AKADEMIA GÓRNICZO-HUTNICZA
IM. STANISŁAWA STASZICA W KRAKOWIE

Hierarchia Chomsky'ego
Maszyna Turinga

Języki formalne i automaty

Dr inż. Janusz Majewski
Katedra Informatyki

Gramatyka

Gramatyką *G* nazywamy czwórkę uporządkowaną

$G = \langle V, \Sigma, P, S \rangle$

gdzie:

V – skończony zbiór symboli nieterminalnych (zmiennych językowych),

Σ – zbiór (alfabet) symboli terminalnych,

P – skończony zbiór produkcji, z których każda ma postać $\alpha \to \beta$

$S \in V$ – wyróżniony symbol początkowy (nieterminal)

przy czym:

$P \subseteq (V \cup \Sigma)^+ \times (V \cup \Sigma)^*$

$P = \{\alpha \to \beta \mid \alpha \in (V \cup \Sigma)^+, \ \beta \in (V \cup \Sigma)^* \}$

Czasami produkcje definiuje się inaczej:

$P \subseteq (V \cup \Sigma)^* V (V \cup \Sigma)^* \times (V \cup \Sigma)^*$

$P = \{\alpha \to \beta \mid \alpha \in (V \cup \Sigma)^* V (V \cup \Sigma)^*, \ \beta \in (V \cup \Sigma)^* \}$

Przykład

$$\begin{array}{l} \mathbf{S} \rightarrow \mathbf{aSBa} \mid \mathbf{aba} \\ \mathbf{aB} \rightarrow \mathbf{Ba} \\ \mathbf{bB} \rightarrow \mathbf{bb} \end{array}$$

język: $\{ a^n b^n a^n \mid n \geq 1 \}$

S
aba

S
a**S**Ba
aab**aB**a
aa**bB**aa
aabbaa

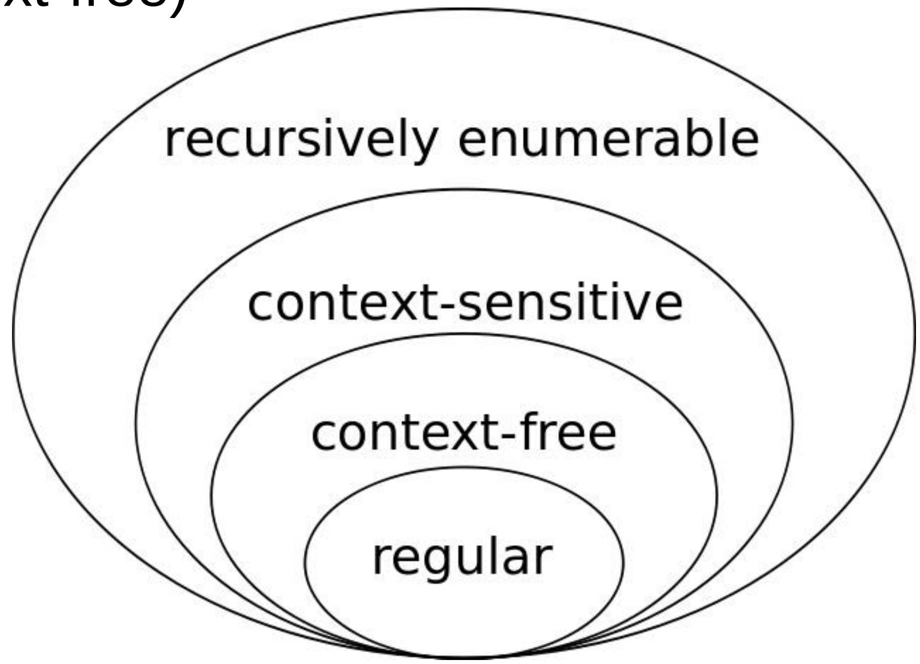
S
a**S**Ba
aa**S**BaBa
aaab**aB**aBa
aaa**bB**aaBa
aaabba**aB**a
aaabb**aB**aa
aaab**bB**aaa
aaabbbaaa

Hierarchia Chomsky'ego

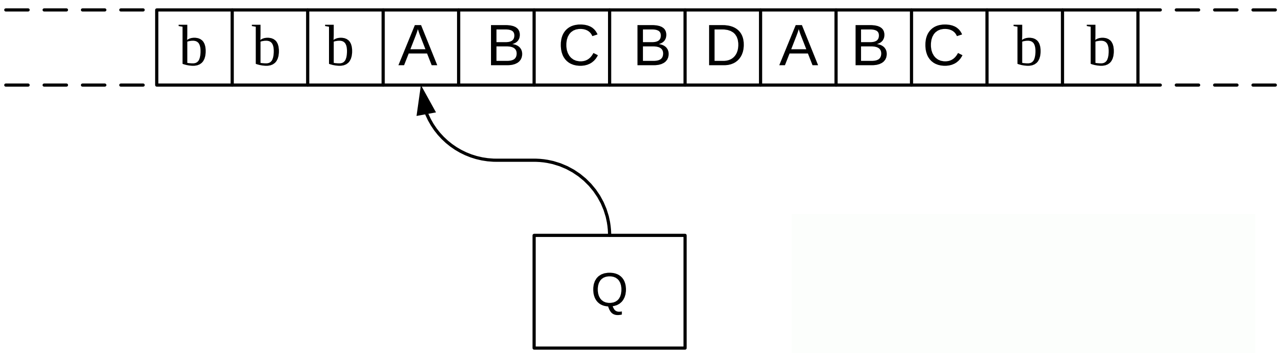
Klasa	Języki	Automaty
0	języki **rekurencyjnie przeliczalne**	maszyna **Turinga**
1	języki **kontekstowe**	automat **liniowo ograniczony**
2	języki **bezkontekstowe**	automat **ze stosem**
3	języki **regularne**	automat **skończony**

Hierarchia Chomsky'ego

Kryterium klasyfikacji: **postać produkcji gramatyki**
Przedmiot klasyfikacji: **języki formalne**
Klasy:
0 – języki **rekurencyjnie przeliczalne** (recursively enumerable)
1 – języki **kontekstowe** (context-sensitive)
2 – języki **bezkontekstowe** (context-free)
3 – języki **regularne** (regular)



Maszyna Turinga (1)

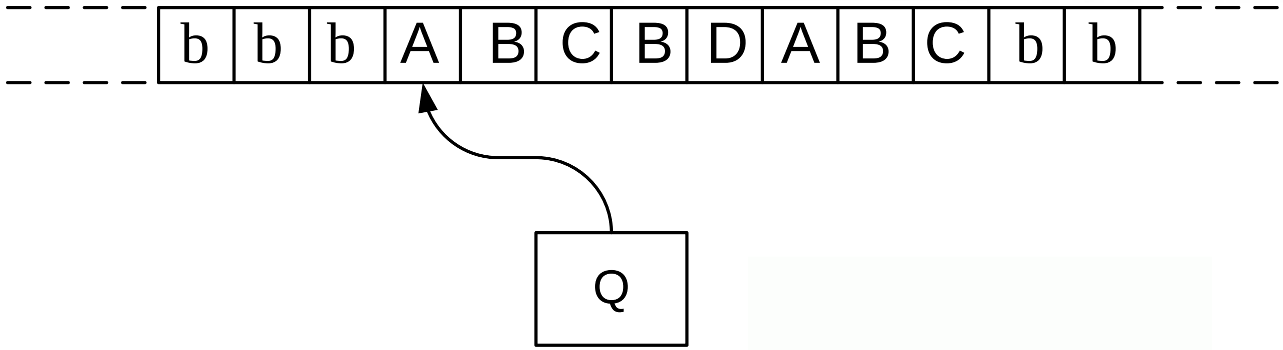


Zależnie od symbolu obserwowanego przez głowicę taśmy oraz stanu sterowania, maszyna Turinga w pojedynczym ruchu:

a) zmienia stan,
b) nadpisuje symbol w obserwowanej komórce taśmy, zastępując nim symbol uprzednio tam wpisany,
c) przesuwa głowicę o jedną komórkę w lewo lub w prawo.

Automat pracuje na NIESKOŃCZONEJ taśmie. Przyjmujemy, że cała taśma wypełniona jest symbolami pustymi („blankami”). W momencie początkowym na środkowej części taśmy zapisane jest badane słowo i głowica ustawiona jest na pierwszym symbolu tego słowa.

Maszyna Turinga (2)



Maszynę Turinga definiujemy jako:

$A = \langle Q, q_0, F, \Gamma, \Sigma, \delta \rangle \in A_T$

Q — skończony zbiór stanów

q_0 – stan początkowy

F – zbiór stanów końcowych

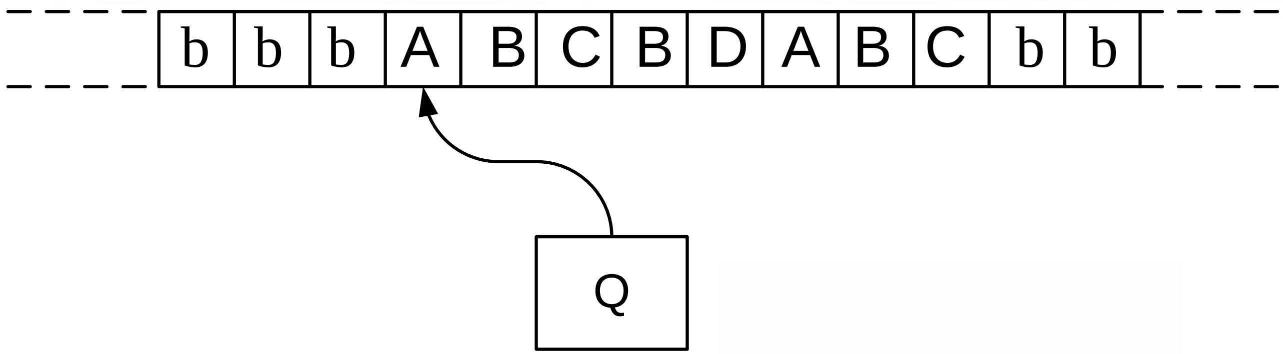
Γ– skończony zbiór symboli taśmy

$\Sigma \subseteq \Gamma$ — alfabet wejściowy

$\boldsymbol{b} \in \Gamma - \Sigma$ — symbol pusty (blank)

δ: $Q \times \Gamma \; \alpha \; 2^{Q \times \Gamma \times \{L,R\}}$ — funkcja przejścia (L–w lewo, R–w prawo)

Maszyna Turinga (3)



Konfiguracja: $(q,\alpha\uparrow\beta)$

q – stan

$\alpha\beta$ – niepusta część taśmy

$\uparrow$ – wskazanie położenia głowicy

<u>Przykład:</u>

Funkcja przejścia: (dla automatu deterministycznego)

$\delta(q_1,C)=(q_2,D,R)$

$(q_1,AB\uparrow CABBBA) \vdash (q_2,ABD\uparrow ABBBA)$

Maszyna Turinga (4)

Konfiguracja początkowa:

$(q_0, \uparrow\alpha), \alpha \in \Sigma^*$

Maszyna Turinga *A* akceptuje język $L \subset \Sigma^*$ gdy:

$L(A) = \{x \in \Sigma^* \mid (\exists\, q \in F)\ (\exists\, y \in \Gamma^*)\ ((q_0, \uparrow x) \vdash_A^* (q, y\uparrow))\}$

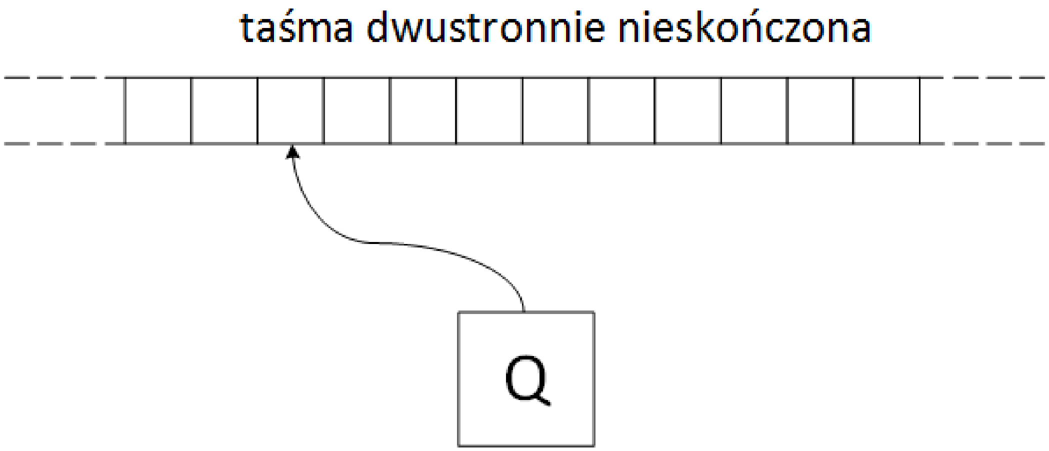
przy czym: $(q, y\uparrow)$ – konfiguracja stopująca

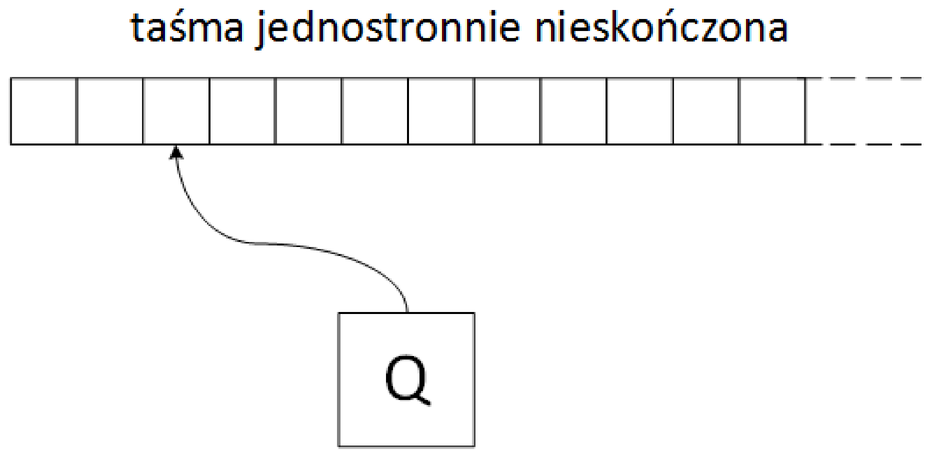
<u>Twierdzenie:</u>

Klasa języków akceptowalnych przez maszyny Turinga L_{MT} jest tożsama z klasą języków rekurencyjnie przeliczalnych L_{RP} – jest to klasa 0 w hierarchii Chomsky'ego

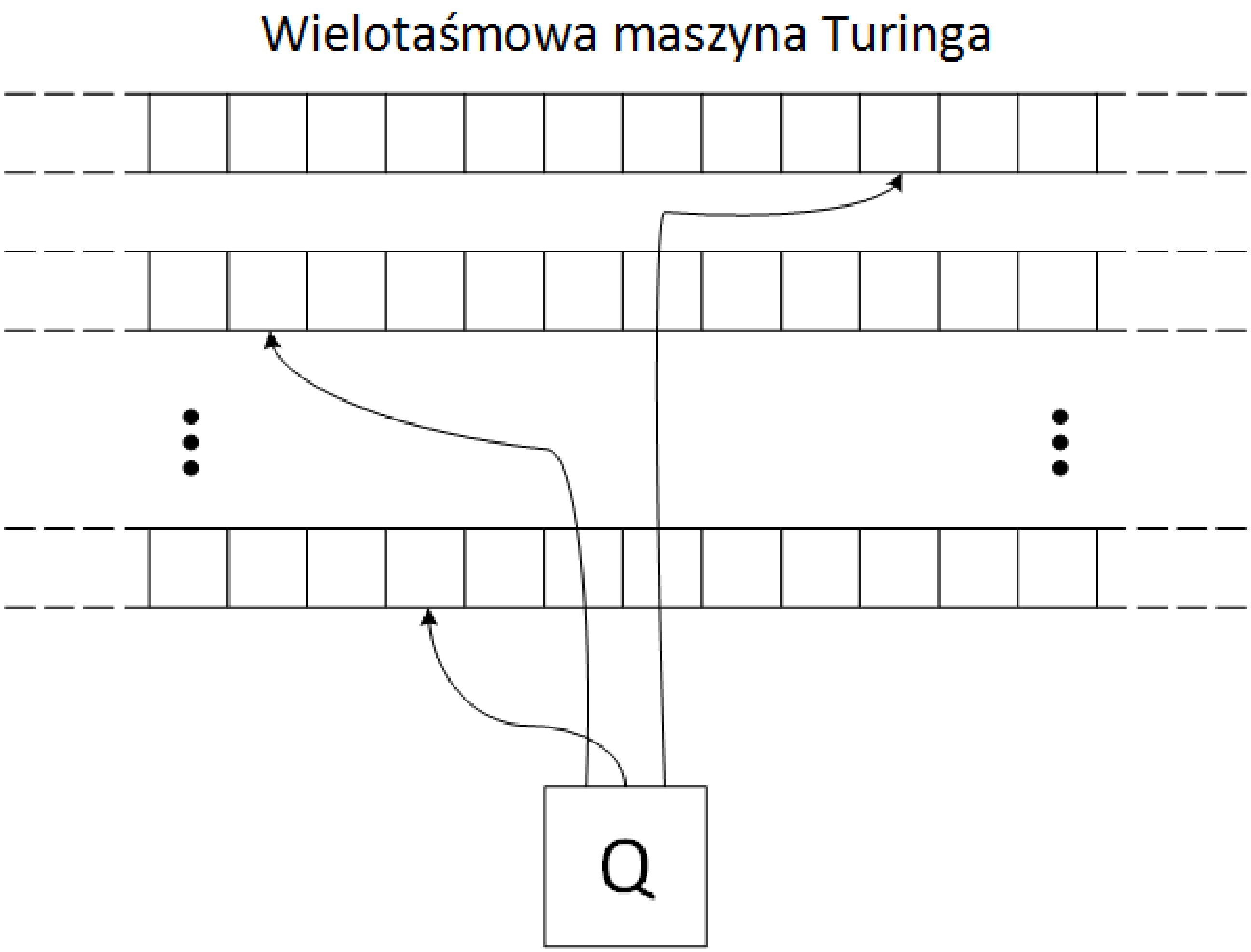
$L_{MT} = L_{RP}$

Równoważne wersje maszyny Turinga





Równoważne wersje maszyny Turinga



Przykład maszyny Turinga

$Q = \{ q_0, q_1, q_2, q_3, q_4 \}$

$F = \{ q_4 \}$

q_0 = stan początkowy

$\Gamma = \{ 0, 1, X, Y, B \}$

$\Sigma = \{ 0, 1 \}$

B - blank

Przykład maszyny Turinga

Stan	Symbol 0	1	X	Y	B
q_0	(q_1, X, P)	–	–	(q_3, Y, P)	–
q_1	$(q_1, 0, P)$	(q_2, Y, L)	–	(q_1, Y, P)	–
q_2	$(q_2, 0, L)$	–	(q_0, X, P)	(q_2, Y, L)	
q_3	–	–	–	(q_3, Y, P)	(q_4, B, P)
q_4	–	–	–	–	–

Rys. 8.9. Maszyna Turinga akceptująca język $\{0^n 1^n \mid n \geq 1\}$

Przykład maszyny Turinga

0	0	0	1	1	1
$q_0\uparrow$					
X	0	0	1	1	1
	$q_1\uparrow$				
X	0	0	1	1	1
		$q_1\uparrow$			
X	0	0	1	1	1
			$q_1\uparrow$		
X	0	0	Y	1	1
		$q_2\uparrow$			
X	0	0	Y	1	1
	$q_2\uparrow$				
X	0	0	Y	1	1
$q_2\uparrow$					

X	0	0	Y	1	1
	$q_0\uparrow$				
X	X	0	Y	1	1
		$q_1\uparrow$			
X	X	0	Y	1	1
			$q_1\uparrow$		
X	X	0	Y	1	1
				$q_1\uparrow$	
X	X	0	Y	Y	1
			$q_2\uparrow$		
X	X	0	Y	Y	1
		$q_2\uparrow$			
X	X	0	Y	Y	1
	$q_2\uparrow$				

Przykład maszyny Turinga

X	X	0	Y	Y	1
		$q_0\uparrow$			
X	X	X	Y	Y	1
			$q_1\uparrow$		
X	X	X	Y	Y	1
				$q_1\uparrow$	
X	X	X	Y	Y	1
					$q_1\uparrow$
X	X	X	Y	Y	Y
				$q_2\uparrow$	
X	X	X	Y	Y	Y
			$q_2\uparrow$		
X	X	X	Y	Y	Y
		$q_2\uparrow$			

X	X	X	Y	Y	Y		
			$q_0\uparrow$				
X	X	X	Y	Y	Y		
				$q_3\uparrow$			
X	X	X	Y	Y	Y		
					$q_3\uparrow$		
X	X	X	Y	Y	Y	B	
						$q_3\uparrow$	
X	X	X	Y	Y	Y	B	B
							$q_4\uparrow$

(akceptacja)

Deterministyczna maszyna Turinga

Dla dowolnej niedeterministycznej maszyny Turinga istnieje równoważna jej maszyna deterministyczna. Maszyna deterministyczna będzie naśladować (symulować) obliczenie maszyny niedeterministycznej, jednak w symulacji należy uniknąć nieskończonego obliczenia. Jeśli maszyna symulująca weszłaby w nieskończone obliczenie, to nie mogłaby sprawdzić innych możliwości obliczeń. Idea symulacji obliczeń maszyny niedeterministycznej wiąże się z przeglądaniem drzewa obliczeń maszyny niedeterministycznej wszerz.

Deterministyczna maszyna Turinga

Jeśli w trakcie symulacji maszyna symulująca osiągnie konfigurację końcową akceptująca w maszynie niedeterministycznej, to maszyna symulująca zatrzyma się i zaakceptuje, w przeciwnym przypadku nastąpi przejście do kolejnego obliczenia wynikającego z przeglądania drzewa obliczeń maszyny niedeterministycznej wszerz.

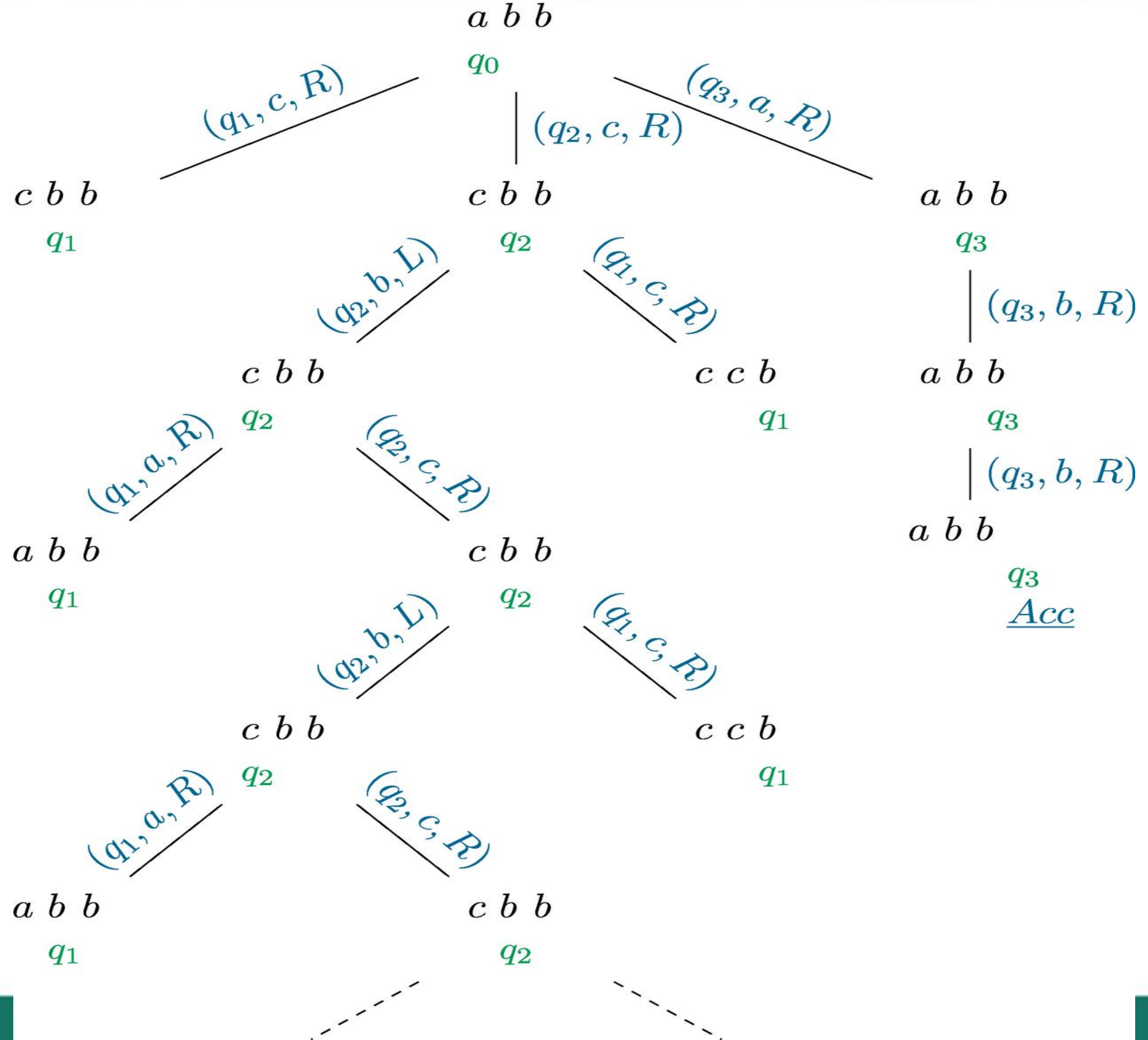
Zastąpienie maszyny niedeterministycznej symulującą ją maszyną deterministyczną zostaje okupione wykładniczym wzrostem złożoności obliczeń.

Niedeterministyczna maszyna Turinga

$$A = \langle Q = \{q_0, q_1, q_2, q_3\}, T = \{a, b\}, \Gamma = \{a, b, c, \flat\}, q_0, F = \{q_3\}, \delta\rangle$$

$\delta:$	a	b	c	$\flat$
q_0	(q_1, c, R) (q_2, c, R) (q_3, a, R)			
q_1	(q_3, a, R)			
q_2		(q_2, b, L) (q_1, c, R)	(q_2, c, R) (q_1, a, R)	
q_3	(q_3, a, R)	(q_3, b, R)		

Drzewo obliczeń niedeterministycznej maszyny Turinga



Języki rozpoznawalne i rozstrzygalne w sensie Turinga

- Maszyna Turinga dla danego wejścia może:
  - zatrzymać się w stanie akceptującym akceptując wejście,
  - zatrzymać się w stanie nie będącym stanem akceptującym, czyli odrzucić wejście,
  - w ogóle nie zatrzymać się (wpaść w nieskończoną pętlę).
- **Język akceptowany (rozpoznawany) przez maszynę Turinga jest zbiorem tych wszystkich słów, dla których maszyna zatrzymuje się w stanie akceptującym. Słowa nie należące do języka mogą zostać odrzucone (maszyna zatrzyma się w stanie nieakceptującym) lub maszyna zapetli się i w ogóle nie zatrzyma się dla takich słów.**
- Takie języki nazywamy także językami rekurencyjnie przeliczalnymi

Języki rozpoznawalne i rozstrzygalne w sensie Turinga

- Maszyna Turinga dla danego wejścia może:
  - zatrzymać się w stanie akceptującym akceptując wejście,
  - zatrzymać się w stanie nie będącym stanem akceptującym, czyli odrzucić wejście,
  - w ogóle nie zatrzymać się (wpaść w nieskończoną pętlę).
- **Maszyny Turinga, które zawsze się zatrzymują się na każdym wejściu nazywamy maszynami rozstrzygającymi lub maszynami z własnością stopu.**
- **Język rozstrzygalny przez maszynę Turinga jest zbiorem tych wszystkich słów, dla których rozstrzygająca maszyna Turinga (maszyna z własnością stopu) zatrzymuje się w stanie akceptującym.**
- Takie języki nazywamy także językami rekurencyjnymi.

Rozstrzygające maszyny Turinga

- **Maszyny Turinga, które zawsze się zatrzymują się na każdym wejściu nazywamy maszynami rozstrzygającymi lub maszynami z własnością stopu.**
- **Nie każda maszyna Turinga jest maszyną rozstrzygającą (z własnością stopu). Istnieją maszyny Turinga, które nie są maszynami rozstrzygającymi.**
- Zbiór wszystkich rozstrzygających maszyn Turinga jest więc podzbiorem właściwym zbioru wszystkich maszyn Turinga.

Kodowanie maszyny Turinga

Ograniczymy się do maszyn Turinga działających na alfabecie binarnym.

- Ponumerujemy stany maszyny zgodnie z ich indeksacją
- Symbolom alfabetu wejściowego *{0,1}* przypiszemy liczby *1* i *2*
- Symbolom alfabetu taśmy *{0,1,B}* przypiszemy liczby *1*, *2* i *3*
- Symbolom kierunku ruchu *{L,R}* przypiszemy liczby *1* i *2*
- Poszczególne liczby reprezentujące symbole opisu maszyny Turinga będziemy zapisywać w postaci ciągu jedynek o długości równej wartości liczby (w systemie jedynkowym)

Kodowanie maszyny Turinga

- Każdy ruch określimy poprzez podanie wartości funkcji przejścia $\delta(q_i, x)=(q_k,Y,A)$, którą będziemy kodować jako ciąg zero-jedynkowy:

$$1^i01^j01^k01^l01^m$$

gdzie: liczby *i, j, k, l,* m są kodami odpowiednio: stanu q_i, symbolu taśmy X, stanu q_k, symbolu taśmy Y, kierunku ruchu głowicy A

Kodowanie maszyny Turinga

- Maszynę Turinga *M* zakodujemy jako ciąg opisów poszczególnych wartości funkcji przejścia oddzielonych dwoma zerami, natomiast początek i koniec zakodowanego opisu maszyny Turinga oznaczymy potrójnymi zerami:

  000 kod-ruchu-1 00 kod-ruchu-2 00 …

  … 00 kod-ruchu-m 000

Problem

- Czy istnieją języki, które nie są rozpoznawane (akceptowane) przez żadną maszynę Turinga?
- Prawdopodobnie tak, bo klasa wszystkich języków jest zbiorem nieprzeliczalnym, natomiast klasa języków akceptowanych przez maszyny Turinga jest przeliczalna (maszyny Turinga można zakodować skończonymi kodami, więc jest ich tylko przeliczalnie wiele).

Język przekątniowy (diagonalizacji)

$L_d = \{ w_i \mid w_i$ <u>nie</u> jest akceptowane przez maszynę Turinga $M_i \}$

Język przekątniowy jest zbiorem tych słów w_i, dla których, jeśli indeksem słowa w porządku standardowym jest liczba *i*, to maszyna Turinga o numerze *i* nie akceptuje słowa *w*.

Język przekątniowy (diagonalizacji)

Rozważymy dwustronnie nieskończoną tablicę o wartościach binarnych. Kolumny tej tablicy indeksowane będą kolejnymi liczbami M_1, M_2,…, których binarna reprezentacja jest poprawnym kodem maszyny Turinga. Wiersze tabeli będą indeksowane kolejnymi słowami nad alfabetem binarnym ustawionymi w porządku standardowym. Wartością pola w wierszu i oraz kolumnie j tej tabeli będzie „tak”, gdy maszyna o numerze j będzie akceptować słowo o numerze i. W przeciwnym wypadku wartością tego pola będzie „nie”.

Język przekątniowy

kody maszyn Turniga (wszystkich) →

		M_1	M_2	M_3	M_4	M_5	...
w_1	ε	Tak	Tak	Tak	Nie	Tak	...
w_2	0	Tak	Nie	Nie	Nie	Nie	...
w_3	1	Nie	Tak	Tak	Tak	Nie	...
w_4	00	Tak	Nie	Tak	Nie	Tak	...
w_5	01	Nie	Nie	Tak	Tak	Nie	...
w_6	10	Tak	Tak	Nie	Tak	Tak	...
w_7	11	Nie	Tak	Nie	Nie	Tak	...
w_8	000	Tak	Nie	Tak	Tak	Nie	...
⋮	⋮	⋮	⋮	⋮	⋮	⋮	...

Język przekątniowy

Język L_d nie jest akceptowany przez żadną maszynę Turinga. Udowodnimy to stwierdzenie przez doprowadzenie do sprzeczności. Załóżmy, że język przekątniowy jest akceptowany przez maszynę Turinga o indeksie k. Rozważmy słowo o indeksie k w porządku kanonicznym, tzn. w_k. Jeśli słowo w_k byłoby akceptowane przez maszynę Turinga M_k, to R_{kk} powinno być równe 1, a to oznacza, że w_k nie może należeć do języka przekątniowego. I na odwrót, jeżeli w_k nie byłoby akceptowane przez maszynę Turinga M_k, to R_{kk} powinno być równe 0, a to oznacza, że w_k należy do języka przekątniowego. Doszliśmy do sprzeczności, tzn. nie może istnieć maszyna Turinga akceptująca język L_d. Zatem język przekątniowy L_d nie jest językiem rekurencyjnie przeliczalnym.

Problem

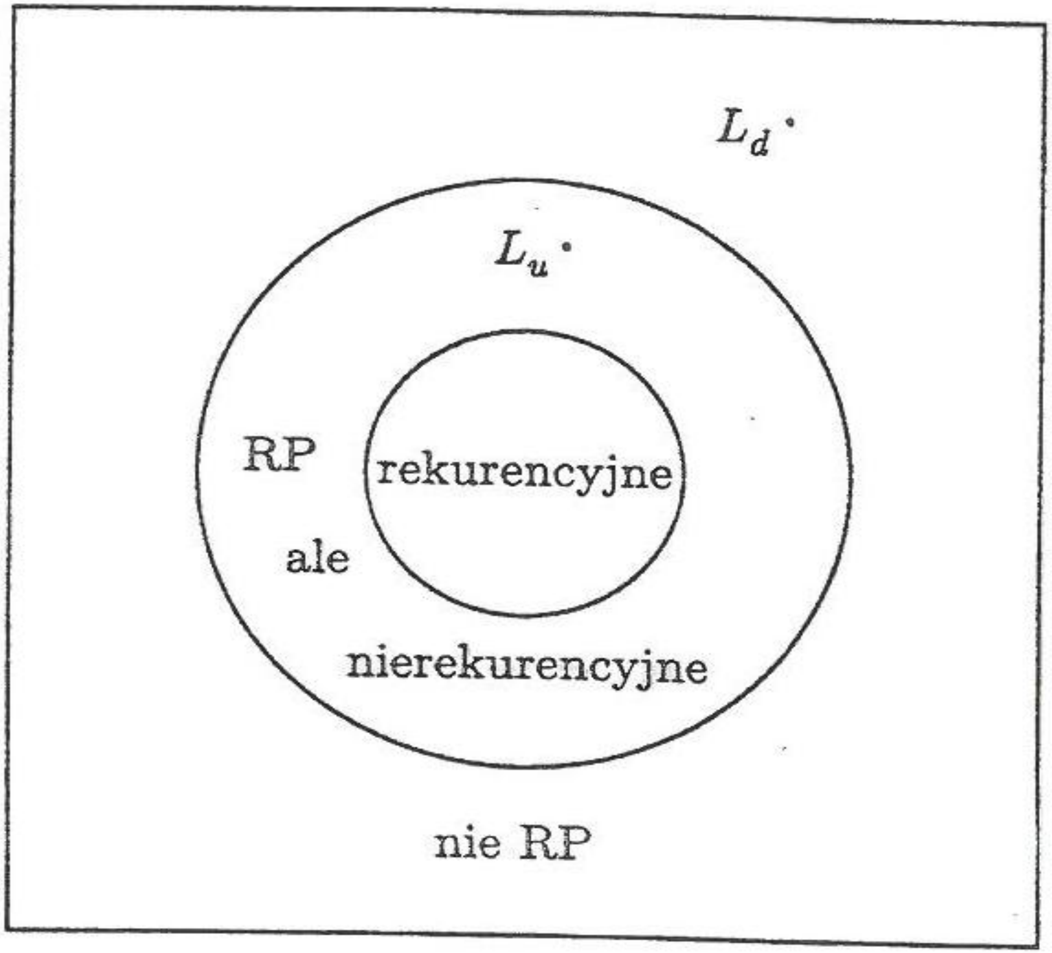
- Czy istnieją języki, które są rozpoznawane (akceptowane) przez maszyny Turinga, ale nie są rozstrzygalne?

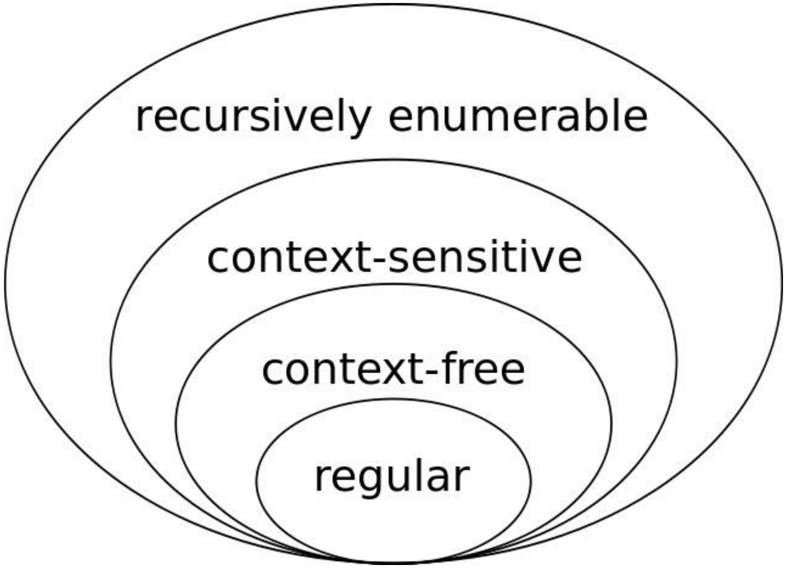
- Prawdopodobnie tak, bo klasa maszyn Turinga z własnością stopu (maszyn rozstrzygających) jest podklasą właściwą klasy wszystkich maszyn Turinga

Język uniwersalny

- L_u = { <*M,w*> | maszyna *M* akceptuje słowo *w* }
- Można skonstruować uniwersalną maszynę Turinga M_u, która na podstawie kodu maszyny *M* będzie symulować jej działanie dla słowa *w.* Wtedy i tylko wtedy, gdy słowo *w* jest akceptowane przez *M*, to maszyna symulująca M_u się zatrzyma i zaakceptuje wejście <*M,w*>. Jeśli *M* nie zatrzymuje się dla *w*, to M_u także się nie zatrzyma.
- L_u jest przykładem języka rozpoznawalnego, ale nie rozstrzygalnego

Maszyna Turinga a języki





Teza Churcha – Turinga

Każdy efektywnie rozwiązywalny (obliczalny) problem algorytmiczny jest rozwiązywalny przez maszynę Turinga.

Każdy problem algorytmiczny, dla którego możemy znaleźć algorytm dający się zaprogramować w pewnym (*dowolnym*) języku programowania, 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.

- Alan M. Turing
- Alonzo Church