

---

# Εισαγωγή στην Επιστήμη των Υπολογιστών

4ο εξάμηνο ΣΗΜΜΥ

---

*3η ενότητα:*

Αυτόματα, τυπικές γλώσσες, γραμματικές

*Επιμέλεια διαφανειών:* Στάθης Ζάχος, Άρης Παγουρτζής

<http://www.corelab.ece.ntua.gr/courses/introcs>

# Μηχανές πεπερασμένων καταστάσεων (FSM)

- Τρόπος κωδικοποίησης αλγορίθμων.
- Τρόπος περιγραφής **συστημάτων πεπερασμένων καταστάσεων**:
  - Συστήματα που έχουν **εσωτερικές καταστάσεις** και **προκαθορισμένο τρόπο μετάβασης** από μία κατάσταση σε άλλη με βάση την **τρέχουσα κατάσταση** και την **είσοδο** (συνήθως ενέργεια κάποιου χρήστη). Μπορεί να έχουν και **έξοδο**.
- Είναι μηχανές Turing χωρίς ταινία (ή, ισοδύναμα, με ταινία εισόδου «read-only» και ταινία εξόδου «write-only»).
- Εφαρμογές σε πλήθος επιστημονικών πεδίων.

# Παράδειγμα: μηχανήμα καφέ (i)

## *Προδιαγραφές*

- Δύο είδη καφέ: ελληνικός ή φρέντο.
- Κόστος καφέ: 40 λεπτά.
- Επιτρέπονται κέρματα 10, 20, ή 50 λεπτών.

# Παράδειγμα: μηχανήμα καφέ (ii)

## Σχεδίαση του συστήματος

- Εσωτερικές καταστάσεις:  $q_0, q_1, q_2, q_3, q_4, q_5$ 
  - $q_i$ : έχουν δοθεί μέχρι στιγμής  $10 \cdot i$  λεπτά
- Δυνατές είσοδοι (ενέργειες):  $P1, P2, P5, K1, K2$ 
  - $P1, P2, P5$ : εισαγωγή κέρματος 10, 20, ή 50 λεπτών
  - $K1, K2$ : κουμπί 1 για ελληνικό καφέ, 2 για φρέντο
- Δυνατές έξοδοι:  $E1, E2, E3, E4, E5, E\Lambda, \Phi P$ 
  - $E_i$ : επιστροφή  $10 \cdot i$  λεπτών
  - $E\Lambda$ : εξαγωγή ελληνικού καφέ
  - $\Phi P$ : εξαγωγή φρέντο

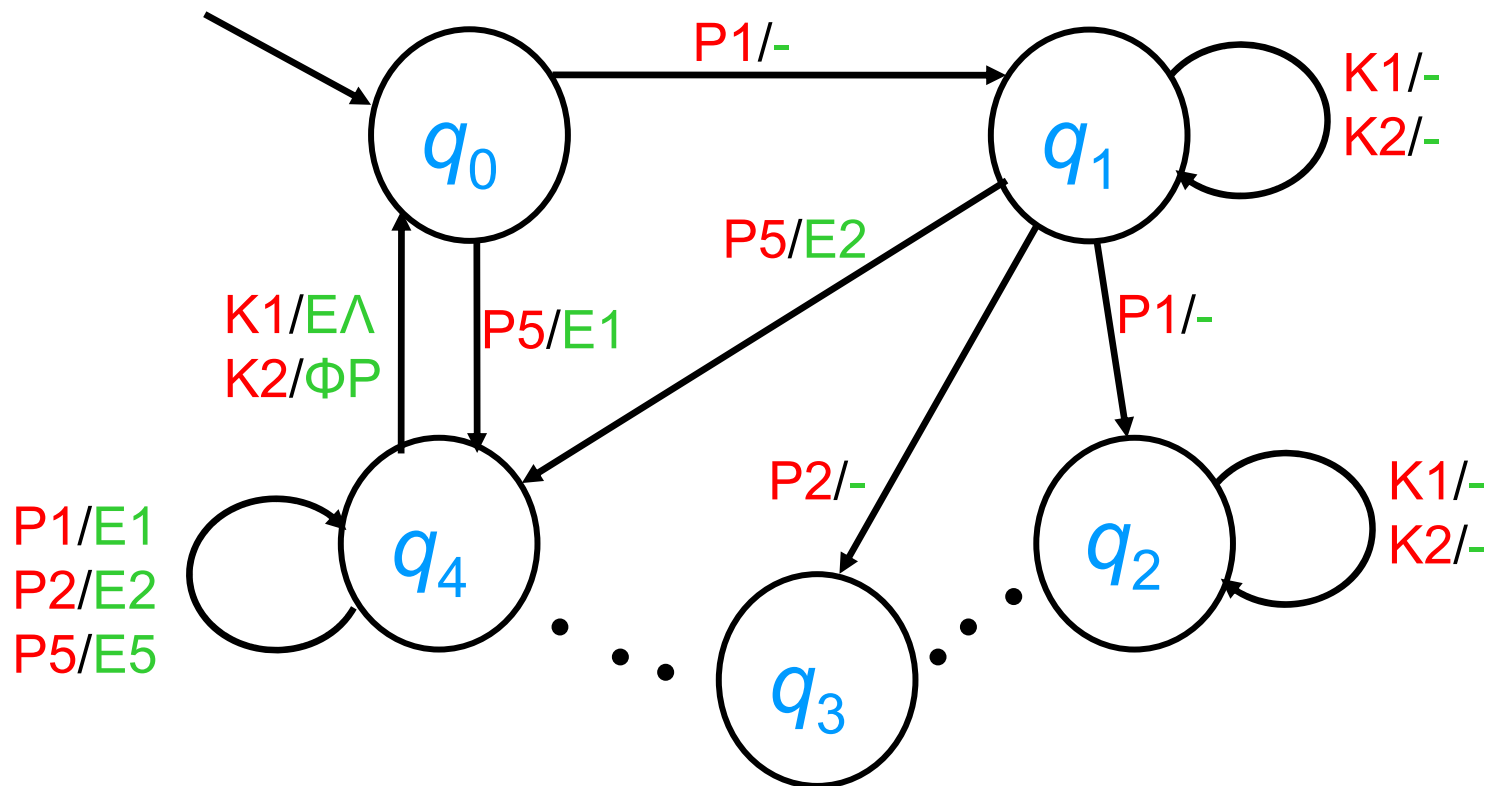
# Παράδειγμα: μηχανήμα καφέ (iii)

- **Πίνακας καταστάσεων:** δείχνει ποια είναι η επόμενη κατάσταση και η έξοδος για κάθε συνδυασμό τρέχουσας κατάστασης και εισόδου. Αρχική κατάσταση:  $q_0$ .

Είσοδος Κατάστ.	P1	P2	P5	K1	K2
$q_0$	$q_1, -$	$q_2, -$	$q_4, E1$	$q_0, -$	$q_0, -$
$q_1$	$q_2, -$	$q_3, -$	$q_4, E2$	$q_1, -$	$q_1, -$
$q_2$	$q_3, -$	$q_4, -$	$q_4, E3$	$q_2, -$	$q_2, -$
$q_3$	$q_4, -$	$q_4, E1$	$q_4, E4$	$q_3, -$	$q_3, -$
$q_4$	$q_4, E1$	$q_4, E2$	$q_4, E5$	$q_0, E\Lambda$	$q_0, \Phi P$

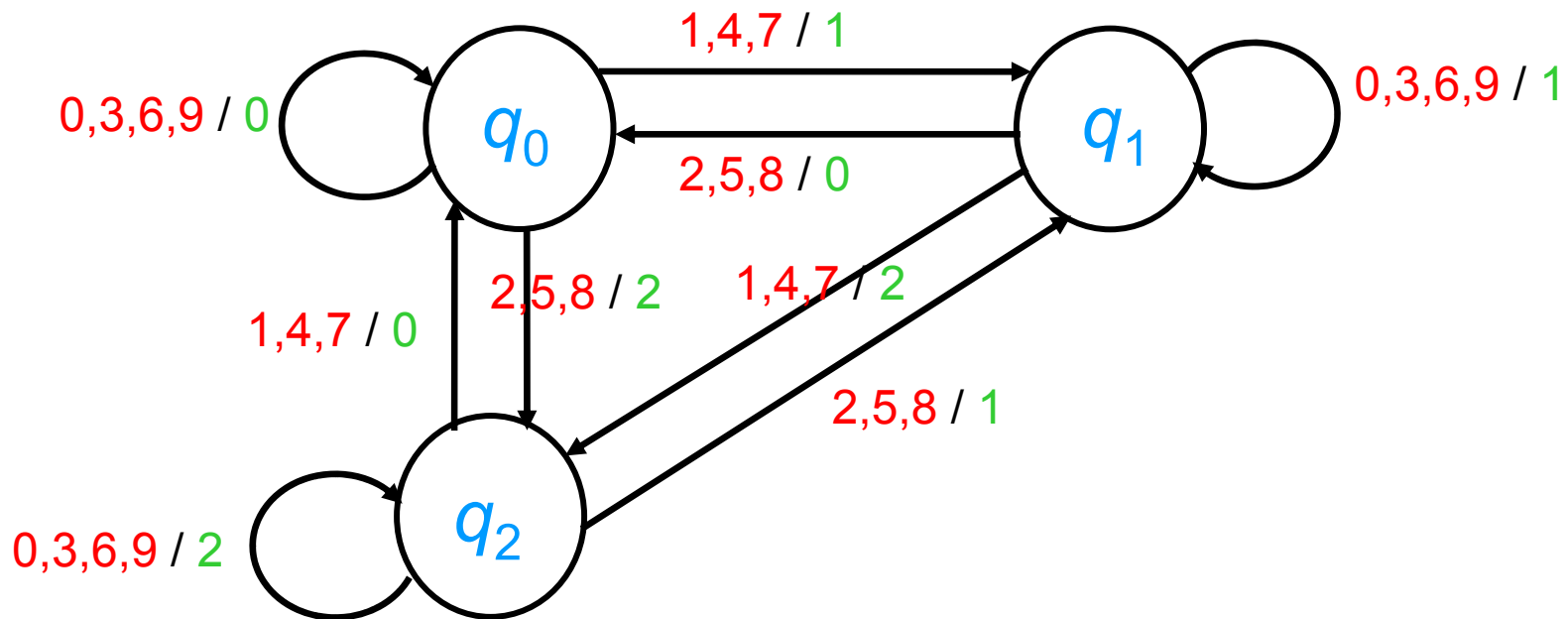
# Παράδειγμα: μηχανήμα καφέ (iv)

- **Διάγραμμα καταστάσεων:** παρέχει τις ίδιες πληροφορίες με τον πίνακα καταστάσεων με πιο εποπτικό τρόπο. Αρχική κατάσταση:  $q_0$  (σημειώνεται με βέλος).



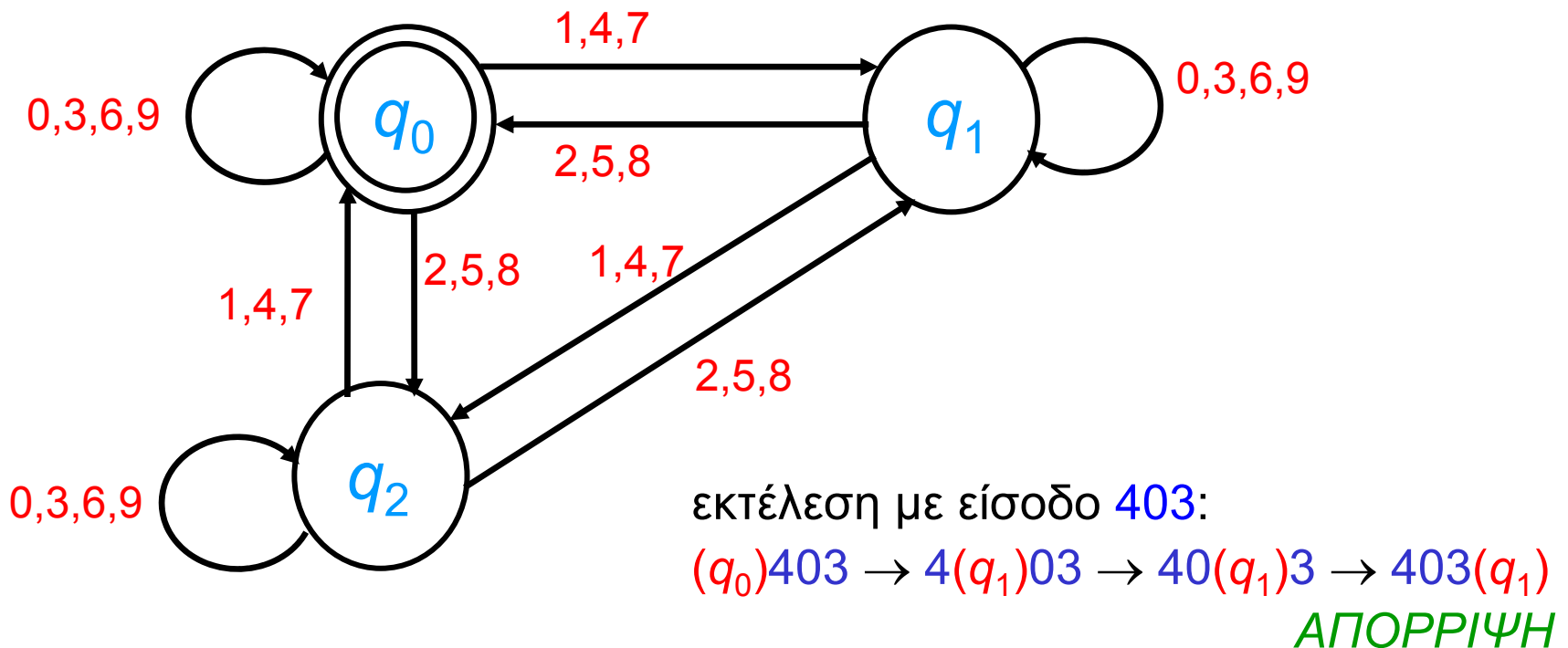
# Παράδειγμα II: αριθμητική modulo

- Κατασκευή μηχανής που να κάνει την πράξη  $n \bmod 3$
- **Πόσες** καταστάσεις χρειαζόμαστε;
- Χρήση ιδιότητας:  $n \bmod 3 = (n_1 + \dots + n_k) \bmod 3$ ,  
 $n_i$  τα (δεκαδικά) ψηφία του  $n$  *Αποδείξτε το!*



# Παράδειγμα II: αριθμητική modulo

- **Απλοποίηση**: αν ενδιαφέρει μόνο η διαιρετότητα με το 3 δεν χρειάζεται έξοδος
- Ορίζουμε **καταστάσεις αποδοχής** (διπλός κύκλος)





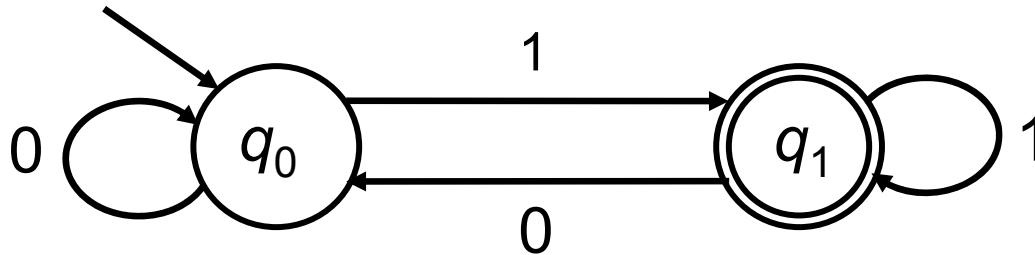
# Αυτόματα

- Μηχανές πεπερασμένων καταστάσεων χωρίς έξοδο: κάποιες καταστάσεις **αποδέχονται**, ενώ οι υπόλοιπες **απορρίπτουν**. Καταστάσεις αποδοχής συμβολίζονται με επιπλέον κύκλο.
- Ένα αυτόματο έχει κάποιες εσωτερικές **καταστάσεις**  $q_0$ ,  $q_1$ ,  $q_7$ ,  $q_{15}$ , ... , και μια **συνάρτηση μετάβασης**  $\delta$  που καθορίζει την **επόμενη κατάσταση** του αυτομάτου με βάση την **τρέχουσα κατάσταση** και την **συμβολοσειρά εισόδου**. Αποδέχεται ή απορρίπτει τη συμβολοσειρά εισόδου.
- **Αναγνωριστές** γλωσσών (δηλαδή επιλύουν **προβλήματα απόφασης**).

# Αυτόματα και τυπικές γλώσσες

- **Τυπικές γλώσσες:** χρησιμοποιούνται για την περιγραφή υπολογιστικών προβλημάτων αλλά και γλωσσών προγραμματισμού.  
Π.χ.  $L = \{x \in \{0,1\}^* \mid x \text{ δυαδική γραφή πρώτου αριθμού}\}$
- **Αυτόματα:** χρησιμεύουν για την αναγνώριση τυπικών γλωσσών και για την κατάταξη της δυσκολίας των αντίστοιχων προβλημάτων:
  - Κάθε αυτόματο αναγνωρίζει μια τυπική γλώσσα: το σύνολο των συμβολοσειρών που το οδηγούν σε κατάσταση αποδοχής.

# Παράδειγμα: αναγνώριση περιττών



- $q_0$ : τελευταίο ψηφίο **διάφορο του 1**
- $q_1$ : τελευταίο ψηφίο **ίσο με 1**
- η  $q_0$  λέγεται **αρχική κατάσταση** ενώ η  $q_1$  λέγεται **κατάσταση αποδοχής (ή τελική)**
- εκτέλεση με είσοδο 0110:  
 $(q_0)0110 \rightarrow 0(q_0)110 \rightarrow 01(q_1)10 \rightarrow 011(q_1)0 \rightarrow 0110(q_0)$  **ΑΠΟΡΡΙΨΗ**
- εκτέλεση με είσοδο 101:  
 $(q_0)101 \rightarrow 1(q_1)01 \rightarrow 10(q_0)1 \rightarrow 101(q_1)$  **ΑΠΟΔΟΧΗ**

# Άλλα αυτόματα

- **Μηχανισμοί**: χωρίς είσοδο – έξοδο:  $\delta(q_i) = q_j$   
εκτέλεση:  $q_0 \rightarrow q_j \rightarrow q_k \rightarrow q_m \dots$
- **Αυτόματα στοίβας (PDA, pushdown automata)**: έχουν πολύ περισσότερες δυνατότητες καθώς μπορούν να χρησιμοποιήσουν **μνήμη** (σε μορφή **στοίβας**).
- **Μηχανές Turing (TM)**: έχουν ακόμη περισσότερες δυνατότητες καθώς έχουν **απεριόριστη μνήμη** (σε μορφή **ταινίας, με δυνατότητα επιστροφής**).
- **Γραμμικά περιορισμένα αυτόματα (LBA)**: είναι TM με μνήμη **γραμμικά περιορισμένη** (ως προς το μήκος της εισόδου).

# Άλλες τυπικές γλώσσες

$$L_1 := \{w \in \{a, b\}^* \mid w \text{ αρχίζει με 'a'}\}$$

$$L_2 := \{w \in \{1, 3\}^* \mid w \text{ περιέχει άρτιο πλήθος '1'}\}$$

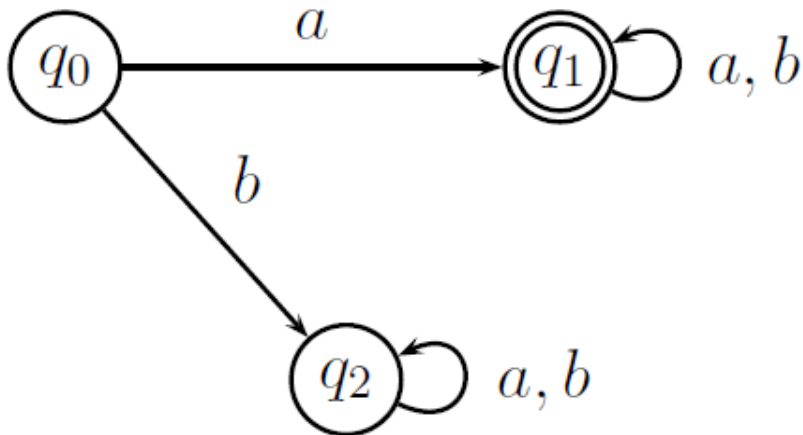
$$L_3 := \{w \in \{a, b\}^* \mid w \text{ είναι παλινδρομική}\}$$

# Ορισμός DFA

- Ντετερμινιστικό πεπερασμένο αυτόματο (Deterministic Finite Automaton, **DFA**): πεντάδα  $M = (Q, \Sigma, \delta, q_0, F)$ 
  - $Q$  : το σύνολο των καταστάσεων του  $M$  (πεπερασμένο), π.χ.  $Q = \{q_0, q_1, q_7, q_{15}\}$
  - $\Sigma$  : πεπερασμένο αλφάβητο εισόδου ( $\Sigma \cap Q = \emptyset$ ), π.χ.  $\Sigma = \{0, 1\}$
  - $\delta : Q \times \Sigma \rightarrow Q$  : συνάρτηση μετάβασης, π.χ.  $\delta(q_i, 0) = q_j$
  - $q_0 \in Q$  : αρχική κατάσταση
  - $F \subseteq Q$  : σύνολο τελικών καταστάσεων (αποδοχής), π.χ.  $F = \{q_1, q_{15}\}$

# Παράδειγμα DFA

$$L_1 := \{w \in \{a, b\}^* \mid w \text{ αρχίζει με 'a'}\}$$



	a	b
q <sub>0</sub>	q <sub>1</sub>	q <sub>2</sub>
q <sub>1</sub>	q <sub>1</sub>	q <sub>1</sub>
q <sub>2</sub>	q <sub>2</sub>	q <sub>2</sub>

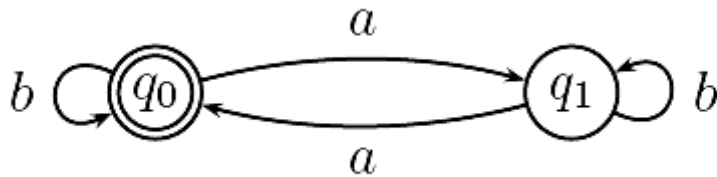
εκτέλεση με είσοδο *abba* :

$(q_0)abba \rightarrow a(q_1)bba \rightarrow ab(q_1)ba \rightarrow abb(q_1)a \rightarrow abba(q_1)$

ΑΠΟΔΟΧΗ

# Γλώσσα με DFA και γλώσσα χωρίς DFA

$$L_2 := \{w \in \{a, b\}^* \mid w \text{ περιέχει άρτιο πλήθος 'a'}\}$$



	a	b
$q_0$	$q_1$	$q_0$
$q_1$	$q_0$	$q_1$

$$L_3 := \{w \in \{a, b\}^* \mid w \text{ είναι παλινδρομική}\}$$

**δεν υπάρχει** DFA που να αναγνωρίζει την  $L_3$ !  
(χρειάζεται μνήμη με μέγεθος που εξαρτάται από την είσοδο – αυστηρή απόδειξη αργότερα)



# Αποδοχή DFA: τυπικοί ορισμοί

- Επέκταση συνάρτησης  $\delta: Q \times \Sigma^* \rightarrow Q$

Η επεκτεταμένη  $\delta$  δέχεται ως ορίσματα μια κατάσταση  $q$  και μια συμβολοσειρά  $u$  και δίνει την κατάσταση όπου θα βρεθεί το αυτόματο αν ξεκινήσει από την  $q$  και διαβάσει την  $u$ .

- Ορισμός επεκτεταμένης  $\delta$  (σχήμα *πρωταρχικής αναδρομής*):

$$\begin{cases} \delta(q, \varepsilon) = q \\ \delta(q, wa) = \delta(\delta(q, w), a) \end{cases}$$

όπου  $w$  είναι συμβολοσειρά οποιουδήποτε μήκους, ενώ  $a$  απλό σύμβολο του αλφαβήτου

# Αποδοχή DFA: τυπικοί ορισμοί

- Ένα DFA αποδέχεται μία συμβολοσειρά  $u$  αν  $\delta(q_0, u) \in F$

- Ένα DFA  $M$  αποδέχεται τη γλώσσα

$$L(M) = \{w \mid \delta(q_0, w) \in F\}$$

- Οι γλώσσες που γίνονται αποδεκτές από DFA λέγονται **κανονικές**

# Μη ντετερμινιστικά αυτόματα

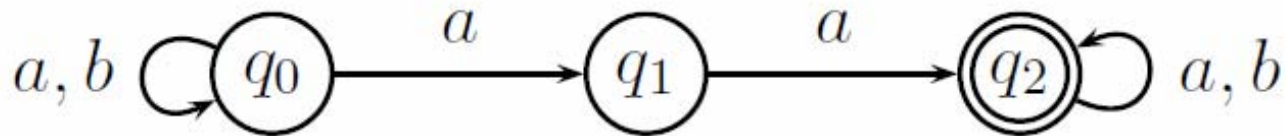
- **Ντετερμινιστικά αυτόματα:** για κάθε συνδυασμό κατάστασης / συμβόλου εισόδου υπάρχει **μοναδική** επόμενη κατάσταση
- **Μη-ντετερμινιστικά αυτόματα:**
  - για κάθε συνδυασμό κατάστασης / συμβόλου εισόδου υπάρχει **επιλογή** από σύνολο δυνατών επόμενων καταστάσεων
  - αποδοχή αν **κάποια** σειρά επιλογών οδηγεί σε αποδοχή

# Μη ντετερμινιστικά πεπερασμένα αυτόματα

- **NFA** (Non-deterministic Finite Automaton): για κάθε κατάσταση και σύμβολο εισόδου επιλέγεται μία από ένα **σύνολο δυνατών επόμενων** καταστάσεων.
- **NFA $\epsilon$**  (NFA με  **$\epsilon$ -κινήσεις**): μπορεί να αλλάζει κατάσταση **χωρίς ανάγνωση** επόμενου συμβόλου.

# Παράδειγμα NFA

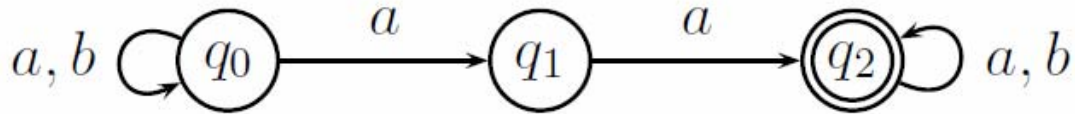
$L_4 := \{w \in \Sigma^* \mid w \text{ περιέχει δύο συνεχόμενα } a\}$



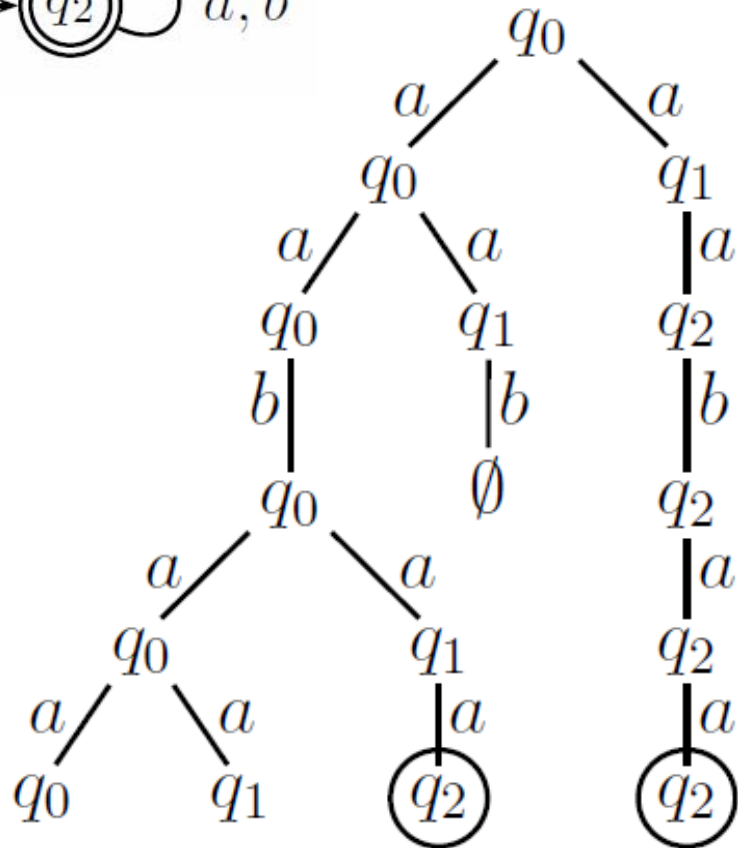
	a	b
q <sub>0</sub>	{q <sub>0</sub> , q <sub>1</sub> }	{q <sub>0</sub> }
q <sub>1</sub>	{q <sub>2</sub> }	∅
q <sub>2</sub>	{q <sub>2</sub> }	{q <sub>2</sub> }

Στη συνάρτηση μετάβασης, **κενό σύνολο** σημαίνει ότι η τρέχουσα εκτέλεση **απορρίπτει** (προσοχή: **μπορεί κάποια άλλη να αποδέχεται**).

# Παράδειγμα NFA



Δένδρο υπολογισμού  
για είσοδο *aaba*



ΑΠΟΔΟΧΗ

# Τυπικός ορισμός NFA

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

- $Q$  : το σύνολο των καταστάσεων του  $M$  (πεπερασμένο)
- $\Sigma$  : πεπερασμένο αλφάβητο εισόδου ( $\Sigma \cap Q = \emptyset$ )
- $\delta : Q \times \Sigma \rightarrow \text{Pow}(Q)$  : συνάρτηση μετάβασης,  
π.χ.  $\delta(q_j, 1) = \{q_j, q_k, q_m\}$
- $q_0 \in Q$  : αρχική κατάσταση
- $F \subseteq Q$  : σύνολο τελικών καταστάσεων (αποδοχής)

Στη συνάρτηση μετάβασης, **κενό σύνολο** σημαίνει ότι η τρέχουσα εκτέλεση **απορρίπτει** (**προσοχή**: μπορεί κάποια άλλη να αποδέχεται).

# Αποδοχή NFA: τυπικοί ορισμοί

- Ένα NFA αποδέχεται συμβολοσειρά  $w$  αν  $\delta(q_0, w) \cap F \neq \emptyset$
- Ένα NFA  $M$  αποδέχεται τη γλώσσα

$$L(M) = \{w \mid \delta(q_0, w) \cap F \neq \emptyset\}$$

- Σημείωση: η συνάρτηση  $\delta$  είναι **επεκτεταμένη** ώστε να δέχεται σαν ορίσματα μια κατάσταση  $q$  και μια συμβολοσειρά  $w$  και να δίνει το **σύνολο των καταστάσεων** όπου μπορεί να βρεθεί το αυτόματο αν ξεκινήσει από την  $q$  και διαβάσει την  $w$



# Ισοδυναμία DFA και NFA (i)

- **Θεώρημα Rabin-Scott:** για κάθε NFA υπάρχει ένα DFA που αποδέχεται την ίδια γλώσσα.
- Επομένως τα DFA και τα NFA αναγνωρίζουν ακριβώς την ίδια κλάση γλωσσών (**κανονικές γλώσσες**).

# Ισοδυναμία DFA και NFA (ii)

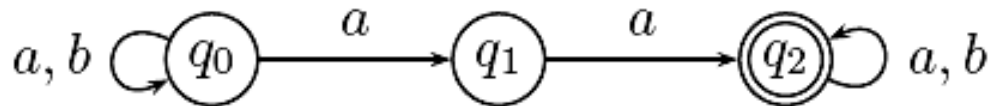
Έστω το NFA  $M = (Q, \Sigma, q_0, F, \delta)$ .

Ένα ισοδύναμο DFA  $M' = (Q', \Sigma, q'_0, F', \delta')$ , ορίζεται ως εξής:

- $Q' = \text{Pow}(Q)$ , δηλαδή οι καταστάσεις του  $M'$  είναι όλα τα υποσύνολα καταστάσεων του  $M$ .
- $q'_0 = \{q_0\}$ ,
- $F' = \{R \in Q' \mid R \cap F \neq \emptyset\}$ , δηλαδή μια κατάσταση του  $M'$  είναι τελική αν περιέχει μια τελική κατάσταση του  $M$ .
- $\delta'(R, a) = \{q \in Q \mid q \in \delta(r, a) \text{ για } r \in R\}$ , δηλαδή  $\delta'(R, a)$  είναι το σύνολο των καταστάσεων όπου μπορεί να βρεθεί το  $M$  ξεκινώντας από οποιαδήποτε κατάσταση του  $R$  και διαβάζοντας  $a$ .

# Εύρεση ισοδύναμου DFA (i)

NFA για τη γλώσσα  $L_4$  ("2 συνεχόμενα  $a$ ")



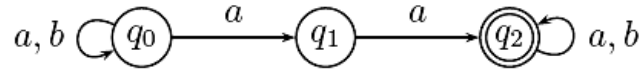
	$a$	$b$
$q_0$	$\{q_0, q_1\}$	$\{q_0\}$
$q_1$	$\{q_2\}$	$\emptyset$
$q_2$	$\{q_2\}$	$\{q_2\}$

DFA για τη γλώσσα  $L_4$

$Q' \setminus \Sigma$	$a$	$b$
$\emptyset$	$\emptyset$	$\emptyset$
$\{q_0\}$	$\{q_0, q_1\}$	$\{q_0\}$
$\{q_1\}$	$\{q_2\}$	$\emptyset$
$\{q_2\}$	$\{q_2\}$	$\{q_2\}$
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	$\{q_0\}$
$\{q_0, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_2\}$
$\{q_1, q_2\}$	$\{q_2\}$	$\{q_2\}$
$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_2\}$

# Εύρεση ισοδύναμου DFA (ii)

NFA για τη γλώσσα  $L_4$



	a	b
q <sub>0</sub>	{q <sub>0</sub> , q <sub>1</sub> }	{q <sub>0</sub> }
q <sub>1</sub>	{q <sub>2</sub> }	∅
q <sub>2</sub>	{q <sub>2</sub> }	{q <sub>2</sub> }

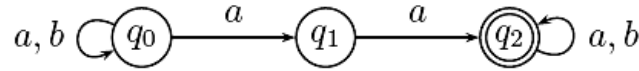
DFA για τη γλώσσα  $L_4$

$Q' \setminus \Sigma$	a	b
∅	∅	∅
✓ {q <sub>0</sub> }	{q <sub>0</sub> , q <sub>1</sub> }	{q <sub>0</sub> }
{q <sub>1</sub> }	{q <sub>2</sub> }	∅
{q <sub>2</sub> }	{q <sub>2</sub> }	{q <sub>2</sub> }
{q <sub>0</sub> , q <sub>1</sub> }	{q <sub>0</sub> , q <sub>1</sub> , q <sub>2</sub> }	{q <sub>0</sub> }
{q <sub>0</sub> , q <sub>2</sub> }	{q <sub>0</sub> , q <sub>1</sub> , q <sub>2</sub> }	{q <sub>0</sub> , q <sub>2</sub> }
{q <sub>1</sub> , q <sub>2</sub> }	{q <sub>2</sub> }	{q <sub>2</sub> }
{q <sub>0</sub> , q <sub>1</sub> , q <sub>2</sub> }	{q <sub>0</sub> , q <sub>1</sub> , q <sub>2</sub> }	{q <sub>0</sub> , q <sub>2</sub> }

Υπόδειξη: μπορούμε να εξετάζουμε μόνο **προσβάσιμα** σύνολα καταστάσεων.

# Εύρεση ισοδύναμου DFA (iii)

NFA για τη γλώσσα  $L_4$



	a	b
q <sub>0</sub>	{q <sub>0</sub> , q <sub>1</sub> }	{q <sub>0</sub> }
q <sub>1</sub>	{q <sub>2</sub> }	∅
q <sub>2</sub>	{q <sub>2</sub> }	{q <sub>2</sub> }

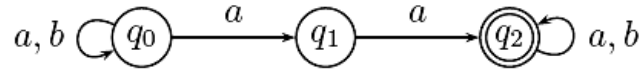
DFA για τη γλώσσα  $L_4$

$Q' \setminus \Sigma$	a	b
∅	∅	∅
✓ {q <sub>0</sub> }	{q <sub>0</sub> , q <sub>1</sub> }	{q <sub>0</sub> }
{q <sub>1</sub> }	{q <sub>2</sub> }	∅
{q <sub>2</sub> }	{q <sub>2</sub> }	{q <sub>2</sub> }
✓ {q <sub>0</sub> , q <sub>1</sub> }	{q <sub>0</sub> , q <sub>1</sub> , q <sub>2</sub> }	{q <sub>0</sub> }
{q <sub>0</sub> , q <sub>2</sub> }	{q <sub>0</sub> , q <sub>1</sub> , q <sub>2</sub> }	{q <sub>0</sub> , q <sub>2</sub> }
{q <sub>1</sub> , q <sub>2</sub> }	{q <sub>2</sub> }	{q <sub>2</sub> }
{q <sub>0</sub> , q <sub>1</sub> , q <sub>2</sub> }	{q <sub>0</sub> , q <sub>1</sub> , q <sub>2</sub> }	{q <sub>0</sub> , q <sub>2</sub> }

Υπόδειξη: μπορούμε να εξετάζουμε μόνο **προσβάσιμα** σύνολα καταστάσεων.

# Εύρεση ισοδύναμου DFA (iv)

NFA για τη γλώσσα  $L_4$



	a	b
q0	{q0, q1}	{q0}
q1	{q2}	$\emptyset$
q2	{q2}	{q2}

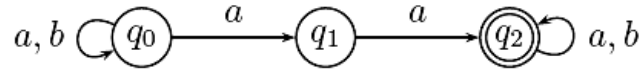
DFA για τη γλώσσα  $L_4$

$Q \setminus \Sigma$	a	b
$\emptyset$	$\emptyset$	$\emptyset$
✓ {q0}	{q0, q1}	{q0}
{q1}	{q2}	$\emptyset$
{q2}	{q2}	{q2}
✓ {q0, q1}	{q0, q1, q2}	{q0}
{q0, q2}	{q0, q1, q2}	{q0, q2}
{q1, q2}	{q2}	{q2}
✓ {q0, q1, q2}	{q0, q1, q2}	{q0, q2}

Υπόδειξη: μπορούμε να εξετάζουμε μόνο **προσβάσιμα** σύνολα καταστάσεων.

# Εύρεση ισοδύναμου DFA (v)

NFA για τη γλώσσα  $L_4$



	a	b
q0	{q0, q1}	{q0}
q1	{q2}	$\emptyset$
q2	{q2}	{q2}

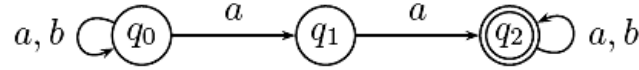
DFA για τη γλώσσα  $L_4$

Υπόδειξη: μπορούμε να εξετάζουμε μόνο **προσβάσιμα** σύνολα καταστάσεων.

$Q' \setminus \Sigma$	a	b
$\emptyset$	$\emptyset$	$\emptyset$
✓ {q0}	{q0, q1}	{q0}
{q1}	{q2}	$\emptyset$
{q2}	{q2}	{q2}
✓ {q0, q1}	{q0, q1, q2}	{q0}
✓ {q0, q2}	{q0, q1, q2}	{q0, q2}
{q1, q2}	{q2}	{q2}
✓ {q0, q1, q2}	{q0, q1, q2}	{q0, q2}

# Εύρεση ισοδύναμου DFA (vi)

NFA για τη γλώσσα  $L_4$

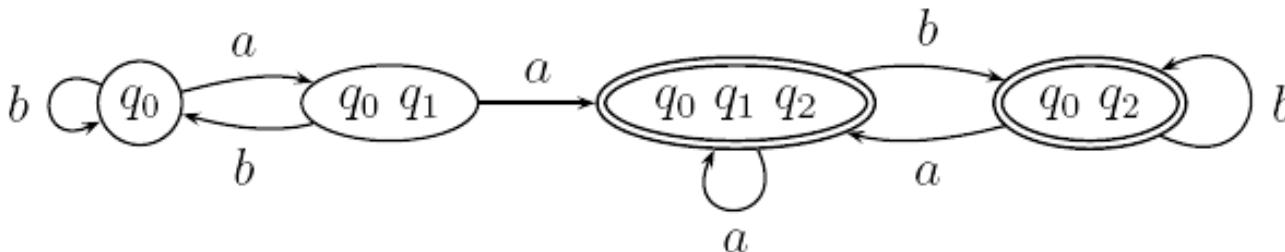


	a	b
q0	{q0, q1}	{q0}
q1	{q2}	$\emptyset$
q2	{q2}	{q2}

DFA για τη γλώσσα  $L_4$

$Q' \setminus \Sigma$	a	b
$\emptyset$	$\emptyset$	$\emptyset$
✓ {q0}	{q0, q1}	{q0}
<del>{q1}</del>	<del>{q2}</del>	<del><math>\emptyset</math></del>
<del>{q2}</del>	<del>{q2}</del>	<del>{q2}</del>
✓ {q0, q1}	{q0, q1, q2}	{q0}
✓ {q0, q2}	{q0, q1, q2}	{q0, q2}
<del>{q1, q2}</del>	<del>{q2}</del>	<del>{q2}</del>
✓ {q0, q1, q2}	{q0, q1, q2}	{q0, q2}

Υπόδειξη: μπορούμε να εξετάζουμε μόνο *προσβάσιμα* σύνολα καταστάσεων.

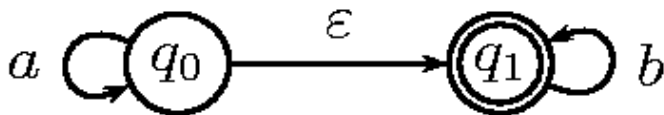




# Αυτόματα με $\epsilon$ -κινήσεις: $NFA_{\epsilon}$

- Επιτρέπουν **μεταβάσεις χωρίς να διαβάζεται σύμβολο** (ισοδύναμα: με είσοδο το κενό string  $\epsilon$ ).
- Αποδέχονται τις συμβολοσειρές που μπορούν να οδηγήσουν σε τελική κατάσταση, χρησιμοποιώντας ενδεχομένως και  $\epsilon$ -κινήσεις.

**Παράδειγμα:**  $NFA_{\epsilon}$  για  $L_5 := \{a^*b^*\} = \{a^n b^m \mid n, m \in \mathbb{N}\}$



	$a$	$b$	$\epsilon$
$q_0$	$\{q_0\}$	$\emptyset$	$\{q_0, q_1\}$
$q_1$	$\emptyset$	$\{q_1\}$	$\{q_1\}$

# Ισοδυναμία $NFA_\epsilon$ με DFA

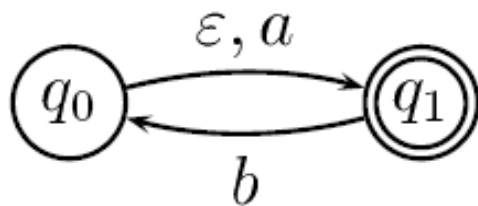
Έστω το  $NFA_\epsilon M = (Q, \Sigma, q_0, F, \delta)$ .

Ένα ισοδύναμο DFA  $M' = (Q', \Sigma, q'_0, F', \delta')$ , ορίζεται ως εξής:

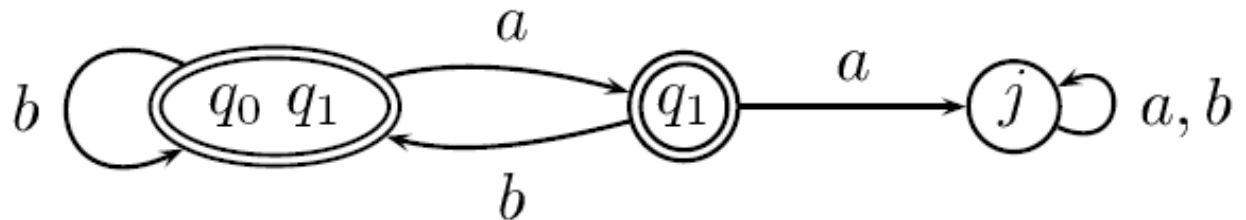
- $Q' = \text{Pow}(Q)$ , δηλαδή οι καταστάσεις του  $M'$  είναι όλα τα υποσύνολα καταστάσεων του  $M$ .
- $q'_0 = \epsilon\text{-κλείσιμο}(q_0) = \{p \mid p \text{ προσβάσιμο από } q_0 \text{ μόνο με } \epsilon\text{-κινήσεις}\}$ ,
- $F' = \{R \in Q' \mid R \cap F \neq \emptyset\}$ , δηλαδή μια κατάσταση του  $M'$  είναι τελική αν περιέχει μια τελική κατάσταση του  $M$ .
- $\delta'(R, a) = \{q \in Q \mid q \in \epsilon\text{-κλείσιμο}(\delta(r, a)) \text{ για } r \in R\}$ , δηλαδή  $\delta'(R, a)$  είναι το σύνολο των καταστάσεων όπου μπορεί να βρεθεί το  $M$  ξεκινώντας από **οποιαδήποτε κατάσταση του  $R$**  και διαβάζοντας  $a$  και χρησιμοποιώντας στη συνέχεια  $\epsilon\text{-κινήσεις}$ .

# Παράδειγμα ισοδυναμίας $NFA_\varepsilon$ με DFA

$NFA_\varepsilon$  για  $\overline{L_4}$  (δηλαδή “όχι δύο συνεχόμενα  $a$ ”):

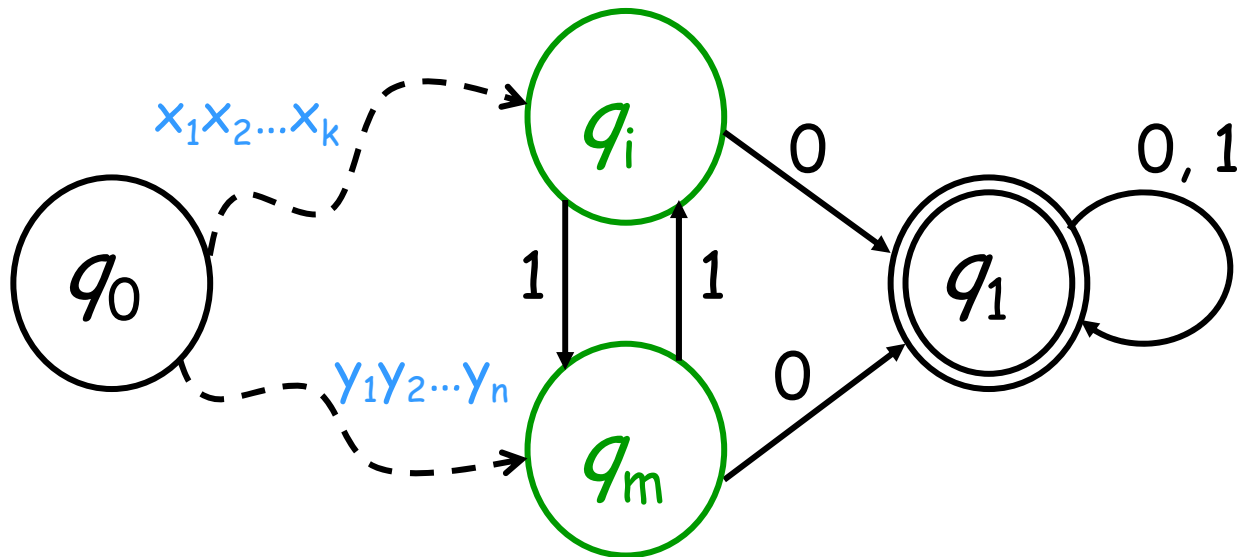


DFA για  $\overline{L_4}$ :



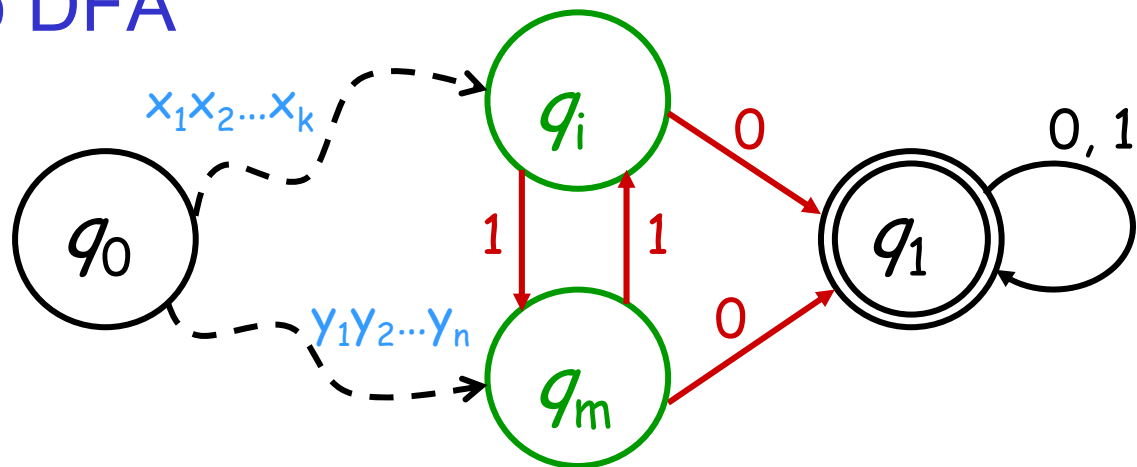
# Ελαχιστοποίηση DFA (i)

- Δύο καταστάσεις μπορούν να **συγχωνευτούν σε μία** (είναι **ισοδύναμες**) αν:  
*οδηγούν με ίδιες συμβολοσειρές σε ίδιο αποτέλεσμα*

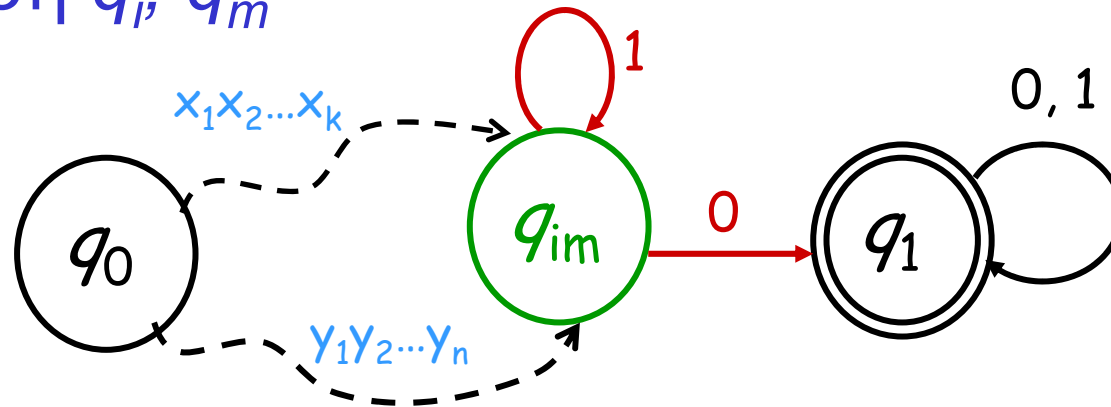


# Ελαχιστοποίηση DFA (ii)

Αρχικό DFA

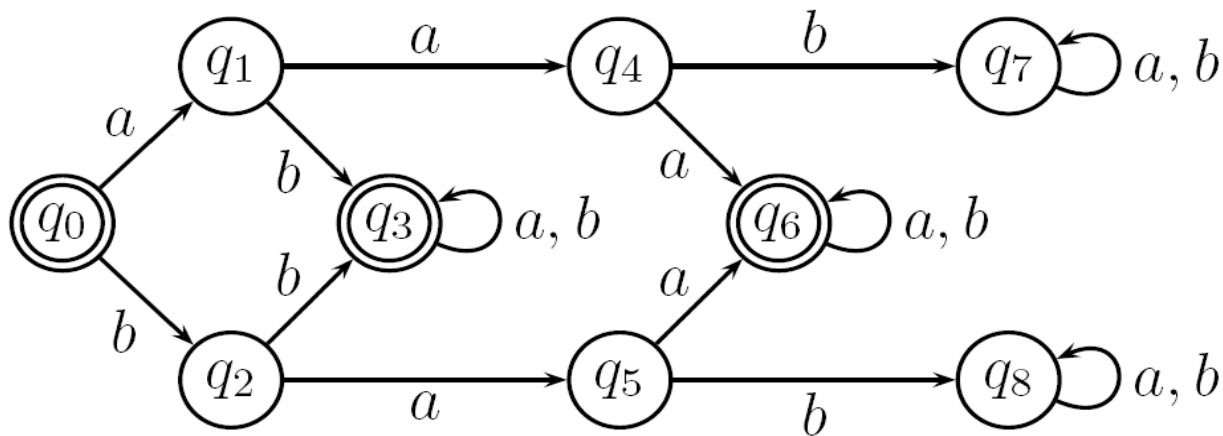


Συγχώνευση  $q_i, q_m$

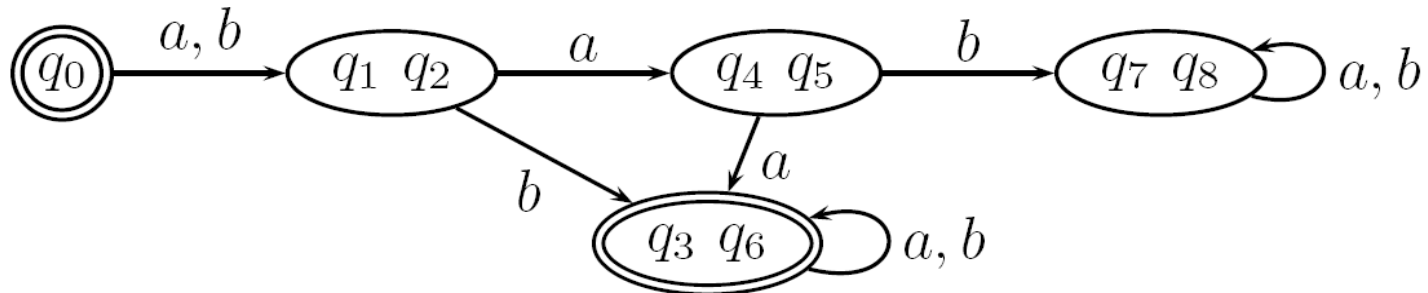


# Ελαχιστοποίηση DFA: παράδ/μα

## Αρχικό DFA



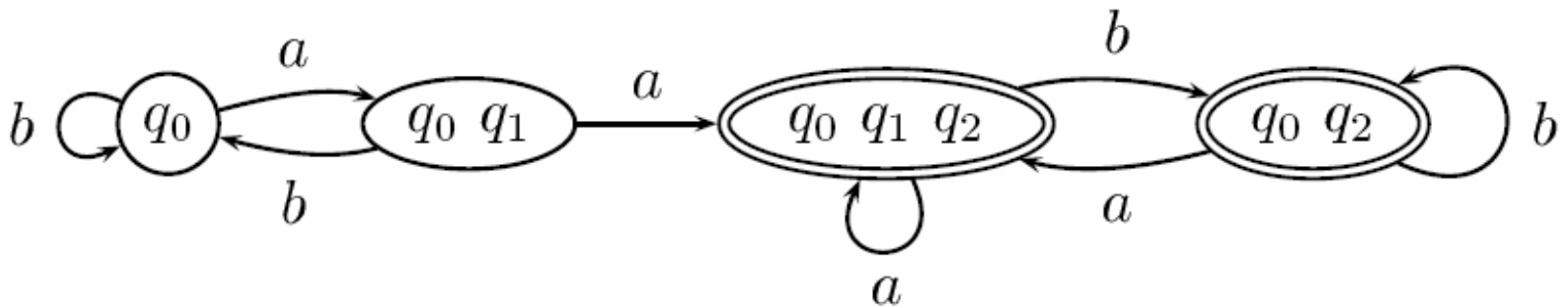
## Ελάχιστο DFA



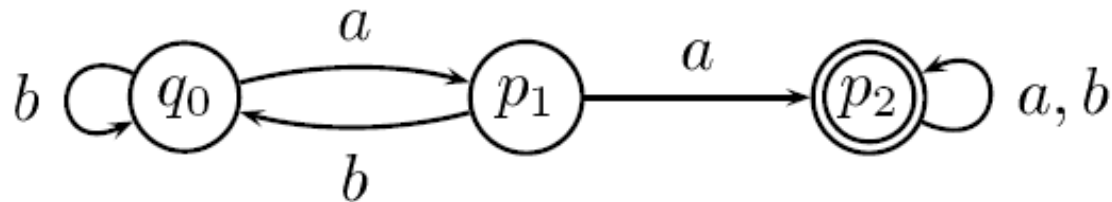
# Ελαχιστοποίηση DFA: 2<sup>ο</sup> παράδ/μα

$L_4 = \{ w \in \{a,b\}^* \mid w \text{ περιέχει } 2 \text{ συνεχόμενα } a \}$ :

## Αρχικό DFA



## Ελάχιστο DFA



# Μέθοδος ελαχιστοποίησης DFA

- Δύο καταστάσεις λέγονται  **$k$ -διακρίσιμες** αν με **κάποια** **συμβολοσειρά** μήκους  $k$  οδηγούν σε **διαφορετικό αποτέλεσμα**. Έτσι, δύο καταστάσεις είναι:
  - **$0$ -διακρίσιμες** αν η μία είναι τελική ενώ η άλλη όχι.
  - **$(i+1)$ -διακρίσιμες** αν με **κάποιο σύμβολο** οδηγούν σε  **$i$ -διακρίσιμες** καταστάσεις.
- Δύο καταστάσεις λέγονται **ισοδύναμες** αν δεν είναι  $k$ -διακρίσιμες για **οποιοδήποτε**  $k$ .
- Ιδέα μεθόδου: για κάθε  $i = 0, 1, 2, \dots$  εντοπίζουμε τα  $i$ -**διακρίσιμα** ζεύγη καταστάσεων έως ότου να μην προκύπτουν άλλα. Τα υπόλοιπα ζεύγη είναι **ισοδύναμα**.

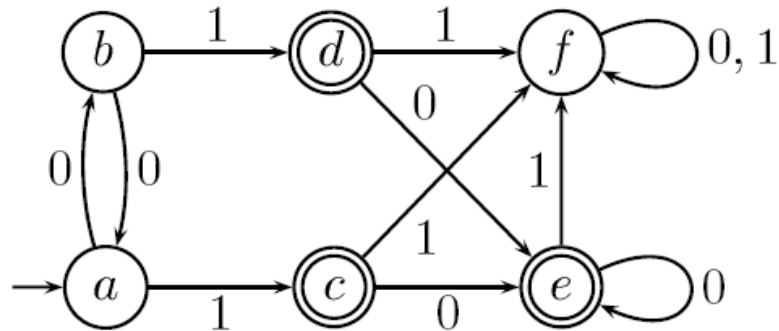


# Η μέθοδος συστηματικά

Κατασκευάζουμε **τριγωνικό πίνακα** για να συγκρίνουμε κάθε ζεύγος καταστάσεων. Γράφουμε  $X_k$  στην αντίστοιχη θέση του πίνακα κάθε φορά που διαπιστώνουμε ότι δύο καταστάσεις είναι  **$k$ -διακρίσιμες**, ως εξής:

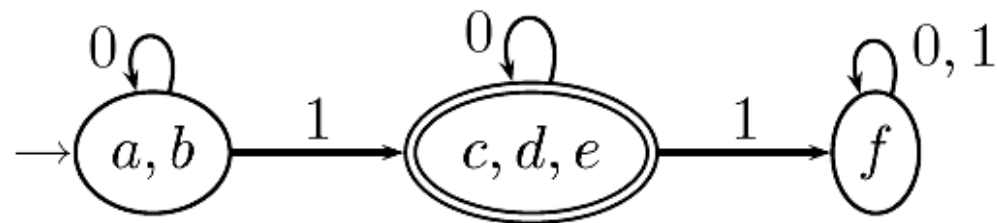
- Αρχικά γράφουμε  $X_0$  σε όλα τα ζεύγη καταστάσεων που είναι **0-διακρίσιμες** γιατί η μία είναι τελική και η άλλη όχι.
- Στη συνέχεια, στον "**γύρο**"  $i+1$ , εξετάζουμε όλα τα μη σημειωμένα ζεύγη και γράφουμε  $X_{i+1}$  σε ένα ζεύγος αν από τις δύο αυτές καταστάσεις με κάποιο σύμβολο οδηγούμαστε σε  **$i$ -διακρίσιμες** καταστάσεις (σημ/νες με  $X_i$ ).
- Επαναλαμβάνουμε μέχρι που σε κάποιο γύρο  $k$  να μην υπάρχει ζεύγος που να σημειωθεί με  $X_k$ .
- Τα μη σημειωμένα ζεύγη αντιστοιχούν σε **ισοδύναμες** καταστάσεις (που επομένως **συγχωνεύονται**).

# Παράδειγμα εφαρμογής της μεθόδου



$b$					
$c$	$X_0$	$X_0$			
$d$	$X_0$	$X_0$			
$e$	$X_0$	$X_0$			
$f$	$X_1$	$X_1$	$X_0$	$X_0$	$X_0$
	$a$	$b$	$c$	$d$	$e$

Τελικά οι ισοδύναμες καταστάσεις είναι  $a \equiv b$ ,  $c \equiv d \equiv e$ .  
 Το ελάχιστο αυτόματο φαίνεται στο παρακάτω σχήμα.



# Γλώσσες, αυτόματα, γραμματικές

- **Τυπικές γλώσσες:** χρησιμοποιούνται για την περιγραφή υπολογιστικών προβλημάτων αλλά και γλωσσών προγραμματισμού.
- **Αυτόματα:** χρησιμεύουν για την αναγνώριση τυπικών γλωσσών και για την κατάταξη της δυσκολίας των αντίστοιχων προβλημάτων.
- **Τυπικές γραμματικές:** άλλος τρόπος περιγραφής τυπικών γλωσσών. Κάθε τυπική γραμματική παράγει μια τυπική γλώσσα.

# Θεωρία γλωσσών και γραμματικών

Εφαρμογές σε:

- Ψηφιακή Σχεδίαση,
- Γλώσσες Προγραμματισμού,
- Μεταγλωττιστές,
- Τεχνητή Νοημοσύνη,
- Θεωρία Πολυπλοκότητας

Ιστορικά **σημαντικοί ερευνητές**:

- Chomsky, Backus, Rabin, Scott, Kleene, Greibach, κ.α.

# Τυπικές γλώσσες

- Πρωταρχικές έννοιες: **σύμβολα**, **παράθεση**.
- **Αλφάβητο**: πεπερασμένο σύνολο συμβόλων. Π.χ.  $\{0,1\}$ ,  $\{x,y,z\}$ ,  $\{a,b\}$ .
- **Λέξη** (ή συμβολοσειρά, ή πρόταση) ενός αλφαβήτου: πεπερασμένου μήκους ακολουθία συμβόλων του αλφαβήτου. Π.χ.  $011001$ ,  $abbbab$ .
- $|w|$  = **μήκος** λέξης  $w$ .
- $\epsilon$  = **κενή** λέξη,  $|\epsilon| = 0$ .
- Άλλες έννοιες: **πρόθεμα** (prefix), **κατάληξη** (suffix), **υποσυμβολοσειρά** (substring), **αντίστροφη** (reversal), **παλινδρομική** ή **καρκινική** (palindrome).

# Τυπικές γλώσσες (συν.)

- $vw$  = παράθεση λέξεων  $v$  και  $w$ .
- Ισχύει:  $\varepsilon x = x\varepsilon = x$ , για κάθε συμβολοσειρά  $x$ .
- ορισμός  $x^n$  με πρωταρχική αναδρομή:

$$\begin{cases} x^0 = \varepsilon \\ x^{k+1} = x^k x \end{cases}$$

- $\Sigma^*$ : το σύνολο όλων των λέξεων του αλφαβήτου  $\Sigma$ .
- Γλώσσα από το αλφάβητο  $\Sigma$ : κάθε σύνολο συμβολοσειρών  $L \subseteq \Sigma^*$ .

# Τυπικές γραμματικές

- Συστηματικός τρόπος μετασχηματισμού συμβολοσειρών μέσω **κανόνων παραγωγής**.
- **Αλφάβητο**: **τερματικά** και **μη τερματικά** σύμβολα και ένα **αρχικό σύμβολο** (μη τερματικό).
- Πεπερασμένο σύνολο κανόνων της μορφής  $\alpha \rightarrow \beta$ : ορίζουν δυνατότητα αντικατάστασης της συμβολοσειράς  $\alpha$  με την συμβολοσειρά  $\beta$ .
- Κάθε τυπική γραμματική **παράγει μια τυπική γλώσσα**: το σύνολο των συμβολοσειρών (με τερματικά μόνο) που παράγονται ξεκινώντας από το αρχικό σύμβολο.
- Λέγονται και **συστήματα μεταγραφής** (rewriting systems) αλλά και **γραμματικές δομής φράσεων** (phrase structure grammars).

# Παράδειγμα γραμματικής για την γλώσσα των περιττών αριθμών

$$S \rightarrow A 1$$
$$A \rightarrow A 0$$
$$A \rightarrow A 1$$
$$A \rightarrow \varepsilon$$

$S$ : το **αρχικό** σύμβολο

$A$ : **μη τερματικό** σύμβολο

$0, 1$ : **τερματικά** σύμβολα

$\varepsilon$ : η **κενή** συμβολοσειρά

- Τα  $S$  και  $A$  **αντικαθίστανται** με βάση τους κανόνες.
- Κάθε περιττός προκύπτει από το  $S$  με κάποια σειρά **έγκυρων** αντικαταστάσεων.
- **Κανονική** παράσταση:  $(0+1)^*1$



# Τυπικές γραμματικές: ορισμοί (i)

Μια τυπική γραμματική  $G$  αποτελείται από:

- ένα αλφάβητο  $V$  από *μη τερματικά* σύμβολα (μεταβλητές),
- ένα αλφάβητο  $T$  από *τερματικά* σύμβολα (σταθερές),  
τ.ω.  $V \cap T = \emptyset$ ,
- ένα πεπερασμένο σύνολο  $P$  από *κανόνες παραγωγής*, δηλαδή διατεταγμένα ζεύγη  $(\alpha, \beta)$ , όπου  $\alpha, \beta \in (V \cup T)^*$  και  $\alpha \neq \varepsilon$  (σύμβαση: γράφουμε  $\alpha \rightarrow \beta$  αντί για  $(\alpha, \beta)$ ),
- ένα *αρχικό σύμβολο* (ή αξίωμα)  $S \in V$ .

# Τυπικές γραμματικές: ορισμοί (ii)

Σύμβαση για τη χρήση γραμμάτων:

- $a, b, c, d, \dots \in T$  : πεζά λατινικά, τα αρχικά του αλφαβήτου, συμβολίζουν τερματικά
- $A, B, C, D, \dots \in V$  : κεφαλαία λατινικά συμβολίζουν μη τερματικά
- $u, v, w, x, y, z \dots \in T^*$  : πεζά λατινικά, τα τελευταία του αλφαβήτου, συμβολίζουν συμβολοσειρές τερματικών
- $\alpha, \beta, \gamma, \delta, \dots \in (V \cup T)^*$  : ελληνικά συμβολίζουν οποιοσδήποτε συμβολοσειρές (τερματικών και μη)

# Τυπικές γραμματικές: ορισμοί (iii)

Ορισμοί για τις παραγωγές:

- Λέμε ότι  $\gamma_1\alpha\gamma_2$  παράγει  $\gamma_1\beta\gamma_2$ , και συμβολίζουμε με  $\gamma_1\alpha\gamma_2 \Rightarrow \gamma_1\beta\gamma_2$ , αν ο  $\alpha \rightarrow \beta$  είναι κανόνας παραγωγής (δηλαδή  $(\alpha, \beta) \in P$ ).
- Συμβολίζουμε με  $\xRightarrow{*}$  το ανακλαστικό, μεταβατικό κλείσιμο του  $\Rightarrow$ , δηλαδή,  $\alpha \xRightarrow{*} \beta$  («το  $\alpha$  παράγει το  $\beta$ ») σημαίνει ότι υπάρχει μια ακολουθία:  
 $\alpha \Rightarrow \alpha_1 \Rightarrow \alpha_2 \Rightarrow \dots \alpha_k \Rightarrow \beta$ .
- Γλώσσα που παράγεται από τη γραμματική  $G$ :  
$$L(G) := \{w \in T^* \mid S \xRightarrow{*} w\}$$
- γραμματικές  $G_1, G_2$  *ισοδύναμες* αν  $L(G_1) = L(G_2)$ .

# Παράδειγμα τυπικής γραμματικής

$$G: V = \{S\}, T = \{a, b\}, P = \{S \rightarrow \varepsilon \mid aSb\}$$

$S \rightarrow \varepsilon \mid aSb$ : σύντμηση των  $S \rightarrow \varepsilon$  και  $S \rightarrow aSb$

Μία δυνατή ακολουθία παραγωγής:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbbb$$

Γλώσσα που παράγεται:

$$L(G) = \{a^n b^n \mid n \in \mathbb{N}\}$$

# Ιεραρχία Γραμματικών Chomsky

- τύπου 0: γενικές γραμματικές (general, phrase structure, semi-Thue).

$$\alpha \rightarrow \beta, \alpha \neq \varepsilon$$

- τύπου 1: γραμματικές με συμφραζόμενα ή μονοτονικές (context sensitive, monotonic).

$$\alpha \rightarrow \beta, |\alpha| \leq |\beta| \text{ (επιτρέπεται και: } S \rightarrow \varepsilon)$$

- τύπου 2: γραμματικές χωρίς συμφραζόμενα (context free).

$$A \rightarrow \alpha \quad (A \in V)$$

- τύπου 3: κανονικές γραμματικές (regular).

δεξιογραμμικές:  $A \rightarrow w, A \rightarrow wB$  ( $w \in T^*, A, B \in V$ ) ή

αριστερογραμμικές:  $A \rightarrow w, A \rightarrow Bw$  ( $w \in T^*, A, B \in V$ )



Γνήσια ιεράρχηση: τύπου 3  $\subset$  τύπου 2  $\subset$  τύπου 1  $\subset$  τύπου 0

# Ιεραρχία Chomsky: μια εκπληκτική σύμπτωση (;)

- τύπου 0  $\leftrightarrow$  **TM** (μηχανές Turing)
- τύπου 1  $\leftrightarrow$  **LBA** (γραμμικά περιορισμένα αυτόματα)
- τύπου 2  $\leftrightarrow$  **PDA** (pushdown automata)
- τύπου 3  $\leftrightarrow$  **DFA** (και NFA)

# Κανονικές Γραμματικές

- Οι **κανονικές γραμματικές** είναι γραμματικές όπου όλοι οι κανόνες είναι της μορφής:

- Δεξιογραμμικοί (right linear)

$$A \rightarrow wB \text{ ή } A \rightarrow w$$

- Αριστερογραμμικοί (left linear)

$$A \rightarrow Bw \text{ ή } A \rightarrow w$$

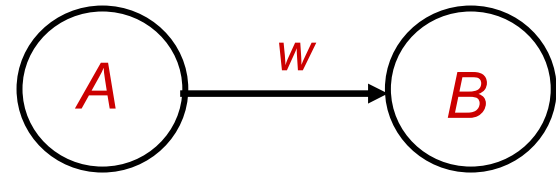
(όπου  $w$  είναι μια ακολουθία από τερματικά σύμβολα της γλώσσας)

- **Θεώρημα:** *οι κανονικές γλώσσες ταυτίζονται με τις γλώσσες που παράγονται από κανονικές γραμματικές.*

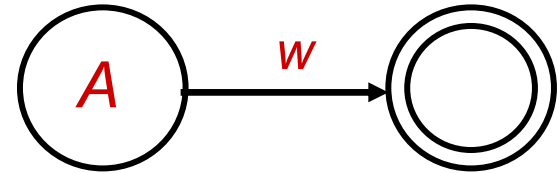
# Ισοδυναμία κανονικών γραμματικών και DFA

- Χρησιμοποιούμε τη δεξιογραμμική μορφή:

■  $A \rightarrow wB$  αντιστοιχεί με



■  $A \rightarrow w$  αντιστοιχεί με



■  $S$  αντιστοιχεί με  $q_0$



---

# Μία ακόμη ισοδυναμία!

**Θεώρημα:** *οι κανονικές γλώσσες ταυτίζονται με τις γλώσσες που περιγράφονται από κανονικές παραστάσεις.*

# Κανονικές παραστάσεις (regular expressions)

Έστω  $L, L_1, L_2$  γλώσσες επί του ίδιου αλφαβήτου  $\Sigma$ .

- $L_1 L_2 := \{uv \mid u \in L_1 \wedge v \in L_2\}$ : παράθεση
- $L_1 \cup L_2 := \{w \mid w \in L_1 \vee w \in L_2\}$ : ένωση
- $L_1 \cap L_2 := \{w \mid w \in L_1 \wedge w \in L_2\}$ : τομή
- $L^0 := \{\varepsilon\}, L^{n+1} := LL^n$
- $L^* := \bigcup_{n=0}^{\infty} L^n$ : άστρο του Kleene
- $L^+ := \bigcup_{n=1}^{\infty} L^n$

# Ορισμός κανονικών παραστάσεων

*Κανονικές παραστάσεις:* παριστάνουν γλώσσες που προκύπτουν από απλά σύμβολα ενός αλφαβήτου με τις πράξεις παράθεση, ένωση, και άστρο του Kleene.

- $\emptyset$  : παριστάνει κενή γλώσσα
- $\varepsilon$  : παριστάνει  $\{\varepsilon\}$
- $\alpha$  : παριστάνει  $\{\alpha\}$ ,  $\alpha \in \Sigma$
- $(r+s)$  : παριστάνει  $R \cup S$ ,  $R = L(r)$ ,  $S = L(s)$
- $(rs)$  : παριστάνει  $RS$ ,  $R = L(r)$ ,  $S = L(s)$
- $(r^*)$  : παριστάνει  $R^*$ ,  $R = L(r)$

όπου  $L(t)$  η γλώσσα που παριστάνεται από καν. παρ.  $t$

# Παραδείγματα κανονικών παραστάσεων

$$L_1 = a(a + b)^*$$

$$L_2 = (b^*ab^*a)^*b^* = (b + ab^*a)^*$$

$L_3$  δεν είναι δυνατόν να παρασταθεί με κανονική παράσταση

$$L_4 = (a + b)^*aa(a + b)^* \quad (\text{τουλάχιστον δύο συνεχόμενα } a)$$

$$\overline{L_4} = (a + \varepsilon)(ba + b)^* \quad (\text{όχι συνεχόμενα } a)$$

$$L_5 = a^*b^*$$

**Προτεραιότητα** τελεστών:

- άστρο Kleene
- παράθεση
- ένωση

# Ισοδυναμία κανονικών παραστάσεων και αυτομάτων

**Θεώρημα:** Μια γλώσσα  $L$  μπορεί να παρασταθεί με **κανονική παράσταση** ανν είναι **κανονική** (δηλαδή  $L=L(M)$  για κάποιο πεπερασμένο αυτόματο  $M$ )

Ιδέα απόδειξης:

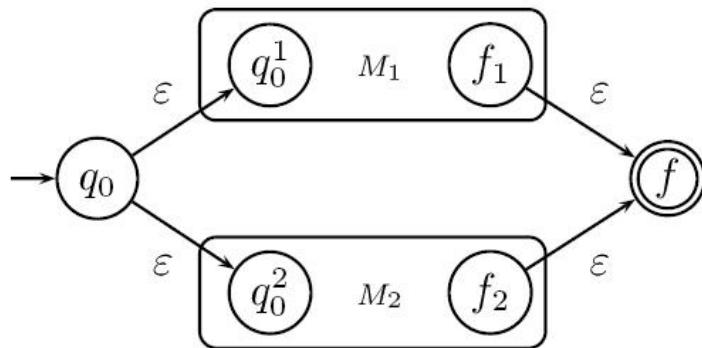
‘ $\Rightarrow$ ’: Επαγωγή στη δομή της κανονικής παράστασης  $r$ :

1. Επαγωγική Βάση:

$$r = \varepsilon: \rightarrow \textcircled{\textcircled{q_0}}, \quad r = \emptyset: \rightarrow \textcircled{q_0} \quad \textcircled{\textcircled{q_f}}, \quad r = a \in \Sigma: \rightarrow \textcircled{q_0} \xrightarrow{a} \textcircled{\textcircled{q_f}}$$

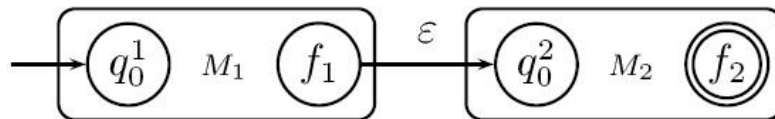
2. Επαγωγικό βήμα. Έστω ότι για  $r_1, r_2$  έχουμε αυτόματα  $M_1, M_2$ , με τελικές καταστάσεις  $f_1, f_2$ :

Περίπτωση α:  $r = r_1 + r_2$



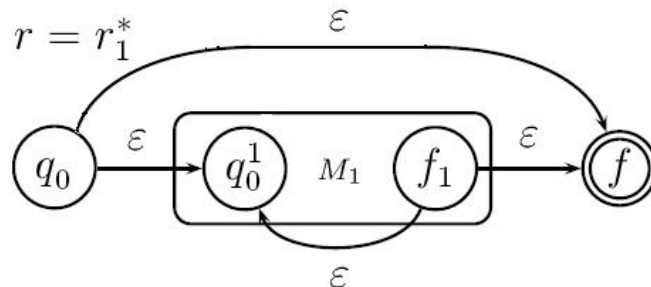
$$L(M) = L(M_1) \cup L(M_2)$$

Περίπτωση β:  $r = r_1 r_2$



$$L(M) = L(M_1)L(M_2)$$

Περίπτωση γ:  $r = r_1^*$

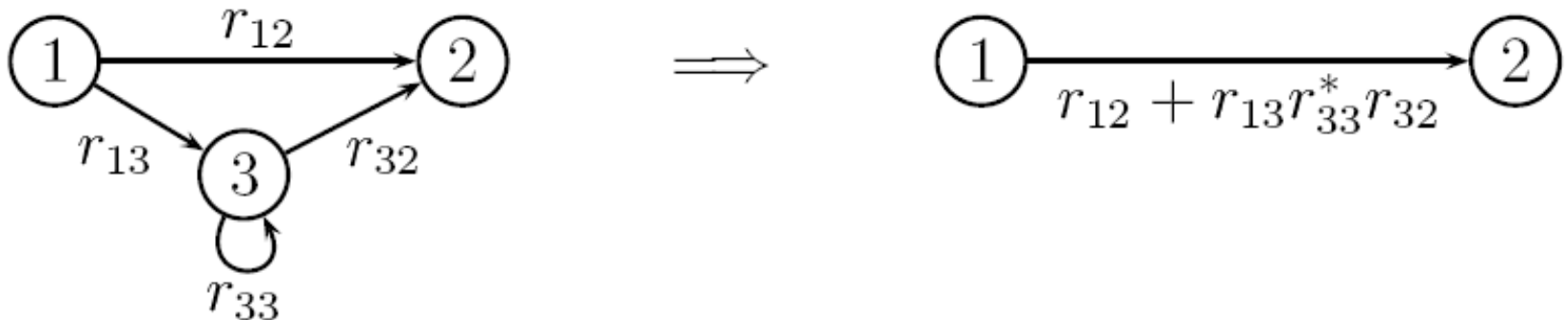


$$L(M) = L(M_1)^*$$

# Ισοδυναμία κανονικών παραστάσεων και αυτομάτων (συν.)

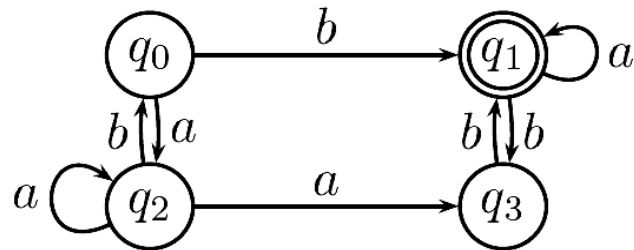
' $\leq$ ': Κατασκευή κανονικής παράστασης από FA (GNFA).

Απαλείφουμε ενδιάμεσες καταστάσεις σύμφωνα με το σχήμα:

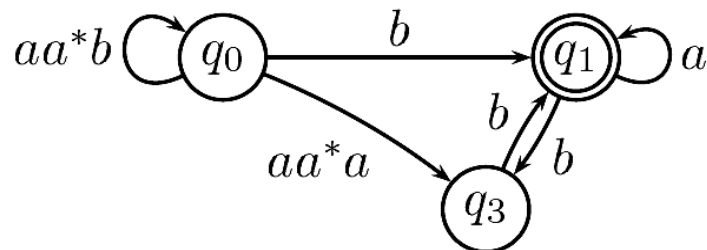


# Παράδειγμα κατασκευής κανονικής παράστασης από FA

Παράδειγμα 4.3.29. Έστω πως έχουμε το ακόλουθο FA:



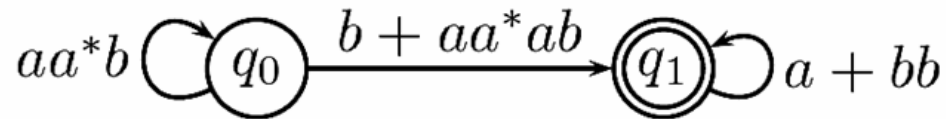
Επιλέγουμε να διαγράψουμε τη κατάσταση  $q_2$ , οπότε ενημερώνουμε τις μεταβάσεις των άλλων καταστάσεων:



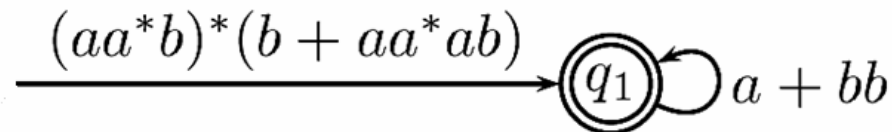


# Παράδειγμα κατασκευής κανονικής παράστασης από FA

Επιλέγουμε να διαγράψουμε τη κατάσταση  $q_3$ , οπότε ενημερώνουμε υπόλοιπα:



Επιλέγουμε να διαγράψουμε τη κατάσταση  $q_0$ :



Τελικά, η κανονική έκφραση που προκύπτει είναι:

$$(aa^*b)^*(b + aa^*ab)(a + bb)^*$$

# Ποιες γλώσσες είναι κανονικές;

- Όλες οι **πεπερασμένες**.
- Όσες σχηματίζονται από κανονικές με τις πράξεις: παράθεση, ένωση, άστρο Kleene,
- αλλά και συμπλήρωμα, τομή, αναστροφή (άσκηση), κ.ά.
- **Γινόμενο αυτομάτων** (product of automata): εναλλακτικός τρόπος κατασκευής DFA για τομή (αλλά και ένωση) κανονικών γλωσσών.

# Γινόμενο αυτομάτων

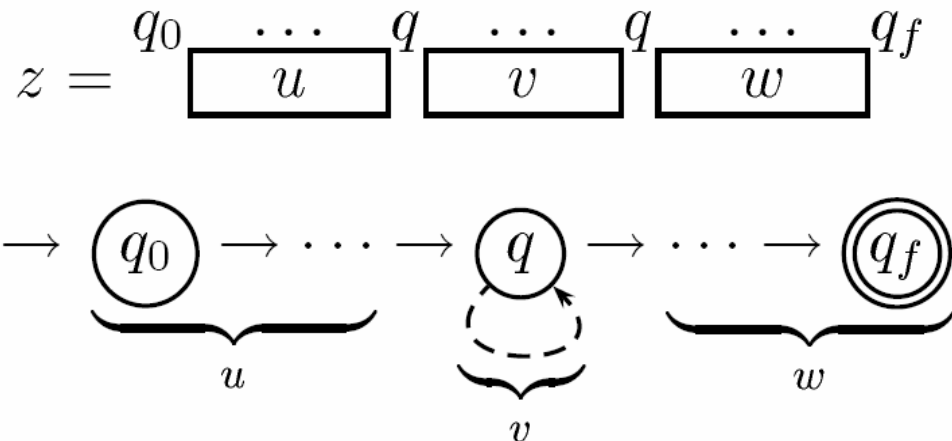
- Έστω δύο DFA  $M_1, M_2$  με  $n, m$  καταστάσεις αντίστοιχα ( $Q_1 = \{q_0, \dots, q_{n-1}\}, Q_2 = \{p_0, \dots, p_{m-1}\}$ ) και κοινό αλφάβητο, που αναγνωρίζουν γλώσσες  $L_1, L_2$  αντίστοιχα.
- Το γινόμενό τους είναι ένα DFA με  $m \cdot n$  καταστάσεις, μία για κάθε ζεύγος καταστάσεων του αρχικού αυτομάτου (σύνολο καταστάσεων  $Q = Q_1 \times Q_2$ ), το **ίδιο αλφάβητο** και αρχική κατάσταση  $(q_0, p_0)$ .
- Συνάρτηση μετάβασης:  $\delta'((q_i, p_j), \sigma) = (q_i', p_k') \Leftrightarrow \delta(q_i, \sigma) = q_i' \wedge \delta(p_j, \sigma) = p_k'$
- **Τελικές καταστάσεις**: ανάλογα με την πράξη μεταξύ  $L_1, L_2$  που θέλουμε. Για **τομή** θέτουμε ως τελικές ζεύγη όπου και οι δύο τελικές στα  $M_1, M_2$ , για **ένωση** ζεύγη που περιέχουν μία τουλάχιστον τελική.
- Παρατήρηση: εύκολη υλοποίηση και άλλων πράξεων μεταξύ  $L_1, L_2$  (διαφορά, συμμετρική διαφορά) με κατάλληλο ορισμό τελικών καταστάσεων.

# Είναι όλες οι γλώσσες κανονικές;

- Η απάντηση είναι «**όχι**»
- Για να το αποδείξουμε χρησιμοποιούμε ένα σημαντικό θεώρημα που λέγεται **Pumping Lemma** (Λήμμα Άντλησης)

# Pumping Lemma (διαίσθηση)

- Αν μια γλώσσα  $L$  είναι κανονική τότε την αποδέχεται ένα DFA με πεπερασμένο αριθμό καταστάσεων, έστω  $n$ .
- Έστω λέξη  $z$  με  $|z| \geq n$  που ανήκει στη γλώσσα, άρα γίνεται αποδεκτή από το αυτόματο.
- Καθώς επεξεργαζόμαστε το  $z$ , το αυτόματο πρέπει να περάσει ξανά από κάποια κατάσταση (αρχή περιστέρων):



- Αφού  $z = uvw \in L$  θα πρέπει και  $uv^i w \in L$ , για κάθε  $i \in \mathbb{N}$

# Pumping Lemma (με λόγια)

Έστω κανονική γλώσσα  $L$ . Τότε:

- υπάρχει ένας φυσικός  $n$  (= πλήθος καταστάσεων του DFA) ώστε:
- για κάθε  $z \in L$  με μήκος  $|z| \geq n$
- υπάρχει «σπάσιμο» του  $z$  σε  $u, v, w$ , δηλαδή  $z = uvw$ , με  $|uv| \leq n$  και  $|v| > 0$
- ώστε για κάθε  $i = 0, 1, 2, \dots$ :

$$uv^i w \in L$$

# Απόδειξη ότι μια γλώσσα δεν είναι κανονική

Χρήση του Pumping Lemma για να δείξουμε ότι μια (μη πεπερασμένη) γλώσσα  $L$  *δεν είναι κανονική*:

αν η  $L$  ήταν κανονική τότε:

- το PL λέει ότι υπάρχει  $n$ . Εμείς *για κάθε*  $n$
- *επιλέγουμε* κατάλληλο  $z \in L$  με μήκος  $|z| \geq n$
- το PL λέει ότι υπάρχει «σπάσιμο»  $z = uvw$ , με  $|uv| \leq n$  και  $|v| > 0$ . Εμείς *για κάθε* «σπάσιμο»  $z = uvw$
- *επιλέγουμε*  $i$  ώστε η λέξη  $uv^i w$  να μην είναι στη γλώσσα  $L$

ΑΤΟΠΟ

*(adversary argument)*

# Παράδειγμα χρήσης Pumping Lemma (i)

- **Θεώρημα.** Η γλώσσα  $L = \{z \mid z \text{ έχει το ίδιο πλήθος } 0 \text{ και } 1\}$  δεν είναι κανονική.

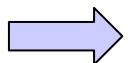
- **Απόδειξη:** Έστω  $L$  κανονική. Τότε:

- το PL λέει ότι υπάρχει  $n$ . Εμείς για κάθε  $n$

- επιλέγουμε  $z = 0^n 1^n \in L$  με μήκος  $|z| = 2n > n$

$$z = \underbrace{000000000\dots 0}_{n} \underbrace{111111111\dots 1}_{n}$$

- το PL λέει ότι υπάρχει «σπάσιμο»  $z = uvw$ , με  $|uv| \leq n$  και  $|v| > 0$ . Εμείς για κάθε «σπάσιμο»  $z = uvw$





# Παράδειγμα χρήσης Pumping Lemma (ii)

- παρατηρούμε ότι αναγκαστικά  $v = 0^k$  για κάποιο  $k$ :

$$w = \underbrace{0\dots0}_u \underbrace{0\dots0}_v \underbrace{0\dots011111111\dots1}_w$$

- και επιλέγουμε  $i = 2$ , διαπιστώνοντας ότι  $uv^i w = uv^2 w$  δεν ανήκει στην  $L$ .

ΑΤΟΠΟ

- Επομένως η  $L$  δεν είναι κανονική.

## Δεύτερο παράδειγμα χρήσης PL (i)

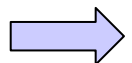
**Θεώρημα.** Η γλώσσα  $L = \{z \mid z=0^i1^j, i > j\}$  δεν είναι κανονική.

Απόδειξη: Έστω  $L$  κανονική. Τότε:

- το PL λέει ότι υπάρχει  $n$ . Εμείς για κάθε  $n$
- επιλέγουμε  $z = 0^{n+1}1^n \in L$  με μήκος  $|z| = 2n+1 > n$

$$\begin{array}{c} \text{■ } z = \underbrace{000000000\dots0}_{n+1} \underbrace{11111111\dots1}_n \end{array}$$

- το PL λέει ότι υπάρχει «σπάσιμο»  $z = uvw$ , με  $|uv| \leq n$  και  $|v| > 0$ . Εμείς για κάθε «σπάσιμο»  $z = uvw$



## Δεύτερο παράδειγμα χρήσης PL (ii)

- παρατηρούμε ότι αναγκαστικά  $v = 0^k$  για κάποιο  $k$ :

$$z = \underbrace{0\dots00}_{u} \underbrace{\dots00}_{v} \underbrace{\dots011111111\dots1}_{w}$$

- όμως, η επανάληψη του  $v$  δίνει λέξεις της γλώσσας
- από πρώτη άποψη αυτό φαίνεται προβληματικό...
- το λήμμα ορίζει ότι θα πρέπει για κάθε  $i \geq 0$ :  $uv^i w \in L$
- επιλέγουμε  $i = 0$ : η λέξη  $uv^0 w$  δεν είναι στην  $L$ .
- Επομένως η  $L$  δεν είναι κανονική.

ΑΤΟΠΟ

# Προσοχή στη χρήση του PL!

- Το Pumping Lemma είναι **αναγκαία** αλλά **όχι και ικανή** συνθήκη για να είναι μια γλώσσα κανονική.
- Υπάρχουν μη κανονικές γλώσσες που ικανοποιούν τις συνθήκες του!
- Επομένως χρησιμεύει **μόνο** για **απόδειξη μη κανονικότητας**.

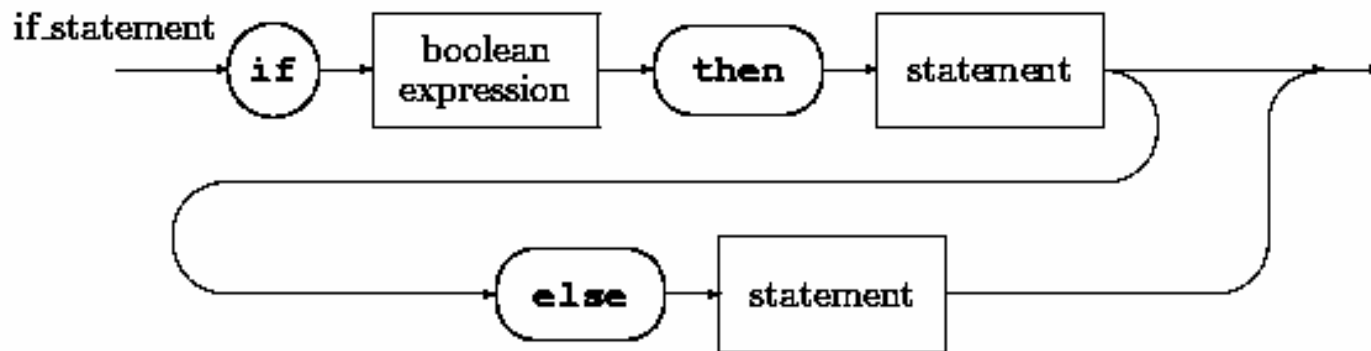
# Γραμματικές για μη κανονικές γλώσσες

- **Χωρίς συμφραζόμενα** (context free, CF): τύπου 2, αντιστοιχία με **αυτόματα στοίβας** (pushdown automata, PDA)
- **Με συμφραζόμενα** (context sensitive, CS): τύπου 1, αντιστοιχία με **γραμμικά περιορισμένα αυτόματα** (linear bounded automata, LBA)
- **Γενικές** (general): τύπου 0, αντιστοιχία με **μηχανές Turing** (Turing machines, TM)

# Γραμματικές χωρίς συμφραζόμενα (Context Free) (i)

Εφαρμογές σε:

- συντακτικό γλωσσών προγραμματισμού (Pascal, C, C++, Java)



- συντακτικό γλωσσών περιγραφής σελίδων web (HTML, XML), editors, ...

# Γραμματικές χωρίς συμφραζόμενα (Context Free) (ii)

- **Μορφή κανόνων:**  $A \rightarrow \alpha$ ,  $A$  μη τερματικό
- Παράδειγμα:

$$G_1: \quad V = \{S\}, \quad T = \{a, b\}, \quad P = \{S \rightarrow \varepsilon, S \rightarrow aSb\}$$

Δυνατή ακολουθία παραγωγής:

$$S \Rightarrow aSb \Rightarrow aaSbb \Rightarrow aaaSbbb \Rightarrow aaabbbb$$

Γλώσσα που παράγεται:

$$L(G_1) = \{a^n b^n \mid n \in \mathbb{N}^*\}$$

# Γραμματικές χωρίς συμφραζόμενα (Context Free) (iii)

- 2<sup>ο</sup> παράδειγμα:

$$G_2: T = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, *\}, V = \{S\}$$

και το  $P$  περιέχει τους κανόνες:

$$S \rightarrow S + S, \quad S \rightarrow S * S, \quad S \rightarrow 0|1|2|3|4|5|6|7|8|9$$

Δυνατές ακολουθίες παραγωγής:

$$S \Rightarrow 3, \quad S \Rightarrow S+S \Rightarrow 3+S \Rightarrow 3+S*S \Rightarrow 3+4*7$$



# Γραμματικές χωρίς συμφραζόμενα (Context Free) (iv)

- 3<sup>ο</sup> παράδειγμα:

$G_3$ :  $V = \{S, A, B\}$ ,  $T = \{a, b\}$ , και  $P$  περιέχει:

$S \rightarrow aB \mid bA$ ,  $A \rightarrow a \mid aS \mid bAA$ ,  $B \rightarrow b \mid bS \mid aBB$

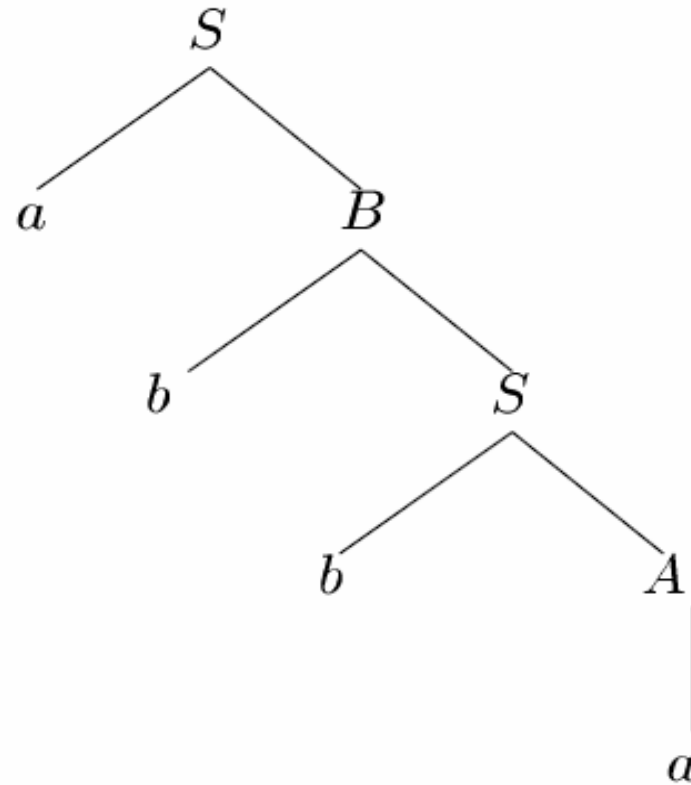
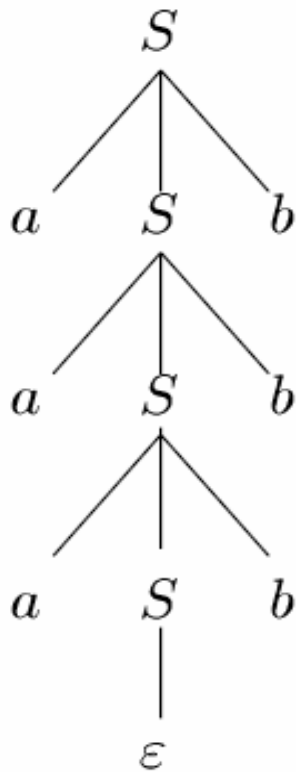
Δυνατή ακολουθία παραγωγής:

$S \Rightarrow aB \Rightarrow abS \Rightarrow abbA \Rightarrow abba$

Γλώσσα που παράγεται (όχι προφανές):

$$L(G_3) = \{w \in T^+ \mid w \text{ έχει ίσο αριθμό } a \text{ και } b\}$$

# Συντακτικά Δένδρα (parse trees) (i)



Φύλλωμα (leafstring):  $aaabbb$  και  $abba$  αντίστοιχα.

## Συντακτικά Δένδρα (parse trees) (ii)

Έστω  $G = \{V, T, P, S\}$  μια γραμματική χωρίς συμφραζόμενα.

Ένα δένδρο είναι **συντακτικό δένδρο της  $G$**  αν:

- Κάθε κόμβος του δένδρου έχει **επιγραφή**, που είναι σύμβολο (τερματικό ή μη τερματικό ή  $\epsilon$ ).
- Η επιγραφή της **ρίζας** είναι το  $S$ .
- Αν ένας εσωτερικός κόμβος έχει επιγραφή  $A$ , τότε το  $A$  είναι μη τερματικό σύμβολο. Αν τα παιδιά του, από αριστερά προς τα δεξιά, έχουν επιγραφές  $X_1, X_2, \dots, X_k$  τότε ο  $A \rightarrow X_1, X_2, \dots, X_k$  είναι κανόνας παραγωγής.
- Αν ένας κόμβος έχει επιγραφή  $\epsilon$ , τότε είναι φύλλο και είναι το μοναδικό παιδί του γονέα του.

## Συντακτικά Δένδρα (parse trees) (iii)

**Θεώρημα.** Έστω  $G = \{V, T, P, S\}$  μια γραμματική χωρίς συμφραζόμενα. Τότε  $S \xRightarrow{*} \alpha$  αν και μόνο αν υπάρχει συντακτικό δένδρο της  $G$  με φύλλωμα  $\alpha$ .

*Απόδειξη:*

‘ $\leq$ ’ : Με επαγωγή ως προς τον αριθμό των εσωτερικών κόμβων.

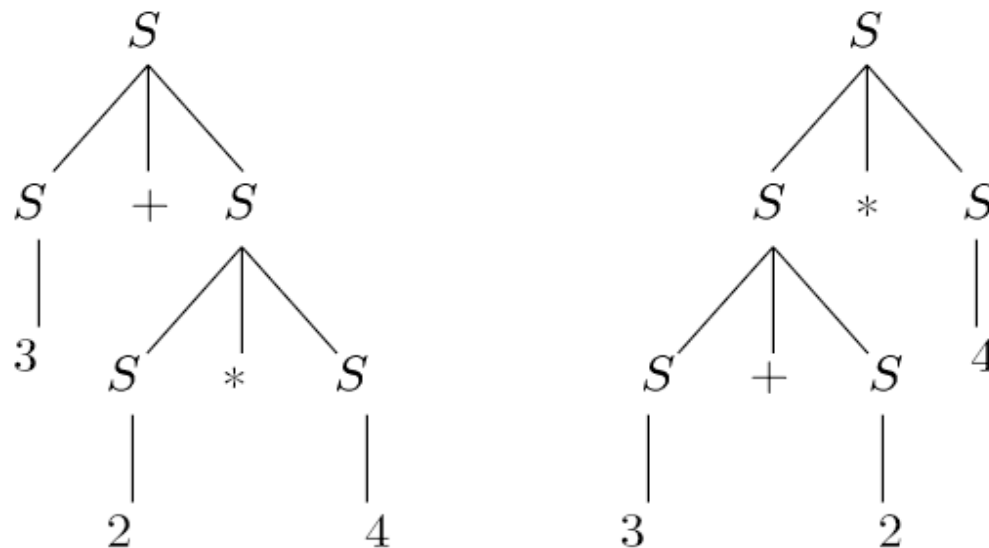
‘ $\Rightarrow$ ’ : Με επαγωγή ως προς τον αριθμό των βημάτων της ακολουθίας παραγωγών (άσκηση).

# Διφορούμενες γραμματικές

Μια γραμματική  $G$  ονομάζεται **διφορούμενη (ambiguous)** αν υπάρχουν δύο συντακτικά δένδρα με το ίδιο φύλλωμα  $w \in L(G)$

Παράδειγμα:

$G_2: T = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, *\}, V = \{S\}$ , και  $P$   
περιέχει:  $S \rightarrow S + S, S \rightarrow S * S, S \rightarrow 0|1|2|3|4|5|6|7|8|9$



# Αλγόριθμος αναγνώρισης για CF γραμματικές: CYK

- Με εξαντλητικό τρόπο μπορούμε να αποφασίσουμε αν μια συμβολοσειρά  $x$  παράγεται από μια γραμματική CF (χωρίς συμφραζόμενα) σε **εκθετικό όμως χρόνο**.
- Οι ιδιότητες της κανονικής μορφής Chomsky επιτρέπουν ταχύτερη αναγνώριση μιας συμβολοσειράς.
- **Αλγόριθμος CYK (Cocke, Younger, Kasami):** αποφασίζει αν μια συμβολοσειρά  $x$  παράγεται από μια γραμματική σε **χρόνο  $O(|x|^3)$** , αρκεί η γραμματική να δίνεται σε Chomsky Normal Form.

# Αυτόματα Στοίβας (PDA) (i)

- Έχουν ταινία εισόδου μιας κατεύθυνσης (όπως και τα FA) αλλά επιπλέον **μνήμη** υπό μορφή **στοίβας**.
- Πρόσβαση μόνο στην κορυφή της στοίβας με τις λειτουργίες:
  - **push**: τοποθετεί στοιχείο στην κορυφή της στοίβας
  - **pop**: διαβάζει και αφαιρεί στοιχείο από την κορυφή της στοίβας

# Αυτόματα Στοίβας (PDA) (ii)

- Παράδειγμα: PDA για αναγνώριση της γλώσσας

$$L = \{w c w^R \mid w \in (0 + 1)^*\}$$

1. push( $a$ ) στην στοίβα για κάθε 0 που συναντάς στην είσοδο, push( $b$ ) στην στοίβα για κάθε 1 που συναντάς στην είσοδο, μέχρι να συναντήσεις το  $c$
2. διάβασε το  $c$ ,  
κάνε pop τα στοιχεία της στοίβας και διάβαζε παράλληλα την είσοδο, αποδέξου αν υπάρχει συμφωνία των στοιχείων εισόδου με τα στοιχεία της στοίβας ( $a$  με 0,  $b$  με 1).



# Αυτόματα Στοιβάς (PDA) (iii)

**Ορισμός 4.4.12.** Ένα αυτόματο στοιβάς (push down automaton ή για συντομία PDA) είναι μία πλειάδα  $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ , όπου:

- $Q$ : πεπερασμένο σύνολο καταστάσεων,
- $\Sigma$ : αλφάβητο εισόδου,
- $\Gamma$ : αλφάβητο στοιβάς,
- $\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \rightarrow Pow(Q \times \Gamma^*)$  (πεπερασμένα υποσύνολα), η συνάρτηση μετάβασης (επιτρέπονται  $\varepsilon$ -κινήσεις και μη ντετερμινισμός),
- $q_0 \in Q$ : αρχική κατάσταση,
- $Z_0 \in \Gamma$ : αρχικό σύμβολο στην στοιβά,
- $F \subseteq Q$ : τελικές καταστάσεις.

# Αυτόματα Στοίβας (PDA) (iv)

Υπάρχουν δύο είδη PDA ως προς το αποδέχεται:

1. αποδέξου όταν βρίσκεσαι σε τελική κατάσταση αφού έχεις διαβάσει όλη την ταινία εισόδου, ανεξάρτητα από το τι υπάρχει στην στοίβα, ή,
2. αποδέξου όταν η στοίβα είναι άδεια αφού έχεις διαβάσει όλη την ταινία εισόδου, ανεξάρτητα από την κατάσταση στην οποία ευρίσκεσαι (σύμβολο:  $F = \emptyset$ ).

Αντίστοιχα ορίζουμε και την γλώσσα που αποδέχεται ένα PDA:

1. Γλώσσα που αποδέχεται σε τελική κατάσταση  $L_f(M)$
2. Γλώσσα που αποδέχεται με κενή στοίβα  $L_e(M)$

# Αυτόματα Στοιβάς (PDA) (v)

- Για να γίνει αποδεκτή η γλώσσα

$$L_1 = \{ww^R \mid w \in (0 + 1)^*\}$$

χωρίς δηλαδή ειδικό μεσαίο σύμβολο 'c' χρειαζόμαστε απαραίτητα **μη ντετερμινιστικό PDA**.

- Τα μη ντετερμινιστικά PDA είναι **γνησίως πιο ισχυρά** από τα ντετερμινιστικά.
- Με τον όρο PDA αναφερόμαστε συνήθως στα μη-ντετερμινιστικά PDA.

# Ισοδυναμία CF γραμματικών και PDA

**Θεώρημα.** *Τα παρακάτω είναι ισοδύναμα για μια γλώσσα  $L$ :*

1.  $L = L_f(M_2)$ ,  $M_2$  είναι PDA.
2.  $L = L_e(M_1)$ ,  $M_1$  είναι PDA.
3. Η  $L$  είναι γλώσσα χωρίς συμφραζόμενα (*context free*).

# Ποιες γλώσσες είναι Context Free;

- Όλες οι **κανονικές**.
- Επίσης όσες σχηματίζονται από γλώσσες CF με τις πράξεις: **παράθεση, ένωση, άστρο Kleene**.
- Αλλά όχι απαραίτητα με τις πράξεις **τομή, συμπλήρωμα**:  
π.χ. η γλώσσα  $\{a^n b^n c^n \mid n \in \mathbf{N}\}$  δεν είναι CF, ενώ είναι τομή δύο CF γλωσσών:

$$\{a^n b^n c^n \mid n \in \mathbf{N}\} = \{a^n b^n c^m \mid n, m \in \mathbf{N}\} \cap \{a^k b^n c^n \mid k, n \in \mathbf{N}\}$$

# Είναι όλες οι γλώσσες Context Free;

- Η απάντηση είναι «όχι».
- Για να το αποδείξουμε χρησιμοποιούμε ένα άλλο λήμμα άντλησης, το Pumping Lemma για γλώσσες χωρίς συμφραζόμενα.
- Βασίζεται στο συντακτικό δένδρο (περισσότερα σε άλλα μαθήματα: «Διακριτά», «Υπολογισιμότητα»).

# Γενικές Γραμματικές (i)

τύπου 0: γενικές γραμματικές (general, phrase structure, semi-Thue).

$$\alpha \rightarrow \beta, \alpha \neq \varepsilon$$

Παράδειγμα:  $\{a^{2^n} \mid n \in \mathbf{N}\}$

$$S \rightarrow AaCB$$

$$CB \rightarrow E \mid DB$$

$$aE \rightarrow Ea$$

$$AE \rightarrow \varepsilon$$

$$aD \rightarrow Da$$

$$AD \rightarrow AC$$

$$Ca \rightarrow aaC$$

# Γενικές Γραμματικές (ii)

**Θεώρημα.** Τα ακόλουθα είναι ισοδύναμα:

1. *Η  $L$  γίνεται αποδεκτή από μια μηχανή Turing*
2.  *$L=L(G)$ , όπου  $G$  είναι γενική γραμματική*

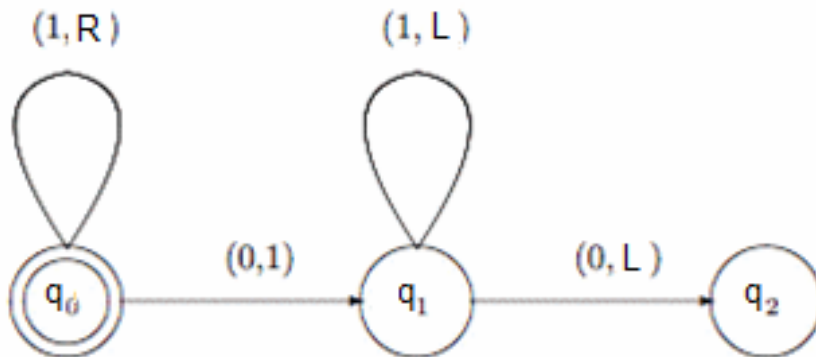
Μια τέτοια γλώσσα λέγεται και *αναδρομικά αριθμήσιμη* (recursively enumerable).



# Μηχανές Turing

Αυτόματα με **απεριόριστη ταινία**. Η είσοδος είναι αρχικά γραμμένη στην ταινία, η κεφαλή μπορεί να κινείται αριστερά-δεξιά, καθώς και να αλλάζει το σύμβολο που διαβάζει.

Παράδειγμα **συνάρτησης μετάβασης**:



$\langle q_0, 1, q_0, R \rangle$

$\langle q_0, 0, q_1, 1 \rangle$

$\langle q_1, 1, q_1, L \rangle$

$\langle q_1, 0, q_2, R \rangle$

# Γραμματικές με Συμφραζόμενα (context sensitive) (i)

τύπου 1: γραμματικές με συμφραζόμενα ή μονοτονικές (context sensitive, monotonic).

$$\alpha \rightarrow \beta, \quad |\alpha| \leq |\beta| \text{ (επιτρέπεται και: } S \rightarrow \varepsilon) \quad \alpha \neq \varepsilon$$

Λέγονται «με συμφραζόμενα» γιατί μπορούν να τεθούν στην εξής κανονική μορφή:

$$\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2, \quad \text{όπου } A: \text{ μη τερματικό και } \beta \neq \varepsilon$$

$\swarrow \quad \nearrow$   
*context*

# Γραμματικές με Συμφραζόμενα (context sensitive) (ii)

Παράδειγμα 4.7.1.  $1^n 0^n 1^n$ . C.s. γραμματική:

$$S \rightarrow 1Z1$$

$$Z \rightarrow 0 \mid 1Z0A$$

$$A0 \rightarrow 0A$$

$$A1 \rightarrow 11$$

*Μετατροπή σε κανονική μορφή:*

$$A0 \rightarrow H0$$

$$H0 \rightarrow HA$$

$$HA \rightarrow 0A$$

Άλλα παραδείγματα τέτοιων γλωσσών:  $\{a^n b^n c^n\}$ ,  $\{a^i b^j c^k \mid 0 \leq i \leq j \leq k\}$ ,  $\{ww \mid w \in \Sigma^*\}$ ,  $0^n 1^n 0^n 1^n$ .

# Ισοδυναμία γραμματικών CS και LBA

*Γραμμικά φραγμένο αυτόματο* (Linear Bounded Automaton, LBA) είναι μια μη ντετερμινιστική μηχανή Turing που η κεφαλή της είναι περιορισμένη να κινείται μόνο στο τμήμα που περιέχει την αρχική είσοδο.

**Θεώρημα.** Τα ακόλουθα είναι ισοδύναμα ( $L$  χωρίς  $\epsilon$ ):

1. Η  $L$  γίνεται *αποδεκτή από LBA*.
2.  $L$  είναι *context sensitive*.

# Ιεραρχία κλάσεων γλωσσών

Θεώρημα Ιεραρχίας. (γλώσσες χωρίς  $\epsilon$ ):

*regular*  $\subsetneq$  *context free*  $\subsetneq$  *context sensitive*  $\subsetneq$  *r.e.*

(*r.e.* = *recursively enumerable*)