AGH

AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

# Automaty skończone

## Teoria automatów i języków formalnych

**Dr inż. Janusz Majewski**
**Katedra Informatyki**

AGH

# Przykład (1)

Rozważamy język nad alfabetem binarnym $\Sigma = \{0, 1\}$ składający się z łańcuchów zero-jedynkowych o tej własności, że liczba zer w każdym łańcuchu jest parzysta i liczba jedynek w każdym łańcuchu też jest parzysta. Wszystkie łańcuchy binarne możemy podzielić na cztery grupy:

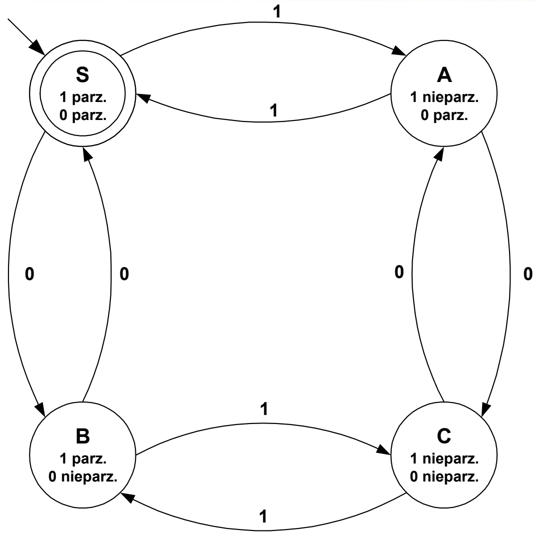
- S – łańcuchy z parzystą liczbą jedynek i parzystą liczbą zer,
- A – łańcuchy z parzystą liczbą jedynek i nieparzystą liczbą zer,
- B – łańcuchy z nieparzystą liczbą jedynek i parzystą liczbą zer,
- C – łańcuchy z nieparzystą liczbą jedynek i nieparzystą liczbą zer.

Analizujemy łańcuch zero-jedynkowy symbol po symbolu od lewej strony. Przed rozpoczęciem analizy jesteśmy w grupie S (łańcuch pusty zawiera zero jedynek i tyleż zer, więc liczba jedynek i liczba zer w tym łańcuchu są parzyste). Jeśli pierwszym symbolem jest jedynka – przechodzimy do grupy A (wtedy liczba jedynek jest nieparzysta, a liczba zer jest dalej parzysta), zaś jeśli pierwszym symbolem jest zero – przechodzimy do grupy B (wtedy liczba zer jest nieparzysta, a liczba jedynek jest dalej parzysta). Zapisujemy to w postaci produkcji:

$S \to 1A \mid 0B$

Dalej analizujemy podobnie kolejne symbole łańcucha. Np. jeśli jesteśmy w grupie A i przeczytamy zero – przechodzimy do grupy C, w której zarówno liczba jedynek, jak i zer są nieparzyste (odpowiedni zapis: $A \to 0C$)

# Przykład (2)




# Przykład (3)

Wreszcie, gdy przeczytaliśmy cały łańcuch, sprawdzamy, czy zatrzymaliśmy się w grupie S. Jeśli tak – badany łańcuch spełnia nałożony nań warunek parzystej liczby jedynek i parzystej liczby zer. Wówczas należy wyeliminować z wyprowadzenia symbol S (odpowiedni zapis: $S \to \varepsilon$). Ostatecznie gramatyka naszego języka ma postać:

$S \to 1A \mid 0B \mid \varepsilon$

$A \to 1S \mid 0C$

$B \to 1C \mid 0S$

$C \to 1B \mid 0A$

Opisana procedura i zamieszczony wcześniej rysunek grafu ilustrują właściwie nie tyle proces konstruowania gramatyki dla pewnego języka, co proces odpowiadania na pytanie: czy dany łańcuch jest słowem należącym do danego języka.

## Przykład – automat skończony

Jest to pewien (w naszym przypadku deterministyczny) algorytm postępowania, polegający na czytaniu badanego łańcucha symbol po symbolu i przechodzenia od jednego stanu do drugiego. Stany reprezentowane są przez kółka (węzły grafu), zaś przejścia pomiędzy stanami, to skierowane krawędzie, opisane (etykietowane) odpowiednimi symbolami alfabetu, z którego pochodzą symbole łańcucha. Jeden ze stanów jest wyróżniony jako stan początkowy (na rysunku – jest to stan oznaczony krótką strzałką dochodząc do niego z zewnątrz). Od tego stanu zawsze rozpoczynamy „wędrówkę" po grafie. Niektóre stany są traktowane jako stany końcowe – akceptujące (są one zaznaczone kółkami rysowanymi podwójną linią). Jeśli w trakcie naszej „wędrówki" po grafie zatrzymamy się w takim stanie, przeczytawszy wejście do końca, to akceptujemy badany łańcuch, jeśli zatrzymamy się w stanie nie będącym stanem końcowym – nie akceptujemy analizowanego łańcucha. Zatrzymanie się w naszym przypadku może być tylko spowodowane przeczytaniem badanego słowa do końca i stwierdzeniem, że już nic nie pozostało do przeczytania.

Opisany powyżej algorytm nosi nazwę automatu skończonego (a dokładnie deterministycznego i zupełnego automatu skończonego).

## Definicja automatu skończonego

Automatem skończonym nazywamy piątkę:

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

gdzie:

$\Sigma$ – zbiór symboli terminalnych (alfabet wejściowy)

$Q$ – zbiór stanów, $\#Q < \infty$

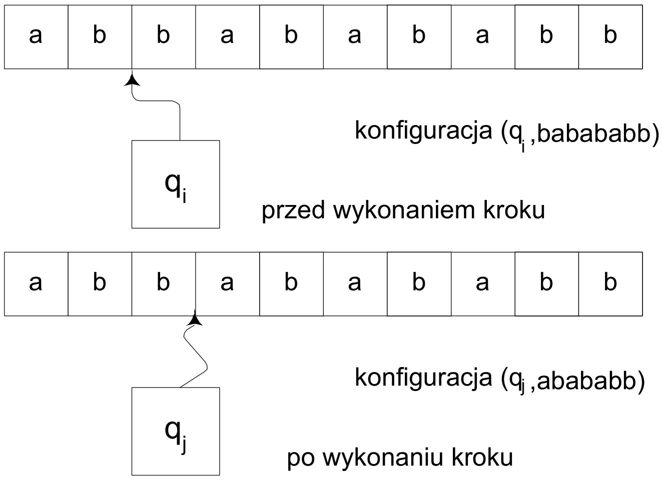
$F \subseteq Q$ – zbiór stanów końcowych

$q_0 \in Q$ – stan początkowy

$\delta$ – funkcja przejścia

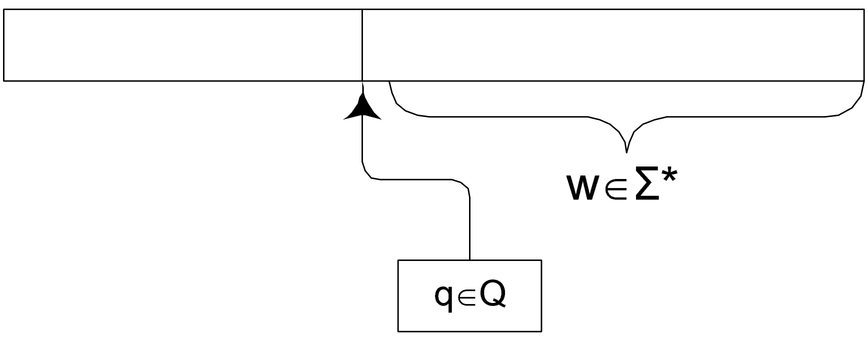
$\delta: Q \times (\Sigma \cup \{\varepsilon\}) \mapsto 2^Q$

# Konfiguracja automatu (1)



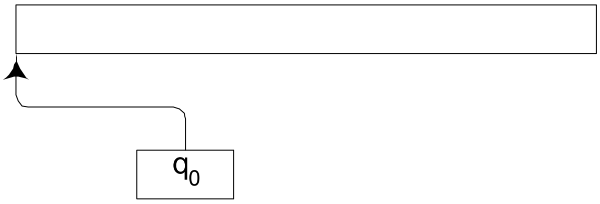

# Konfiguracja automatu (2)

Konfiguracja automatu to dwójka: (q, w), gdzie *q* jest aktualnym stanem, zaś *w* jest nieprzeczytaną przez automat częścią słowa zapisanego na taśmie wejściowej

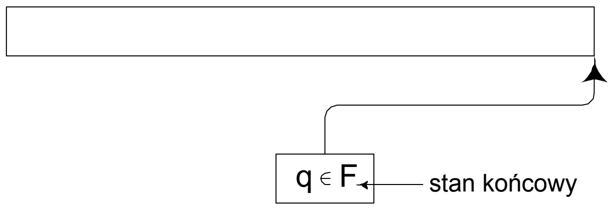


# Konfiguracja automatu (3)

Konfiguracja początkowa:




Konfiguracja końcowa akceptująca:




# Wyprowadzenie bezpośrednie i pośrednie

Wyprowadzenie bezpośrednie:

$(q, ax) \vdash_A (q', x)$

gdzie: $q,q' \in Q$, $a \in (\Sigma \cup \{\varepsilon\})$, $x \in \Sigma^*$ , $q' \in \delta(q,a)$

Wyprowadzenia pośrednie $\vdash_A^+$ i $\vdash_A^*$ są odpowiednio przechodnim oraz przechodnim i zwrotnym domknięciem relacji wyprowadzenia bezpośredniego $\vdash_A$:

$(q, w) \vdash_A^+ (q', w') \Leftrightarrow \exists\, p_0, \ldots, p_n$ ($p_i$ – konfiguracje), takie że:

$q_0=(q, w)$,

$q_n= (q', w')$,

$p_i \vdash_A p_{i+1}$ dla $i=0, 1, \ldots, n-1$

$(q, w) \vdash_A^* (q' , w') \Leftrightarrow (q, w) \vdash_A^+ (q', w') \lor (q, w)=(q', w')$

# Akceptacja języka przez automat

$x \in \Sigma^*$ jest słowem akceptowanym przez automat A (skończony) $\Leftrightarrow$

$(\exists q \in F)\ ((q_0, x) \vdash_A^* (q, \varepsilon))$

Język L jest akceptowany przez automat A (co oznaczamy L(A) ) $\Leftrightarrow$

$L = L(A) = \{ x \in \Sigma^* \mid x \text{ jest akceptowane przez A} \}$

Konfiguracja blokująca:

$(q,w)$ jest blokująca $\Leftrightarrow \neg\, (\exists (q', w'))\ \ ((q,w) \vdash_A (q', w'))$

# Przykład (1)

Przykład: Automat niedeterministyczny akceptujący język regularny opisany wyrażeniem regularnym **(a|b)*abb**

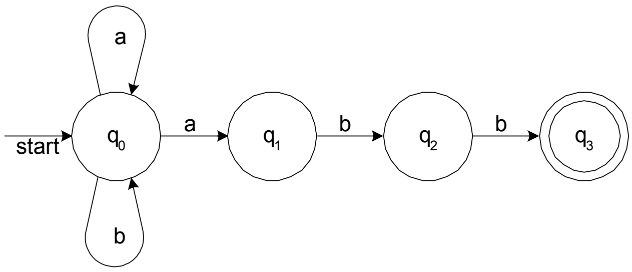
$\Sigma = \{a,b\}$

$Q = \{q_0, q_1, q_2, q_3\}$

$F = \{q_3\}$

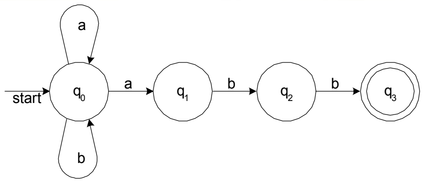
$q_0$ – stan początkowy

$\delta$ - funkcja przejścia:




| stan | a | b |
|---|---|---|
| $q_0$ | $\{q_0, q_1\}$ | $\{q_0\}$ |
| $q_1$ | $\varnothing$ | $\{q_2\}$ |
| $q_2$ | $\varnothing$ | $\{q_3\}$ |
| $q_3$ | $\varnothing$ | $\varnothing$ |

# Przykład (2)




Analizowane słowo : *aabb* $\in$ L( **(a|b)*abb** )

Możliwe wyprowadzenia:

- ( $q_0$ , aabb ) $\vdash$ ( $q_0$, abb ) $\vdash$ ( $q_1$, bb ) $\vdash$ ( $q_2$, b ) $\vdash$ ( $q_3$, $\varepsilon$) – konfiguracja końcowa akceptująca)
- ( $q_0$ , aabb ) $\vdash$ ( $q_0$, abb ) $\vdash$ ( $q_0$, bb ) $\vdash$ ( $q_0$, b ) $\vdash$ ( $q_0$, $\varepsilon$) – konfiguracja blokująca, bo $q_0 \notin F$
- ( $q_0$ , aabb ) $\vdash$ ( $q_1$, abb ) – konfiguracja blokująca, słowo nie zostało przeczytane do końca

Automat nie jest deterministyczny. Jednak istnieje wyprowadzenie (ciąg kroków), które doprowadza do konfiguracji akceptującej, więc zgodnie z definicją automat akceptuje to słowo.

# Własności automatów skończonych

Automat skończony A jest zupełny $\Leftrightarrow$
$(\forall a \in \Sigma)\ (\forall q \in Q)\ \ (\#\delta(q,a) \geq 1)$

Automat skończony A jest deterministyczny $\Leftrightarrow$

(i) $(\forall q \in Q)\ (\#\delta(q,\varepsilon) = 0)$ oraz

(ii) $(\forall a \in \Sigma)\ (\forall q \in Q)\ \ (\#\delta(q,a) \leq 1)$

Automat skończony A jest deterministyczny i zupełny $\Leftrightarrow$

(i) $(\forall q \in Q)\ \ (\#\delta(q,\varepsilon) = 0)$ oraz

(ii) $(\forall a \in \Sigma)\ (\forall q \in Q)\ \ (\#\delta(q,a) = 1)$

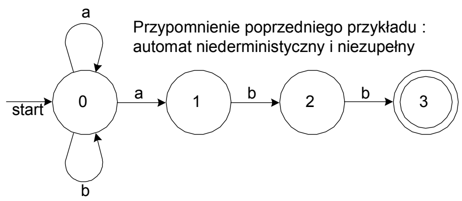
Automat skończony zupełny nazywamy automatem Rabina-Scotta.

Automat skończony, deterministyczny i zupełny nazywamy deterministycznym automatem Rabina-Scotta

# Przykład

Przykład: Deterministyczny zupełny automat akceptujący język opisany wyrażeniem regularnym **(a|b)*abb**

Przypomnienie poprzedniego przykładu : automat niederministyczny i niezupełny




$\Sigma = \{a, b\}$

$Q = \{0, 1, 2, 3\}$

$F = \{3\}$

$q_0 = 0$

$\delta$ - funkcja przejścia:

| Stan | a | b |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 1 | 2 |
| 2 | 1 | 3 |
| 3 | 1 | 0 |