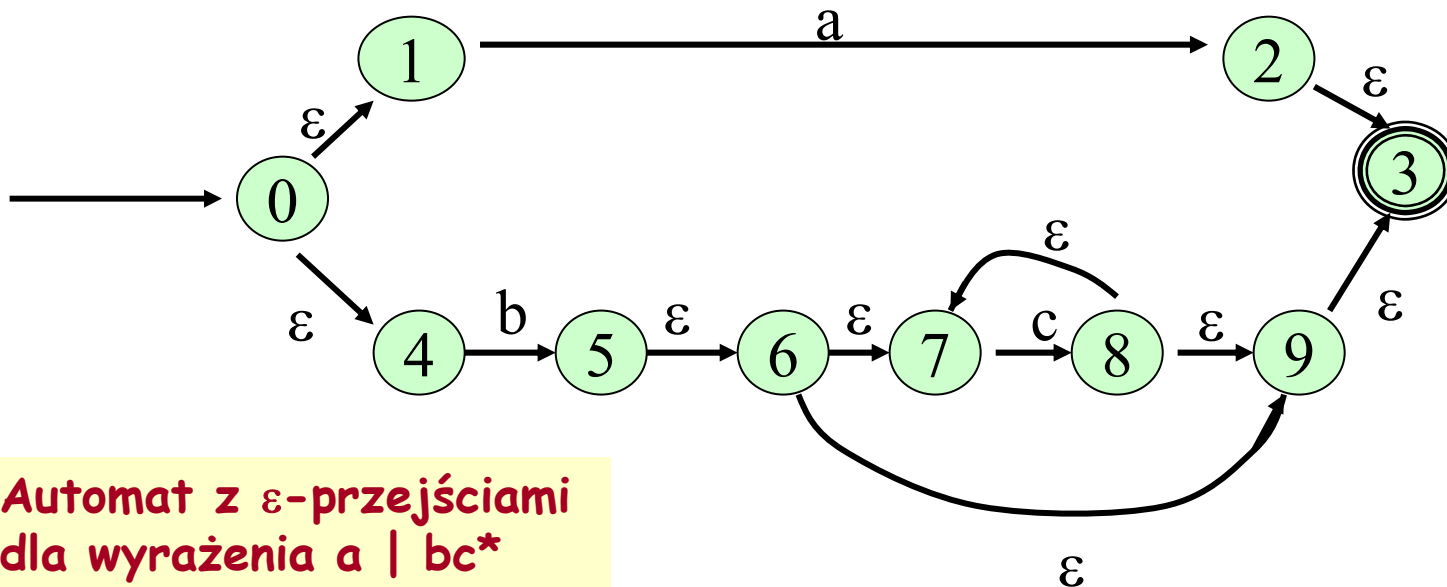
Posiadając informacje o osiągalności, możemy skonstruować równoważny automat nie posiadający ε -przejsć. Stany do których przechodzi się krawędziami zaetykietowanymi symbolami rzeczywistymi nazywamy stanami istotnymi. W nowym automacie chcemy zawrzeć tylko te stany oraz stany początkowe dla zbioru jego własnego zbioru stanów. Należy też zdecydować które stany będą stanami akceptującymi.



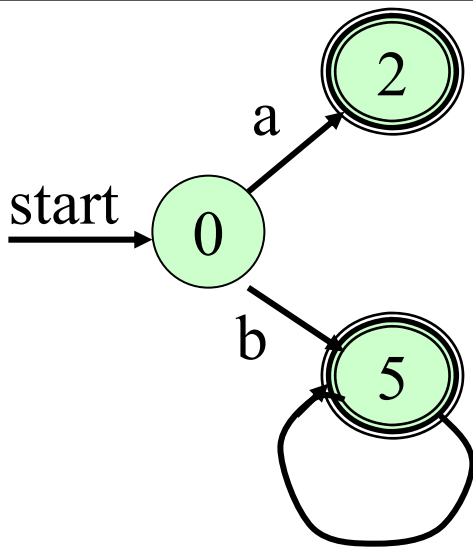
Automat z ϵ -przejściami dla wyrażenia $a \mid bc^*$



Automat skonstruowany na podstawie eliminacji ϵ -przejęć. Automat akceptuje wszystkie ciągi języka $L(a \mid bc^*)$.



Automat z ϵ -przejściami dla wyrażenia $a \mid bc^*$

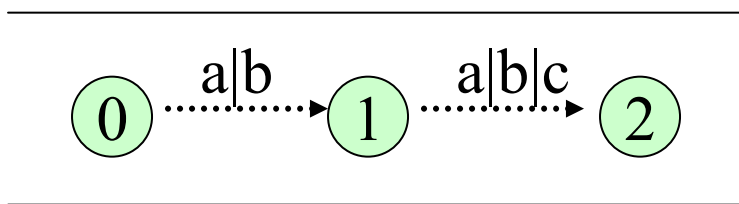


Prostsza wersja automatu dla języka $L(a \mid bc^*)$. Wykorzystano obserwację że stany (5) i (8) są równoważne i mogą zostać połączone.

Od automatów do wyrażeń regularnych.

Dla każdego automatu istnieje A wyrażenie regularne, którego język dokładnie odpowiada zbiorowi ciągów znaków akceptowanych przez automat A. Konstrukcja polega na eliminacji stanów automatów. Etykiety krawędzi, które są zbiorami znaków, zastępuje się bardziej skomplikowanymi wyrażeniami regularnymi. Jeżeli dla pewnej krawędzi istnieje etykieta $\{x_1, x_2, \dots, x_n\}$, zastępuje się ją wyrażeniem regularnym $x_1 \mid x_2 \mid \dots \mid x_n$, które reprezentuje ten sam zbiór symboli. Etykietę ścieżki można postrzegać jako złożenie wyrażeń regularnych opisujących krawędzie tej ścieżki, lub jako język zdefiniowany przez złożenie tych wyrażeń.

Przykład: Wyrażenia regularne etykietujące krawędzie to $a \mid b$ i $a \mid b \mid c$. Zbiór znaków etykietujących tę ścieżkę składa się z tych, które występują w języku zdefiniowanym przez wyrażenia regularne: $(a \mid b)(a \mid b \mid c)$ czyli $\{aa, ab, ac, ba, bb, bc\}$.

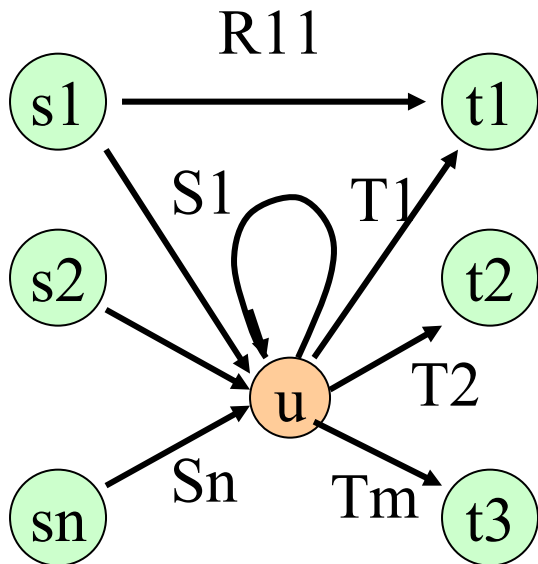


Ścieżka z wyrażeniami regularnymi jako etykietami. Etykieta ścieżki należy do wyrażeń regularnych utworzonych w wyniku złożenia.

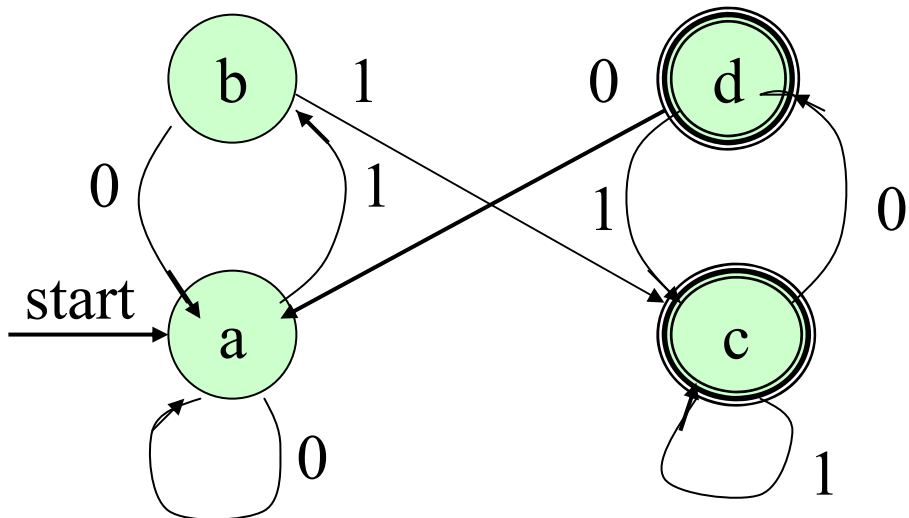
Konstrukcja eliminacji stanów.

Kluczowym etapem konwersji z postaci automatu na wyrażenie regularne jest eliminacja stanów. Chcemy wyeliminować stan u , ale chcemy zachować etykiety krawędzi występujące w postaci wyrażień regularnych, tak aby zbiór etykiet ścieżek między dowolnymi pozostałymi stanami nie uległ zmianie.

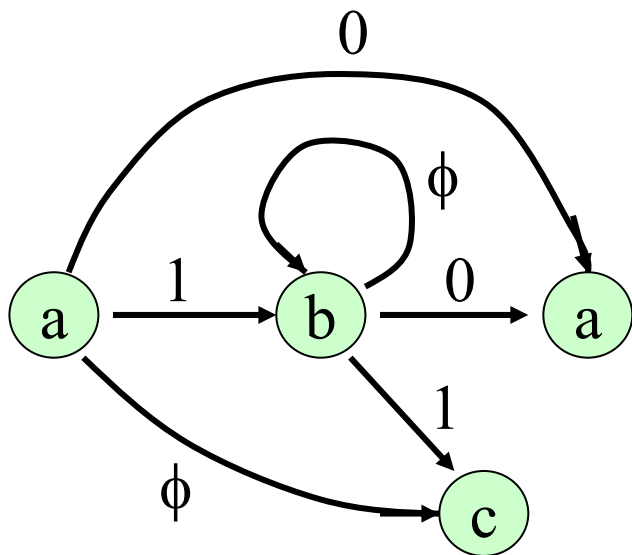
Poprzednik stanu u to s_1, s_2, \dots, s_n zaś następniki stanu u to t_1, t_2, \dots, t_n (mogą też istnieć stany wspólne).



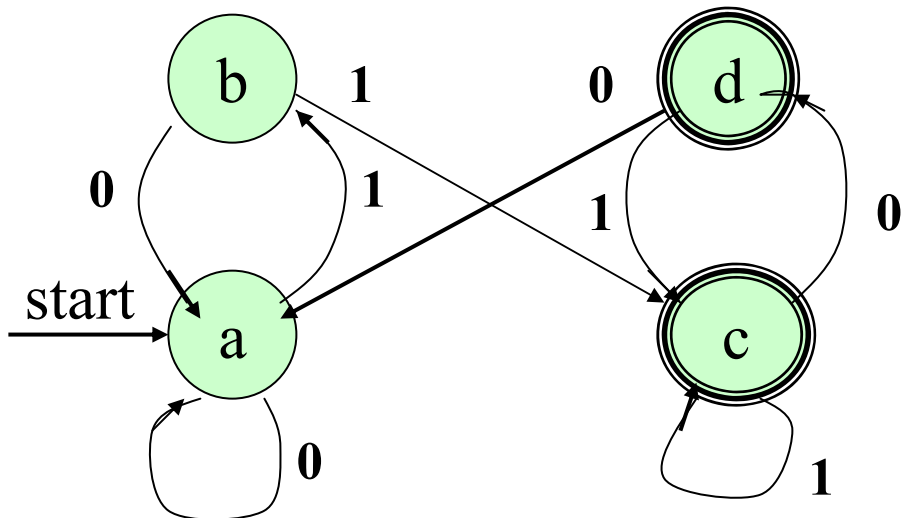
Zbiór ciągów znaków etykietujących ścieżki wiodące z wierzchołków s_i do wierzchołka u , łącznie z ścieżkami biegnącymi kilkakrotnie wokół pętli $u \rightarrow u$, oraz z wierzchołka u do wierzchołka t_j , jest opisany za pomocą wyrażenia regularnego $S_i U^* T_j$. Po eliminacji wierzchołka u należy zastąpić etykietę R_{ij} , czyli etykietę krawędzi $s_i \rightarrow t_j$ przez etykietę $R_{ij} \mid S_i U^* T_j$.



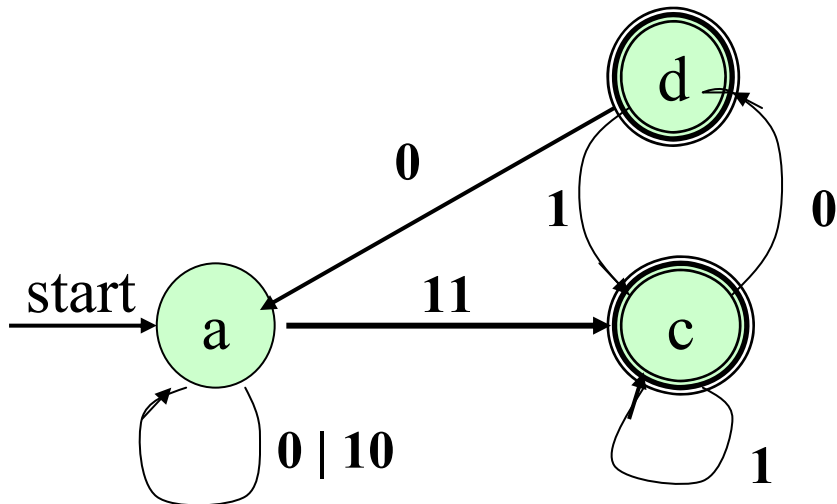
**Automat skończony
filtra odbijającego.**



**Stan b, jego poprzedniki
oraz następniki.**



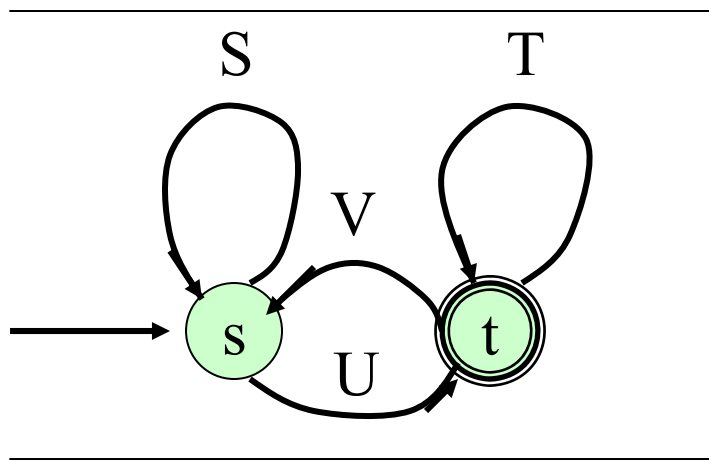
Automat skończony filtra odbijającego.



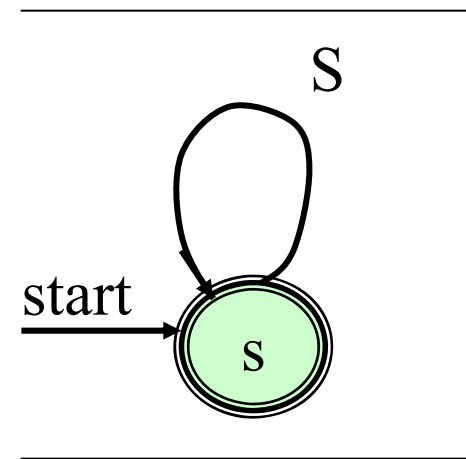
Automat skończony filtra odbijającego po wyeliminowaniu stanu b.

Redukcja zupełna automatu

W celu otrzymania wyrażenia regularnego określającego wszystkie ciągi znaków akceptowane przez automat A i żadne inne, należy rozpatrzyć po kolei każdy stan akceptujący t automatu A . Każdy ciąg znaków akceptowany przez automat A jest akceptowany dlatego, że etykietuje on ścieżkę wiodącą ze stanu początkowego s do pewnego stanu akceptującego t .



**Automat zredukowany do dwóch stanów.
Wyrażenie regularne: $S^*U(T | VS^*U)^*$**



**Automat posiadający jedynie stan początkowy.
Wyrażenie regularne: S^***

Posumowanie

⇒ Trzy sposoby określania języków dają te same możliwości wyrażania:

- (1) Istnieje pewien automat deterministyczny, akceptujący wszystkie ciągi znaków języka L i żadne inne.
- (2) Istnieje pewien, być może niedeterministyczny automat, akceptujący wszystkie ciągi znaków języka L i żadne inne.
- (3) Język L jest językiem $L(R)$ pewnego wyrażenia regularnego.

Konstrukcja podzbiorów pokazuje, że (2) implikuje (1).

Stwierdzenie (1) implikuje (2) gdyż automat deterministyczny jest szczególnym rodzajem automatu niedeterministycznego.

Przechodzenie od wyrażeń regularnych do automatów oznacza że (3) implikuje (2).

Przechodzenie od automatów do wyrażeń regularnych oznacza że (2) implikuje (3).

⇒ Automaty deterministyczne mogą być używane jako podstawa programów, które rozpoznają wiele różnych rodzajów wzorców w ciągach znaków.

⇒ Wyrażenia regularne są często przydatną konwencją tonacyjną opisywania wzorców. Istnieje formalizm praw algebraicznych dla wyrażeń regularnych.