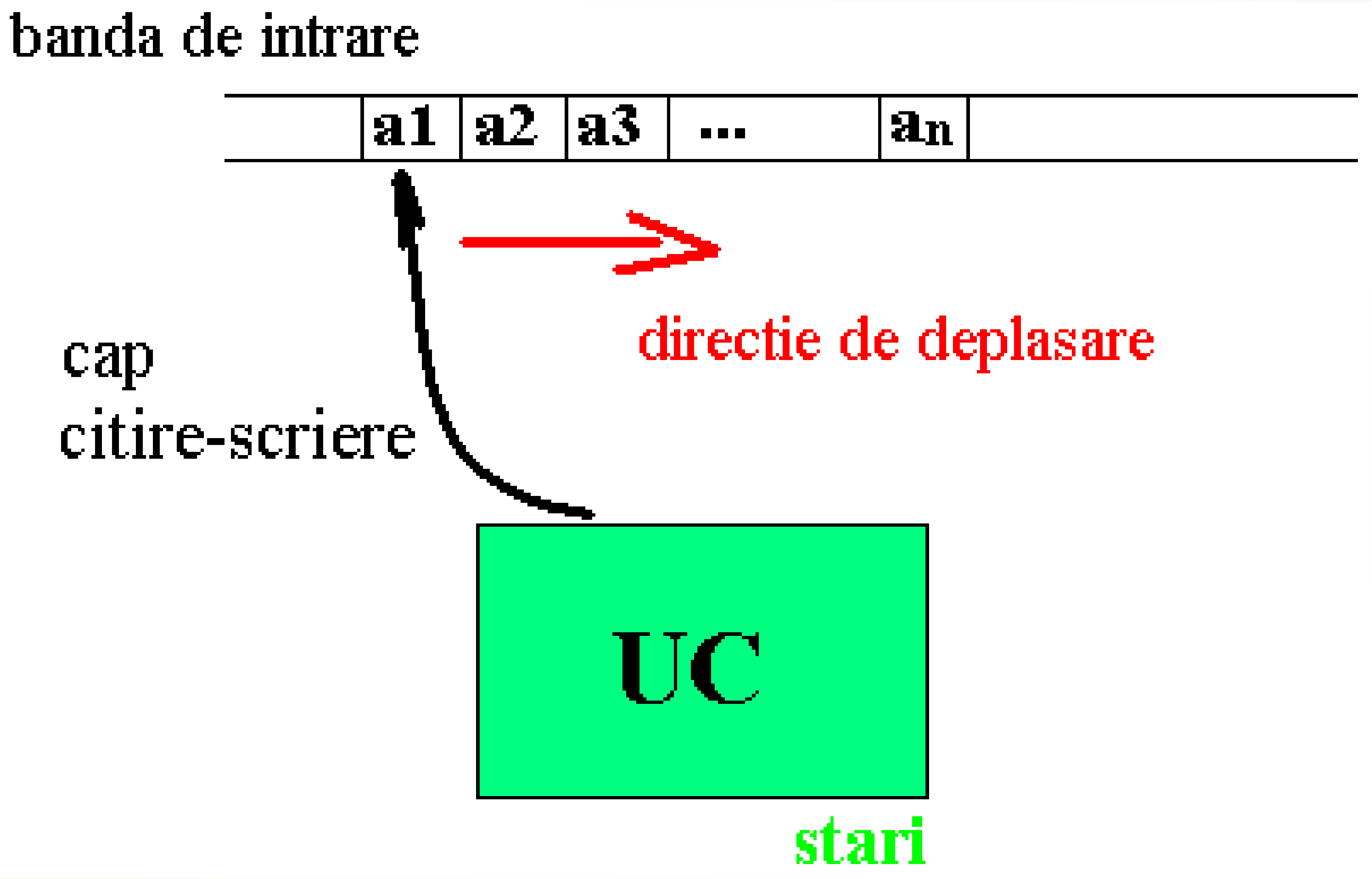
Automate

- Recapitulare, Exemple, Aplicatii

- Translatoare
- Masini Turing

Automat finit: model fizic



Automat finit: model matematic

- Un ***automat finit*** este un ansamblu

  $M = (Q, \Sigma, \delta, q_0, F)$:

AF cu numar minim de stari

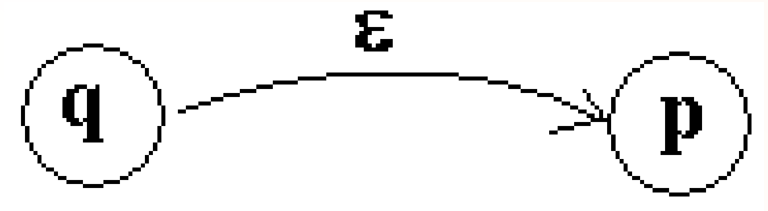
- Q – alfabetul starilor
- Σ – alfabet de intrare
- $\delta : Q x \Sigma \rightarrow \mathcal{P}(Q)$ functie de tranzitie
- $q_0 \in Q$ - stare initialã
- $F \subseteq Q$ multimea stãrilor finale

Automate finite cu ε-miscari

$M = (Q, \Sigma, \delta, q_0, F)$:

- $\delta : Q x(\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}(Q)$ functia de tranzitie

Putem avea si ε-tranzitii
(automate cu ε-tranzitii)



Teorema:
Pentru orice automat finit cu ε-miscari
exista un automat finit echivalent.

<u>Obs</u>. *Conform def., automatele finite sunt fara ε-miscari*

Limbaje regulare. Echivalente

- putere de exprimare

AF: AFN ⇔ AFD

AF ⇔ (m.regulare ⇔ expr.reg.)

AF ⇔ gr.regulare

Proprietati de inchidere

Cursul 3, constructii

Teorema:

Daca

L_1, L_2 sunt limbaje regulare peste alfabetul Σ

atunci:

$\mathbf{L_1 \cup L_2, L_1 \cap L_2, L_1L_2, L_1^*}$, **complement($L_1$)**

sunt limbaje regulare peste alfabetul Σ

Proprietãti de închidere ale limbajelor independente de context

Teoremã.

Dacã L_1si L_2 sunt limbaje independente de context atunci:

$$L_1 \cup L_2 \ , \ L_1 L_2 \ , \ L_1^*$$

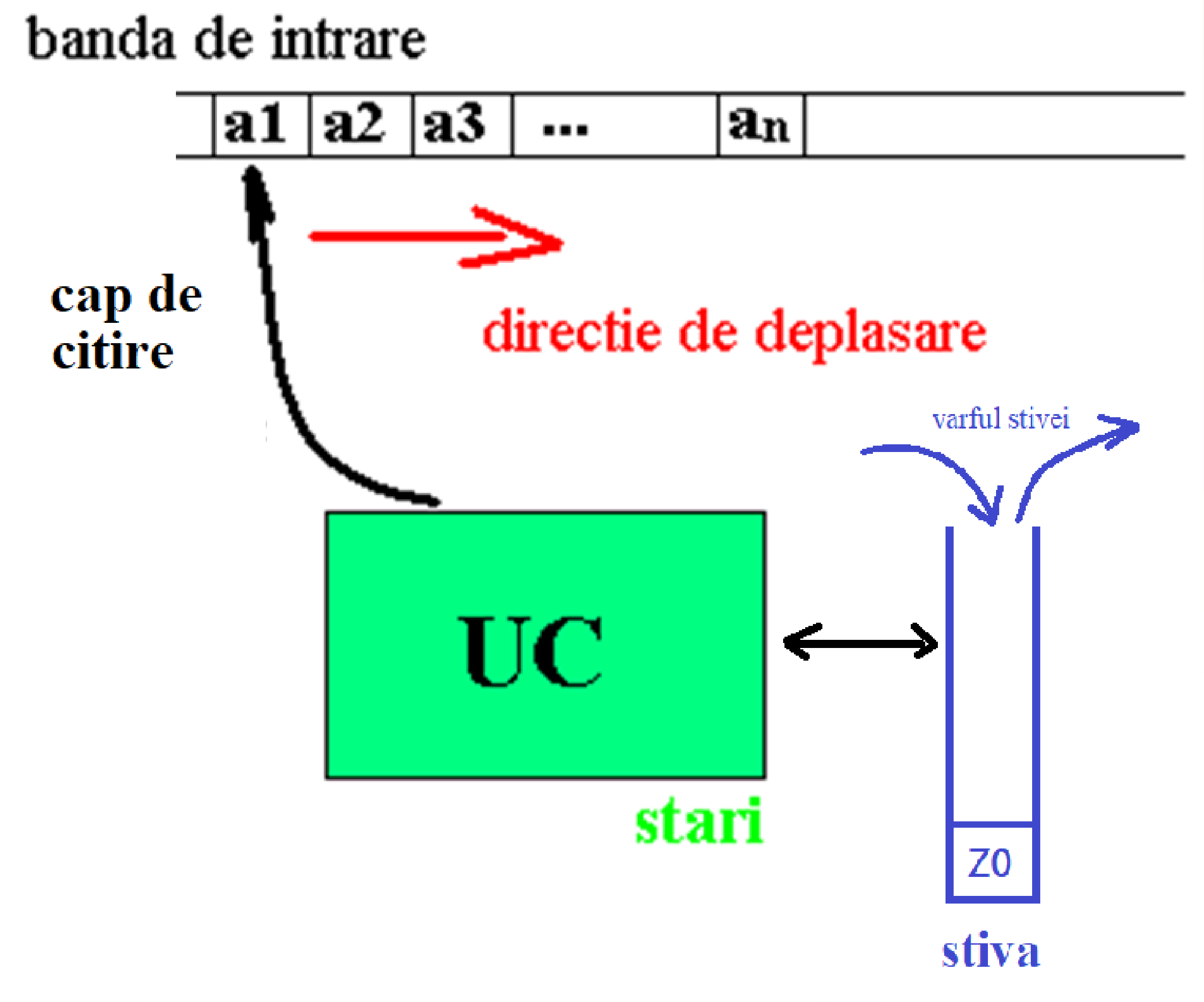
sunt limbaje independente de context.

Observatie:

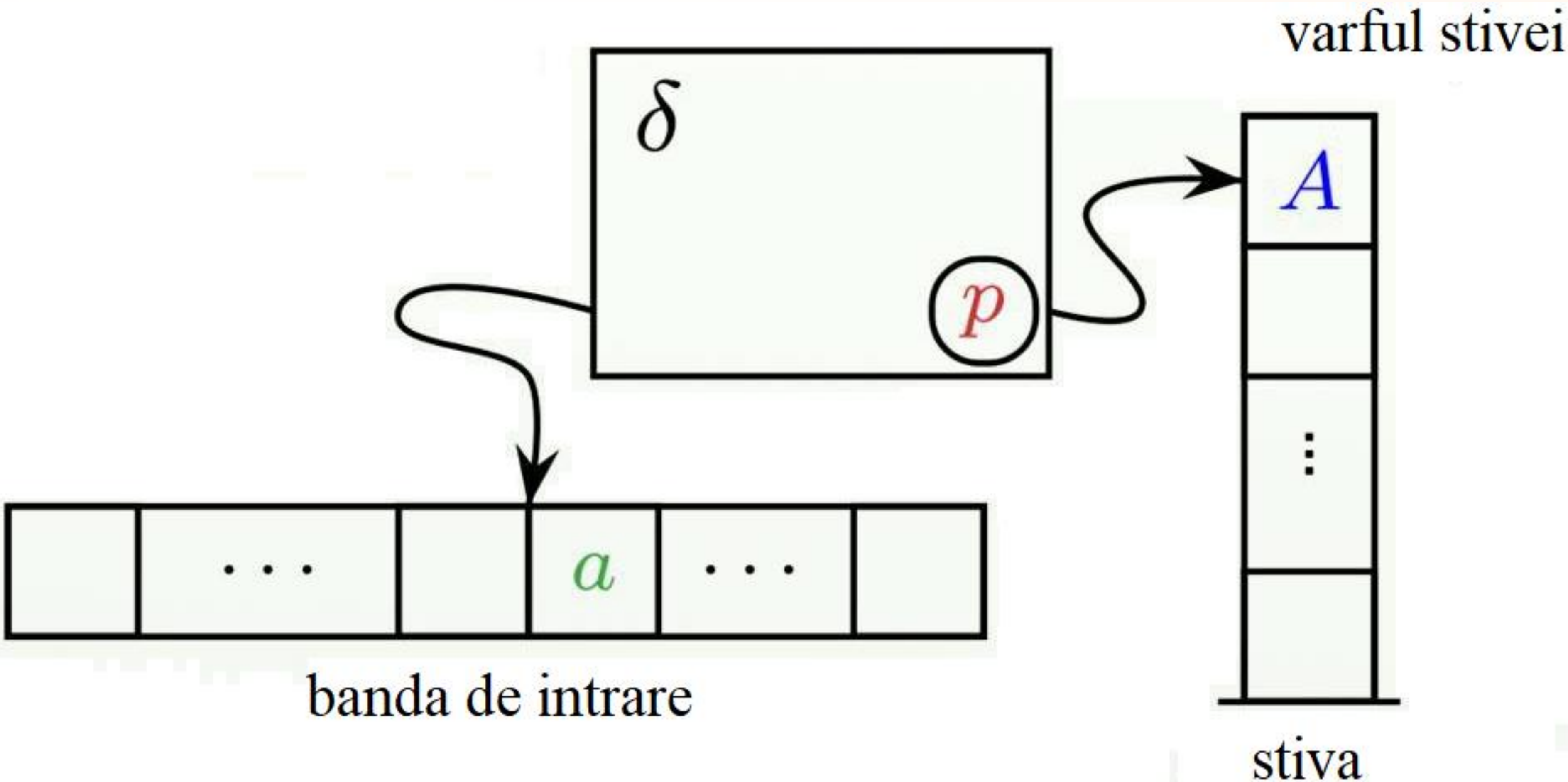
$L1 \cap L2$, compl(L1) :

nu sunt neaparat limbaje independente de context

Automat push-down (APD)



Automat push-down (APD)



https://en.wikipedia.org/wiki/File:Pushdown-overview.svg

Automat push-down (APD)

$M = (Q, \Sigma, \Gamma, \delta, q_o, Z_o, F)$

- Q alfabetul stărilor;
- Σ alfabetul de intrare;
- Γ alfabetul memoriei stivă;
- $q_o \in Q$ stare iniţială;
- $Z_o \in \Gamma$ simbolul de start al memoriei stivă;
- $F \subseteq Q$ mulţimea stărilor finale;
- $\delta : Q x(\Sigma \cup \{\varepsilon\}) x \Gamma \rightarrow \mathcal{P}_0 (Q x \Gamma^*)$ funcţia de tranziţie
  – are ca valori submulţimi finite din $Q x \Gamma^*$ (posibil multimea vida)

Determinism

$\mathbf{M = (Q, \Sigma, \Gamma, \delta, q_o, Z_o, F)}$ este ***determinist*** dacă:

$\forall \mathbf{Z} \in \Gamma$, $\forall \mathbf{q} \in \mathbf{Q}$, $\forall \mathbf{a} \in \Sigma$

1) $|\delta(\mathbf{q},\varepsilon,\mathbf{Z})| = \mathbf{0}$ si $|\delta(\mathbf{q},\mathbf{a},\mathbf{Z})| \mathbf{<=1}$

2) $|\delta(\mathbf{q},\varepsilon,\mathbf{Z})| = \mathbf{1}$ si $|\delta(\mathbf{q},\mathbf{a},\mathbf{Z})| = \mathbf{0}$

In caz contrar, automatul nu este determinist

Multimea limbajelor acceptate de APD nedeterministe este strict mai larga decat multimea limbajelor acceptate de APD deterministe

Limbaje independente de context. Teoreme de echivalenta

Teoremă.

Fie automatul push-down **M**. Există întotdeauna un automat push-down **M'** astfel încât $L_\varepsilon(M') = L_f(M)$; si reciproc.

Teoremă.

Oricare ar fi G – o gramatica independenta de context, există un automat push-down **M** astfel încât $L_\varepsilon(M) = L(G)$;

si reciproc.

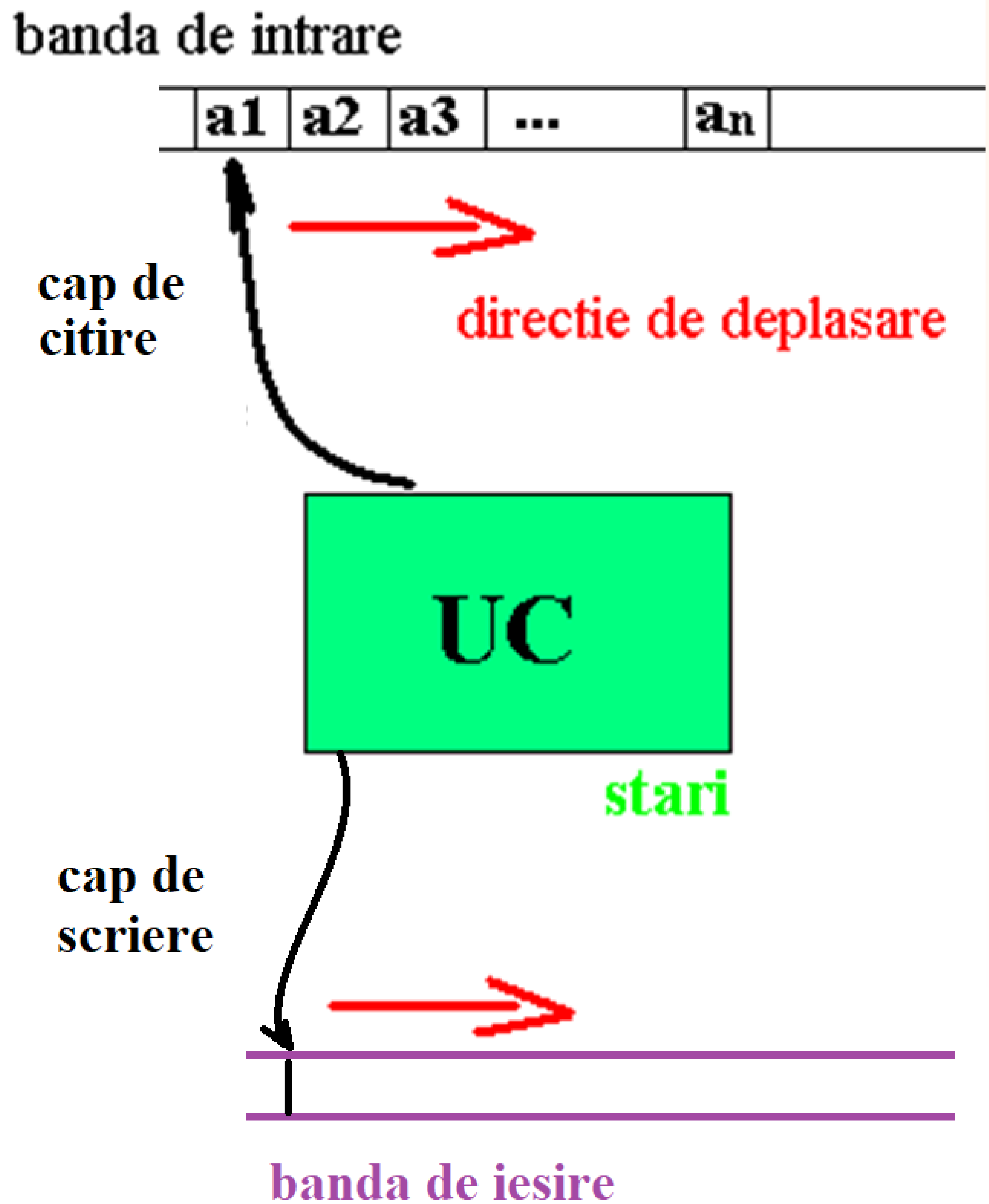
Translator finit

$M = (Q, \Sigma, D, \delta, q_o, F)$

- Q alfabetul stărilor;
- Σ alfabetul de intrare;
- D alfabetul de iesire;
- $q_o \in Q$ stare iniţială;
- $F \subseteq Q$ mulţimea stărilor finale;
- $\delta : Q x (\Sigma \cup \{\varepsilon\}) \rightarrow \mathcal{P}_0 (Q \text{ x } D^*)$

multimea partilor finite

Translator finit



Translator finit

Exemplu:

$M = (\quad Q, \qquad \Sigma, \quad D, \quad \delta, \quad q_o, \quad F \ )$

$M = (\{q_0, q_1\}, \quad \{a\}, \{b\}, \ \delta, \quad q_0, \quad \{q_1\} \ )$

$\delta(q_0, a) = \{q_1, \{b\}\}$

$d(q_1, \varepsilon) = \{q_1, \{b\}\}$

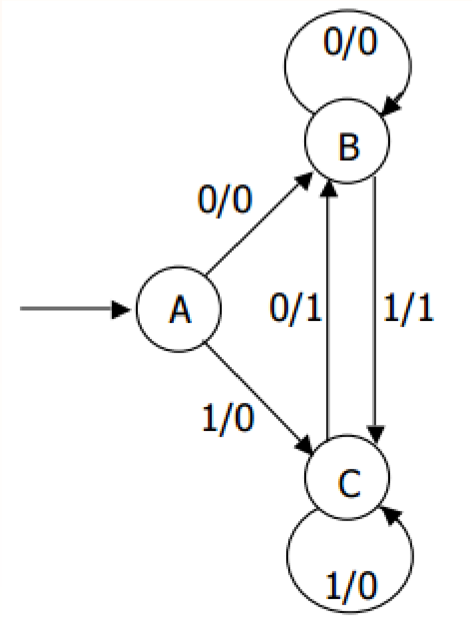
Translatarea definita de M:

$T(M)=\{(x,y) | \ x \in \Sigma^* , \ y \in D^*, \ (q_0,x,\varepsilon) \ |\text{-}^* \ (q,\varepsilon,y), \ q \in F)$

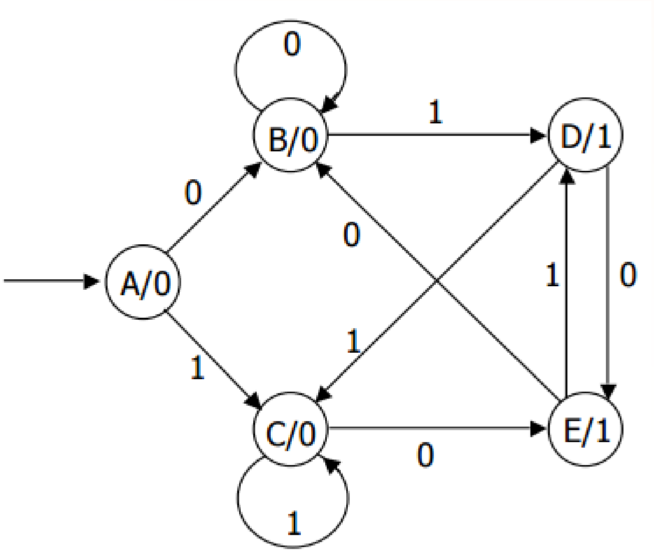
Alte automate

In literatura mai gasim:

- **Automatele Mealy**: la fiecare tranzitie se produce un simbol de ieşire.



- **Automatele Moore**: fiecarei stari (intrare intr-o stare) i se asociaza un simbol de ieşire.



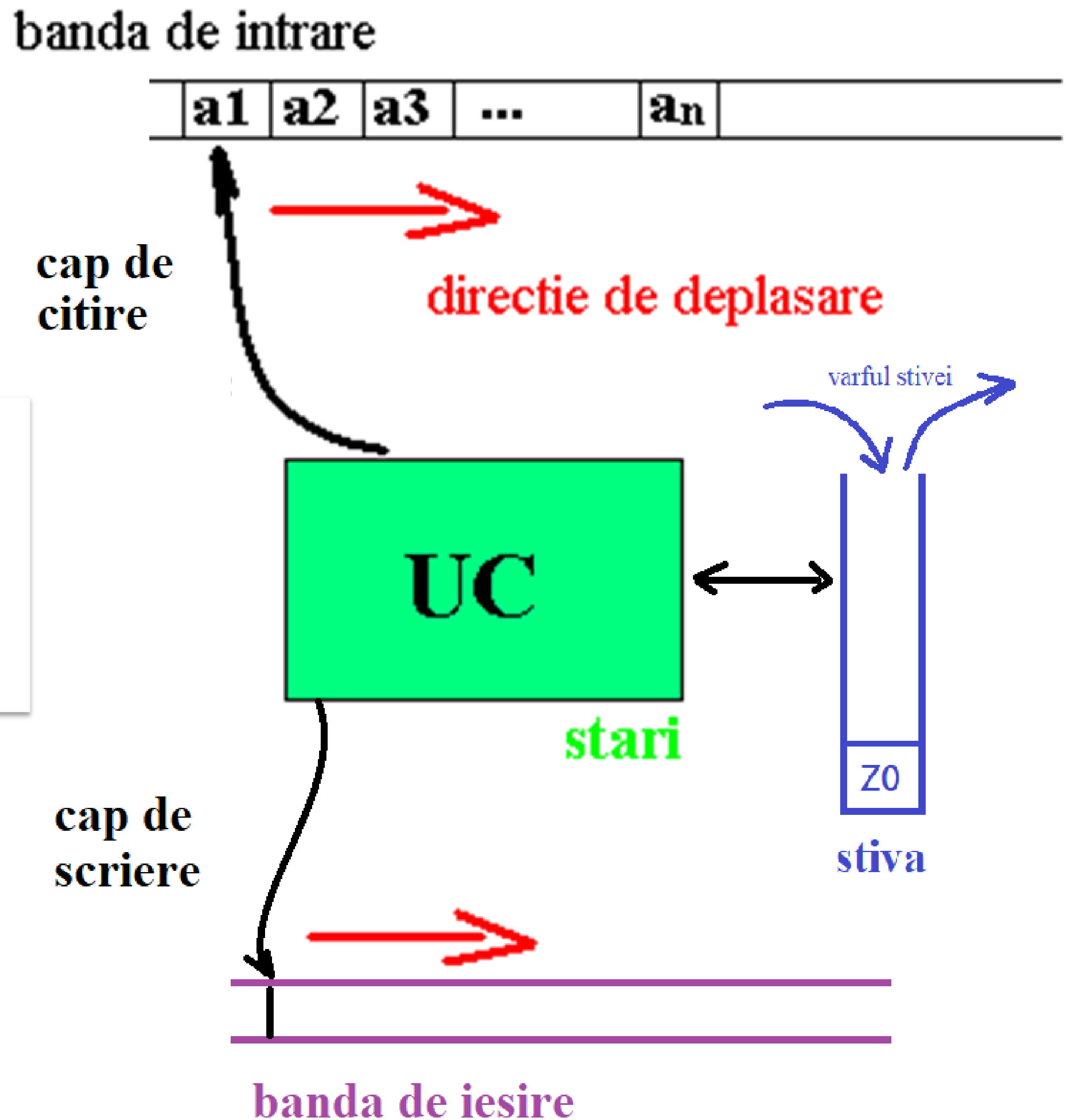
Translator push-down

$M = (Q, \Sigma , \Gamma, D, \delta, q_o, Z_0, F)$

- Q alfabetul stărilor;
- Σ alfabetul de intrare;
- Γ alfabetul memoriei stivă;
- D alfabetul de iesire;
- $q_o \in Q$ stare iniţială;
- $Z_o \in \Gamma$ simbolul de start al memoriei stivă;
- $F \subseteq Q$ mulţimea stărilor finale;
- $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow \mathcal{P}_0 (Q \times \Gamma^* \times D^*)$

multimea partilor finite

Translator push-down



Translator push-down

$M = (Q, \Sigma , \Gamma, D, \delta, q_o, Z_0, F)$

$Q = \{q\}$

$\Sigma = \{a,+,*\}$

$\Gamma = \{E, +, *\}$

$D = \{a, +, *\}$

$q_o=q$

$Z_o=E$

$\delta (q, a, E) = \{(q, \varepsilon, a)\}$

$\delta (q, +, E) = \{(q, EE+, \varepsilon)\}$

$\delta (q, *, E) = \{(q, EE*, \varepsilon)\}$

$\delta (q, \varepsilon, +) = \{(q, \varepsilon, +)\}$

$\delta (q, \varepsilon, *) = \{(q, \varepsilon, *)\}$

Considerand criteriul stivei vide,
descrieti translatarea pe care acesta o defineste .

… am lucrat si cu alte translatoare

Vezi:

LL(1)

LR(*)

Ne reamintim: Analizorul LL(1)

- Automat: (α,β,Π)
  - banda de intrare: α
  - stiva β (stiva de lucru)
  - banda de iesire Π => sirul regulilor de productie
- config. initiala: $(w\$, S\$, \varepsilon)$
- config. finala: $(\$, \$, \Pi)$
- tranzitii
  - push $(\mathbf{a}x\$, \mathbf{A}\beta, \Pi) \vdash (ax\$,\alpha\beta,\Pi\mathbf{i})$ dc.: $M(A,a) = (\alpha,i)$
  - pop $(\mathbf{a}x\$, \mathbf{a}\beta, \Pi) \vdash (x\$, \beta, \Pi)$
  - acc $(\$, \$, \Pi) \vdash acc$
  - err in celelalte cazuri

Automatul LL(1)
ca translator push-down (modificat)

[*Moldovan*]

Translatorul push-down modificat este:

$M = (\{q\}, \Sigma, \Gamma, D, \delta, q, S, \varnothing)$

Σ- *alfabet de intrare şi este acelaşi cu alfabetul* Σ *din G;*

D – *alfabet de ieşire,* $D = \{1,2,\ldots,m\}$

$i \in D$ *reprezintă nr. de ordine al producţiilor din P,* $i : A \to \alpha$, $i = \overline{1,m}$

Γ - *alfabetul memoriei push-down* $\Gamma = N \cup \Sigma$

q - *starea internă (stare iniţială)*

S – *simbolul de start în memoria push-down,* $S \in \Gamma$

δ – *funcţia de tranziţie modificată*

$\delta : \Sigma \times \Gamma \to P(\Gamma^* \times (D \cup \{\varepsilon\}) \times \{0,1\})$

$\delta(a,a) = \{(\varepsilon,\varepsilon,1)\} \quad \forall a \in \Sigma$

$\delta(a,A) = \{(x,i,0)\}$, $a \in \Sigma$, $A \in N$

dacă $(\exists)i : A \to x$ *şi dacă* $a \in \varphi(x)$, *sau*

dacă $\varepsilon \in \varphi(x) \Rightarrow a \in \varphi(A)$

În celelalte cazuri $\delta(.,.) = \varnothing$

1 : semnifică înaintarea benzii cu o poziţie;

0 : semnifică staţionarea benzii.

Limbajul acceptat se defineste dupa criteriul stivei vide.

Nu se da la examen

Ne reamintim:

Automatul LR

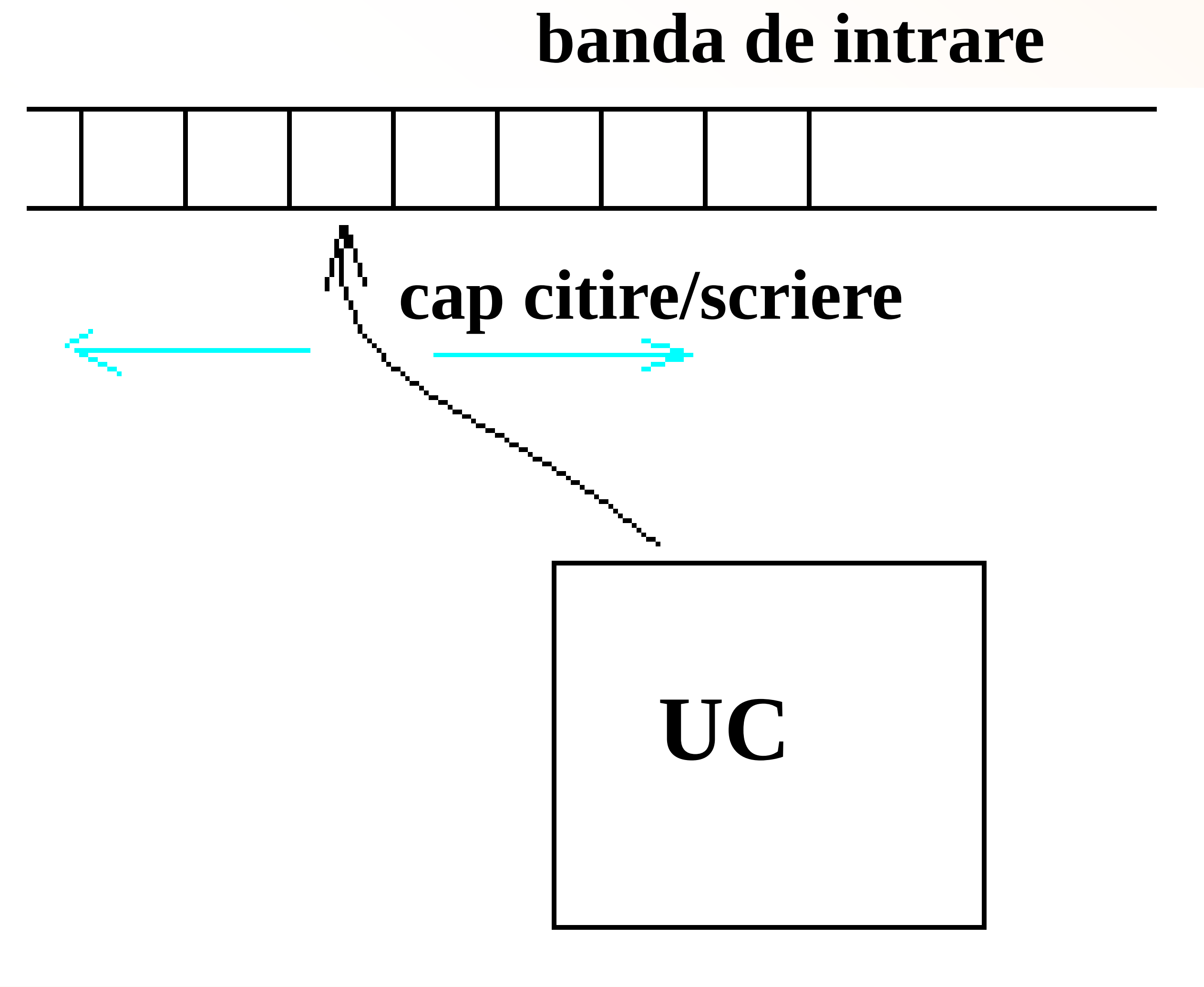
- configuratie:

  (α,β,Π)

  (stiva_de_lucru, banda_de_intrare, banda_de_iesire)
- pe stiva: prefixe viabile, stari ale analizorului

- config. initiala: $(\$0,w\$, \varepsilon)$
- config. finala: $(\$0S\ I_{acc}, \$, \Pi)$

Masini Turing



Masini Turing

O miscare a masinii Turing consta din:

- se schimba starea
- se inlocuieste simbolul curent de pe banda de intrare
- capul citire/scriere se muta cu o pozitie la stanga sau la dreapta

Masina Turing cu banda infinita

O masina Turing este: $M = (Q, \Sigma, \Gamma, \delta, q_0, \#, F)$

- Q – multime finita de stari
- Γ – multimea simbolurilor benzii
- # - un simbol din Γ, numit simbolul *blanc*
- Σ – o submultime a lui $\Gamma - \{\#\}$
- δ – este functia de tranzitie

$$\delta: Q \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L,R\})$$

- q_0 – starea initiala
- $F \subset Q$ – multimea starilor finale

Masina Turing cu banda infinita

- configuratie

  $\alpha_1\ q\ \alpha_2$ $\qquad$ $\alpha_1\alpha_2 \in \Gamma^*$, separate de capul de citire

  pana la cel mai din stanga/dreapta simbol ne-blank

- tranzitie

$(p,Y,L) \in \delta(q,X_i)$

$$X_1X_2\ldots X_{i-1}qX_i\,X_{i+1}\ldots X_n \vdash X_1X_2\ \ldots\ X_{i-2}\ p\ X_{i-1}YX_{i+1}\ldots X_n$$

$(p,Y,R) \in \delta(q,X_i)$

$$X_1X_2\ldots X_{i-1}qX_i\,X_{i+1}\ldots X_n \vdash X_1X_2\ \ldots\ X_{i-1}Y\ p\ X_{i+1}\ldots X_n$$

- limbaj acceptat

$$\{w \in \Sigma^* \mid q_0w \vdash \alpha_1q\alpha_2\ ,\ q \in F,\ \alpha_1\alpha_2 \in \Gamma^*\}$$

Exemplu: masina Turing

Functia de tranzitie

0011 ?

	Σ		$\Gamma - \Sigma$			
	0	**1**	**X**	**Y**	**#**	
q_0	(q_1,X,R)			(q_3,Y,R)		0
q_1	$(q_1,0,R)$	(q_2,Y,L)		(q_1,Y,R)		0
q_2	$(q_2,0,L)$		(q_0,X,R)	(q_2,Y,L)		0
q_3				(q_3,Y,R)	$(q_4,\#,R)$	0
q_4						1

Masini Turing

- Masina Turing cu o singura banda

  versus Masina Turing cu mai multe benzi

  O mașină Turing cu k benzi

  nu este mai puternică decât o mașină Turing standard
- Mașină Turing deterministă (MTD)

  versus mașină Turing nedeterministă (MTND)

  Cele două sunt computațional echivalente,

  adică orice MTND se poate transforma într-o MTD

  (și invers).

Masini Turing

Teza lui Church

- Logicianul Alonzo Church a emis ipoteza că maşina Turing este modelul cel mai general de calcul care poate fi propus.
  - maşina Turing este la fel de puternică ca orice alt model de calcul
  - nu înseamnă că poate calcula la fel de repede ca orice alt model de calcul, ci că poate calcula aceleaşi lucruri
- Acest enunţ nu este demonstrabil în sens matematic.

Dacă avem un model de calcul, putem defini precis ce înţelegem prin complexitate:

- **Timpul** de calcul pentru un şir dat la intrare: este numărul de mutări făcut de maşina Turing înainte de a intra în starea ``terminat";
- **Spaţiul** consumat pentru un şir de intrare: este numărul de căsuţe de pe bandă pe care algoritmul le foloseşte în timpul execuţiei sale.