Maszyna Turinga (Algorytmy Część III)

wer. 11 z drobnymi modyfikacjami!

Wojciech Myszka

2023-11-28 07:31:43 +0100

Politechnika Wrocławska

Upraszczanie danych

- Komputery są coraz szybsze i sprawniejsze.
- Na potrzeby rozważań naukowych — potrzebne są modele uogólniające i jak najprostsze...
- Struktura pamięci komputera jest bardzo prosta — **liniowa**

Upraszczanie danych

- Zmienne i wektory — bardzo łatwo dają się „zlinearyzować”
- Tablice dwu (ale i wielowymiarowe) stosunkowo łatwo można zapisać w taki sposób: trzeba się tylko umówić czy zapisujemy dane wierszami czy kolumnami

Przykład

Upraszczanie danych

- Zmienne i wektory — bardzo łatwo dają się „zlinearyzować”
- Tablice dwu (ale i wielowymiarowe) stosunkowo łatwo można zapisać w taki sposób: trzeba się tylko umówić czy zapisujemy dane wierszami czy kolumnami

Przykład

```
11  12  13
21  22  23
31  32  33
41  42  43
```

Upraszczanie danych

- Zmienne i wektory — bardzo łatwo dają się „zlinearyzować”
- Tablice dwu (ale i wielowymiarowe) stosunkowo łatwo można zapisać w taki sposób: trzeba się tylko umówić czy zapisujemy dane wierszami czy kolumnami

Przykład

11 12 13
21 22 23
31 32 33
41 42 43

może być zapisane tak: 11; 12; 13 * 21; 22; 23 * 31; 32; 33 * 41; 42; 43 *

Upraszczanie danych

- Zmienne i wektory — bardzo łatwo dają się „zlinearyzować”
- Tablice dwu (ale i wielowymiarowe) stosunkowo łatwo można zapisać w taki sposób: trzeba się tylko umówić czy zapisujemy dane wierszami czy kolumnami

Przykład

11 12 13
21 22 23
31 32 33
41 42 43

może być zapisane tak: 11; 12; 13 * 21; 22; 23 * 31; 32; 33 * 41; 42; 43 *
albo tak: 11; 21; 31; 41 * 12; 22; 32; 42 * 13; 23; 33; 43 *

Upraszczanie danych

- Zmienne i wektory — bardzo łatwo dają się „zlinearyzować"
- Tablice dwu (ale i wielowymiarowe) stosunkowo łatwo można zapisać w taki sposób: trzeba się tylko umówić czy zapisujemy dane wierszami czy kolumnami

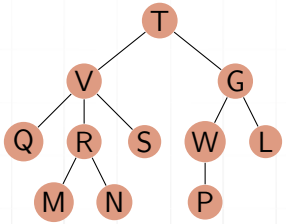
Przykład

11 12 13
21 22 23
31 32 33
41 42 43

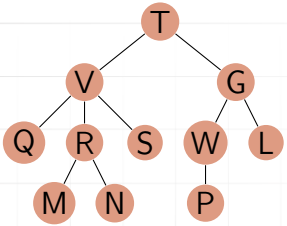
może być zapisane tak: 11; 12; 13 * 21; 22; 23 * 31; 32; 33 * 41; 42; 43 *
albo tak: 11; 21; 31; 41 * 12; 22; 32; 42 * 13; 23; 33; 43 *
albo tak: {{11, 21, 31, 41}, {12, 22, 32, 42}, {13, 23, 33, 43}} czyli jako lista!

Upraszczanie danych

- Podobnie będzie i ze stosem i z kolejką
- Baza danych — to właściwie tak bardziej rozbudowana tablica; ale nie znaczy że będzie łatwo...
- Co z drzewem?

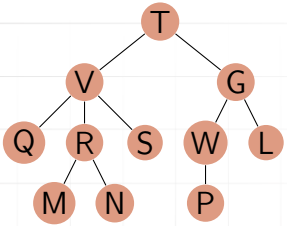


Upraszczanie danych

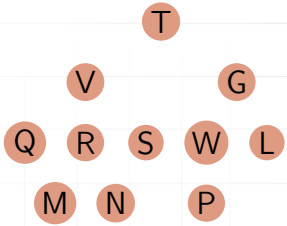


Pokazane drzewo można próbować zapisać tak: T ** V; G ** Q; R; S; W; L ** M; N; P** (gwiazdki oznaczają tu koniec kolejnych „poziomów" drzewa). Ale jedno co można odtworzyć to...

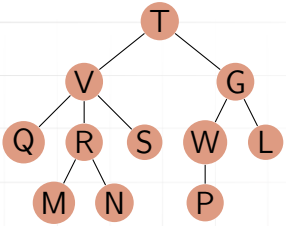
Upraszczanie danych



Pokazane drzewo można próbować zapisać tak:
T ** V; G ** Q; R; S; W; L ** M; N; P**
(gwiazdki oznaczają tu koniec kolejnych „poziomów" drzewa). Ale jedno co można odtworzyć to...



Upraszczanie danych



Jeżeli jednak trochę zapis skomplikować to można uzyskać coś więcej:
(T) (V; G) (Q; R; S) (W; L) () (M; N) () (P) ()
powyższy zapis jest bardzo zwarty, ale też trochę skomplikowany: nawiasy grupują **tylko** węzły będące potomkami tego samego węzła. Poziomy drzewa nie są zaznaczane w żaden specjalny sposób i muszą być wyliczane.
Uwaga: jest to zapis listowy!

Upraszczanie danych — teza

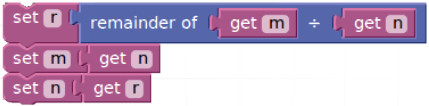
Przyjmujemy, że **każdą** strukturę danych można zapisać w postaci liniowej, na przykład na odpowiednio długiej „taśmie" złożonej z „krateczek" w których zapisane są pojedyncze dane.

Upraszczanie programu

- Czym jest program (algorytm) komputerowy?

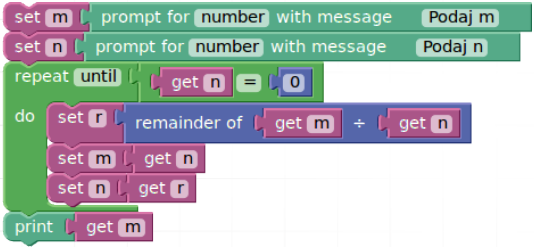
Upraszczanie programu

- Czym jest program (algorytm) komputerowy?



Upraszczanie programu

- Czym jest program (algorytm) komputerowy?



Upraszczanie programu

- Czym jest program (algorytm) komputerowy?
  rodzaj „struktury sterującej" określającej w jaki sposób i w jakiej kolejności przetwarzane są dane wejściowe

Upraszczanie programu

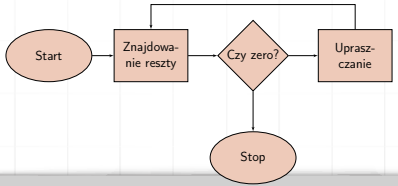
- Czym jest program (algorytm) komputerowy?
  rodzaj „struktury sterującej" określającej w jaki sposób i w jakiej kolejności przetwarzane są dane wejściowe
- Każdy algorytm jest skończony.

Upraszczanie programu

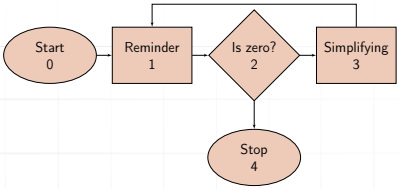
- Czym jest program (algorytm) komputerowy?
  rodzaj „struktury sterującej" określającej w jaki sposób i w jakiej kolejności przetwarzane są dane wejściowe
- Każdy algorytm jest skończony.
- Przejścia pomiędzy instrukcjami (albo blokami algorytmu) odbywają się albo sekwencyjnie albo są sterowane danymi (wewnętrznymi albo zewnętrznymi).

Upraszczanie programu

- Czym jest program (algorytm) komputerowy?
  rodzaj „struktury sterującej" określającej w jaki sposób i w jakiej kolejności przetwarzane są dane wejściowe
- Każdy algorytm jest skończony.
- Przejścia pomiędzy instrukcjami (albo blokami algorytmu) odbywają się albo sekwencyjnie albo są sterowane danymi (wewnętrznymi albo zewnętrznymi).



Przykład



State	m	n	r
0	10	18	—
1	10	18	10
2	10	18	10
3	18	10	10
1	18	10	8
2	18	10	8
3	10	8	8
1	10	8	2
2	10	8	2
3	8	2	2
1	8	2	0
2	8	2	0
4	8	2	0

Upraszczanie programów — teza

W pewnym uproszczeniu możemy na program (algorytm) komputerowy popatrzyć jak na rodzaj „skrzyni biegów": urządzenia posiadającego jedynie skończoną liczbę stanów.
Każdy program można zapisać w postaci zestawu „stanów" i odpowiednich „przejść" między nimi (sterowanych danymi).

Maszyna Turinga

Maszyna Turinga składa się z:

1. skończonego **alfabetu** symboli;
2. skończonego zbioru **stanów** z wyróżnionymi **stanami końcowymi** (po osiągnięciu których maszyna zatrzymuje się);
3. nieskończonej **taśmy** z zaznaczonymi kwadratami, z których każdy może zawierać pojedynczy symbol;
4. ruchomej **głowicy** odczytująco-zapisującej, która może wędrować wzdłuż taśmy przesuwając się o jeden kwadrat na raz;
5. **diagramu przejść między stanami** (zazwyczaj zwanego **diagramem przejść** po prostu) zawierającego instrukcje, które powodują, że zmiany następują przy każdym zatrzymaniu się.

Maszyna Turinga

Opis formalny maszyny Turinga

Hopcroft i Ullman zdefiniowali formalnie maszyną Turinga jako następującą „siódemkę" (*7-tuple*): $M = \langle Q, \Gamma, b, \Sigma, \delta, q_0, F \rangle$ gdzie:

- Q to skończny zbiór stanów,
- Γ to skończony zbiór symboli alfabetu
- $b \in \Gamma$ jest symbolem pustym
- $\Sigma \subseteq \Gamma \setminus \{b\}$ to zbiór symboli wejściowych
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, P\}$ jest „funkcją przejścia", L onacza przesunięcie w lewo, a P przesunięcie w prawo.
- $q_0 \in Q$ To stan początkowy
- $F \subseteq Q$ jest zbiorem stanów końcowych

Każdy obiekt spełniający powyższe założenia nazywany być może maszyną Turinga.

Maszyna Turinga

Diagram przejść

Diagram przejść prezentować będziemy jako to graf skierowany, którego wierzchołki reprezentują stany.

Krawędź prowadzącą ze stanu s do t nazywa się **przejściem** i etykietuje kodem postaci: a/b, *kierunek*; gdzie a i b są symbolami (alfabetu), natomiast *kierunek* to albo „w prawo" albo „w lewo" (P, L), albo nic — co oznacza pozostanie w miejscu.

Część a jest **wyzwalaczem**[1] przejścia, a część b, *kierunek* **akcją**.[2]

[1] . . . jeżeli na taśmie spotkasz a. . .
[2] . . . to zapisz w kratce b i przesuń się w *kierunku*. . .

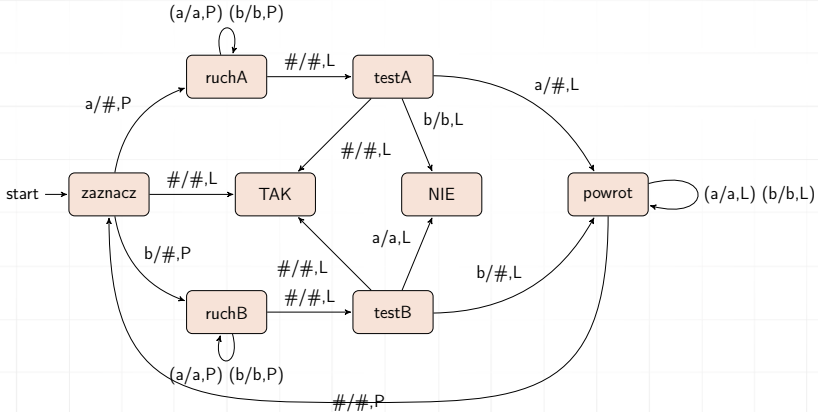
Przykładowa maszyna Turinga

- **Alfabet** — trzy symbole: **a**, **b** i # (co oznacza symbol pusty, „nic")
- **Dane**

...	#	#	a	b	b	a	#	#	...

Przykładowa maszyna Turinga

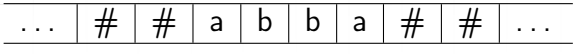
Diagram przejść, stany

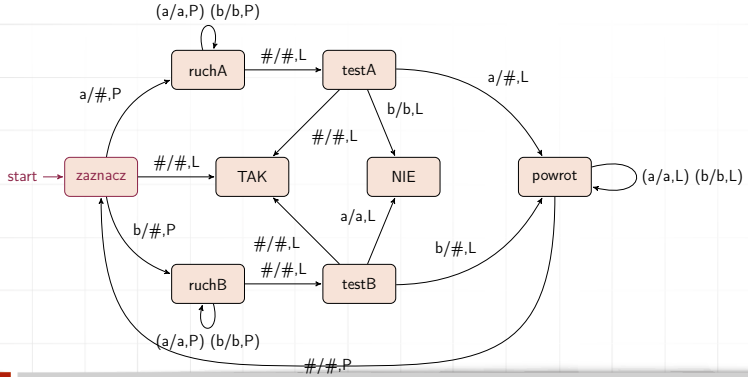


Stany końcowe to stany opisane jako „TAK” i „NIE”

Maszyna Turinga

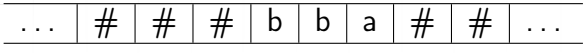
Przykład (1)

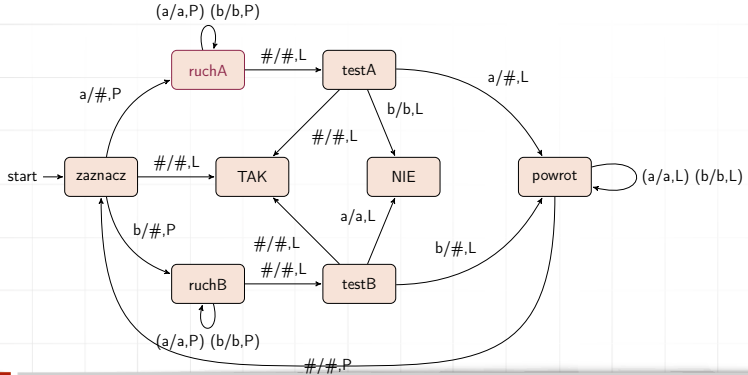




Maszyna Turinga

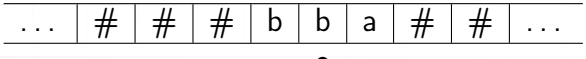
Przykład (1)

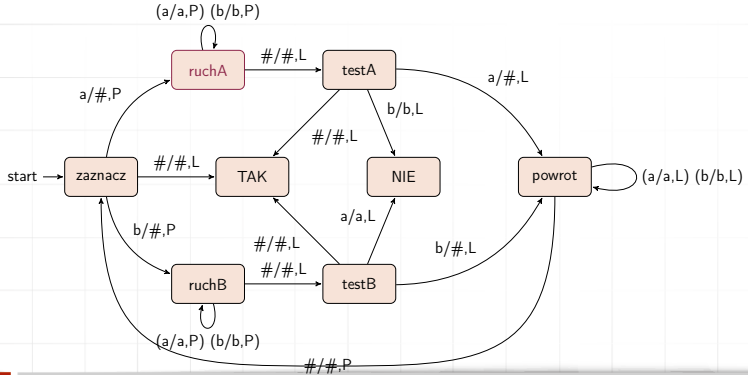




Maszyna Turinga

Przykład (1)

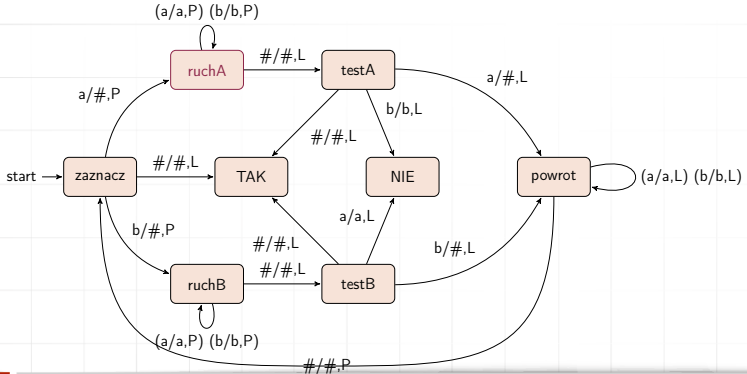




Maszyna Turinga

Przykład (1)

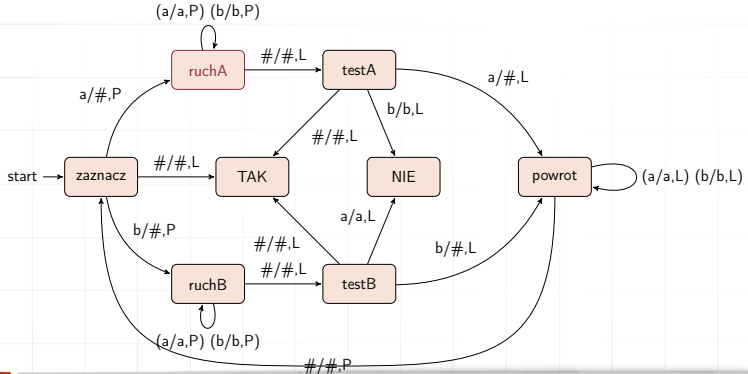
...	#	#	#	b	b	a	#	#	...



Maszyna Turinga

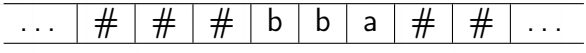
Przykład (1)

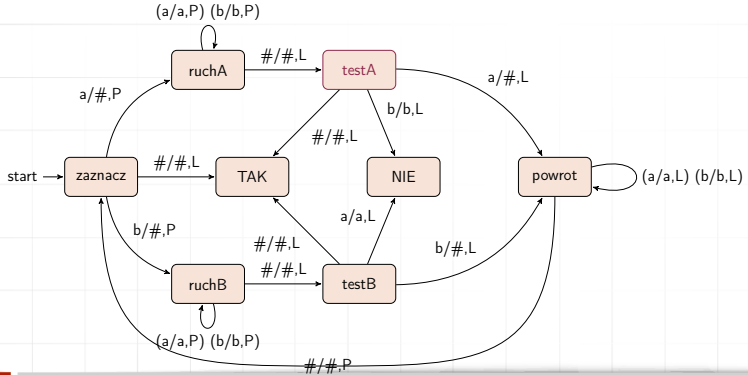
...	#	#	#	b	b	a	#	#	...



Maszyna Turinga

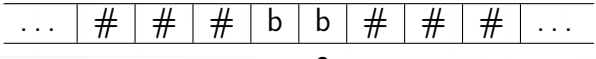
Przykład (1)

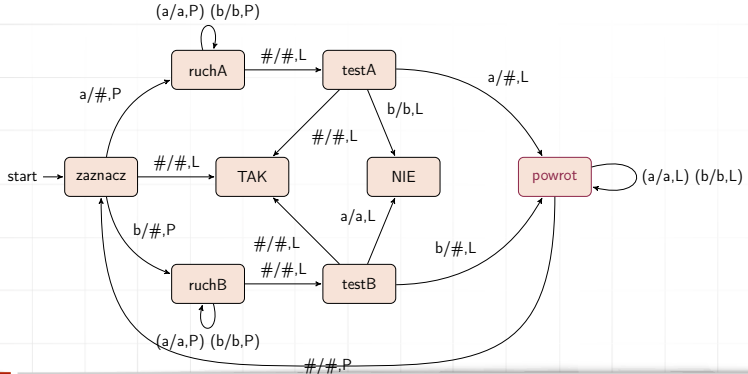




Maszyna Turinga

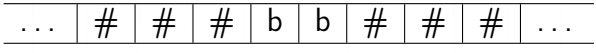
Przykład (1)

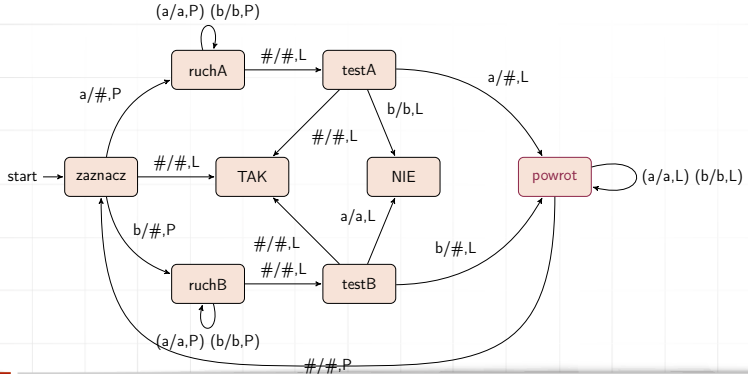




Maszyna Turinga

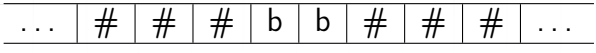
Przykład (1)

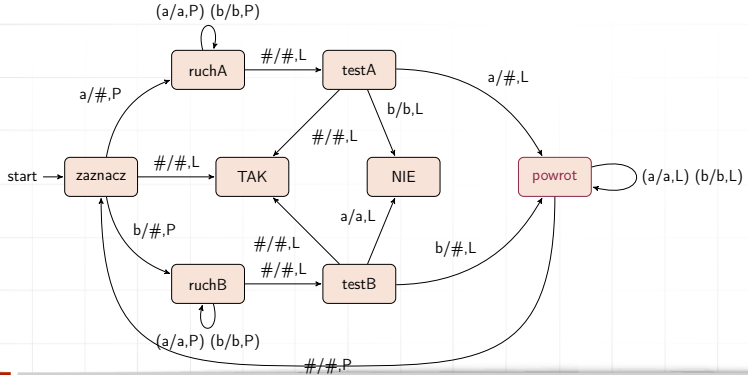




Maszyna Turinga

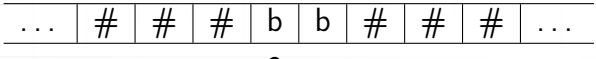
Przykład (1)

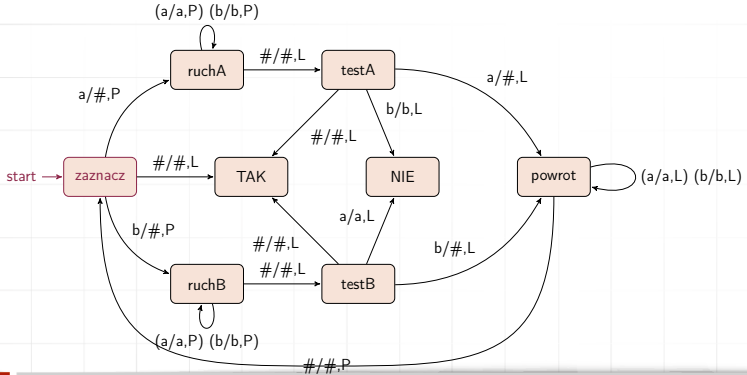




Maszyna Turinga

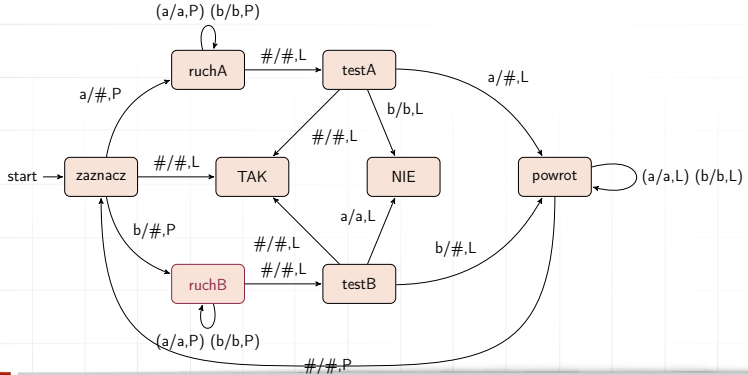
Przykład (1)





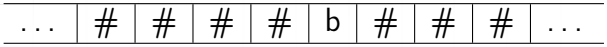
Maszyna Turinga

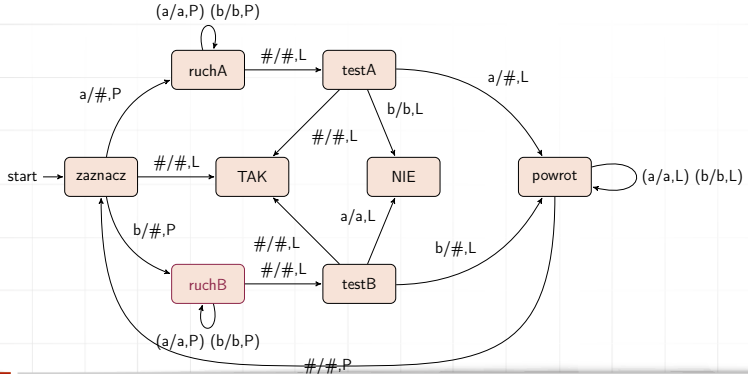
Przykład (1)



Maszyna Turinga

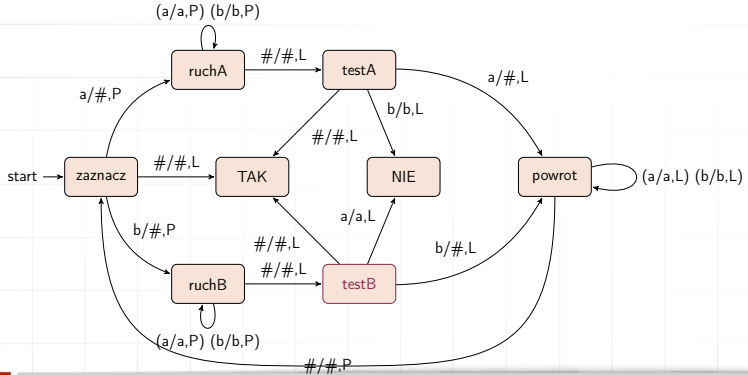
Przykład (1)





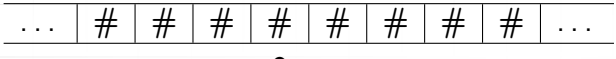
Maszyna Turinga

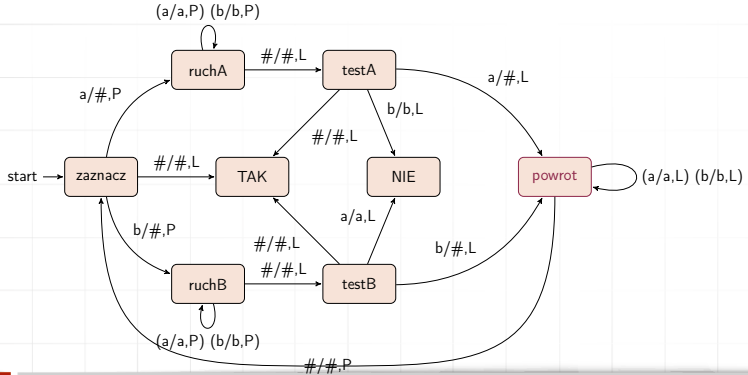
Przykład (1)



Maszyna Turinga

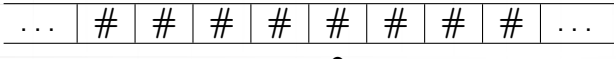
Przykład (1)

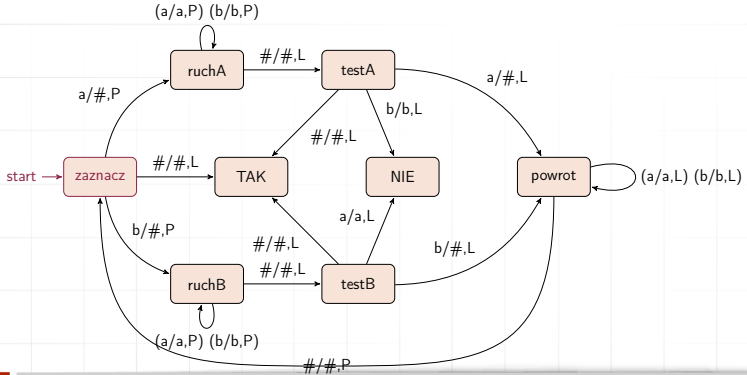




Maszyna Turinga

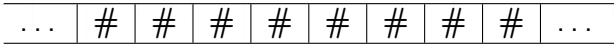
Przykład (1)

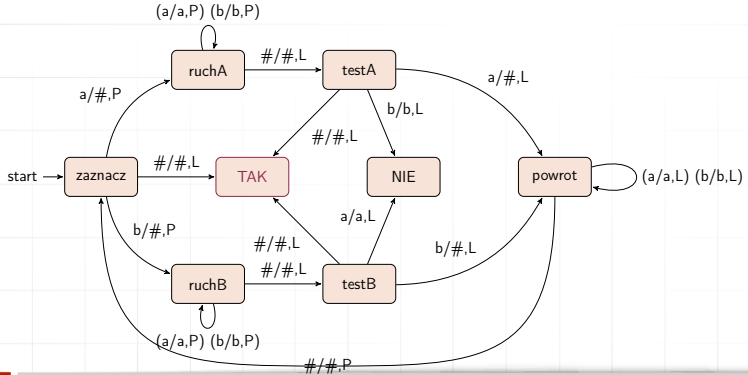




Maszyna Turinga

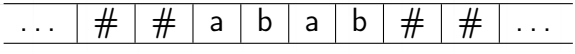
Przykład (1)

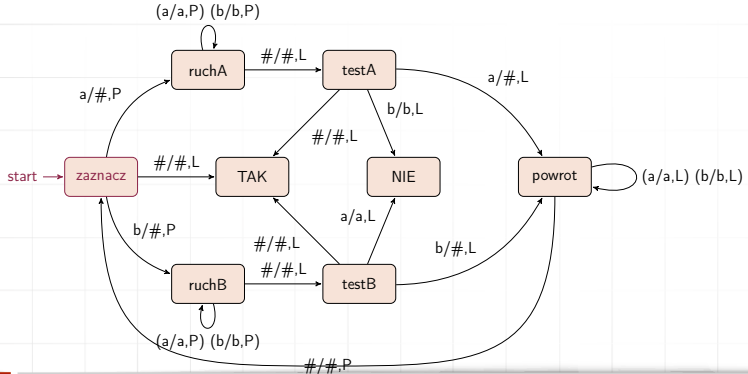




Maszyna Turinga

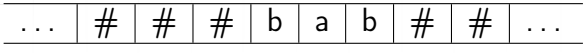
Przykład (2)

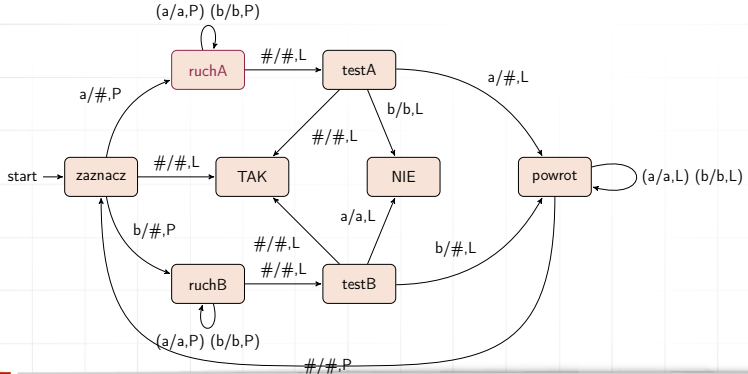




Maszyna Turinga

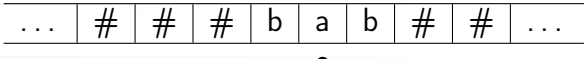
Przykład (2)

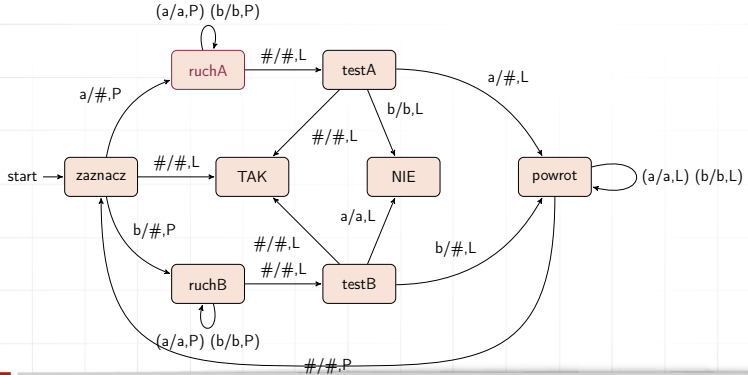




Maszyna Turinga

Przykład (2)

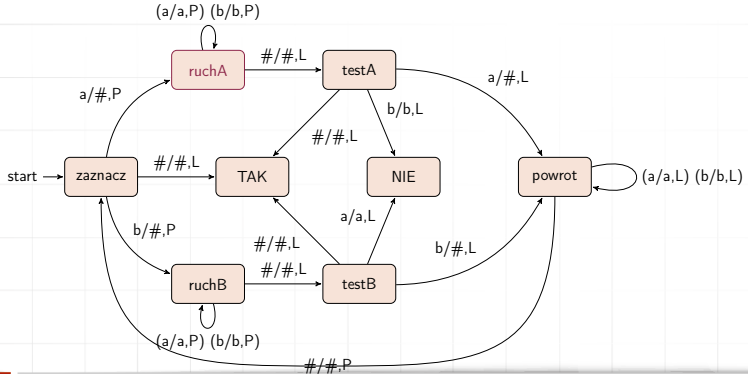




Maszyna Turinga

Przykład (2)

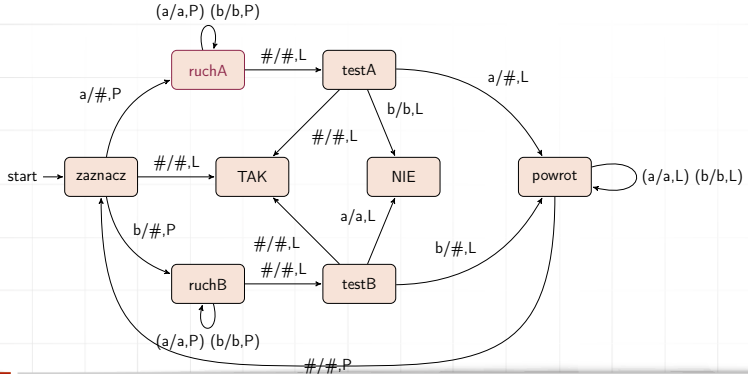
...	#	#	#	b	a	b	#	#	...



Maszyna Turinga

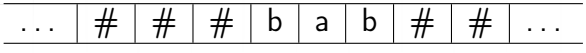
Przykład (2)

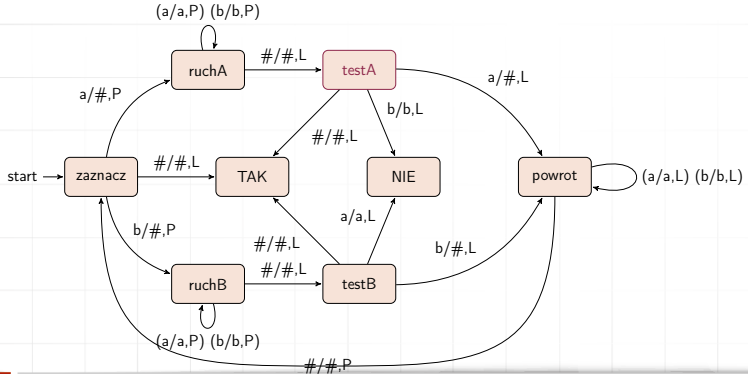
...	#	#	#	b	a	b	#	#	...



Maszyna Turinga

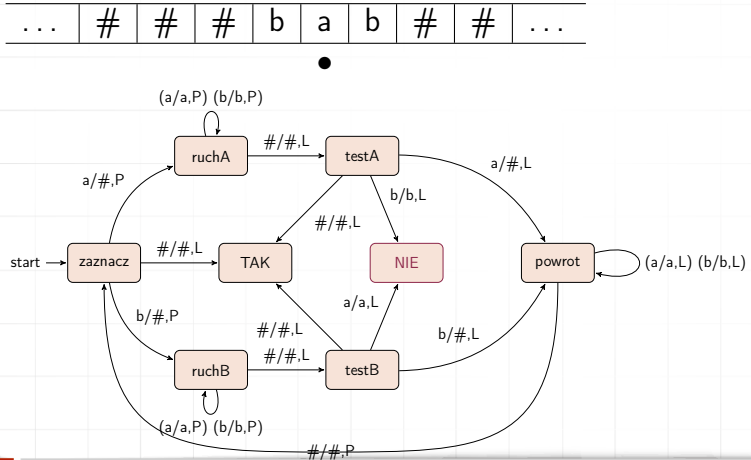
Przykład (2)





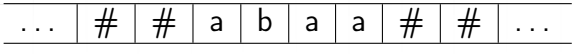
Maszyna Turinga

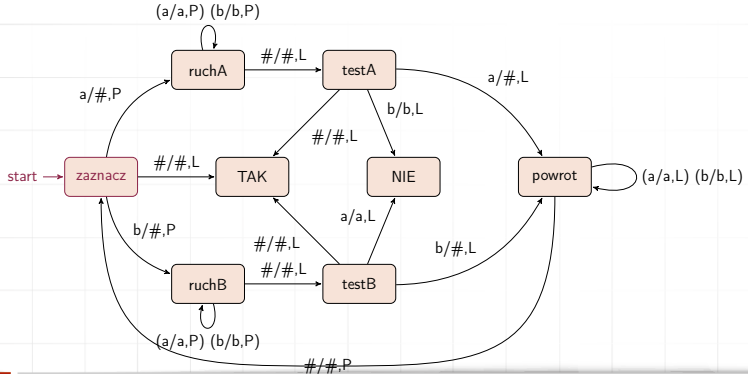
Przykład (2)



Maszyna Turinga

Przykład (3)

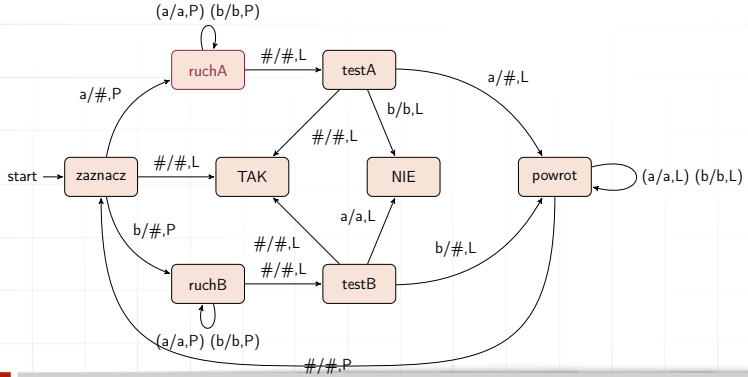




Maszyna Turinga

Przykład (3)

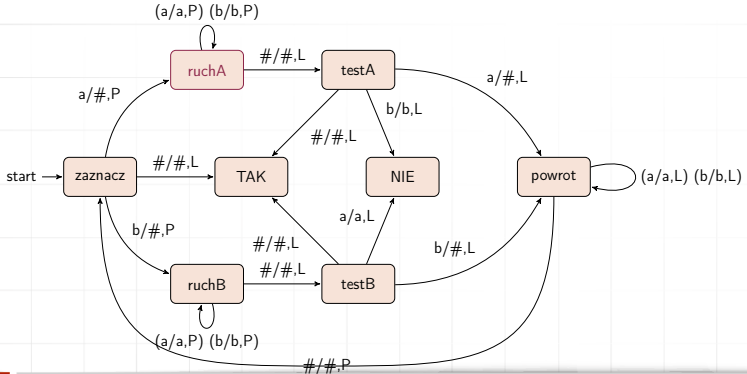
...	#	#	#	b	a	a	#	#	...



Maszyna Turinga

Przykład (3)

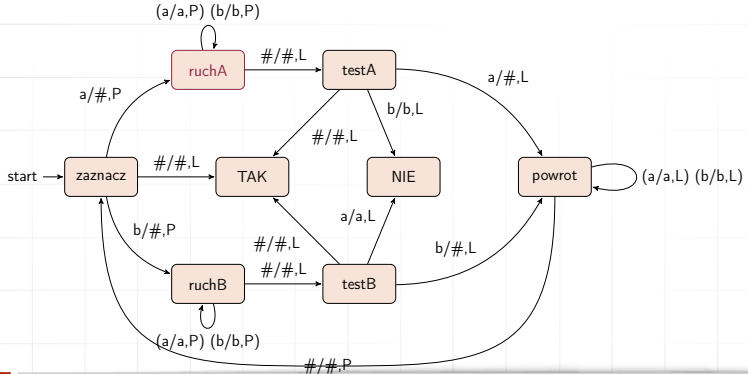
...	#	#	#	b	a	a	#	#	...



Maszyna Turinga

Przykład (3)

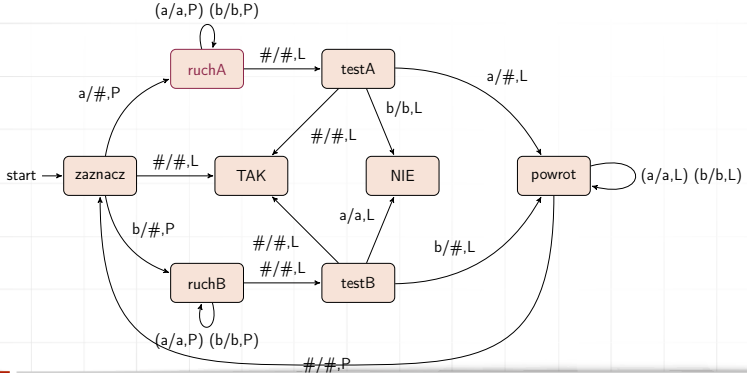
...	#	#	#	b	a	a	#	#	...



Maszyna Turinga

Przykład (3)

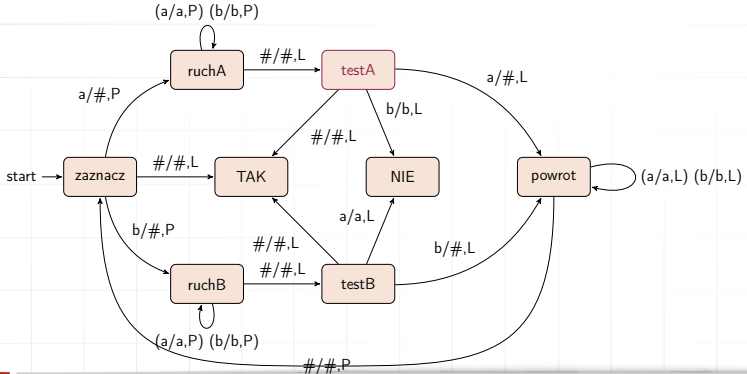
...	#	#	#	b	a	a	#	#	...



Maszyna Turinga

Przykład (3)

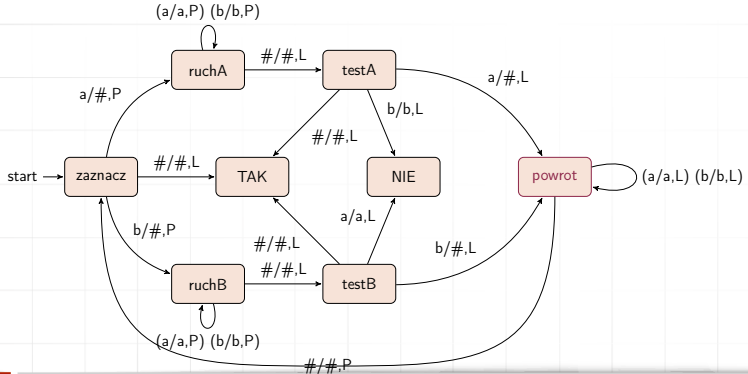
...	#	#	#	b	a	a	#	#	...



Maszyna Turinga

Przykład (3)

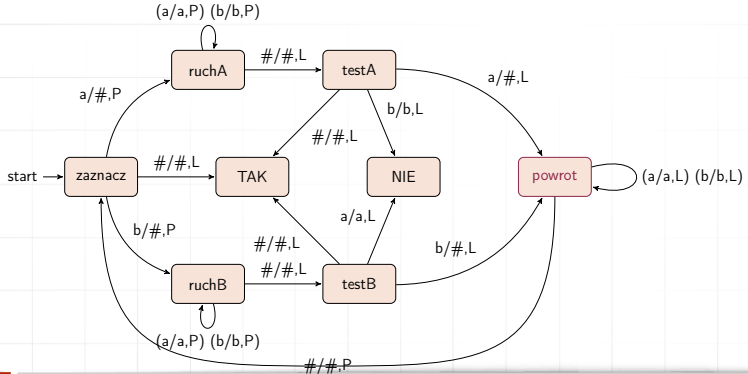
...	#	#	#	b	a	#	#	#	...



Maszyna Turinga

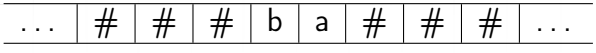
Przykład (3)

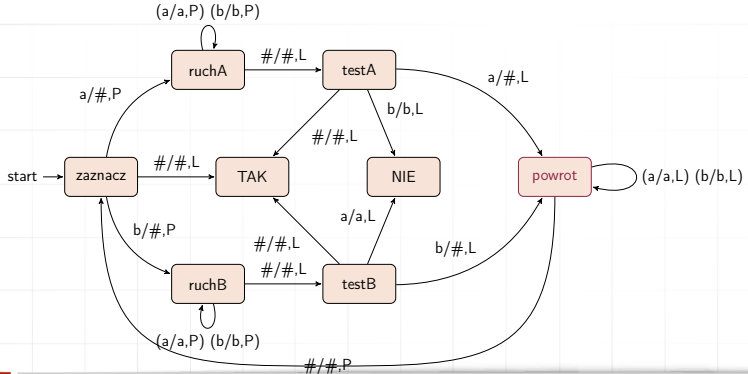
...	#	#	#	b	a	#	#	#	...



Maszyna Turinga

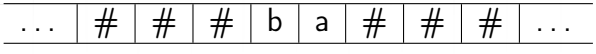
Przykład (3)

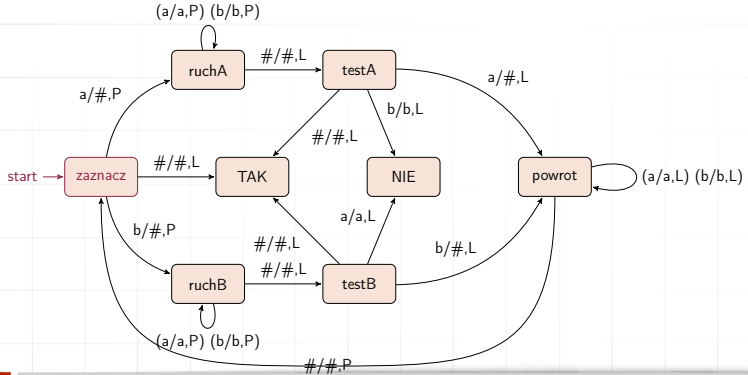




Maszyna Turinga

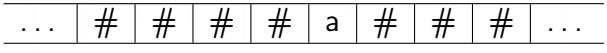
Przykład (3)

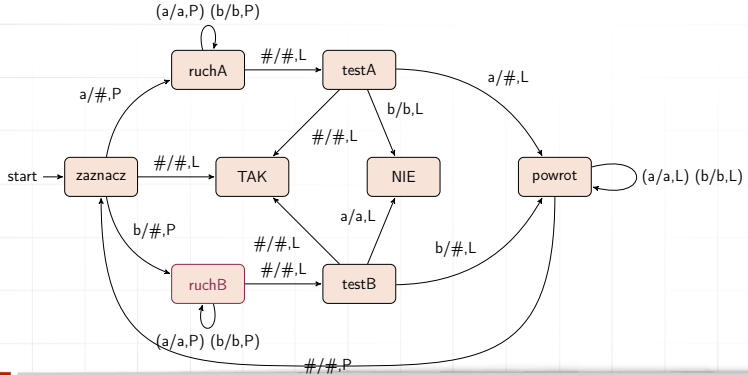




Maszyna Turinga

Przykład (3)

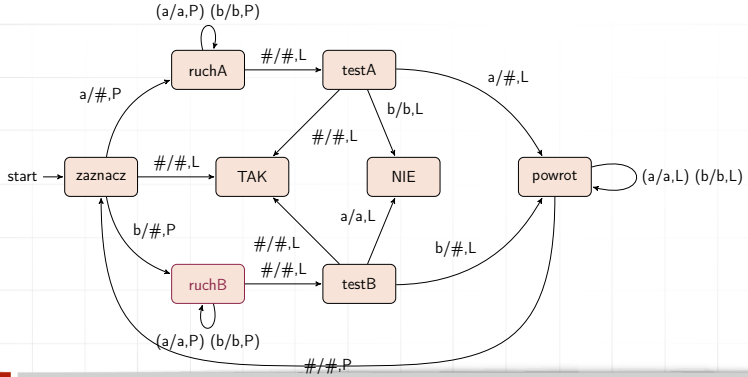




Maszyna Turinga

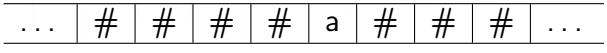
Przykład (3)

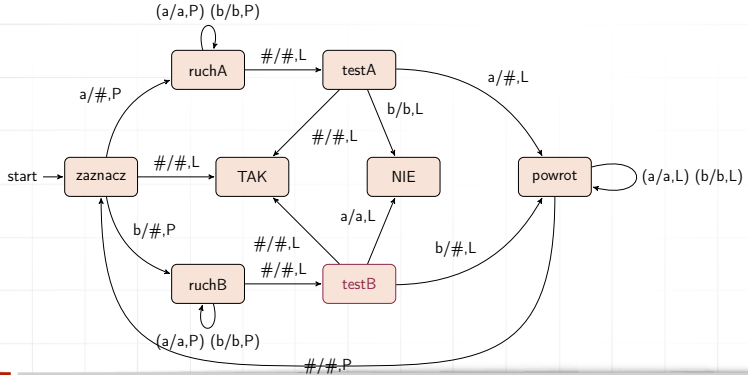
...	#	#	#	#	a	#	#	#	...



Maszyna Turinga

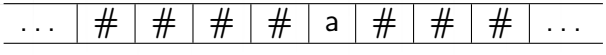
Przykład (3)

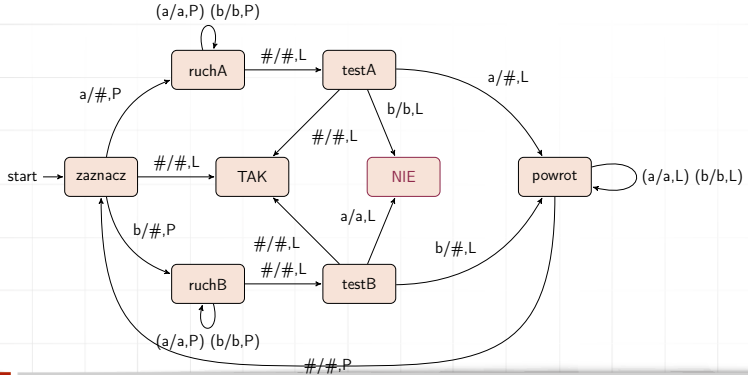




Maszyna Turinga

Przykład (3)





Maszyna Turinga

Warianty

- Taśma „jednostronnie" nieskończona (ma początek, nie ma końca).
- Dwie taśmy („wejściowa" i „wyjściowa").
- Taśma dwuwymiarowa.
- Maszyna niedeterministyczna (w wariancie deterministycznym określony wyzwalacz w danym stanie powoduje zawsze taką samą reakcję; w wariancie niedeterministycznym dopuszcza się dla jednego wyzwalacza kilka różnych akcji wybieranych losowo).

Maszyna Turinga

Warianty

Wszystkie tak zmodyfikowane maszyny Turinga są sobie równoważne!

Maszyna Turinga

Warianty

Wszystkie tak zmodyfikowane maszyny Turinga są sobie równoważne!
(to znaczy każdy problem, który można rozwiązać na jednej z nich można rozwiązać na każdej innej; albo inaczej: każda z tych maszyn może emulować każdą inną).

Maszyna Turinga

Warianty

Wszystkie tak zmodyfikowane maszyny Turinga są sobie równoważne!
(to znaczy każdy problem, który można rozwiązać na jednej z nich można rozwiązać na każdej innej; albo inaczej: każda z tych maszyn może emulować każdą inną).

Pokazano, że komputer zaprojektowany przez Babbage'a — maszyna analityczna — jest równoważny z maszyną Turinga.

Maszyna Turinga

Do czego służy

1. Maszyna Turinga ma tylko skończoną liczbę stanów.
2. Programowanie jej nie musi być łatwe (spróbujcie skonstruować maszyną mnożącą liczby!)
3. Jej działanie zapewne będzie bardzo powolne, a sama symulacja (ręczna) jest dosyć nużąca.
4. Ale co tak na prawdę maszyna Turinga może zrobić?

Teza Churcha-Turinga

Każdy problem algorytmiczny, dla którego możemy znaleźć algorytm dający się zaprogramować w pewnym — dowolnym języku, wykonujący się na pewnym, dowolnym komputerze, nawet na takim, którego jeszcze nie zbudowano, ale można zbudować, i nawet na takim, który wymaga nieograniczonej ilości czasu i pamięci dla coraz większych danych, jest także rozwiązywalny przez maszynę Turinga.

Teza Churcha-Turinga

Każdy problem algorytmiczny, dla którego możemy znaleźć algorytm dający się zaprogramować w pewnym — dowolnym języku, wykonujący się na pewnym, dowolnym komputerze, nawet na takim, którego jeszcze nie zbudowano, ale można zbudować, i nawet na takim, który wymaga nieograniczonej ilości czasu i pamięci dla coraz większych danych, jest także rozwiązywalny przez maszynę Turinga.

Uwaga!

Jest to TYLKO teza, nie twierdzenie!
Powyższe sformułowanie jest jednym z wielu!

Konsekwencje tezy Churcha-Turinga

1. „Domowy komputer" jest równoważny superkomputerowi z wielkiego centrum obliczeniowego (co nie znaczy, że rozwiąże ten sam problem w tym samym czasie!)
2. Wszystkie języki programowania są sobie równoważne (to znaczy to co można zaprogramować w jednym — można w każdym innym)
3. ...

Programy licznikowe

Dopuszcza się jedynie trzy typy elementarnych operacji:

- $X \leftarrow 0$,

Programy licznikowe

Dopuszcza się jedynie trzy typy elementarnych operacji:

- $X \leftarrow 0$,
- $X \leftarrow Y + 1$,

Programy licznikowe

Dopuszcza się jedynie trzy typy elementarnych operacji:

- $X \leftarrow 0$,
- $X \leftarrow Y + 1$,
- $X \leftarrow Y - 1$ (zakładamy, że jeżeli Y jest już równe 0 to $Y - 1$ również jest zero.

Programy licznikowe

Dopuszcza się jedynie trzy typy elementarnych operacji:

- $X \leftarrow 0$,
- $X \leftarrow Y + 1$,
- $X \leftarrow Y - 1$ (zakładamy, że jeżeli Y jest już równe 0 to $Y - 1$ również jest zero.
- Z instrukcji sterujących dopuszczamy tylko skok warunkowy: **jeśli** $X = 0$ **skocz-do** L

Programy licznikowe

Dopuszcza się jedynie trzy typy elementarnych operacji:

- $X \leftarrow 0$,
- $X \leftarrow Y + 1$,
- $X \leftarrow Y - 1$ (zakładamy, że jeżeli Y jest już równe 0 to $Y - 1$ również jest zero.
- Z instrukcji sterujących dopuszczamy tylko skok warunkowy: **jeśli** $X = 0$ **skocz-do** L

Takie zmienne, których wartość może się zmieniać o plus lub minus jeden nazywamy licznikami.

Programy licznikowe

Dopuszcza się jedynie trzy typy elementarnych operacji:

- $X \leftarrow 0$,
- $X \leftarrow Y + 1$,
- $X \leftarrow Y - 1$ (zakładamy, że jeżeli Y jest już równe 0 to $Y - 1$ również jest zero.
- Z instrukcji sterujących dopuszczamy tylko skok warunkowy: **jeśli** $X = 0$ **skocz-do** L

Takie zmienne, których wartość może się zmieniać o plus lub minus jeden nazywamy licznikami.
Zwracam uwagę, że powyższy model jest znacznie bliższy modelowi komputera zaproponowanemu przez von Neumanna niż maszyna Turinga!

Programy licznikowe

Przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

W powyższym X i Y to dane wejściowe, Z — dana wyjściowa; nieistniejąca etykieta L — koniec programu

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
2	3		

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
2	3		0

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
2	3		0

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3		0

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	4	0

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	3	0

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	3	0

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	2	0

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	2	1

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	Y	V	Z
1	3	2	1

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	2	1

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	1	1

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	1	2

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	Y	V	Z
1	3	1	2

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	1	2

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	0	2

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	0	3

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	Y	V	Z
1	3	0	3

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	0	3

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
1	3	0	3

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	0	3

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	4	3

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	3	3

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	3	3

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	2	3

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	2	4

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	Y	V	Z
0	3	2	4

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	2	4

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	1	4

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	1	5

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	Y	V	Z
0	3	1	5

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	1	5

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	0	5

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	0	6

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	Y	V	Z
0	3	0	6

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	0	6

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	0	6

Programy licznikowe

Działający przykład

$U \leftarrow 0$
$Z \leftarrow 0$
A: **jeśli** $X = 0$ **skocz-do** L
$X \leftarrow X - 1$
$V \leftarrow Y + 1$
$V \leftarrow V - 1$
B: **jeśli** $V = 0$ **skocz-do** A
$V \leftarrow V - 1$
$Z \leftarrow Z + 1$
jeśli $U = 0$ **skocz-do** B

X	**Y**	**V**	**Z**
0	3	0	6

KONIEC

Programy licznikowe

Można pokazać, że program licznikowy może symulować działanie maszyny Turinga oraz, że maszyna Turinga może zasymulować program licznikowy.

Programy licznikowe

Można pokazać, że program licznikowy może symulować działanie maszyny Turinga oraz, że maszyna Turinga może zasymulować program licznikowy.

Program licznikowy jest tak samo dobry jak Maszyna Turinga i jak każdy inny komputer (z dokładnością do czasu i przestrzeni).

Automaty skończone

Dla dociekliwych:

Deterministyczny automat skończony może zostać jednoznacznie opisany przez przez piątkę (A, Q, q_0, F, d), gdzie

1. A jest alfabetem
2. Q jest zbiorem stanów
3. q_0 jest wyróżnionym stanem początkowym należącym do Q
4. F jest zbiorem stanów akceptujących (końcowych), będącym podzbiorem Q
5. d jest funkcją przejścia, przypisującą parze (q, a) nowy stan p, w którym znajdzie się automat po przeczytaniu symbolu a w stanie q.

Proszę porównać powyższą definicję z definicją maszyny Turinga!

Kolofon

Prezentacja złożona w systemie LaTeX 2_ε z wykorzystaniem klasy beamer. Użyto fontu MS Trebuchet. Ilustracja na stronie tytułowej jest fragmentem zdjecia, przedstawiającego pomnik Alana Mathisona Turinga znajdujący się w Bletchley Park — centrum brytyjskiej kryptografii z czasów Drugiej Wojny Światowej. Coakley, Garrett. 2009. Alan Mathison Turing. Lipiec 18. Flickr. http://www.flickr.com/photos/garrettc/3777325029/.